Integrationsaspekte der Simulation:
Technik, Organisation und Personal
Gert Zülch & Patricia Stock (Hrsg.)
Karlsruhe, KIT Scientific Publishing 2010

Using Agent-based Simulation and Distributed Computing to solve Vehicle Routing Problems

Einsatz der Agenten-basierten Simulation und des Verteilten Rechnens bei der Tourenplanung

Angel A. Juan, Josep M. Marull, Josep Jorba, Josh Hester,
Joan M. Marquès, Xavi Vilajosana
Open University of Catalonia, Barcelona (Spain)
ajuanp@uoc.edu, jmarull@uoc.edu, jjorbae@uoc.edu,
jmarquesp@uoc.edu, xvilajosana@uoc.edu, jchester@mit.edu

Abstract: This paper presents a hybrid approach for solving the Capacitated Vehicle Routing Problem. After introducing the problem and some related work, the basic ideas of this approach are explained. Our base algorithm uses Monte Carlo simulation to induce a biased random behavior into a classical heuristic. Moreover, it also uses splitting and cache techniques to accelerate convergence to pseudo-optimal solutions. The efficiency of our methodology is enriched further by considering an agent-based simulation approach in which multiple instances or clones of the base algorithm are executed in parallel. Each of these instances uses a different seed for the random number generator. The multi-agent model is tested against a set of classical benchmarks and its performance is analyzed for different scales of agents. Finally, a distributed computing model is proposed as a potential methodology that can add significant value to the decision-making process.

1 The Vehicle Routing Problem

The Capacitated Vehicle Routing Problem (CVRP) is a NP-hard problem in which a set of customer demands must be served by a fleet of homogeneous vehicles departing from a depot, which initially holds all available resources (TOTH, VIGO 2002). There are some tangible costs associated with the distribution of these resources from the depot to the customers. In particular, it is usual for the model to explicitly consider costs due to moving a vehicle from one node – customer or depot – to another. The classical goal here consists of determining the optimal set of routes that minimizes those tangible costs under the following set of constraints: (i) all routes begin and end at the depot; (ii) each vehicle has a maximum load capacity, which is considered to be the same for all vehicles; (iii) each customer has a well-

known demand that must be satisfied; (iv) each customer is supplied by a single vehicle; and (v) a vehicle cannot stop twice at the same customer.

Even though this problem and some of its variants have been studied for decades, they still are 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 metaheuristics for solving combinatorial problems. As a matter of fact, different approaches to deterministic and stochastic VRPs have been explored in recent years (GOLDEN et al. 2008). These approaches range from the use of pure optimization methods such as linear programming for solving small- to medium-size problems with relatively simple constraints, to the use of heuristics and metaheuristics that provide pseudo-optimal solutions for larger problems with more complex constraints (CORDEAU et al. 2004). Most of these methods focus on minimizing an aprioristic cost function, which usually models tangible costs subject to a set of well-defined and simple constraints. However, real-life problems can be very complex, with intangible costs, fuzzy constraints and desirable solution properties that are difficult to model (KANT et al. 2008). In other words, it is not always straightforward to construct an initial model which takes into account all possible costs, (environmental costs, work risks, etc.), constraints and desirable solution properties (time or geographical restrictions, balanced work load among routes, solution attractiveness, etc.). For this reason, there is a need for new methods that are able to provide a large set of alternative pseudo-optimal solutions with different properties so that decision-makers can choose among alternative solutions according to their specific needs and preferences, i.e., according to their utility function, which is usually unknown to the researcher. All in all, as some VRP specialists have pointed out already, there is a need for more simple and flexible methods to solve the problem, methods that can be used to handle – in reasonable computing times– the numerous side constraints that arise in practice (LAPORTE 2007).

