Metaheuristic Optimization - CAT 2

Biased Randomization

Óscar Gómez Álvarez
Juan José Rodriguez Aldavero

September 23, 2022

Contents

1 Reading and assessment of papers

1.a Biased randomization of heuristics using skewed probability distributions: A survey and some applications

A large number of computational problems are untreatable by means of exact and deterministic classical procedures because they are NP-hard and the solution space grows exponentially with the size of the problem. This motivates the development of alternative techniques among which heuristic and metaheuristic procedures can be highlighted.

Randomized heuristics are useful to solve a myriad of these problems, intractable through classical procedures. In this paper, the authors briefly review the main applications of biased randomization over the field of metaheuristic algorithms. The biased randomization procedure can be applied to metaheuristic algorithms in order to deal with the problem of locality (not getting stuck in local optima). The two main techniques are the use of empirical bias functions and skewed theoretical probability distributions. Empirical bias functions consist on the selection of the pool items based on a tailored bias function that assigns weights to each item in the list, normalized to form a probability distribution. The purpose of the bias function is to balance the exploration and exploitation of the algorithm in order to adapt it to the problem at hand. An instance of this method is the Parameterized Biased Random Sampling, on which the values assigned by the bias function are based on a set of priority rules, which are specifically formulated for each problem. As an example of this we have the β parameter methodology (where β represents the maximum assigned probability).

The use of biased theoretical probability distributions, as opposed to uniform distributions, allows keeping the greedy properties of an algorithm and at the same time increasing its exploration properties. Examples of these kind of distributions are the geometrical or the triangular. In addition, unlike empirical distributions, theoretical distributions are more efficient because analytical expressions are known for them which allows assigning probabilistic values to each element in the pool by simply evaluating the analytical expression, and not by using the less efficient bias function. This type of methodologies allows us to find competitive solutions in a very short period of time, which can be even shorter by the use of parallelization techniques.

The authors approach several combinatorial optimization problems using this algorithm. However, for simplicity we will only explain its application to the arc routing problem, which is similar in many ways to the CVRP. In this case, we can apply biased randomization to the CWS Savings algorithm, giving the so called SHARP algorithm developed for solving the arc routing problem. For the computational benchmark, the authors used a geometric distribution with a β parameter adapted to the problem, and implemented a multi-start method in order to run many instances in parallel. Then, each instance was run 100 times with different time steps in order to select the best solution among them. The results can be seen in the following chart:

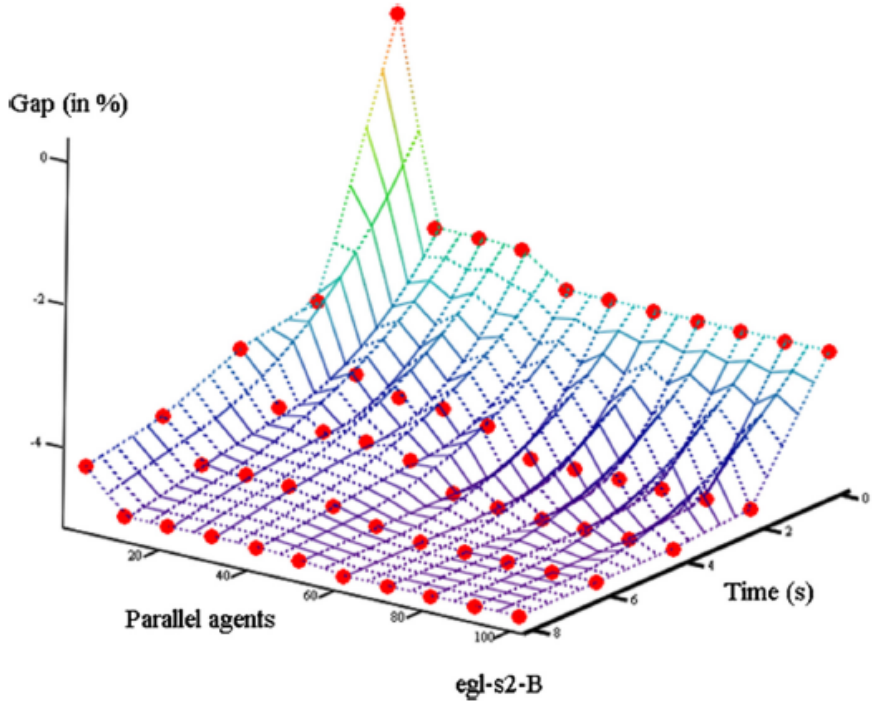

Figure 1.1: *Gap evolution for an ARP instance*

We can see how running more parallel agents decreases the time needed to reach the best solutions. In brief, combining skewed probability distributions with random sampling, heuristics can be improved without losing their previous, good properties while at the same being able to run many instances of the algorithm sequentially or in parallel to obtain different promising solutions to the original problem, thus increasing the probability of obtaining better and diversified solutions.

