A multi-space sampling heuristic for the vehicle routing problem with stochastic demands

Jorge E. Mendoza, Juan Villegas

A multi-space sampling heuristic for the vehicle routing problem with stochastic demands*

Jorge E. Mendoza[1] Juan G. Villegas[2]

September 28, 2011

[1] Équipe Optimisation des Systèmes de Production et Logistiques, LISA (EA CNRS 4094).
Université Catholique de l'Ouest, 3 Place André Leroy 49008 Angers, France.
jorge.mendoza@uco.fr

[2] Departamento de Ingeniería Industrial – Facultad de Ingeniería – Universidad de Antioquia.
Calle 70 No. 52 - 21, 050010 Medellin, Colombia.
jvillega@udea.edu.co

Abstract

The vehicle routing problem with stochastic demands consists in designing transportation routes of minimal expected cost to satisfy a set of customers with random demands of known probability distribution. This paper proposes a novel heuristic approach that uses randomized heuristics for the traveling salesman problem, a tour partitioning procedure, and a set-partitioning formulation to sample the solution space and find high-quality solutions for the problem. Computational experiments on benchmark instances from the literature show that the proposed approach outperforms the state-of-the-art algorithm for the problem in terms of both accuracy and efficiency.

1 Introduction

The vehicle routing problem with stochastic demands (VRPSD) can be defined on a complete and undirected graph $G = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{0, \ldots, n\}$ is the vertex set and $\mathcal{E}$ the edge set. Vertices $v = 1, \ldots, n$ represent the customers and vertex $v = 0$ represents the depot. A distance d_e is associated to edge $e = (v, u) = (u, v) \in \mathcal{E}$ and it represents the travel cost between vertices v and u. Each customer v has a random demand ξ_v for a given product. Customer demands are serviced using an unlimited fleet of homogeneous vehicles with capacity Q located at the depot. The exact quantities demanded by each customer are only known upon the vehicle's arrival to the customer location. It is assumed, however, that each customer's demand follows an independent and known probability distribution and that all demand realizations (actual quantities) are nonnegative and less than the capacity of the vehicle.

Different frameworks can be applied to model and solve the VRPSD (for a compact survey the reader is referred to [14]). Among these frameworks, the most widely used in the literature, and the one selected for this research, is two-stage stochastic programming. As the name suggests, in two-stage stochastic programming the problem is solved in two stages. In the first stage, a set $\mathcal{R}$ of *planned* routes is designed. Each route $r \in \mathcal{R}$ is a sequence of vertices $r = (0, v_1, \ldots, v_i, \ldots, v_{n_r}, 0)$, where $v_i \in \mathcal{V} \setminus \{0\}$ and n_r is the number of customers serviced by the route. During the planning phase, routes are designed so the total expected demand they service does not exceed the capacity of the vehicle (i.e., $\sum_{v \in r \setminus 0} E[\xi_v] \leq Q \ \forall \ r \ \in \mathcal{R}$). In the second stage, each planned route is executed until a *route failure* occurs, that is, whenever the capacity of the vehicle is exceeded. Upon failure, a *recourse action* is applied to recover the feasibility of the failing route. The recourse action is classically defined as a return trip to the depot to restore the capacity of the vehicle, followed by a trip back to the customer location to complete the service. After service completion, the route is resumed from that point on as originally planned. It is worth noting at this point that the literature accounts for more sophisticated recourse actions (see for instance [1, 4, 10, 16]); nonetheless, we decided to keep the classical policy because it is simple, suitable to many practical applications, and it allows a more direct comparison with previously published results. The second stage solution is then the actual set of routes traveled by the vehicles. The problem is to determine in the first stage the set of planned routes $\mathcal{R}$ that minimizes the expected cost $E[C]$ of the second stage solution given by:

*Under review since Sep. 28, 2011

$$E[C] = \sum_{r \in \mathcal{R}} E\left[l_r + G_r\left(\vec{\xi}\right)\right] = \sum_{r \in \mathcal{R}} l_r + \sum_{r \in \mathcal{R}} E\left[G_r\left(\vec{\xi}\right)\right] \quad (1)$$

where l_r denotes the planned length (planned cost) and $E\left[G_r\left(\vec{\xi}\right)\right]$ the expected length of the returning trips to the depot, or cost of recourse, caused by route failures for each route $r \in \mathcal{R}$. The planned cost of a route is given by the sum of the lengths of the arcs traversed by the route. On the other hand, the estimation of the expected cost of recourse is slightly more complicated. Under the selected recourse action, the expected cost of the failures in a route is given by:

$$E\left[G_r\left(\vec{\xi}\right)\right] = 2 \times \sum_{i=2}^{n_r} \sum_{l=1}^{i-1} Pr\left(\sum_{j=2}^{i-1} \xi_{v_j} \leq l \cdot Q < \sum_{j=2}^{i} \xi_{v_j}\right) \times d_{v_i,0} \quad (2)$$

where the probability term represents the probability of having the l_{th} failure while servicing customer v_i. The expected cost of failures in (2) can be efficiently computed when customer demands follow a probability function with the cumulative property. This property states that the sum of two independent and Ψ distributed random variables is also Ψ distributed, as it is the case for the normal, Poisson, and Gamma distributions. For details on the derivation of (2) the reader is referred to [3, 7].

The two-stage stochastic programming formulation with the classic recourse action presented above has been the main block to build solution methods of different nature for VRPs with stochastic demands. Exact methods based on this formulation include that of Laporte et al. [7] who proposed an implementation of the L-Shaped algorithm and solved to optimality instances of up to 50 and 100 customers with Normally and Poisson distributed demands of a VRPSD variant with limited fleet. In the same vein, Rei et al. [12] proposed an implementation of the L-Shaped algorithm with local branching cuts for a variant in which a single route servicing all customers is to be designed (this problem is usually referred in the literature as the SVRPSD). The authors reported optimal solutions for instances of up to 90 customers with uniformly distributed demands. Christiansen and Lysgaard [3] proposed a branch-and-price algorithm to tackle the classical formulation. Their approach successfully solved instances of up to 60 customers with Poisson distributed demands. In the segment of heuristic approaches, Gendreau et al. [5] proposed a tabu search (TS) algorithm, known as *tabustoch*, designed to tackle an extension of the classical formulation in which, in addition to the demands, customers are also stochastic (i.e., they are present, or not, with a given probability). Yang et al. [16] introduced two constructive heuristics for another extended formulation in which preventive trips to the depot are allowed. A similar formulation was proposed by Bianchi et al. [2] in the context of the SVRPSD. To solve their problem, these authors introduced a set of metaheuristics comprising simulated annealing (SA), iterated local search, ant colony optimization, evolutionary algorithms, and TS. More recently, Rei et al. [13] introduced a hybrid Monte Carlo local branching approach for the SVRPSD. Another extension to the classical formulation was introduced by Mendoza et al. [8] who generalized the problem to consider the case in which each customer demands several incompatible products that are transported on different vehicle compartments. By setting the number of products and compartments to 1, they obtain a classical VRPSD. These authors proposed two memetic algorithms [8], and a set of look-ahead heuristics [9] for their problem. Goodson et al. [6] introduced a hybrid simulated annealing that embeds sophisticated neighborhood schemes into a local search procedure. To the best of our knowledge, the latter approach currently holds the best known solutions for most of the instances proposed by Christiansen and Lysgaard [3].

