Journal of Natural Sciences and Mathematics
Qassim University, Vol. 13, PP 01-11 (January 2020)

Using Randomisation into Iterated Greedy algorithm in order to Solve Capacitated Vehicle Routing Problem

Abdulwahab Almutairi

School of Mathematics, University of Qassim

a.abdulwahab.a@gmail.com

§1.1 Abstract

In this paper, we develop a heuristic algorithm for solving a Capacitated Vehicle Routing Problem (CVRP). In general, VRP is a well-known problem in which a number of vehicles are located at a central depot; each vehicle has a limited capacity and has to serve a number of geographically dispersed customers whose actual demands are known in advance. The contribution of the paper is to implement the Clarke and Wright Saving (CWS) algorithm and Iterated Greedy (IG) algorithm that used in the research efforts. Our results show that both CWS and IG combined with randomisation is a powerful algorithm for the well-known benchmark instances problem. Also, the proposed methodology is capable of finding useful trade-off solutions for the problem. We report the best solutions for 55 instances. Therefore, the results obtained are quite competitive when compared to the other algorithms found. Also, the results at best have been highly promising and useful for decision makers.

Keywords: The Capacitated Vehicle Routing Problem (CVRP), Randomized Clarke and Wright Saving algorithm (CWS), and Randomized Iterated Greedy algorithm (IG).

§1.2 Introduction

The Vehicle Routing Problem introduced by [2]. The decision maker has to serve a given set of n customers with known demand from s single depot using a given number of k of vehicles. Also, these vehicles have the same capacity. Some constraints have to applied such as each customer has to be served by exactly one vehicle, actual demand of each customer cannot be split and have be entirely satisfied using exactly one vehicle. The total demand assigned to each vehicle cannot exceed the vehicle capacity. The main objective of the problem is to minimize the total distance traveled by the vehicles. We refer to [2], [3], [14], [13], and [20] for a survey of vehicle routing

applications, model extensions, and solution methods. There are many researches that study probabilistic or randomized algorithms. For example, [10] has a review of biased randomization of heuristic. The aim of Angel (2017) paper is in the subset of randomized algorithms that include some type of bias in any of their random processes. [10] "A randomized algorithm uses random bits to make random choices during its execution. Unlike deterministic algorithms, different solutions are obtained every time the procedure is executed. The most successful approaches to solve large combinatorial problems take advantage of this feature to perform several iterations and collect the best overall output". Iterated Greedy algorithm is a meta-heuristic approach that is able to solve combinatorial optimization problems by iterating over greedy constructive heuristics. This algorithm is well-known in the literature of different fields such as the computer science and operational research, due to their simplicity and promising results. One of the key advantages of using Iterated Greedy algorithm is the use of a straightforward extension of an iterated local search to the context of greedy construction heuristics. In addition, it obtains good results in several applications.

The contribution of this paper is to implement/develop Iterated Greedy algorithm in order to solve Capacitated Vehicle Routing Problem. From this model, an efficient Iterated Greedy Algorithm composed of destruction and reconstruction procedures is implemented to generate feasible solutions. We added randomization to further improve the solution quality given by the destruction-reconstruction method. Empirical work indicates the effectiveness of the proposed heuristic. This paper is organized as follows: in Section 1.3 we discuss some literature about work done on Iterated Greedy algorithm. In Section 1.4, we propose the methodology in more detail in order to solve the problem. Then, Section 1.5 shows the computational results. Finally, in Section 1.6, we draw some conclusions based on the results of this work.

§1.3 Literature review

The vehicle routing problem and scheduling is considered as a very active research area on heuristic, meta-heuristic and hyper-heuristic techniques, mainly because the difficulty encountered by exact methods to find the optimal or near optimal solution for a medium or large instance. Some of these approaches are able to provide excellent effectiveness and efficiency at the expense of being utterly complicated. Iterated Greedy algorithm is considered as one of the interesting approaches that are used to solve these instances. According to [19], the Iterated Greedy algorithm has been proposed in order to solve scheduling problems and they explained the steps of the approach with more details. From review of the computational results, the results

display its great promise and the overall performance of this approach can be considered as a very good effort. In [7] aimed to deal with the minimization of the maximum completion time of the jobs. They tried to solve a problem called unrelated parallel machine scheduling, where the processing time depends on a machine. Also, they presented a set of simple Iterated Greedy local search based meta-heuristics that generated very good solutions in terms of short amount of time, as shown in the computational results. Different research proposed Iterated Greedy algorithm to find the minimization of total weighted completion time for flexible flow line with sequence dependent set-up times [16]. They also find their final results are effective and that is demonstrated through comparison.