In my opinion, the application of these biased randomization techniques opens the door to the development of a large number of improvements to classical optimization algorithms, as well as to the development of new ones. I believe that the incorporation of parallelization techniques on the set random solutions obtained by these kind of algorithms allows us to exploit the power of specialized hardware units such as GPUs, and can lead to many new applications. In the algorithms explained below, we will explore in greater detail the improvements that this type of methodology has to offer.

1.b The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem

The CVRP problem consists on finding routes which, starting from a central depot node, travel through a set of nodes representing consumers in closed routes in such a way as to satisfy their demands. This is done through a fleet of similar vehicles with a limited capacity. The cost of going from one node to another is reflected in a cost matrix that is usually symmetrical, and usually it is imposed that each node can only be visited once by a single vehicle. New conditions can be added to increase the complexity of the problem, giving rise to a category of problems called Rich VRP. This problem is NP-hard, therefore it is very difficult to find good solutions

with classical approaches, let alone the optimal solution. Also, it is a useful problem for testing the efficiency of new optimization algorithms.

The authors present the modification of a classical heuristic called CWS (Clarke and Wright's Savings algorithm) by Monte Carlo simulation and biased randomization techniques. This is made possible by the existence of extremely long period random number generators. Without going into much detail for now, we can anticipate that the result is an improvement in the execution time and the construction of competitive solutions. Also, this methodology allows to find new solutions in each execution (CWS is deterministic) and to easily incorporate new contour conditions.

In further detail, the CWS algorithm is based on optimizing a cost function through the exchange of edges between two given nodes by means of an indicator called 'savings' . In other words, given a list of possible edge changes and their respective savings amount, the classical algorithm sorts it from highest to lowest savings and always selects the edge change with the highest amount (greedy behavior). The authors propose including random behaviors in the selection process by means of biased probability distributions. This way, the edge switch with largest savings is not always chosen because all the possible shifts are assigned a probability different from zero (although exponentially decreasing).

To further increase the degree of randomization, for each selection process a α parameter is chosen from a uniform or Gaussian probability distribution that modifies the properties of the theoretical one, so that it is different for each construction step.

In order to study the efficiency of the algorithm, an experimental test has been carried out where solutions of the CWS algorithm are compared with solutions of this modified algorithm. The result is that it is quite likely to obtain superior solutions to the classical ones (although as it is a probabilistic process this is not always the case) and in addition this occurs in very short times (milliseconds in a conventional computer).

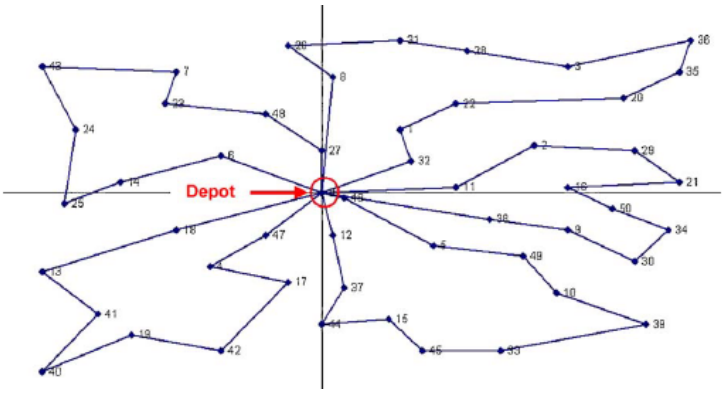

Figure 1.2: *Example of a CVRP solution found using the SR-GCWS algorithm*

In my opinion, this type of modifications can lead to a substantial improvement on the quality and execution time of classical algorithms, promising considerable gains on a large number of potential applications. In addition, this approach improves the flexibility of the original algorithm, allowing it to be generalized to more complex problems with more restrictive contour conditions. As a final property, it allows the creation of solutions in almost real time, which is very attractive for applications on problems that need a low latency.

1.c On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics

On this paper, the authors proposed the SR-GCWS-CS as a random biased adaptation of the classical Clarke and Wright's Savings algorithm. Its development has been possible thanks to the development of high quality random number generators that have a very long period, allowing a deep search in the solution space. In essence, the authors implement Monte Carlo simulation on the CWS algorithm, in particular on the parallelized version.

The objective is to introduce random biased behaviours on the CWS in order to generate new solutions other than the classical one. In order to do this, a randomization procedure is introduced on the construction phase. For each step, a α parameter is chosen from a uniform distribution of probability $P(a, b) \in (0, 1)$ and from this parameter the algorithm draws a biased distribution of probability (which is usually geometric, but can also be triangular or pseudo-geometric) which assigns exponentially decreasing selection probabilities to each edge. Since it is a probability distribution, the sum of all selection probabilities equals one, as shown in the following cumulative probability graph. This method alone leads to an improvement of the algorithm, but the authors do not stop there and introduce two more procedures, one to store routes in the cache and another to split the problem into subsets (divide-and-conquer strategy).