2 Monte Carlo Simulation and the CVRP

Monte Carlo simulation (MCS) can be defined as a set of techniques that make use of random numbers and statistical distributions to solve certain stochastic or deterministic problems (LAW 2007). MCS has proved to be extremely useful for obtaining numerical solutions to complex problems, which cannot be efficiently solved by using analytical approaches. One of the fundamental ideas behind our previous work in the CVRP arena has been to add a biased random behavior to a classical heuristic (FAULIN et al. 2008). In order to do this, we combined MCS techniques with one of the best-known classical heuristics for the CVRP, namely the Clarke and Wright savings (CWS) heuristic (CLARKE, WRIGHT 1964). The CWS procedure uses the concept of cost savings, which are determined for each possible edge connecting two nodes. Generally speaking, the CWS first constructs a starting feasible solution, which is then improved by using an iterative process in which routes are merged under certain constraints. Specifically, at each step of this process the edge with the highest savings is selected from the list of possible edges and, if no constraint is violated, the two corresponding routes are merged. This process continues until the list of edges is empty. Since the CWS algorithm always chooses the

edge with the highest savings value, it is a deterministic methodology, i.e.: the output of the CWS will always be the same no matter how many times it is run. In order to develop a stochastic version of the CWS, we proposed a procedure which assigns random probabilities during the edge-selection process (JUAN et al. 2009). This randomization of the CWS – which provides a different output each time it is run – is based on the use of right-skewed probability distributions, such as the geometric distribution or the decreasing triangular distribution. By assigning decreasing probabilities to the sorted savings list of edges, those edges with higher savings will be more likely to be selected from the list than those with lower savings. This approach was enriched by adding some cache and splitting techniques. Figure 1 summarizes the main steps of this algorithm (a more detailed description, including extensive validation tests, can be found in JUAN et al. (2009)): (1) initially, a CVRP instance is given; (2) the corresponding CWS solution is obtained; (3) a random solution is generated by executing the randomized CWS procedure; (4) the resulting random solution is compared against the CWS one; if the randomized CWS solution does not outperform the CWS one, then the process starts again from step 3; (5) otherwise, the randomized CWS solution is divided into two sets of routes, the front set and the back set; the front set is then considered as a new CVRP problem with the same initial constraints but with less nodes –which reduces the problem difficulty; (6) now, an inner loop starts; at each iteration of that loop, the new subproblem is solved by using the randomized CWS algorithm; once this inner loop has finished, the resulting solution is sorted and saved in a database, and the process restarts from step 3 (outer loop); notice that this restarting factor is decisive in order to avoid that the methodology gets trapped into a local minimum.

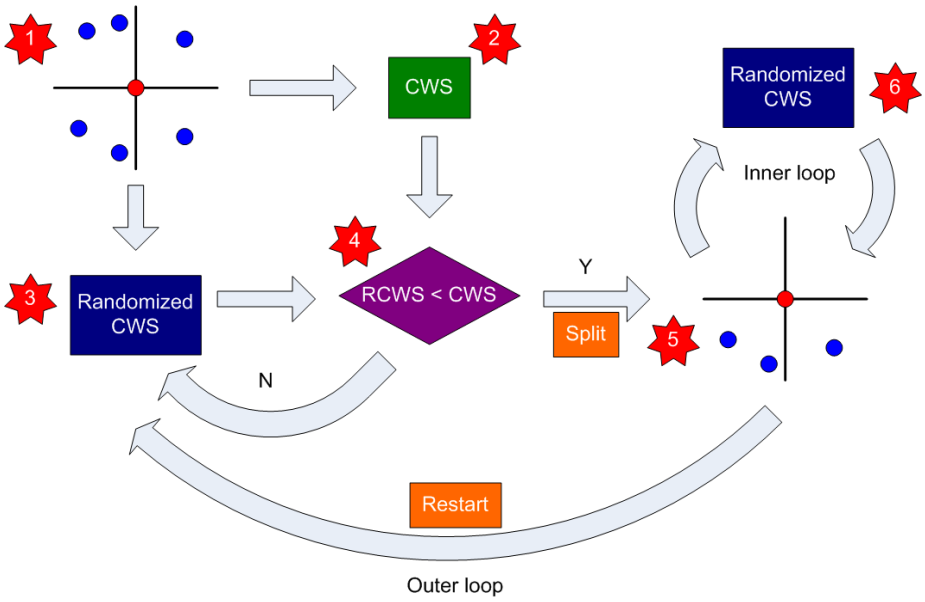

Figure 1: *Overview of the randomized base algorithm including splitting techniques*