[9] proposed a variable greedy algorithm with differential evolution in order to find a solution for the no-idle permutation flow-shop scheduling problem. The fundamental idea behind this approach is to apply differential evolution to determine two main parameters for the Iterated Greedy algorithm. In paper [17] they used an effective iterated greedy algorithm for mixed no-idle permutation flow-shop scheduling problem. A generalization of both the regular permutation flow-shop and no-idle permutation flow-shop scheduling problem has been proposed for the first time in this paper. The idea behind no-idle flow-shop is that machines cannot be empty after finishing one job and before starting the next job. Therefore, start times of jobs have to be delayed in order to guarantee this constraint. They generated a good initial solution by using an NEH heuristic. Also, they implemented a local search to emphasize intensification and exploration in the Iterated Greedy algorithm. Iterated Greedy algorithm can be used to solve other problems such as Market Segmentation Problem with Multiple Attributes which is proposed by [11].

Nowadays, Iterated Greedy algorithm has been successfully implemented in order to deal/solve a variety of combinatorial optimization problems and other case studies. [8] proposed Iterated Greedy algorithm for solving the blocking flow-shop scheduling problem (BFSP) with the make-span criterion. In terms of the computational results, the performance of the Iterated Greedy algorithm depended on the speed-up methods employed. Also, the results showed that the Iterated Greedy algorithm with the speed-up methods is equivalent to the best performance algorithms from the literature. [1] introduced an effective Iterated Greedy algorithm for unrelated parallel machines with different capacities and unequal ready times in order to minimize the make-span. The problem can be described as follows: a machine can process a number of jobs simultaneously as a batch ready time, as long as capacity of the machine has not been exceeded. The results ~~that~~

obtained in the paper showed that the Iterated Greedy algorithm has outperformed compared to other meta-heuristics for similar problems. [6] attempted to solve the make-span permutation flow-shop problem by using an Iterated Greedy algorithm with optimization. The main step of this paper is to explore the possibility of re-optimizing partial solutions that achieved, after the solution destruction step, in an Iterated Greedy algorithm for permutation flow-shop problem.

Recently, many researchers implemented Iterated Greedy algorithm with different methods in order to solve a specific problem. For example, [12] presented an Iterated Greedy algorithm for solving a market segmentation problem with multiple attributes. [15] proposed IG algorithm in order to solve the industrial problem, especially in reprocessing shops of remanufacturing systems. The aim of using this approach in this problem is to minimize the total family flow time for job-shop scheduling, with job families and sequence-dependent set-ups. An effective IG algorithm has been used to solve flow-shop scheduling problems with time lags [21].

§1.4 Methodology

The proposed algorithm in this paper is based on Iterated Greedy algorithm. Iterated Greedy algorithm considers as one of heuristic that used to solve combinatorial optimization problems by iterating over greedy constructive heuristic. [19] has introduced the Iterated Greedy algorithm in order to solve the scheduling side. The process in this paper is to use randomization into the Iterated Greedy algorithm to solve the capacitated vehicle routing problem. Iterated Greedy algorithm generates a sequence of solutions by iterating over greedy constructive heuristics using two main phases: namely destruction and construction. During the destruction phase some customers are removed from a constructed complete candidate solution. Then, the construction procedure applies a greedy constructive heuristic to construct a complete solution. Iterated Greedy iterates over these steps until some stopping criterion is met. The aim in this paper is to try to extend the work by adding a randomization inside the Iterated Greedy algorithm in order to improve the solutions of the problem.

The probability can be assigned to the destruction phase then continue to implement the IG algorithm approach. The above explanation of the IG algorithm is when the demand is deterministic and the safety stocks in all vehicles are not considered. The solutions for the deterministic case were computed using the IG algorithm. Randomization was then added by

assigning different probabilities to each potential customer. Customers with high probability are then chosen first, before customers with low probability, although all customers are selected in the end. It is also important to note that in doing so, an attempt is made to avoid the issue of selecting the appropriate size of the restricted list, and to provide a guarantee that the probabilities of being selected are always proportional to the position of each customer in the route.