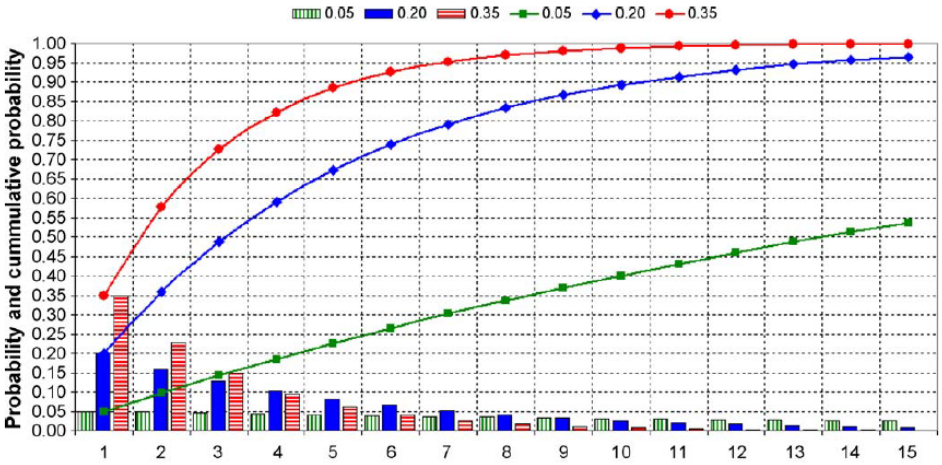

Figure 1.3: *Cumulative distribution of probability for geometric distributions of different α parameter*

The cache method is based on storing the order between nodes of a route that minimizes the cost, so that it can be used later. Every time a better ordering is found, this cache is updated, and this order is used whenever possible to improve the built solutions. It is important to see that these types of procedures increase the exploitation of the algorithm in the solution space at the expense of worsening the exploration.

The partitioning method consists in dividing the problem into subsets of nodes and applying the algorithm to these subsets. There are different partitioning procedures, but they are all based on the same principles: the geometric centres of individual routes are calculated and then the subsets are calculated by comparing the proximity of these centres to the geometric centre of the problem as a whole.

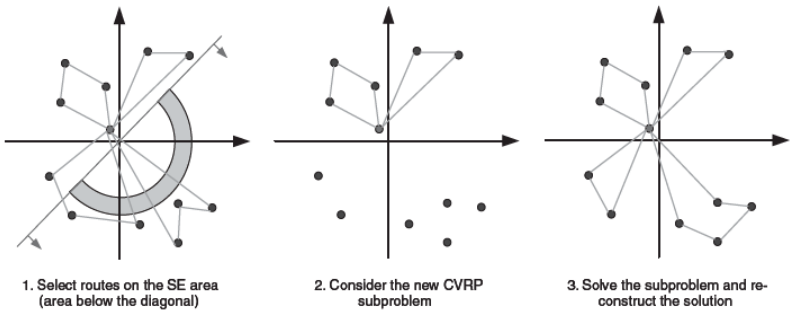

Figure 1.4: *An example of a splitting scheme*

In order to test the efficiency of this modified algorithm, an experimental test has been carried out on a hybrid parallelized architecture in Java and C. For this purpose, the authors have chosen 33 CVRP problems and their classical CWS solutions, and they have been compared by means of a gap indicator with the modified algorithm solutions. The outcome is that a significant number of these 33 problems achieve a better solution with the modified algorithm (negative gap) and in a very short time (few seconds).

In summary, it can be stated that this algorithm is a good alternative to other metaheuristics for the CVRP problem: it has the advantage of being very simple compared to other alternatives (fine-tuning is not necessary) and it can be generalized for other combinatorial optimization problems beyond the CVRP thus opening the door for new potential applications. In particular, the authors raise the possibility of using CUDA parallelization techniques on GPUs to further increase speed, as well as trying to include more restrictive contour conditions to simulate real problems.

Personally, I think that modifying classical heuristic and metaheuristic algorithms with randomization techniques can pave the way for a significantly more efficient and faster collection of algorithms. In addition, the fact that this technique is relatively new means that there are a multitude of applications that have not yet been discovered. Who knows if there is any classical algorithm that can dramatically benefit from the incorporation of this type of techniques, causing a significant increase in efficiency over actual problems.

2 Design and development of an algorithm

2.a Title

Biased Randomized CWS for Capacitated Vehicle Routing Problem

2.b Introduction

Transportation activities have an increasing economic, environmental, and social impact on our society. Moreover, its contribution to the GDP and the number of people it employs are significant in most countries. On the one hand, optimizing logistics activities is essential for many enterprises to remain competitive. On the other hand, governments aim at improving citizens' welfare by optimizing urban logistics, e.g., by limiting the maximum velocity they expect to reduce CO2 emissions and noise, banning big trucks in the city center for safety reasons, etc.

Therefore, in the last years, logistics and transportation companies are facing growingly demanding situations with fewer available resources. Market instability and the competitive business environment have caused an increasing optimization of logistic processes. Several fields of research have directed their efforts to conceive techniques to fulfil this purpose, like applied mathematics, operations management and computer sciences. The main challenge for these theoretical domains is the consideration of real contexts including real constraints into their approaches.