3 Adding a Multi-agent Approach

The base algorithm described above has proven to be capable of offering pseudo-optimal solutions for the CVRP without needing to be adjusted or configured for the specific instance being studied. However, its efficiency can still be significantly improved by using parallelization techniques. Offering pseudo-optimal solutions to complex problems in real time and without adjustments beforehand still presents a challenge. In spite of this, we have noticed that it is possible to significantly reduce the execution time that the algorithm needs to obtain good solutions, depending on the seed that is chosen for the pseudo-random number generator. There are new processor design paradigms based on gaining computation capacity through the parallel execution of multiple processes and threads (multi-core). Also, new and affordable Graphic Processing Units (GPUs) have recently been introduced on the market, offering the capacity of executing hundreds – or even thousands – of threads concurrently. The goal is to execute multiple instances of the algorithm at the same time, each with a different seed. As shown in Figure 2, each of these instances can be considered as a cloned agent that is searching the solution space by using MCS.

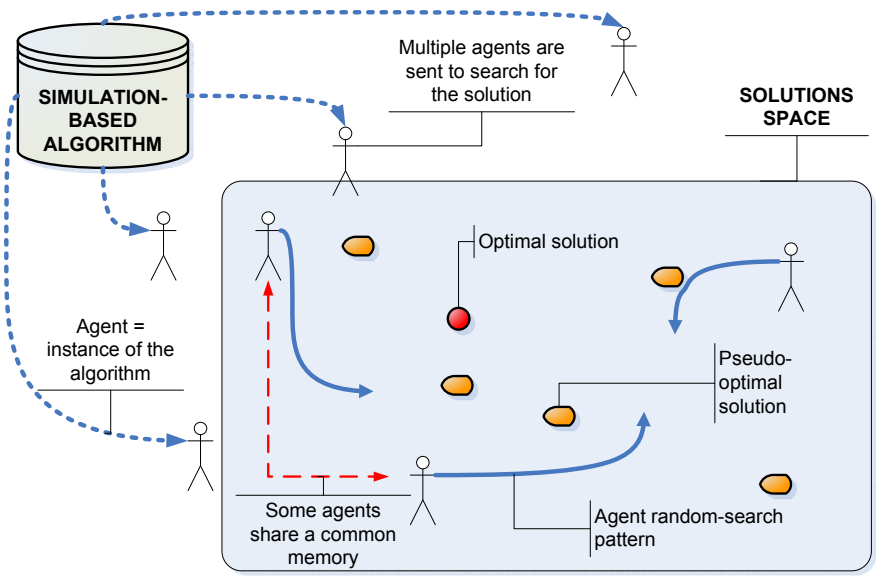

Figure 2: *Using a Multi-Agent Approach to perform the search*

Notice that several policies can be then considered regarding the level of communication and/or memory sharing among these cloned agents. In fact, communication processes among these agents can become quite complex to design and manage, and they constitute one of our future research lines. In this paper we only consider memory-sharing policies among some agents. In particular, agents running in the same machine will share a common hash table. This way, different agents are able to check the shared table and look up for the best-known route covering a given set of nodes (agents use this valuable information as part of a local search process). Also, agents are able to update the shared table whenever they find a better route for

visiting a given set of nodes. As stated before, other communication policies among agents could be considered, but the main goal of this paper is to show that pseudo-optimal solutions can be obtained for most small- and mid-size VRP instances (up to 120 nodes) in "real-time" (less than 10 seconds) by parallelizing the simulation-based algorithm explained in the previous section. Another important goal of this paper is to estimate the number of agents needed to attain the aforementioned pseudo-optimal solutions in "real-time".