§1.5 Computational results

In this part, the methodology described in this paper was implemented as a Java application. In the experiments, a Macintosh HD on 2.3 GHz Intel HD Graphics 4000 1024 MB with Intel Core i7 with 4 GB 1600 MHz DDR3 was used to perform the computational experiments. The instance considered a set of 55 classical Capacitated Vehicle Routing Problem instances, namely instances "A", "B", "E", "F", and "P". The main characteristics of these instances, such as vehicle capacity, customer location, number of both vehicles and customer are described online at https://www.coinor.org/SYMPHONY/branchandcut/VRP/data/index.htm.old. We compare the performance of Randomized Clarke and Wright Saving algorithm to Randomized Iterated Greedy local search algorithm for the Capacitated Vehicle Routing Problem benchmark instances. To facilitate the comparison, we utilize the same problem instances as in the website. The name of each problem instance indicates the number of customers (including the depot) and the number of vehicles that can be used. For example, A-n32-k5 indicates that A represents the instance class and 32 presents the number of customers in this class, whilst 5 presents the number of the vehicles that will be used to serve these customers.

In this paper, several computational experiments designed to test the Vehicle Routing Problem instances were conducted. During the implementation, the vehicle capacity has been considered as 100% and also safety stock has not been used, therefore there is a chance that solutions can suffer from route failures. Tables below summarise the complete computational results of both approaches for all 55 instances; the best results were found to use two different approaches for solving VRP with the same well-known benchmark problems. The following abbreviations are used:

- RIG: Randomised Iterated Greedy algorithm.
- RCWS: Randomised Clarke and Wright algorithm.

The results achieved from the preliminary analysis of both approaches are shown in these tables and the second column of the table shows the name of the instances used for all the approaches. The randomisation is used to improve the solutions in both approaches. Column 3 is related to solutions obtained with the Randomised Clarke and Wright Saving algorithm (RCWS). The third column is related to solutions obtained by Randomised Iterated Greedy algorithm (RIG) approach. The randomisation is used in both CWS and IG to improve the solutions for the instances. The last column shows the % improvement between the RCWS and RIG algorithm. Comparative results between the approaches are shown in the last column whilst the bold numbers represent the instances that improved. The improvement rate is defined as: $Improvement\ rate = \frac{Final\ best\ solution - previous\ best\ solution}{Final\ best\ solution} * 100$. The bottom row in Table 2 shows the average value of the results for each approach for all instances of the problem. Overall, our algorithm was able to solve instances with up to 121 customers. Furthermore, a comparison of the solutions is shown; the solution for the RIG is actually able to outperform the solution related to the RCWS. In addition, the standard version of RIG approach improves the solutions for the RCWS on all 55 instances.