Vehicle routing is a complex logistics management problem and represents a key phase for the logistic optimization. There are many variations for the routing problem. The capacitated vehicle routing problem (CVRP) is probably the most popular routing problem in the literature on combinatorial optimization. It is a NP-hard problem and its the basic goal is to find an optimal set of routes for a fleet of vehicles so that a set of customers' demands is satisfied. Usually all vehicles are considered to be identical (homogeneous fleet). All routes begin and end at one depot, where all resources are initially located. Typically each vehicle has a maximum loading capacity, a single vehicle supplies each customer, and a vehicle cannot stop twice at the same customer. The classical objective is to minimize the costs related to distances traveled by vehicles and/or the times spent during the distribution process, while satisfying the associated constraints.

The CWS heuristic is among the most popular heuristic for solving the CVRP because it is relatively simple to implement and fast om terms of computation time. And according to Cordeau et al. (2002), a powerful heuristic for the VRP should posses these qualities: accuracy, simplicity, flexibility and speed. Additionally, the CWS heuristics has been shown more powerful when integrated with biased randomisation techniques.

In this work, an approach based on the Biased Randomisation of the Clarke and Wright Savings heuristics (BR-CWS) for solving CVRP is presented.

2.c Literature review

CVRP and many of its variants have been well studied in the literature since its introduction by Dantzig and Ramser (1959) but it is still attracting a great amount of attention from top researchers worldwide due to its potential applications, both to real-life scenarios and also to the development of new algorithms, optimization methods and meta-heuristics for solving combinatorial problems. Some examples can be found in Ruiz et al. (2018) in which a web solution is developed for solving NP-hard combinatorial optimization problems in "real time" (usually a few seconds) or in Iori et al. (2007) in which adds twodimensional loading additional constraints combining the CVRP, with a loading problem that is closely related to the well-known Two-Dimensional Bin Packing Problem (2BPP).

In particular, the routing part of our approach employs an enhanced and biased-randomized version of the well-known savings heuristic (Clarke and Wright (1964)). This randomized version makes use of a pseudogeometric distribution as initially proposed in Juan et al. (2010) in which we can find one example of the randomization of the CWS for the CVRP with the SR-GCWS algorithm. In Gonzalez et al. (2012) the RandSHARP algorithm is introduced for the CARP. This algorithm is a biased randomized version of the SHARP algorithm. The randomized algorithm uses solutions produced by the savings-based SHARP heuristic, and then iteratively it generates a new randomized solution by introducing a probabilistic criterion when selecting edges from the savings list. The algorithm uses a geometric probabilistic criterion. In Dominguez et al. (2014), the authors propose an algorithm to solve the 2L-HFVRP, which can be seen as a variant of the heterogeneous VRP where two-dimensional loading constraints have been incorporated to deal with 'large-size' items – which are usually, assumed to be rectangular shaped. The problem represents an important challenge since it combines a heterogeneous VRP with vehicle packing problems. The combination of these two classical problems is found in practical applications of some real-world transportation activities. Their approach relies on the biased randomization of routing and packing heuristics, which are integrated inside a multi-start framework.

There are also combinations of Monte Carlo simulation (MCS) and CWS algorithm. The first author in use it with CVRP was Buxey (1979). This method was revisited by Faulin and Juan (2008), who introduced an entropy function to guide the random selection of nodes. MCS has also been used by other authors to solve the CVRP (Fernández de Córdoba et al. (2000)).

2.d Algorithm developed

The proposed algorithm is performed by improvement and modification of the randomisation version of CWS heuristic. This algorithm introduces Randomised CWS heuristic in order to generate the best routes to satisfy objectives and constraints of CVRP. A feasible solution consists of a round-trip route in which a vehicle starts and ends at the same depot. Furthermore, each customer must be visited by the vehicle exactly once.

Characteristically, CWS heuristic uses a priority list of potential movements (savings) which is traversed in an iterative manner, i.e., at each iteration, the next constructive movement is selected from the list, which is sorted such that the most saving is on the top list. Thus, the CWS heuristic could be seen as an iterative greedy procedure, which constructs a feasible 'good' solution to the problem at hand by selecting, at each iteration, the 'best' option from a list, sorted according to some logical criterion.