This research introduces a new heuristic approach that uses randomized heuristics for the traveling salesman problem (TSP), a route partitioning procedure, and a set-partitioning model, to sample the solution space and find high-quality solutions for the problem. Computational experiments carried on instances from the literature show that the proposed approach outperforms the state-of-the-art heuristic method for the problem. The paper is organized as follows. Sections 2 outlines the proposed heuristic. Section 3 discusses extensive computational experiments conducted on a set of benchmark instances and presents a detailed analysis of the components of the proposed approach. Finally, Section 4 concludes the paper and outlines future research perspectives.

2 Multi-space sampling heuristic

When solving vehicle routing problems one usually works with elements belonging to four different spaces or sets: the set of TSP-like tours (i.e., giant tours visiting all customers), $\mathcal{P}$; the set of all feasible routes (i.e., routes that verify all the side constraints of the problem), $\mathcal{T}$; the set of all clusters of customers from which feasible routes may be build, $\mathcal{C}$; and, the set of all feasible solutions to the problem, $\mathcal{S}$. The

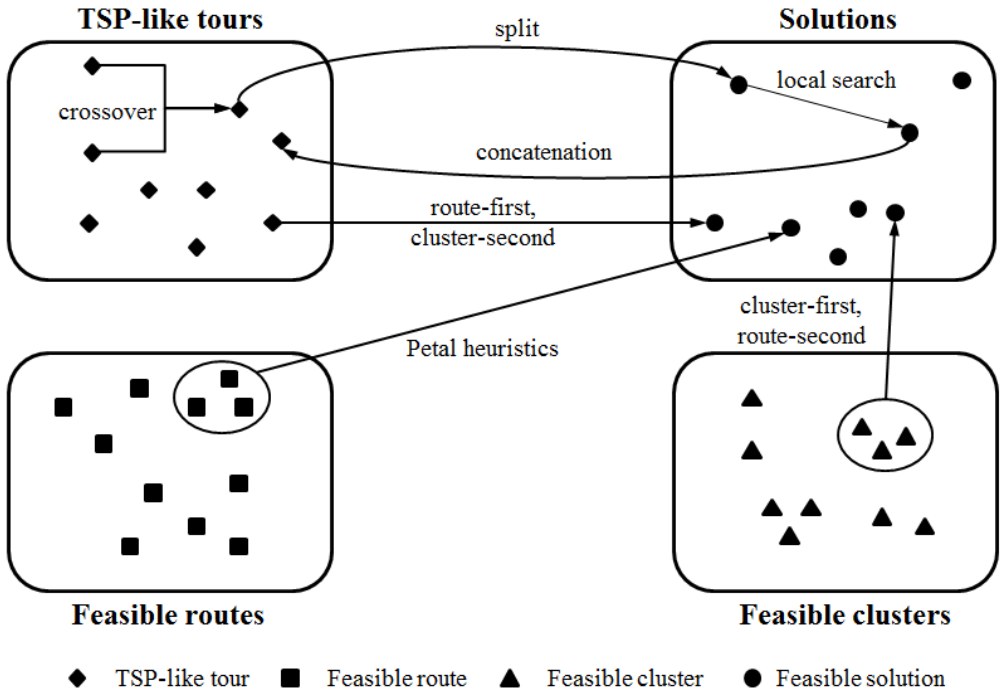


Figure 1: Multi-space search for vehicle routing problems

ultimate goal of a solution approach is to find the best possible element in the latter set. To accomplish their goal, many vehicle routing heuristics exploit the intimate relation existing between elements from different sets, that is, the fact that an element from one set may be easily mapped to an element belonging to another set. Figure 1 depicts some examples of these heuristics. For instance, cluster-first route-second approaches use some procedure to select a sub-set of elements from $\mathcal{C}$ and then apply another procedure to map the selected elements to an element in $\mathcal{S}$ that is eventually reported as output. Similarly, petal heuristics identify a sub-set of elements from $\mathcal{T}$ and then select elements from that sub-set to build a single solution, and route-first cluster-second approaches search for an element in $\mathcal{P}$ and then map it to an element in $\mathcal{S}$. More elaborate approaches search on different spaces in an iterative fashion. One good example is Prins' well-known evolutionary algorithm for the capacitated VRP [11]. At each iteration, the algorithm selects two elements from $\mathcal{P}$ and applies to them a crossover operator to obtain a new element in the same set. The new element is mapped to an element in $\mathcal{S}$ using a procedure called *split*. Then, local search is applied to the mapped element trying to identify a new (better) element in the set of solutions. When local search is completed, the new element is mapped back to $\mathcal{P}$ using a concatenation procedure and the whole routine starts over. In any case, single or multiple-space search, the effectiveness of the methods is highly dependent on the procedures used to select elements from one set and map them to another. The idea behind the multi-space sampling heuristic proposed in this paper is to take advantage of existing procedures that are able to effectively map elements between different sets in the context of the VRPSD and embed them into an approach that samples different spaces to find solutions for the problem.

The proposed heuristic works in two phases. In the first phase (sampling phase), it uses a set H of *sampling heuristics* to draw T elements from the set of TSP-like tours (i.e., $\mathcal{P}$) and maps each sampled element p^t to a sub-set Ω^t in the set of feasible routes (i.e., $\mathcal{T}$). In the second phase (mapping phase), the approach maps a set $\Omega \subset \mathcal{T}$, where $\Omega = \Omega^1 \cup \ldots \cup \Omega^T$, to one element $s \in \mathcal{S}$ by solving a set-partitioning formulation of the problem. Algorithm 1 presents the general structure of the proposed heuristic.

To draw elements from $\mathcal{P}$ (line 7 in Algorithm 1), we use randomized versions of 4 different TSP constructive heuristics: randomized nearest neighbor (RNN), randomized nearest insertion (RNI), randomized best insertion (RBI), and randomized farthest insertion (RFI). Although the strategies used to generate the randomized versions of the four heuristics are rather intuitive, for the sake of completeness we briefly describe them here.