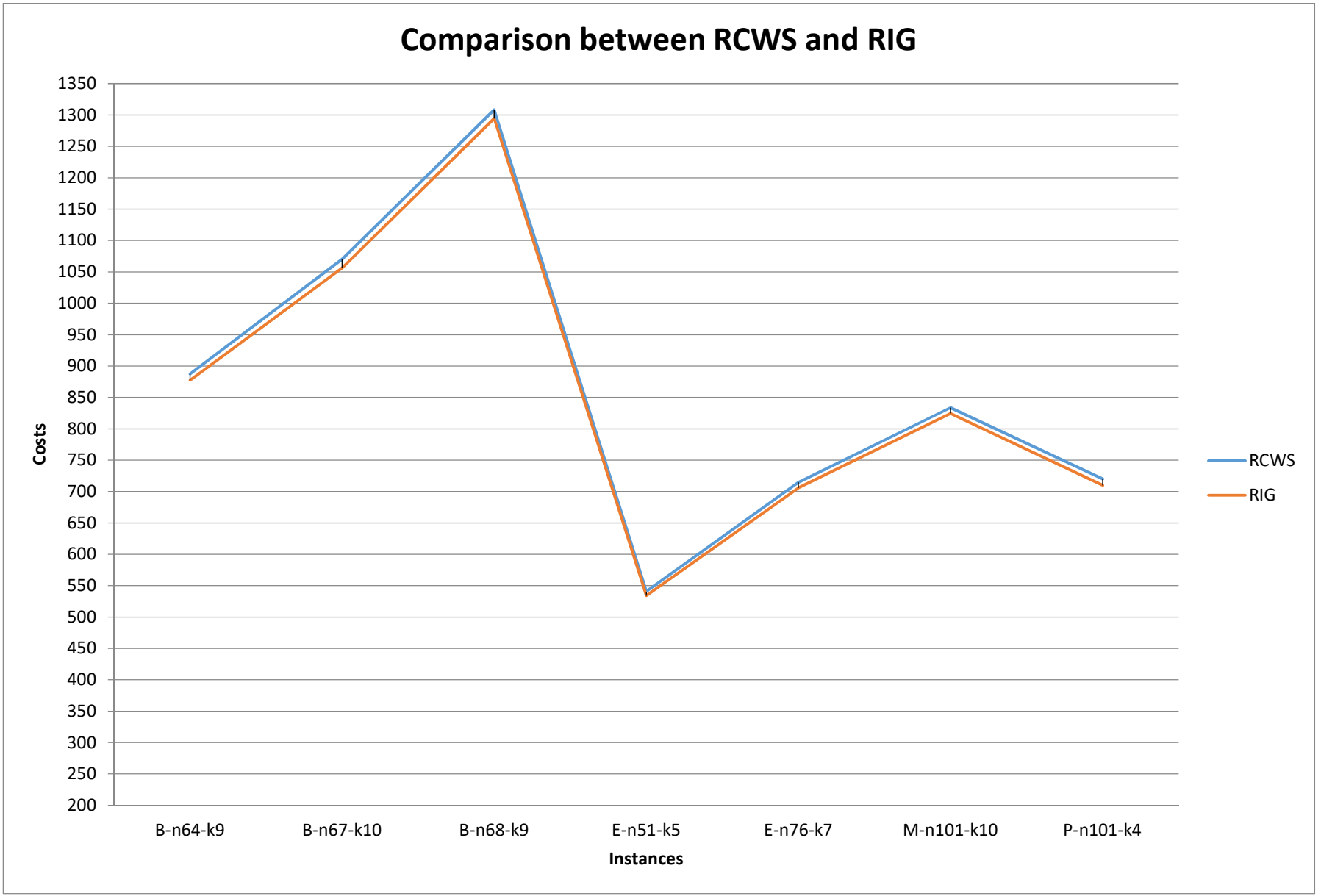
As presented in Tables 1 and 2, the last columns are the most interesting, which shows the improvement between the approaches. In the experiments on 55 instances, using IG with randomisation led to an improvement in the solutions and thus improved all instances compared to the other approach. From the tables, in terms of the improvement column, in 36 out of 55 instances the improvement of the total cost is less than 1%, in 12 out of 55 instances it is more than 1%. For example, for instances that produced better results, A-n38-k5, A-n45-k6, B-n68-k9, F-n72-k4 and P-n101-k4 were found to be the best solutions and obtained results of above 1%. However, 7 out of 55 instances had no improvement. Notice that the smallest improvement, that is zero, has been omitted from the calculation of the average in the last row. In addition, the average value of the results between RCWS and RIG for all instances of the problem is equal to 0.74%. Therefore, the average solution by RIG approach showed better results than the other approach.

Name of instance		RCWS	RIG	GAP
1	A-n32-k5	787.8	**787.8**	0.00%
2	A-n33-k5	665.8	**663.8**	0.30%
3	A-n33-k6	744.3	**742.9**	0.18%
4	A-n37-k5	679.8	**674.2**	0.83%
5	A-n38-k5	747.1	**735.9**	1.52%
6	A-n39-k6	835.4	**834.3**	0.13%
7	A-n45-k6	968.7	**957.1**	1.21%
8	A-n45-k7	1156.8	**1154.6**	0.19%
9	A-n55-k9	1082.7	**1080.1**	0.24%
10	A-n60-k9	1377.4	**1371.7**	0.41%
11	A-n61-k9	1058	**1051**	0.66%
12	A-n63-k9	1654	**1642.3**	0.71%
13	A-n65-k9	1197.5	**1197.5**	0.00%
14	A-n80-k10	1802.1	**1797.2**	0.27%
15	B-n31-k5	676.1	**676.1**	0.00%
16	B-n35-k5	970.4	**963.7**	0.69%
17	B-n39-k5	556.4	**553.4**	0.54%
18	B-n41-k6	840.3	**837.6**	0.32%
19	B-n45-k5	757.2	**755.2**	0.26%
20	B-n50-k7	748.8	**745**	0.51%
21	B-n52-k7	762	**760.5**	0.19%
22	B-n56-k7	729.3	**720.6**	1.20%
23	B-n57-k9	1609	**1604.3**	0.29%
24	B-n64-k9	887.6	**877.5**	1.15%
25	B-n67-k10	1070	**1056.2**	1.30%
26	B-n68-k9	1308.3	**1294.4**	1.07%
27	B-n78-k10	1263.5	**1257.3**	0.49%