However, a uniform randomization of that list will basically destroy the logic behind the greedy behaviour of the heuristic and, therefore, the output of the randomized algorithm is unlikely to provide a good solution. In fact, we could run the randomized algorithm thousands of times and it is likely that all the solutions generated would be significantly worse than the one provided by the original heuristic. To avoid losing the common sense behind the heuristic,

Festa and Resende (2009) proposes Greedy Randomised Adaptive Search Procedure (GRASP) to consider a restricted list of candidates. This is a sub-list including just some of the most promising movements that is, the ones at the top of the list, and then applying a uniform randomisation in the order of the elements of that restricted list as shown in Figure 2.1 (see left-hand side). The proposed approach goes one step further, and instead of restricting the list of candidates, it assigns different probabilities of being selected to each potential movement in the sorted list. In this way, the elements at the top of the list receive more probabilities of being selected than those at the bottom of the list, but potentially all elements could be selected as shown in Figure 2.1 (see left-hand side) and Figure 2.2. Notice that by doing this, the process does not only avoid the issue of selecting the proper size of the restricted list, but it also guarantees that the probabilities of being selected are always proportional to the position of each element in the list.

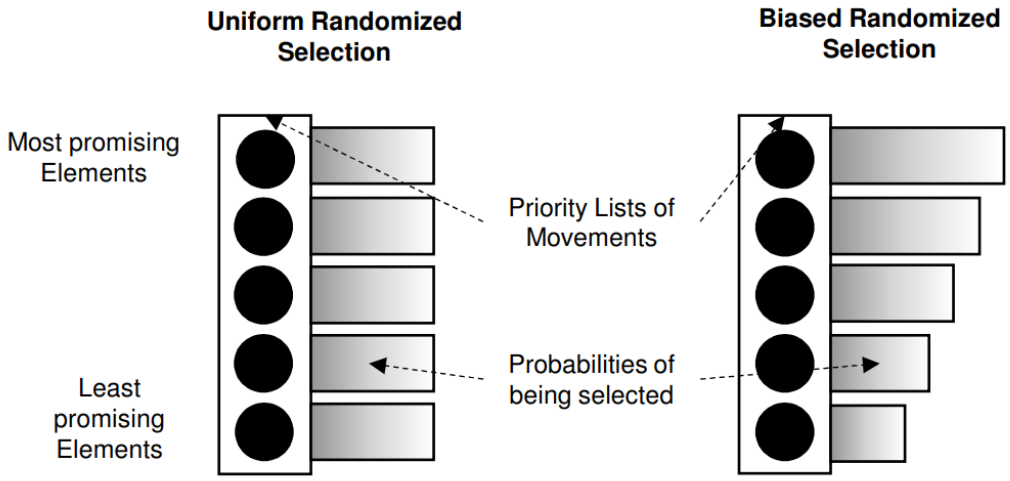

Figure 2.1: *Biased Randomization of Priority Lists*

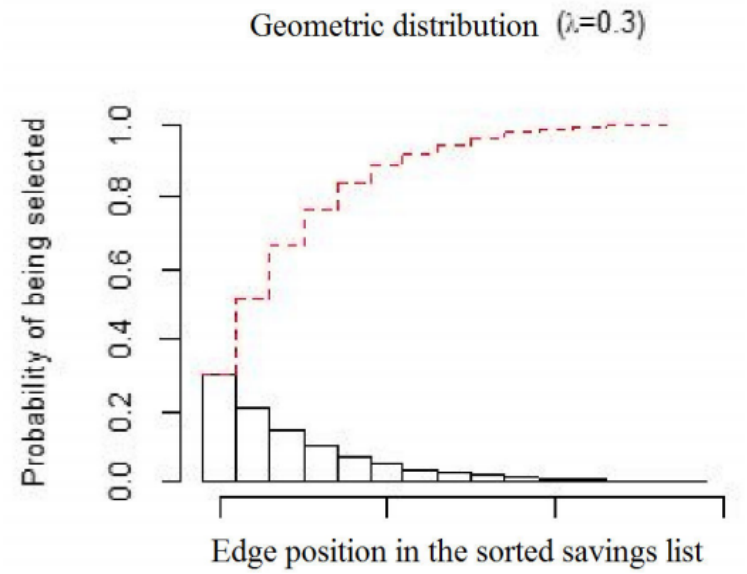

Figure 2.2: *Sorted saving list and geometric distribution with parameter* $\lambda = 0.3$

The step for transforming CWS heuristic into a probabilistic heuristic by means of biased randomisation is using the geometric probability distribution, the normal one (Algorithm 2) and the quasi-geometric (Algorithm 3), although the triangular distribution (Algorithm 4) and uniform distribution (Algorithm 5) have also been used for the experiment. It can give a selection of probabilities in each edge in the saving list. The larger saving value is more likely to be selected from the list than the other lesser saving value.