Let p be the TSP tour being built by a given sampling heuristic, $\mathcal{W}$ the set of vertices visited by p, and $\mathcal{N} = \mathcal{V} \setminus \mathcal{W}$ an ordered set of not-routed vertices. For the sake of simplicity, we assume that sets $\mathcal{W}$ and $\mathcal{N}$ are updated every time a customer is added to p. Let us also define three metrics for every

Algorithm 1 Multi-space sampling heuristic: general structure

```
1: function MultiSpaceSampling(G,H)
2:     t ← 1
3:     Ω ← ∅
4:     while t ≤ T do
5:         for k = 1 to k = |H| do
6:             h ← H_k
7:             p^t ← h(G)
8:             ⟨Ω^t, s^t⟩ ←s-split(G, p^t)
9:             Ω ← Ω ∪ Ω^t
10:            if t = 1 then
11:                s* ← s^t
12:            else if f(s^t) < f(s*) then
13:                s* ← s^t
14:            end if
15:            t ← t + 1
16:        end for
17:    end while
18:    s* ← setPartitioningProblem(G, Ω, s*)
19:    return R ← s*
20: end function
```

customer $v \in \mathcal{N}$, namely, $d_{min}(v) = \min\{d_{v,u} | u \in \mathcal{W}\}$, $d_{max}(v) = \max\{d_{v,u} | u \in \mathcal{W}\}$, and $\Delta_{min}(v) = \min\{d_{u,v} + d_{v,w} - d_{u,w} | (u,w) \in p\}$. Finally, let k be a random integer selected in $[1, \min\{K, |\mathcal{N}|\}]$, where parameter K denotes the *randomization factor* of each heuristic. The four sampling heuristics operate as follows:

- **RNN:** Set $p = \{0\}$ and $u = 0$. At each iteration: identify the vertex v who is the k_{th} nearest vertex to u, append v to p, and set $u = v$. Stop when $|\mathcal{N}| = 0$ and append 0 to p to complete a tour.
- **RNI:** Initialize p as a tour starting at the depot and performing a round trip to a randomly selected customer (henceforth this procedure will be referred simply as initialize p). At each iteration: sort $\mathcal{N}$ in non-decreasing order of $d_{min}(v)$. Insert $v = \mathcal{N}_k$ in the best possible position in the tour (i.e., the position generating the shortest increment in the cost of the tour). Stop when $|\mathcal{N}| = 0$.
- **RFI:** Initialize p. At each iteration: sort $\mathcal{N}$ in non-decreasing order of $d_{max}(v)$ and insert $v = \mathcal{N}_k$ in the best possible position in the tour. Stop when $|\mathcal{N}| = 0$.
- **RBI:** Initialize p. At each iteration: sort $\mathcal{N}$ in non-decreasing order of $\Delta_{min}(v)$ and insert $v = \mathcal{N}_k$ in the best possible position in the tour. Stop when $|\mathcal{N}| = 0$.

To map each sampled element p^t to its corresponding $\Omega^p \subset \mathcal{T}$ (line 8 in Algorithm 1) our approach uses the s-split procedure for the VRPSD [8]. S-split was originally proposed to map an element in $\mathcal{P}$ to an element in $\mathcal{S}$ by optimally partitioning the giant tour into a set of feasible routes that form a VRPSD solution. Nonetheless, s-split accomplishes its mission by running a dynamic programming algorithm that evaluates every single feasible route that can be obtained from the giant tour without altering the order of the customers. In other words, by saving a reference to the set of routes evaluated by s-split while partitioning p^t, we obtain a mapping of p^t to a sub-set $\Omega^t \subset \mathcal{T}$. Since s-split also retrieves a mapping of p^t to a solution $s^t \in \mathcal{S}$, we keep a reference to the best solution found during the sampling phase (lines 10–14 in Algorithm 1) to have a good upper bound in the second phase of our heuristic. The leftmost frame in Figure 2 illustrates the operation of s-split in an execution of the proposed heuristic with $H = \{RFI, RNI\}$ and $T = 2$. For all details needed to implement s-split the reader is referred to [8, 9].

In the second phase, our approach maps the set $\Omega = \Omega^1 \cup \ldots \cup \Omega^T$ to a solution $s^* \in \mathcal{S}$ by solving a set-partitioning formulation of the VRPSD proposed by Christiansen and Lysgaard [3]:

$$\min_{\mathcal{R} \subseteq \Omega} \left\{ \sum_{r \in \mathcal{R}} E[C_r] : \bigcup_{r \in \mathcal{R}} = \mathcal{V}; r_i \bigcap r_j = \{0\} \, \forall r_i, r_j \in \mathcal{R} \right\} \tag{3}$$

The objective is then to select the best sub-set of routes from Ω to build the set of planed routes $\mathcal{R}$ (i.e., solution) ensuring that each customer will be visited by exactly one route. The rightmost frames in Figure 2 illustrate the operation of the mapping phase in our heuristic.

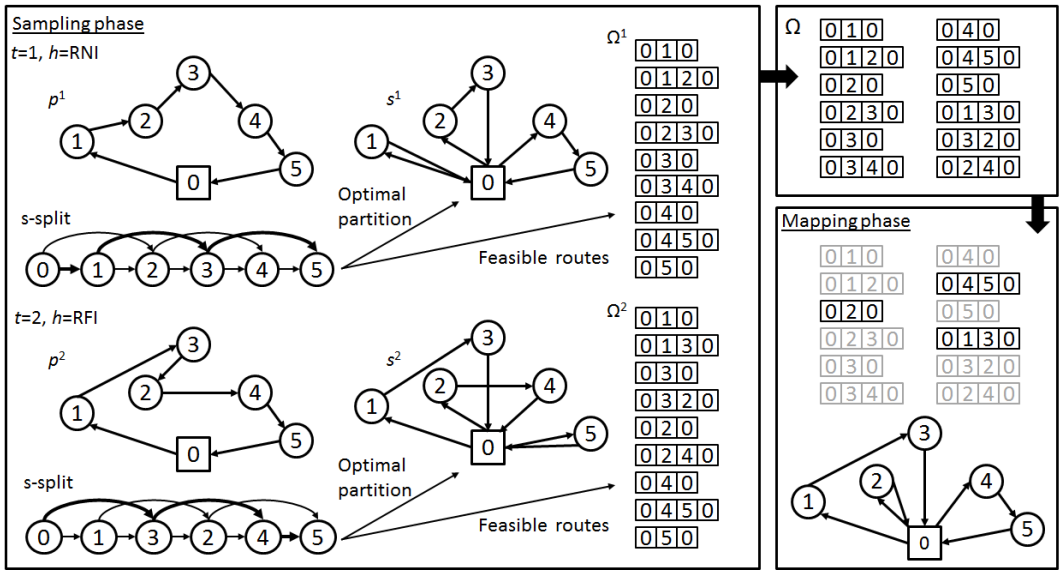


Figure 2: Multi-space sampling heuristic: example of an execution with $H = \{RNI, RFI\}$ and $T = 2$

In most VRP variants, it is possible to replace the set-partitioning formulation of the problem by an equivalent set-covering formulation aiming to gain in computational efficiency. This approach is based in the premise that the cheapest way to cover all customers is to cover them only once because the cost verifies the triangle inequality. Therefore, the optimal solution of the set-covering problem is a feasible VRP solution. In the VRPSD, however, that is not necessarily the case. Indeed, because the expected cost of failing while visiting a customer v depends on the expected demands of all customers previously visited by the route and on its distance to the depot, see Equation (2), there is no guarantee that a route $r = (0, \ldots, u, v, \ldots, 0)$ has a total expected cost $E[C_r]$ that is cheaper than that of a route $r' = (0, \ldots, u, w, v, \ldots, 0)$. Hence, the cheapest way to cover all customers may include covering some customers more than once, which leads to an unfeasible VRPSD solution. One can overcome this difficulty by implementing an ad-hoc verification and reparation procedure. After some preliminary experimentation using a set-covering formulation in the mapping phase, we assessed that the benefits obtained in terms of computational efficiency did not pay off the loss in simplicity of our heuristic, so we kept the original set-partitioning formulation.