Table 1. Comparison of methodologies for 55 selected VRP instances

Name of instance		RWCS	RIG	GAP
28	E-n22-k4	375.3	**375.3**	0.00%
29	E-n30-k3	511.3	**506.7**	0.90%
30	E-n33-k4	843.1	**842.8**	0.03%
31	E-n51-k5	540.7	**534.2**	1.21%
32	E-n76-k7	714.7	**705.8**	1.26%
33	E-n76-k10	867.3	**864.5**	0.32%
34	E-n76-k14	1055.7	**1045.6**	0.96%
35	F-n45-k4	728	**724.1**	0.53%
36	F-n72-k4	249.3	**236.2**	5.54%
37	F-n135-k7	1165	**1160**	0.43%
38	M-n101-k10	833.5	**824.5**	1.09%
39	M-n121-k7	1059.3	**1054.7**	0.43%
40	P-n19-k2	216.1	**212.7**	1.59%
41	P-n20-k2	217.4	**217.4**	0.00%
42	P-n22-k2	218.7	**217.9**	0.36%
43	P-n22-k8	588.8	**588.8**	0.00%
44	P-n40-k5	463	**462.9**	0.02%
45	P-n50-k8	635.3	**632.7**	0.41%
46	P-n50-k10	708.9	**705.3**	0.51%
47	P-n51-k10	747.6	**746.7**	0.12%
48	P-n55-k7	2441.7	**2441.7**	0.00%
49	P-n55-k15	960.6	**959.6**	0.10%
50	P-n60-k10	765.8	**763.2**	0.34%
51	P-n65-k10	820.7	**812.6**	0.99%
52	P-n70-k10	853.6	**846.4**	0.85%
53	P-n76-k4	632	**626.8**	0.82%
54	P-n76-k5	655.5	**651.8**	0.56%
55	P-n101-k4	720.1	**710**	1.42%
	Average	864.09	**859.32**	0.74%

Table 2. Comparison of methodologies for 55 selected VRP instances


Comparison between RCWS and RIG
1350
1300
1250
1200
1150
1100
1050
1000
950
900
850
800
750
700
650
600
550
500
450
400
350
300
250
200
Costs
B-n64-k9
B-n67-k10
B-n68-k9
E-n51-k5
E-n76-k7
M-n101-k10
P-n101-k4
Instances
RCWS
RIG

The above chart shows the best solution obtained by both Randomised CWS and Randomised IG for several instances of the total costs. In brief, the detail along the left side of the graph above shows the total costs. Whereas the bottom of the graph above shows the different classes from the instances. The chart presented the full comparison between total costs obtained by these two approaches. In summary, it is apparent that there was a slight improvement in the solutions to the instances, such as B-n64-k6. Also, a clear improvement can be seen in the results using the RIG algorithm; this approach out-performs the other approach in the improvement of solutions. As shown in the preliminary result tables it compares two types of results which are Randomized Clarke and Wright Saving algorithm (RCWS) and Randomized Iterated Greedy algorithm (RIG). We finished the section verifying that the RIG provided better results to the problem than RCWS. It is significant to notice that time is considered to be an important part and for this reason this methodology will use the local search implicitly with time constraint in the future work in order to improve the solutions that were obtained by this method.

§1.6 Conclusion

This paper has introduced a randomisation into IG algorithm. In this paper, IG algorithm has been extended by using randomisation. Also, randomised IG has been proposed and analysed the solutions on these instances. In terms of the experiments, the randomised parallel version of Clarke and Wright algorithm provided results not as good as Randomised IG algorithm, as can be seen in the preliminary results tables where it compares two types of results. In this paper, it is significant to note that results of these algorithms do not employ any local search process. Future work will consider different variants such as stochastic demand. In practice, this type of problem occurs when customer demand is not known in advance, but is known when the vehicle arrives at a customer location.