Algorithm 1 Randomized CWS procedure to generate a random initial solution

```
1: procedure FEASIBLESOLUTION(inputs, priorityList, paramAlgorithm)
2:     List ← copyList(priorityList)
3:     sol ← constructInitialSol(inputs)
4:     while List is no empty do
5:         edge ← selectEdgeAtRandom(List,rng)
6:         iNode ← getOrigen(edge)
7:         jNode ← getEnd(edge)
8:         iR ← getOrigen(iR, sol)
9:         jR ← getOrigen(jR, sol)
10:        if feasiblemerging(sol, iR, jR, inputs) then
11:            sol ← mergingRoutes(sol, iR, jR)          ▷ CWS heuristic
12:        end if
13:        jR ← deleteEdge(edge, List)
14:    end while
15: return sol
16: end procedure
```

Algorithm 2 Randomized edge-selection procedure

```
1: procedure SELECTEDGEATRANDOM(list, rng)
2:     beta ← generateRandomNumber(rng, a, b)        ▷ e.g.: a = 0.05 and b = 0.25
3:     randomValue ← generateRandomNumber(rng, 0, 1)
4:     pos ← floor(log(randomValue/log(1-beta))      ▷ Geometric distribution
5:     pos ← pos mod listSize                        ▷ Random position from the list
6: return getEdgeAtPosition(pos)
7: end procedure
```

To construct an initial solution, the CWS algorithm is used (see Algorithm 1). The algorithm uses the concept of savings associated with each edge to be considered for merging. At each step, the edge with the greatest saving is selected, if and only if the two-corresponding routes can be combined into a new feasible route. Also, the selected edge is composed of nodes that are directly connected to the depot. Thereafter, a new savings list of edges is obtained by randomising the original savings list through the use of a geometric probability distribution as shown in Algorithm 2. This allows different outputs at each iteration. The logic behind the randomised CWS heuristic is that edges with higher savings are more likely to be selected from the list than those with lower savings. Then, until the savings list gets empty, an iterative process begins in which the edge at the top of randomised list is most likely to be extracted.

Algorithm 3 Quasi-Geometric distribution

```
1: procedure selectEdgeAtRandom(list, rng)
2:     beta ← generateRandomNumber(rng, a, b)          ▷ e.g.: a = 0.05 and b = 0.25
3:     randomValue ← generateRandomNumber(rng, 0, 1)
4:     n ← 0
5:     cumulativeProbability ← 0.0
6:     for each edge e in sorted list do
7:         eProbability ← beta * (1 - beta)^n                    ▷ Geometric distribution
8:         cumulativeProbability ← eProbability + cumulativeProbability
9:         if randomValue < cumulativeProbability then
10:            return e
11:        else
12:            n ← n + 1
13:        end if
14:    end for
15:    k ← generateIntegerRandomNumber(rng, 0, listSize)
16: return getEdgeAtPosition(k)
17: end procedure
```

Algorithm 4 Triangular distribution

```
1: procedure selectEdgeAtRandom(list, rng)
2:     randomValue ← generateRandomNumber(rng, 0, 1)
3:     pos ← (int) (listSize * (1 - Math.sqrt(randomValue)))    ▷ Triangular distribution
4: return getEdgeAtPosition(pos)
5: end procedure
```

Algorithm 5 Uniform distribution

```
1: procedure selectEdgeAtRandom(list, rng)
2:     k ← generateIntegerRandomNumber(rng, 0, listSize)        ▷ Uniform distribution
3: return getEdgeAtPosition(k)
4: end procedure
```

2.e Computational experiment

This section presents a set of extensive computational experiments carried out to test the BR-CWS algorithm for CVRP. Firstly, the section introduces the instances that will be used to test our approach. Secondly, the algorithm parameters are discussed. Finally, the computational results are provided, they will be fully analyzed in the next section. The algorithm has been implemented as a Java application. A standard personal computer with an Intel Core i7 CPU at 1.8 GHz and 16 GB RAM has been employed to perform all the experiments.

2.e.1 Benchmark instances

As a benchmark for the test, a set of 20 instances originally proposed by the assessment are selected. Each instance describes the capacity of the vehicle, the coordinates of each node and also its demand. These instances are large enough to solve by CWS heuristic, that is, they cannot be solved by using exact methods.

2.e.2 Parameters setting

One of the advantages of the BR-CWS algorithm is that it does not require a complex fine-tuning process. In fact, after some quick trial-and-error experiments, the following values were set for each parameter:

- The biased-randomized selection during the construction process was generated by using a geometric probability distribution with parameter $\beta \in (0.05, 0.25)$, that is, at each iteration a random value inside the previous interval was assigned to β.
- For each instance, the algorithm was run 3 times, each time employing a different seed for the pseudo-random number generator.

2.e.3 Computational results

The results of our experiments comparing the different approaches are summarized in Table 2.1 and Table 2.2. It is structured as follows. Table 2.1 shows the original approach (no randomized), the Geometric distribution approach (Algorithm 2) and the Quasi-geometric distribution approach (Algorithm 3). Table 2.2 shows the Uniform distribution approach (Algorithm 4) and the Triangular distribution approach (Algorithm 5). All approaches have represented their cost and time achieve for each instance. Finally, the average values of cost and time are shown in the last row for each approach.

2.f Analysis of results

Table 2.1 and Table 2.2 summarize the results obtained using the BR-CWS algorithm. The approaches were run using the same parameters setting as described in Section 2.e.1. They show that Geometric distribution approaches obtain slightly better cost results than the original approach. However, they take longer to find the solutions.