It is worth mentioning that although in many VRP variants excellent solutions can be found by solving set-covering/set-partitioning formulations over a reduced set of routes, researchers often oversight the benefits of embedding this kind of component in heuristic procedures. Authors like Villegas et al. [15], however, have shown that applied as a post-optimization procedure, this approach may lead to significant improvements in the quality of final solutions. We consider important to remark that in the proposed heuristic, the set-partitioning model is an integral part of the method rather than only a post-optimization strategy.

3 Computational experiments

We implemented the proposed heuristic in Java (jre V.1.6.0_22-b04) and used the Gurobi Optimizer (version 4.5.1) for solving the set-partitioning model. To test our approach, we ran it on the 40-instance testbed proposed by Christiansen and Lysgaard [3]. These instances range from 16 to 60 customers and assume Poisson distributed customer demands. To assess the effectiveness of our heuristic, we compared our results to the best known solutions (BKSs) for the testbed which, to the best of our knowledge, are due to either [3], [6], [9], or to the three of them. It is worth mentioning that [3] provided optimality certificates for 19 out of the 40 instances. For each instance, we executed 10 runs with 5 different values for parameter T, namely, 1,000; 2,000; 5,000; 10,000; and 20,000. For the 2,000 runs ($= 40 \times 10 \times 5$) we set the value of K to 6 for RNI, RBI, RFI, and to 3 for RNN. All experiments were run in a PC with an Intel Xeon processor running at 2.4 GHz under Windows Server 2008 (64 bits) with 12 GB of RAM.

Table 1 summarizes the results of the experiments: the first column describes the performance metric, the following 5 columns report the results for the 5 different algorithm configurations, and the last column presents the results reported (also over 10 runs) by the simulated annealing approach by Goodson et al.

[6] which to our knowledge is currently the best-performing heuristic approach for the VRPSD. Detailed results for each instance can be found in Online Resources 1–5.

Performance	T					Goodson
Metric	1,000	2,000	5,000	10,000	20,000	et al.
Avg. Gap (%)	0.67	0.49	0.31	0.23	0.15	0.33
Max. Gap (%)	2.52	2.24	1.62	1.34	1.16	1.89
Avg. Gap best (%)	0.36	0.26	0.14	0.09	0.05	0.01
Avg. CPU (s)	12.90	22.75	50.19	93.43	180.78	268.66
Max. CPU (s)	45.00	82.05	179.61	314.09	782.77	603.80
Min. CPU (s)	0.69	1.00	1.90	3.29	5.91	9.00
Matched BKSs	13	17	22	27	29	37
Improved BKSs	1	2	3	4	4	-

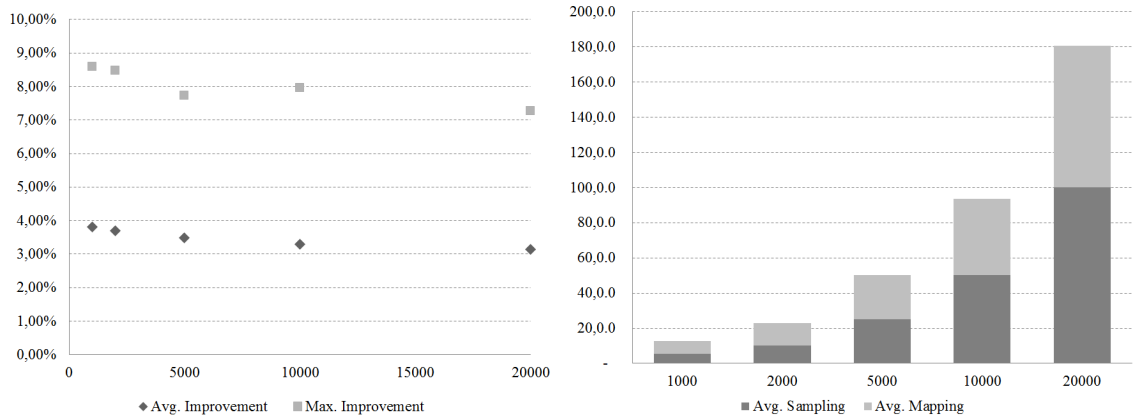
Table 1: Summary of results on the 40-instance testbed by Christiansen and Lysgaard

The results of the experiment show that the proposed approach is competitive to tackle VRPSD in terms of both accuracy and efficiency. Running on its most-effective configuration (i.e., $T = 20,000$), the heuristic matched 29 out of the 40 BKSs (i.e., 72.5% of the instances) and improved another 4 (i.e. 10% of the instance) while delivering solutions with an average gap of 0.15%. Moreover, the latter figure reduces to only 0.05% if we consider only the best solution found over the 10 runs for each instance. In terms of average performance, these results are superior to those reported by the sophisticated SA approach by Goodson et al. [6], although their algorithm reports a slightly better average gap when only the best solution found over the 10 runs is considered. Nonetheless, for $T \geq 5,000$, our approach achieves smaller maximum gaps than their SA, revealing the stability of the proposed multi-space sampling approach. In terms of computational efficiency, the results suggest that our approach outperforms the state-of-the-art SA. Running on its most-effective, yet most-expensive, configuration, our heuristic reports CPU times that are comparable to those reported by Goodson et al.; nonetheless, running in a faster configuration, i.e., $T = 10,000$, our algorithm is able to deliver solutions of a similar quality while investing (on average) less than half of the computational effort. It is fair to say that we did not scale the CPU times reported by Goodson et al. to account for differences in the testing environment, that is, programming language, operating system, processing power, etc. However, the testing environment used on their experiments is in theory at least as performing as ours, so we feel that our conclusion is well-sustained. Finally, it is worth highlighting the good performance of the proposed method when running on its fastest configuration (i.e., $T = 1,000$); while CPU times are under 45 seconds for every instance of the set, the reported solutions have an average gap of only 0.69%. The latter observation tips the balance towards our method when solving the VRPSD in practice.

3.1 Component analysis

To gain insight about our approach, we collected several statistics during our experimentation campaign. This data allowed us to analyze the impact of the different components on the performance of our heuristic. The first analysis focuses on the impact of each phase on the effectiveness and efficiency of the approach. Figure 3a shows the average and maximal improvement of the solution reported at the end of the mapping phase with respect to the best solution found in the sampling phase (over the 400 runs conducted for each value of T). The results show that the capacity of the mapping phase to improve the best solution from the sampling phase decreases with increments in T. This result is explained by the fact that drawing more samples during the sampling phase increases the chances of finding a better solution. Nonetheless, as shown in Figure 3a, even at the largest value of T, the mapping phase is able to improve about 3% (on average) the best solution found during the sampling phase. The latter observation along with the results in Table 1 shed some lights about the synergy between the two phases of our heuristic. On one hand, an approach based only on drawing a large number of solutions using the sampling heuristics would not be competitive enough. On the other hand, the mapping phase needs an extensive sampling phase to have a rich pool of routes (i.e., Ω) as input in order to find more competitive solutions for the problem.