§1.7 Reference

1. Arroyo. J. E,. Joseph, Y., Leung, T., An effective iterated greedy algorithm for scheduling unrelated parallel batch machines with non-idential capacities and unequal ready times. Computers and industrial enginnering 105 (2017) 84-100.
2. Bodin, L., Golden, B.L. Assad, A.A. Ball, M.O. Routing and scheduling of vehicles and crews, the state of the art, Comput. Oper. Res. 10 (2) (1983) 63–212.
3. Christo6des, N. Vehicle routing, in: Lawler, E.L. Lenstra, J.K. Rinnooy Kan, A.H.G. Shmoys D.B., (Eds.), The Traveling Salesman Problem, Wiley, Chichester, 1985, pp. 431–448.
4. Christo6des, N. Mingozzi, A. Toth, P. The vehicle routing problem, in: Christo6des, N. Mingozzi, A. Toth, P. Sandi C. (Eds.), Combinatorial Optimization, Wiley, Chichester, 1979, pp. 315–338.
5. Dantzig, G. B. and Ramser. J. H. The truck dispatching problem. Management Science, 6(1):80–91, 1959.

6. Dubois-Lacoste, J., Pagnozzi, F., Stutzle, Thomas., (2017). An Iterated greedy algorithm with optimisation of partial solutions for the makespan permutation flowshop problem. Computer and Operations Research 81 (2017) 160-166.
7. Fanjul-Peyro, L., Ruiz, R., (2010). Size-reduction heuristics for the unrelated parallel machines scheduling problem.
8. Fatih Tasgetiren, M., Kizilay, A., Pan, Q., Suganthan, P, N., Iterated greedy algorithms for the blocking flowshop scheduling problem with makespan criterion. Computer and operations research 77 (2017) 111-126.
9. Fatih Tasgetiren, M., Pan, Q., Suganthan, P, N., Buyukdagli., O., (2013). A variable iterated greedy algorithm with differential evolution for the no-idle permutation flowshop scheduling problem. Computer and operations research 40(2013) 1729-1743.
10. Grasas, A., Juan, A, A., Faulin, J., Armas, J, D., Ramalhinho., (2017). Biased Randomisation of heuristics using skewed probability distribution: a survey and some applications. Computer and industrial engineering 110 (2017) 216-228.
11. Huerta-Munoz, D., Rios-Mercado, R, Z., Ruiz, R. An iterated greedy heuristic for a market segmentation problem with multiple attributes. European jurnal of operational research 261 (2017) 75-87.
12. Huerta-muñoz, D. L., Rios-Mercado, R, Z., Ruiz, R. (2014). An iterated greedy heuristic for a market segmentation problem with multiple attributes. Report Number: PISIS-2014-02, Affiliation: Graduate Program in Systems Engineering, UANL.
13. Laporte, G. Osman, I.H. Routing problems: a bibliography, Ann. Oper. Res. 61 (1995) 227–262.
14. Magnanti, T.L. Combinatorial optimization and vehicle fleet planning: Perspectives and prospects, Networks 11 (1981) 179–214.
15. Kim, Ji-su., Park, J, Lee, D,. (2016). Iterated greedy algorithms to minimise the total family flow time for job-shop scheduling with job families and sequence-dependent set-ups. Engineering optimization. Vol. 49, No. 10, 1719-1732.
16. Naderi, B., Zandieh, M., Mohammad, S., & Fatemi, T. (2009). An iterated greedy algorithm for flexible flow lines with sequence dependent setup times to minimize total weighted Completion Time. Journal of Industrial Engineering, 3, 33–37.
17. Pan, Q., & Ruiz, R. (2014). An effective iterated greedy algorithm for the mixed no-idle permutation flowshop scheduling problem. Omega, 44(2014)41-50.
18. Ruiz, R., Stutzle, T. (2005). An iterated greedy algorithm for the flowshop problem with sequence dependent setup times. The 6th Metaheuristics International Conference, 817–823.
19. Ruiz, R., Stutzle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European Journal of Operational Research, 177, 2033–2049.
20. Toth, P. Vigo, D. Exact solution of the vehicle routing problem, in: T.G. Crainic, G. Laporte (Eds.), Fleet Management and Logistic, Kluwer Academic Publisher, Boston (MA), 1998, pp. 1–31.
21. Zhao, N., Ye, S., Li, Kaidian., Chen, S., (2017). Effective Iterated greedy algorithm for flow-shop scheduling problems with time lags. Chin, J, Mech, Eng. (2017) 30:652-662.