Triangular distribution gets far worse results than Geometric distribution, in cost and time. Therefore, it has worse results than the original approach. In contrast, Triangular distribution obtains better results than Uniform distribution, being Uniform distribution which has the worst effect on the experiment.

Therefore, the results are consistent with the approaches. The original approach always selects the edge at the position 0 so that it will be faster than apply some distribution. It obtains good results because select the most saving edge. However, it does not apply randomized and thus always obtain the same route.

The Triangular distribution gets worse outcomes than Geometric distribution because its skewness is lesser. Finally, Uniform distribution does not have skewness, then it obtains the worst results.

Table 2.1: *Results of tests executions for different randomized edge-selection implementations.*

	Original		**Geometric dist.**		**Quasi-geometric dist.**	
Test	Cost	Time	Cost	Time	Cost	Time
1	5956.50	0.0316	6083.27	0.0299	5943.29	0.0552
2	9242.38	0.0298	8963.83	0.0359	8916.18	0.0406
3	12594.03	0.0157	12354.19	0.0640	12383.94	0.04300
4	16125.21	0.0718	16139.91	0.0439	16206.08	0.0865
5	7244.21	0.0061	7307.44	0.0076	7268.10	0.0284
6	9392.63	0.0086	9430.74	0.0185	9353.90	0.0329
7	11606.43	0.0094	11619.81	0.0297	11596.25	0.0325
8	13056.34	0.0158	12978.46	0.0410	13177.92	0.0308
9	663.57	0.0072	664.06	0.0161	674.89	0.0180
10	838.91	0.0079	847.80	0.0305	863.98	0.0234
11	1052.12	0.0174	1073.55	0.0410	1059.02	0.0305
12	1270.98	0.0238	1276.98	0.0699	1282.55	0.0402
13	952.74	0.0049	969.13	0.0163	956.81	0.0098
14	1221.68	0.0059	1199.14	0.0201	1224.50	0.0183
15	1512.65	0.0143	1525.59	0.0276	1515.62	0.0294
16	1774.68	0.0164	1849.64	0.0477	1847.25	0.0473
17	771.70	0.0027	785.79	0.0114	780.86	0.0086
18	1069.28	0.0029	1112.66	0.0196	1097.66	0.0145
19	1465.99	0.0045	1484.66	0.0298	1483.05	0.0228
20	1963.46	0.0069	1980.67	0.0373	1996.82	0.0426
Average	4988.77	0.0134	4982.37	0.0321	4981.43	0.0327

Table 2.2: *Results of tests executions for different randomized edge-selection implementations.*

	Uniform dist.		**Triangular dist.**	
Test	Cost	Time	Cost	Time
1	35543.28	1.6454	30803.49	1.4314
2	60148.88	3.9992	56771.09	3.3873
3	97729.74	9.9801	87513.08	8.7749
4	136235.81	19.6250	123766.64	15.6335
5	49576.32	0.4893	42602.04	0.3551
6	71422.89	2.1309	60272.21	1.5908
7	87822.97	5.5184	77922.56	4.5739
8	105873.94	12.8149	93245.65	10.5586
9	3242.13	1.5324	2649.19	1.1178
10	4761.33	3.7306	3886.27	3.0917
11	6409.58	8.6524	5296.90	7.2113
12	8344.27	21.7657	6840.56	14.9974
13	4681.25	1.4064	3691.41	0.9793
14	6408.47	4.0961	5287.31	2.9944
15	9170.43	8.9487	7122.09	7.1996
16	11698.71	20.7617	9183.21	15.8690
17	3702.58	1.1076	2945.03	0.8532
18	5035.81	3.1418	4115.01	2.3215
19	7066.62	5.9060	5681.82	4.6059
20	9690.14	10.6705	7895.74	8.3785
Average	36228.26	7.3961	31874.57	5.7963

2.g Conclusion

In this work a CWS algorithm was presented for solving a CVRP. The proposed methodology for the solving process combines biased randomization with CWS.

The random behavior increases the number of outcomes exponentially. This leads to long computing times, possibly exceeding the solving ability of the available resources.

In the present work the contribution to the provided algorithm consists of several developments. Due to the combination of several methodologies the developed program is able to deal with realistic routes applying randomized. Its flexibility contributes highly to the adaptability to different scenarios of realistic routing problem.

A further significant contribution is that the analysis of probability distributions described in the Section 2.e.3 can be used to new approaches in which shows that the Geometric distribution obtains the best results.

2.h Future work

In the literature review can be appreciate that the biased-randomization of heuristics combined with other techniques (e.g., simulation, parallel and distributed computing, constraint programming) have been useful for addressing a large set of CVRPs. Therefore, as for future work, the following lines of research are suggested:

1. Since the proposed algorithm is based on a biased-randomized selection of elements inside of heuristics, other biased (non-symmetric) probabilistic distributions could be used to measure its performance and its impact on results (sensibility).