Figure 3b shows the total average CPU time for each value of T (over the 400 runs conducted for each value of T) and how the execution time is split among the two phases of the heuristic. The results show that at any given value of T, the CPU time is equally distributed among the sampling and mapping phases. A similar behavior is observed within each instance. Online Resources 6 presents the details for every instance in the experiment conducted with $T = 10,000$. A close look to these results reveals



(a) (b)

Figure 3: Component analysis: impact of the sampling and mapping phases in the performance of the heuristic. Average values over the 400 runs conducted for each value of T

that independently of the size of the instance, the portion of time invested to solve the hard set-covering problem is directly correlated to the portion of time used by the sampling phase. Moreover, fitting a linear regression on the two variables confirms this finding. This observation provides some inside on how we can control the CPU time of our heuristic, since the time needed for the sampling phase can be roughly estimated a priori given the instance data and a value for T.

The second analysis focuses in the capacity of the sampling heuristics to contribute routes to Ω and to the final solution $\mathcal{R}$ reported by our approach. To measure this contribution, we save for each route in Ω a counter over the number of times that the route is generated by each sampling heuristic (i.e, is extracted from a TSP tour built by each sampling heuristic). Using this data, we calculated 5 metrics for each sampling heuristic: *diversity ratio* = (non-repeated routes/generated routes)$\times$100%; *absolute contribution to* Ω = (non-repeated routes/$|\Omega|$)$\times$100%; *relative contribution to* Ω = (exclusive routes/$|\Omega|$)$\times$100%; *absolute contribution to* $\mathcal{R}$ = (routes contributed to $\mathcal{R}/|\mathcal{R}|$)$\times$100%; and *relative contribution to* $\mathcal{R}$ = (exclusive routes contributed to $\mathcal{R}/|\mathcal{R}|$)$\times$100%. In this context we refer to *non-repeated routes* as the number of routes generated by the sampling heuristic after eliminating all duplicates. Similarly, we refer to *exclusive routes* as the number of routes that were not generated by any other sampling heuristic, although they may have been generated more than once by the heuristic. Figures 4a and 4b present for each metric the average value over the 40 test instances for $T = 10,000$.

The results show that RNN is the sampling heuristic that contributes the most to the diversity of Ω. As shown in Figure 4a, 62.71% of the routes in Ω were generated at least once by RNN and 53.48% of the routes in Ω are exclusive of RNN. On the other hand, the contribution to the pool diversity of the insertion-based sampling heuristics is more modest. A plausible explanation for this behavior is that RFI, RBI, and RNI are less sensible to the randomization, probably because they always insert the selected nodes into the best possible position in the route. However, the results in Figure 4b reveal that despite the relatively small number of routes that are generated by RFI, RBI, and RNI with respect to RNN, the former have a greater impact in the final solutions reported by the heuristic. For instance, 6.21% the routes found in the final solutions are exclusive from RFI. In other words, eliminating RFI from H, would have caused our heuristic to miss nearly 6% of the routes that it ended up using to build its high-quality final solutions. A similar observation can be made about RBI and RNI. Based on the contribution of RFI and RNI to the final solutions (81.72% and 79.73%, respectively) we decided perform a simple experiment (for $T = 10,000$) setting $H = \{RFI\}$ and $H\{RNI\}$. While the version of the heuristic that uses only RFI delivered solutions with an average gap of 0.32% with respect to the BKSs, the one using solely RNI produced results with an average gap of 0.37%. In both cases the results are dominated by those obtained by the original multi-space sampling heuristic. In conclusion, embedding different sampling heuristics into the sampling phase fosters diversity and contributes to the effectiveness of the heuristic. The latter is an important observation since VRP heuristics that are based on drawing samples of the solution space (e.g., GRASP) traditionally consider only one randomized heuristic in their design.

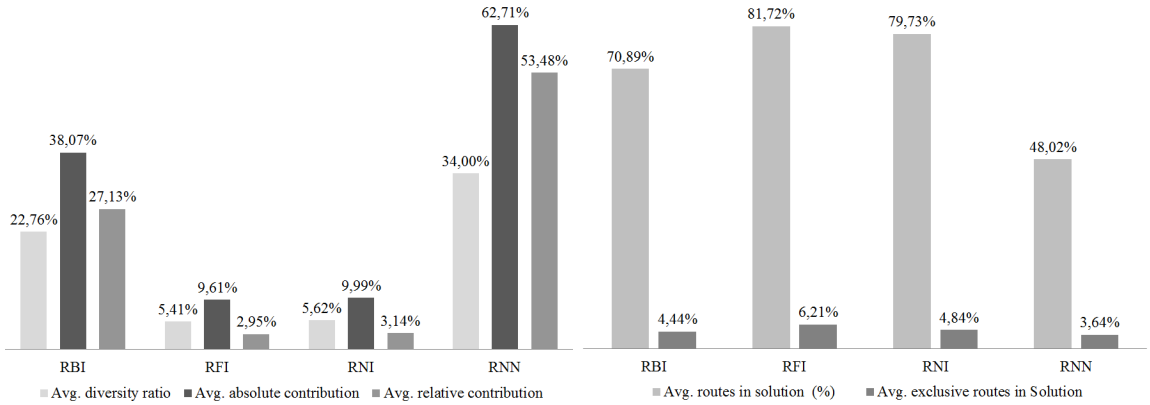


Figure 4: Component analysis metrics: average values on the 40-instance set by Christiansen and Lysgaard for $T = 10,000$

4 Conclusions and perspectives

This paper introduces a new multi-space sampling heuristic for the VRPSD. The approach uses three simple components: a set of four randomized heuristics for the TSP, a tour partitioning procedure, and a set-covering formulation; to sample the solution space and find solutions for the problem. Computational experiments conducted in a set of 40 instances from the literature showed that the proposed heuristics outperforms the state-of-the-art heuristic method for the problem in terms of both solution quality and computational efficiency. Our approach set 4 new best known solutions for the testbed and matched another 29. For the remaining 7 instances, the heuristic reported average gaps with respect to the BKSs ranging from 0.69% to 0.15% depending on its configuration. The paper also presents a detailed analysis of the impact of each algorithmic component in the performance of the heuristic. This analysis provides valuable insight about the proposed method and how its principles can be used to design or enhance vehicle routing heuristics.

Research currently underway includes the extension of our approach to tackle the VRPSD in two new scenarios: heterogeneous fleet and the case in which customer demands are correlated.

References

[1] A. Ak and A. Erera. A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands. *Transportation Science*, 41(2):222–237, 2007.

[2] L. Bianchi, M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto. Hybrid metaheuristics for the vehicle routing problem with stochastic demands. *Journal of Mathematical Modelling and Algorithms*, 5(1):91–110, 2006.

[3] C. Christiansen and J. Lysgaard. A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research Letters*, 35(6):773–781, 2007.