4 Computational Evaluation

With the objective of analyzing how an agent-based approach affects both the quality of the solutions provided by our algorithm and the associated computational time, we selected a subset of 11 large-size classical CVRP instances (each of them containing between 76 and 121 nodes) and performed two experiments: the first one was aimed at answering the following question: "for each instance, which is the best solution that our algorithm can provide in real-time (less than 10 seconds) using different scales of agents and without any fine-tuning processes?"; The second experiment was aimed at answering the following question: "for each instance, which is the best solution that our algorithm can provide in a reasonable time (less than 5 minutes) using different scales of agents and without any fine-tuning processes?". The following instances were selected: A-n80-k10, B-n78-k10, E-n76-k7, E-n76-k10, E-n76-k14, F-n135-k7, M-n101-k10, M-n121-k7, P-n76-k4, P-n76-k5 and P-n101-k4 (more details on these instances can be found at http://branchandcut.org). Five different scales of agents were considered in our tests: M1T10, M25T10, M50T10, M75T10 and M100T10, where MnT10 represents n machines with 10 threads (agents) per machine, which adds up to $10 \cdot n$ agents in total. As discussed in the previous section, for the experiments in this paper we designed a simple multi-agent model in which only those threads running in the same machine share a common memory (a hash table with best-known routes). Therefore, to avoid dealing with complex communication issues –which could be also time-consuming due to latency times, especially in distributed scenarios like the ones proposed in the next section–, agents running in different machines do not communicate among themselves. The tests were done using a multi-core processor Intel® Xeon E5504 and 4 GB of RAM. A multi-thread version of the algorithm implemented in C was executed under Windows® 7 Professional. As explained before, each agent (thread) was an instance of our algorithm defined by a different seed for the pseudo-random number generator. Notice also that no fine-tuning process was carried out, since one of our goals was to prove that our algorithm is robust and can provide efficient solutions to any CVRP problem without any initial adjustments. Figure 3 relates to the first experiment and it shows how the gap between our best "real-time" solution and the best-known solution evolves as the number of agents increases. Observe that, even for the smallest scale (M1T10 with 10 agents), the average gap is just 0.3 %, which is a remarkable result if we consider that: (a) these are relatively large and difficult-to-solve CVRP instances, (b) the maximum time allowed was just 10 seconds, and (c) no fine-tuning process has been performed. Notice also that this average gap is reduced to 0.1 % in the M25T10 scenario (250 agents). At this point, no significant improvement is obtained by further increasing the number of agents. In other words, for relative large CVRP

instances, our parameter-free algorithm is able to provide pseudo-optimal solutions (average gap of 0.1 %) in real-time (less than 10 seconds) by employing 25 machines (250 agents) working in parallel. Alternatively, we could also say that our algorithm is able to provide pseudo-optimal solutions by employing one single machine (10 agents) working for about 250 seconds.

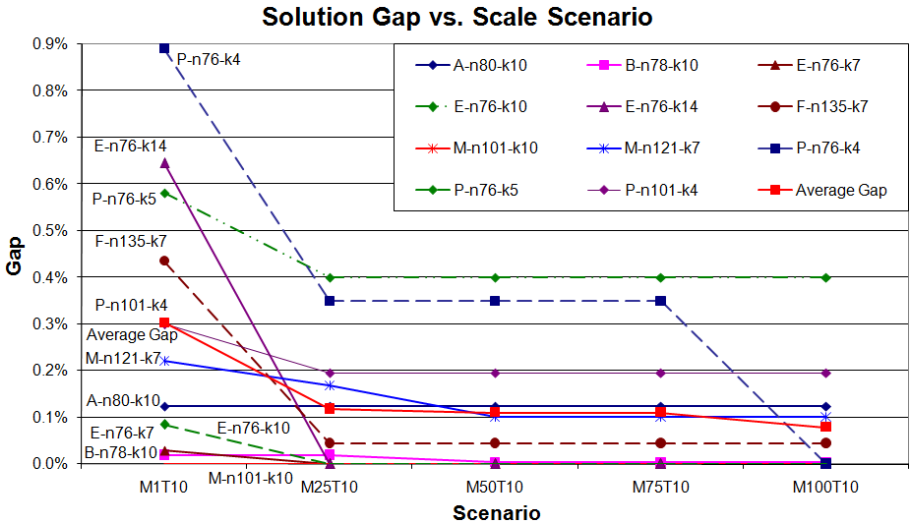

***Figure 3**: Solution gap versus scale scenario for experiment #1 (real-time)*

Figure 4 relates to both experiments and shows average gaps versus average times for each scale scenario.