2. Due to the common nature of combinatorial optimization problems, the proposed algorithm could be applied to efficiently solve other problem scopes like arc routing (González-Martín et al. (2012)), scheduling (Juan and Rabe (2013),Montoya-Torres et al. (2012)), flowshop (Juan et al. (2012), Juan et al. (2013)), clustering (Muñoz-Villamizar et al. (2013)) or green computing (Cabrera et al. (2013)).
3. The implementation and deployment of proposed methodologies in real enterprise environments in order to offer a day-to-day optimization routing planning, including more real constraints into it.
4. Explore other practical ways to apply some parallel and distributed techniques on the proposed algorithm that allows to reuse the computing platform of an enterprise.
5. Investigation of alternative forms of hybridization between heuristic and exact approaches for CVRPs.
6. The combination of routing and environmental aspects represent and promising and interesting trade-off to be studied.
7. Implementation and use of high-quality pseudo-random number generators (L'Ecuyer (2002)), so that the system becomes more reliable.

2 Bibliography

G.M Buxey. The vehicle scheduling problem and monte carlo simulation. *Journal of Operational Research Society*, pages 563–573, 1979.

G. Cabrera, A. Juan, H. Perez-Roses, J. Marques, and J. Faulin. Promoting green internet computing throughout simulation-optimization scheduling algorithms. *Proceedings of the 2013 Winter Simulation Conference*, 2013.

G. Clarke and J.R. Wright. Scheduling of vehicle routing problem from a central depot to a number of delivery points. *Operations Research*, pages 568–581, 1964.

J.-F Cordeau, Michel Gendreau, G Laporte, JY Potvin, and Frédéric Semet. A guide to vehicle routing heuristics. *Journal of the Operational Research Society*, pages 512–522, 2002.

G.B. Dantzig and Ramser. The truck dispatching problem. *Management Science*, pages 81–91, 1959.

O Dominguez, A. Juan, B. Barrios, J. Faulin, and A. Agustin. Using biased randomization for solving the two-dimensional loading vehicle routing problem with heterogeneous flee. *Annals of Operations Research*, 2014.

J. Faulin and A. Juan. The algacea-1 method for the capacitated vehicle routing problem. *International Transactions in Operational Research*, pages 1–23, 2008.

P. Fernández de Córdoba, L.M. García Raffi, A. Mayado, and J.M. Sanchis. A real de-livery problem dealt with monte carlo techniques. *TOP*, pages 57–71, 2000.

P. Festa and M. G. C Resende. Grasp – part i : Algorithms. *International Transactions in Operational Research*, pages 1–24, 2009.

S. Gonzalez, A. Juan, D. Riera, M. Elizondo, and P. P. Fonseca. Sim-randsharp: A hybrid algorithm for solving the arc routing problem with stochastic demands. *Proceedings of the 2012 Winter Simulation Conference*, pages 1–11, 2012.

S. González-Martín, A.A Juan, D. Riera, Q. Castella, R. Muñoz, and A. Pérez. Development and assessment of the sharp and randsharp algorithms for the arc routing problem. *AI Communications*, pages 173–189, 2012.

Manuel Iori, Juan José Salazar González, and Daniele Vigo. An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, pages 253–264, 2007.

A. Juan, J. Faulin, R. Ruiz, B. Barrios, and Caballe. The sr-gcws hybrid algorithm for solving the capacitated vehicle routing problem. *Applied Soft Computing*, pages 215–224, 2010.

A.A. Juan and M. Rabe. Combining simulation with heuristics to solve stochastic routing and scheduling problems. *Proceedings of the 15th ASIM Dedicated Conference*, 2013.

A.A Juan, H.R. Louren¸co, M. Mateo, Q. Castella, and B.B. Barrios. Ils-esp an efficient, simple, and parameterfree algorithm for solving the permutation flow-shop problem. *Proceedings of the 6th Workshop on Statistics, Mathematics and Computation*, 2012.

A.A. Juan, H.R. Louren¸co, M. Mateo, R. Luo, and Q. Castella. Using iterated local search for solving the flow-shop problem: parametrization, randomization and parallelization issues. *AI Communications*, 2013.

P L'Ecuyer. Ssj: A framework for stochastic simulation in java. *Proceedings of the 2002 Winter Simulation Conference*, page 234 – 242, 2002.

J.R. Montoya-Torres, A.A. Juan, L.H. Huatuco, J. Faulin, and G.L. Rodríguez-Verján. Hybrid algorithms for service, computing and manufacturing systems: routing and scheduling solutions. *IGI Globa*, 2012.

A. Muñoz-Villamizar, J.R. Montoya-Torres, A.A. Juan, and J. Cáceres-Cruz. A simulation-based algorithm for the integrated location and routing problem in urban logistics. *Proceedings of the 2013 Winter Simulation Conference*, pages 1–12, 2013.

Xavier Ruiz, Laura Calvet, Jaume Ferrarons, and Angel Juan. Smartmonkey: A web browser tool for solving combinatorial optimization problems in real time. *Applied Mathematics and Computational Intelligence*, pages 74–86, 2018.