[4] A. Erera, M. Savelsbergh, and E. Uyar. Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints. *Networks*, 54(4):270–283, 2009.

[5] M. Gendreau, G. Laporte, and R. Séguin. A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research*, 44(3):469–477, 1996.

[6] J. C. Goodson, J. W. Ohlmann, and B. W. Thomas. Cyclic-order neighborhoods with application to the vehicle routing problem with stochastic demand. *European Journal of Operational Research*, In Press. doi: 10.1016/j.ejor.2011.09.023.

[7] G. Laporte, F. Louveaux, and L. Van Hamme. An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415–423, 2002.

[8] J. E. Mendoza, B. Castanier, C. Guéret, A. L. Medaglia, and N. Velasco. A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, 37(11):1886–1898, 2010.

[9] J. E. Mendoza, B. Castanier, C. Guéret, A. L. Medaglia, and N. Velasco. Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. *Transportation Science*, 45 (3):335–345, 2011.

[10] C. Novoa, R. Berger, J. Linderoth, and R. Storer. A set-partitioning-based model for the stochastic vehicle routing problem. Technical report, Texas State University, 2006.

[11] C. Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002, 2004.

[12] W. Rei, M. Gendreau, and P. Soriano. Local branching cuts for the 0-1 integer L-shaped algorithm. (CIRRELT-2007-23), 2007.

[13] W. Rei, M. Gendreau, and P. Soriano. A hybrid Monte Carlo local branching algorithm for the single vehicle routing problem with stochastic demands. *Transportation Science*, 44(1):136–146, 2010.

[14] N. Secomandi and F. Margot. Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operations Research*, 57(1):214–230, 2009.

[15] J. G. Villegas, C. Prins, C. Prodhon, A. L. Medaglia, and N. Velasco. A matheuristic for the truck and trailer routing problem. Technical report, Department of Industrial Engineering Universidad de Antioquia, 2011.

[16] W. H. Yang, K. Mathur, and R. Ballou. Stochastic vehicle routing with restocking. *Transportation Science*, 34(1):99–112, 2000.

Instance	BKS	Multi-space sampling heuristic running with T=1,000					Goodson et al.				
		CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best E[C]	Gap best E[C]	CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best	Gap
A-n32-k5	**853.6***	6.68	857.57	0.47%	856.28	0.31%	199.80	853.60	0.00%	853.60	0.00%
A-n33-k5	**704.2***	6.20	712.70	1.21%	710.00	0.82%	178.20	705.91	0.24%	704.20	0.00%
A-n33-k6	**793.9***	4.94	795.13	0.15%	794.15	0.03%	141.10	793.95	0.01%	793.90	0.00%
A-n39-k5	**869.18***	11.42	888.81	2.26%	879.07	1.14%	257.60	869.58	0.05%	869.18	0.00%
A-n39-k6	**876.6***	8.55	877.67	0.12%	876.60	0.00%	239.90	883.44	0.78%	876.60	0.00%
A-n45-k7	**1264.83***	14.46	1,282.74	1.42%	1,274.43	0.76%	216.00	1288.70	1.89%	1264.99	0.01%
E-n22-k4	**411.57***	2.13	412.60	0.25%	411.57	0.00%	104.10	411.57	0.00%	411.57	0.00%
E-n33-k4	**850.27***	10.18	867.11	1.98%	861.96	1.37%	371.70	851.24	0.11%	850.27	0.00%
P-n16-k8	**512.82***	0.69	512.82	0.00%	512.82	0.00%	9.00	512.82	0.00%	512.82	0.00%
P-n19-k2	**224.06***	3.07	224.06	0.00%	224.06	0.00%	234.80	224.06	0.00%	224.06	0.00%
P-n20-k2	**233.05***	4.25	236.02	1.27%	235.22	0.93%	269.60	233.05	0.00%	233.05	0.00%
P-n21-k2	**218.96***	4.36	219.06	0.05%	218.96	0.00%	332.70	218.96	0.00%	218.96	0.00%
P-n22-k2	**231.26***	5.66	231.32	0.03%	231.26	0.00%	352.30	231.26	0.00%	231.26	0.00%
P-n22-k8	**681.06***	1.36	681.28	0.03%	681.06	0.00%	28.60	681.06	0.00%	681.06	0.00%
P-n23-k8	**619.52***	1.32	619.99	0.08%	619.65	0.02%	21.10	619.57	0.01%	619.53	0.00%
P-n40-k5	**472.5***	12.28	474.12	0.34%	472.50	0.00%	367.20	472.50	0.00%	472.50	0.00%
P-n51-k10	**809.7***	10.53	813.36	0.45%	810.12	0.05%	159.90	816.95	0.90%	812.74	0.38%
P-n55-k15	**1068.05***	9.97	1,068.36	0.03%	1,068.05	0.00%	117.00	1072.17	0.39%	1068.05	0.00%
P-n60-k15	**1085.49***	12.75	1,086.01	0.05%	1,085.49	0.00%	134.80	1098.11	1.16%	1087.41	0.18%
A-n34-k5	826.87	6.74	827.60	0.09%	826.87	0.00%	236.40	827.26	0.05%	826.87	0.00%
A-n36-k5	858.71	10.92	880.35	2.52%	875.17	1.92%	276.10	859.48	0.09%	858.71	0.00%
A-n37-k5	708.34	13.92	712.55	0.59%	709.51	0.16%	386.90	709.67	0.19%	708.34	0.00%
A-n37-k6	1030.73	7.04	1,033.76	0.29%	1,031.16	0.04%	205.50	1031.74	0.10%	1030.73	0.00%
A-n38-k5	775.14	8.96	778.01	0.37%	777.82	0.35%	313.40	775.25	0.01%	775.14	0.00%
A-n44-k6	1025.48	13.84	1,034.63	0.89%	1,030.34	0.47%	281.40	1029.72	0.41%	1025.48	0.00%
A-n45-k6	1026.73	13.48	1,031.39	0.45%	1,026.87	0.01%	301.40	1027.92	0.12%	1026.73	0.00%
A-n46-k7	1002.22	12.55	1,006.36	0.41%	1,002.22	0.00%	314.10	1003.19	0.10%	1002.22	0.00%
A-n48-k7	1187.14	22.00	1,213.06	2.18%	1,203.61	1.39%	292.00	1188.06	0.08%	1187.14	0.00%
A-n53-k7	1124.27	23.32	1,137.65	1.19%	1,133.65	0.83%	468.80	1129.36	0.45%	1124.27	0.00%
A-n54-k7	1287.07	20.88	1,299.25	0.95%	1,296.66	0.75%	409.40	1292.29	0.41%	1287.07	0.00%
A-n55-k9	1179.11	15.26	1,187.09	0.68%	1,180.33	0.10%	265.50	1184.77	0.48%	1179.11	0.00%
A-n60-k9	1529.82	45.00	1,539.15	0.61%	1,534.85	0.33%	393.70	1535.35	0.36%	1529.82	0.00%
E-n51-k5	552.26	31.26	557.15	0.88%	553.31	0.19%	586.00	552.81	0.10%	552.26	0.00%
P-n45-k5	533.52	20.81	536.51	0.56%	534.52	0.19%	603.80	537.13	0.68%	533.52	0.00%
P-n50-k7	582.37	20.71	590.14	1.33%	588.36	1.03%	343.00	584.95	0.44%	582.37	0.00%
P-n50-k8	669.81	14.12	670.55	0.11%	670.18	0.05%	225.30	674.11	0.64%	669.81	0.00%
P-n50-k10	760.94	11.32	759.15	-0.23%	758.76	-0.29%	150.60	764.12	0.42%	760.94	0.00%
P-n55-k7	588.56	30.27	597.41	1.50%	593.59	0.85%	452.80	593.10	0.77%	588.56	0.00%
P-n55-k10	745.70	21.03	751.36	0.76%	750.25	0.61%	208.90	752.36	0.89%	745.70	0.00%
P-n60-k10	804.24	21.77	806.68	0.30%	805.24	0.12%	296.10	810.22	0.74%	804.24	0.00%
Max.		45.00		2.52%		1.92%	603.80		1.89%		0.38%
Min.		0.69		-0.23%		-0.29%	9.00		0.00%		0.00%
Avg.		12.90		0.67%		0.36%	268.66		0.33%		0.01%
Std. Dev.		9.22		0.70%		0.50%	132.73		0.40%		0.06%