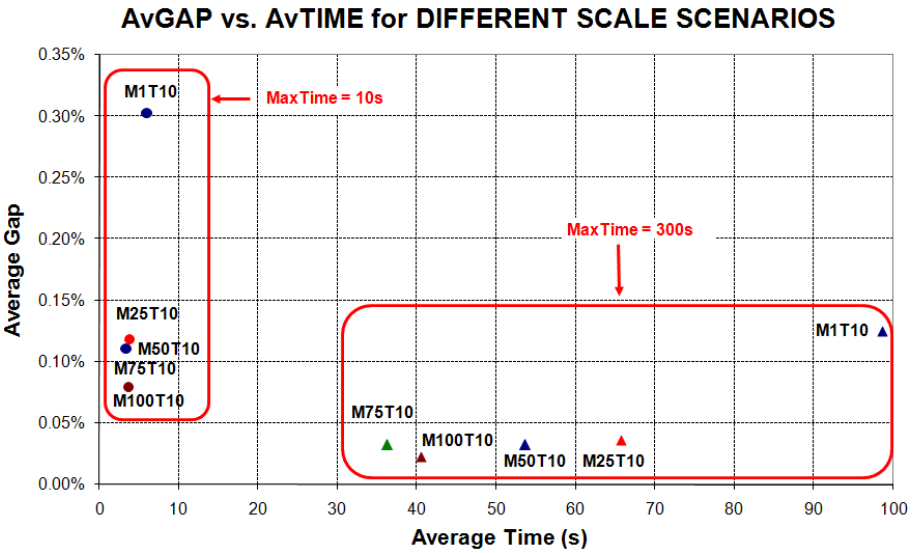

***Figure 4**: Average gap vs. average time for both experiments*

Observe that for all the considered scenarios, the average gap between our solution and the best-known solution is significantly reduced (p-value = 0.04) by increasing the computational time from 10 seconds to 300 seconds. Also, as expected, there is a clear equivalence between the M25T10-10s and the M1T10-300s scenarios. An important conclusion from these experiments is that either of these alternative scenarios (250 agents during 10 seconds or 10 agents during 300 seconds, which have shown to be approximately equivalent as expected by design) is recommended over other scenarios that require many more machines or longer computational times for solving CVRP instances with up to 120 nodes. Of course, larger CVRP instances with several hundreds or even thousand or nodes might require far more computational resources and/or computational times to be satisfactorily solved.

5 The Role of Distributed Computing

Usually, small- and medium enterprises (SMEs) in the logistics business lack technical expertise and high-tech computational resources. It is not likely that they can afford buying expensive software or powerful computer systems to solve their complex routing problems in real time. In such scenarios, one possible approach to allow for real-time decision-making could be to employ modern GPUs. However, use of GPUs requires advanced algorithmic and programming skills. An alternative approach is to use a distributed problem-solving approach: the basic idea here is to construct a middleware that, relying on the so-called "Volunteer Computing", aggregates all existing computational power in a SME –or even better, in a group of federated SMEs– in order to concurrently execute thousands of clones or instances of our algorithm; this way, pseudo-optimal solutions for large and complex real-life problems might be obtained in nearly real time at an inexpensive monetary cost. In effect, a standard SME uses a large number of commodity computers distributed among their different departments and/or facilities. Most of these personal computers offer more computing capabilities than required to complete their daily activities (word processors, spreadsheets, e-mail, etc.). Moreover, they happen to be underutilized during nightly hours. Our proposal gathers the spare resources from each computer and aggregates them into a computational environment where hundreds or even thousands of instances of our algorithm can be run simultaneously. These cloned agents can either use a global collaborative memory or several local collaborative memories, depending on the characteristics of the distributed system. Moreover, resources from a SME may be federated with resources from other SMEs, therefore resulting in an even larger computational system. To avoid interferences with the current tasks executed in each computer, contributed resources may be provided through the use of virtual machines. As more computational resources become available, more agents (algorithm's instances) will be concurrently executed, thus increasing the chances of finding pseudo-optimal solutions in a reduced time-period. The idea of aggregating computational resources from different machines in a network has been successfully explored in several works and real-life applications. In particular, the Volunteer Computing platforms (MARQUES et al. 2007) aggregate computing capacities from the edges of Internet. Those platforms offer tools to create ad-hoc communities that perform massive computation by aggregating the resources of their participants.