Online Resources 1 Multi-space sampling heuristic: detailed results for the 40 instances by Christiansen and Lysgaard running with T=1,000

Mendoza J. E. and Villegas J. G. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Under review since Sep. 28, 2011.

Corresponding Author: Jorge E. Mendoza, Université Catholique de l'Ouest, Angers (France). Email: jorge.mendoza@uco.fr

Instance	BKS	Multi-space sampling heuristic running with T=2,000					Goodson et al.				
		CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best E[C]	Gap best E[C]	CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best	Gap
A-n32-k5	**853.6***	11.99	856.18	0.30%	854.14	0.06%	199.80	853.60	0.00%	853.60	0.00%
A-n33-k5	**704.2***	10.50	711.86	1.09%	709.53	0.76%	178.20	705.91	0.24%	704.20	0.00%
A-n33-k6	**793.9***	8.99	794.81	0.11%	794.15	0.03%	141.10	793.95	0.01%	793.90	0.00%
A-n39-k5	**869.18***	19.48	884.26	1.74%	878.64	1.09%	257.60	869.58	0.05%	869.18	0.00%
A-n39-k6	**876.6***	16.01	877.03	0.05%	876.60	0.00%	239.90	883.44	0.78%	876.60	0.00%
A-n45-k7	**1264.83***	24.94	1,277.21	0.98%	1,273.89	0.72%	216.00	1288.70	1.89%	1264.99	0.01%
E-n22-k4	**411.57***	3.31	412.57	0.24%	411.57	0.00%	104.10	411.57	0.00%	411.57	0.00%
E-n33-k4	**850.27***	17.47	861.73	1.35%	854.74	0.53%	371.70	851.24	0.11%	850.27	0.00%
P-n16-k8	**512.82***	1.00	512.82	0.00%	512.82	0.00%	9.00	512.82	0.00%	512.82	0.00%
P-n19-k2	**224.06***	5.80	224.06	0.00%	224.06	0.00%	234.80	224.06	0.00%	224.06	0.00%
P-n20-k2	**233.05***	7.71	235.92	1.23%	235.22	0.93%	269.60	233.05	0.00%	233.05	0.00%
P-n21-k2	**218.96***	7.91	219.06	0.05%	218.96	0.00%	332.70	218.96	0.00%	218.96	0.00%
P-n22-k2	**231.26***	10.53	231.26	0.00%	231.26	0.00%	352.30	231.26	0.00%	231.26	0.00%
P-n22-k8	**681.06***	2.09	681.06	0.00%	681.06	0.00%	28.60	681.06	0.00%	681.06	0.00%
P-n23-k8	**619.52***	2.15	619.73	0.03%	619.53	0.00%	21.10	619.57	0.01%	619.53	0.00%
P-n40-k5	**472.5***	20.60	472.79	0.06%	472.50	0.00%	367.20	472.50	0.00%	472.50	0.00%
P-n51-k10	**809.7***	18.40	811.98	0.28%	810.12	0.05%	159.90	816.95	0.90%	812.74	0.38%
P-n55-k15	**1068.05***	18.29	1,068.32	0.03%	1,068.05	0.00%	117.00	1072.17	0.39%	1068.05	0.00%
P-n60-k15	**1085.49***	23.64	1,085.91	0.04%	1,085.49	0.00%	134.80	1098.11	1.16%	1087.41	0.18%
A-n34-k5	826.87	12.70	826.87	0.00%	826.87	0.00%	236.40	827.26	0.05%	826.87	0.00%
A-n36-k5	858.71	19.06	877.93	2.24%	873.17	1.68%	276.10	859.48	0.09%	858.71	0.00%
A-n37-k5	708.34	25.74	710.12	0.25%	709.51	0.16%	386.90	709.67	0.19%	708.34	0.00%
A-n37-k6	1030.73	11.75	1,031.68	0.09%	1,030.73	0.00%	205.50	1031.74	0.10%	1030.73	0.00%
A-n38-k5	775.14	16.10	777.74	0.34%	776.79	0.21%	313.40	775.25	0.01%	775.14	0.00%
A-n44-k6	1025.48	25.80	1,032.92	0.73%	1,027.67	0.21%	281.40	1029.72	0.41%	1025.48	0.00%
A-n45-k6	1026.73	24.62	1,029.63	0.28%	1,026.87	0.01%	301.40	1027.92	0.12%	1026.73	0.00%
A-n46-k7	1002.22	22.58	1,005.21	0.30%	1,002.22	0.00%	314.10	1003.19	0.10%	1002.22	0.00%
A-n48-k7	1187.14	36.31	1,209.24	1.86%	1,203.61	1.39%	292.00	1188.06	0.08%	1187.14	0.00%
A-n53-k7	1124.27	43.62	1,135.57	1.01%	1,131.35	0.63%	468.80	1129.36	0.45%	1124.27	0.00%
A-n54-k7	1287.07	38.06	1,296.35	0.72%	1,292.49	0.42%	409.40	1292.29	0.41%	1287.07	0.00%
A-n55-k9	1179.11	29.56	1,185.77	0.57%	1,180.33	0.10%	265.50	1184.77	0.48%	1179.11	0.00%
A-n60-k9	1529.82	82.05	1,535.15	0.35%	1,529.82	0.00%	393.70	1535.35	0.36%	1529.82	0.00%
E-n51-k5	552.26	54.88	554.87	0.47%	553.26	0.18%	586.00	552.81	0.10%	552.26	0.00%
P-n45-k5	533.52	36.77	535.48	0.37%	534.52	0.19%	603.80	537.13	0.68%	533.52	0.00%
P-n50-k7	582.37	36.01	587.84	0.94%	586.19	0.66%	343.00	584.95	0.44%	582.37	0.00%
P-n50-k8	669.81	22.99	670.02	0.03%	669.23	-0.09%	225.30	674.11	0.64%	669.81	0.00%
P-n50-k10	760.94	18.93	758.98	-0.26%	758.76	-0.29%	150.60	764.12	0.42%	760.94	0.00%
P-n55-k7	588.56	49.60	595.55	1.19%	592.36	0.64%	452.80	593.10	0.77%	588.56	0.00%
P-n55-k10	745.70	33.86	749.63	0.53%	746.17	0.06%	208.90	752.36	0.89%	745.70	0.00%
P-n60-k10	804.24	38.14	805.74	0.19%	805.24	0.12%	296.10	810.22	0.74%	804.24	0.00%
Max.		82.05		2.24%		1.68%	603.80		1.89%		0.38%
Min.		1.00		-0.26%		-0.29%	9.00		0.00%		0.00%
Avg.		22.75		0.49%		0.26%	268.66		0.33%		0.01%
Std. Dev.		16.32		0.58%		0.43%	132.73		0.40%		0.06%