6 Conclusion

We have presented a multi-agent approach for solving small- and mid-size Vehicle Routing Problems (up to 120 nodes) in "real-time" (10 seconds). The base algorithm (agent) in which our approach relies presents some advantages over some other metaheuristics: (a) it can be easily implemented, (b) it does not require any fine-tuning process, (c) it is able to provide a large set of alternative good solutions, and (d) multiple instances (agents) can be concurrently executed by changing the seed of the random number generator. Even in a single-workstation scenario, our approach has provided pseudo-optimal solutions in reasonable times for all considered benchmarks, so we expect to be able to solve even larger problems when using a cluster or a grid. We also discussed the possibility of using distributed computing as an economical and efficient way to solve large-scale vehicle routing problems.

References

CLARKE, C.; WRIGHT, J.: Scheduling of Vehicles from a central Depot to a Number of Delivering Points. In: Operations Research, Hanover, MD; 12(1964)4, pp. 568-581.

CORDEAU, J.; GENDREAU, M.; HERTZ, A.; LAPORTE, G.; SORMANY, J.: New Heuristics for the Vehicle Routing Problem. In: Logistics Systems: Design and Optimization. Eds.: LANGEVIN, A; RIOPEL, D. Boston: Kluwer, 2004, pp. 313-329.

FAULIN, J.; GILIBERT, M.; JUAN, A.; RUIZ, R.; VILAJOSANA, X. (2008): A Simulation-based Algorithm for the Capacitated Vehicle Routing Problem. In: Proceedings of the Winter Simulation Conference. Eds.: MASON, S.; HILL, R.; MONCH, L.; ROSE, O.; JEFFERSON, T.; FOWLER, J. Piscataway, NJ: IEEE, 2008, pp. 2708-2716.

GOLDEN, B.; RAGHAVAN, S.; WASIL, E. (Eds.): The Vehicle Routing Problem: Latest Advances and New Challenges. New York: Springer, 2008.

JUAN, A.; FAULIN, J.; RUIZ, R.; BARRIOS, B.; CABALLE, S.: The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem. In: Applied Soft Computing, Amsterdam et al., 10(2010)1, pp. 215-224.

KANT, G.; JACKS, M.; AANTJES, C.: Coca-Cola Enterprises Optimizes Vehicle Routes for Efficient Product Delivery. In: Interfaces, Linthicum, MD, 38(2008)1, pp. 40-50.

LAPORTE, G.: What you should know about the Vehicle Routing Problem. Naval Research Logistics, New York, 54(2007)8, pp. 811-819.

LAW, A.: Simulation Modeling & Analysis. Boston, MA: McGraw-Hill, 2007.

MARQUES, J.; VILAJOSANA, X.; DARADOUMIS, T.; NAVARRO, L.: LaCOLLA: Middleware for Self-Sufficient Online Collaboration. In: IEEE Internet Computing, New York, 11(2007)2, pp. 56-64.

TOTH, P.; VIGO, D.: The Vehicle Routing Problem. New York: SIAM Monographs on Discrete Mathematics and Applications, 2002.