Online Resources 2 Multi-space sampling heuristic: detailed results for the 40 instances by Christiansen and Lysgaard running with T=2,000

Mendoza J. E. and Villegas J. G. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Under review since Sep. 28, 2011.

Corresponding Author: Jorge E. Mendoza, Université Catholique de l'Ouest, Angers (France). Email: jorge.mendoza@uco.fr

Instance	BKS	Multi-space sampling heuristic running with T=5,000					Goodson et al.				
		CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best E[C]	Gap best E[C]	CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best	Gap
A-n32-k5	**853.6***	25.80	855.41	0.21%	853.60	0.00%	199.80	853.60	0.00%	853.60	0.00%
A-n33-k5	**704.2***	24.45	710.64	0.91%	709.40	0.74%	178.20	705.91	0.24%	704.20	0.00%
A-n33-k6	**793.9***	18.55	794.46	0.07%	794.15	0.03%	141.10	793.95	0.01%	793.90	0.00%
A-n39-k5	**869.18***	41.73	878.27	1.05%	869.45	0.03%	257.60	869.58	0.05%	869.18	0.00%
A-n39-k6	**876.6***	39.18	876.60	0.00%	876.60	0.00%	239.90	883.44	0.78%	876.60	0.00%
A-n45-k7	**1264.83***	56.20	1,273.23	0.66%	1,269.49	0.37%	216.00	1288.70	1.89%	1264.99	0.01%
E-n22-k4	**411.57***	6.56	411.92	0.09%	411.57	0.00%	104.10	411.57	0.00%	411.57	0.00%
E-n33-k4	**850.27***	38.15	856.51	0.73%	854.74	0.53%	371.70	851.24	0.11%	850.27	0.00%
P-n16-k8	**512.82***	1.90	512.82	0.00%	512.82	0.00%	9.00	512.82	0.00%	512.82	0.00%
P-n19-k2	**224.06***	12.17	224.06	0.00%	224.06	0.00%	234.80	224.06	0.00%	224.06	0.00%
P-n20-k2	**233.05***	15.60	235.25	0.94%	233.05	0.00%	269.60	233.05	0.00%	233.05	0.00%
P-n21-k2	**218.96***	17.15	218.96	0.00%	218.96	0.00%	332.70	218.96	0.00%	218.96	0.00%
P-n22-k2	**231.26***	23.30	231.26	0.00%	231.26	0.00%	352.30	231.26	0.00%	231.26	0.00%
P-n22-k8	**681.06***	4.02	681.06	0.00%	681.06	0.00%	28.60	681.06	0.00%	681.06	0.00%
P-n23-k8	**619.52***	4.30	619.61	0.01%	619.53	0.00%	21.10	619.57	0.01%	619.53	0.00%
P-n40-k5	**472.5***	44.07	472.50	0.00%	472.50	0.00%	367.20	472.50	0.00%	472.50	0.00%
P-n51-k10	**809.7***	40.14	810.62	0.11%	810.12	0.05%	159.90	816.95	0.90%	812.74	0.38%
P-n55-k15	**1068.05***	43.45	1,068.05	0.00%	1,068.05	0.00%	117.00	1072.17	0.39%	1068.05	0.00%
P-n60-k15	**1085.49***	56.01	1,085.70	0.02%	1,085.49	0.00%	134.80	1098.11	1.16%	1087.41	0.18%
A-n34-k5	826.87	26.79	826.87	0.00%	826.87	0.00%	236.40	827.26	0.05%	826.87	0.00%
A-n36-k5	858.71	39.53	872.59	1.62%	864.71	0.70%	276.10	859.48	0.09%	858.71	0.00%
A-n37-k5	708.34	55.43	709.51	0.16%	709.51	0.16%	386.90	709.67	0.19%	708.34	0.00%
A-n37-k6	1030.73	25.85	1,031.06	0.03%	1,030.73	0.00%	205.50	1031.74	0.10%	1030.73	0.00%
A-n38-k5	775.14	36.02	777.53	0.31%	776.50	0.17%	313.40	775.25	0.01%	775.14	0.00%
A-n44-k6	1025.48	55.36	1,027.67	0.21%	1,025.96	0.05%	281.40	1029.72	0.41%	1025.48	0.00%
A-n45-k6	1026.73	49.74	1,027.28	0.05%	1,026.87	0.01%	301.40	1027.92	0.12%	1026.73	0.00%
A-n46-k7	1002.22	51.42	1,002.37	0.01%	1,002.22	0.00%	314.10	1003.19	0.10%	1002.22	0.00%
A-n48-k7	1187.14	73.73	1,204.15	1.43%	1,202.34	1.28%	292.00	1188.06	0.08%	1187.14	0.00%
A-n53-k7	1124.27	104.42	1,131.67	0.66%	1,128.61	0.39%	468.80	1129.36	0.45%	1124.27	0.00%
A-n54-k7	1287.07	92.14	1,294.20	0.55%	1,291.54	0.35%	409.40	1292.29	0.41%	1287.07	0.00%
A-n55-k9	1179.11	67.32	1,184.37	0.45%	1,179.11	0.00%	265.50	1184.77	0.48%	1179.11	0.00%
A-n60-k9	1529.82	179.61	1,531.18	0.09%	1,529.82	0.00%	393.70	1535.35	0.36%	1529.82	0.00%
E-n51-k5	552.26	135.60	552.99	0.13%	552.29	0.01%	586.00	552.81	0.10%	552.26	0.00%
P-n45-k5	533.52	80.95	534.64	0.21%	534.52	0.19%	603.80	537.13	0.68%	533.52	0.00%
P-n50-k7	582.37	71.98	587.63	0.90%	586.19	0.66%	343.00	584.95	0.44%	582.37	0.00%
P-n50-k8	669.81	52.49	669.57	-0.04%	669.23	-0.09%	225.30	674.11	0.64%	669.81	0.00%
P-n50-k10	760.94	40.12	758.84	-0.28%	758.76	-0.29%	150.60	764.12	0.42%	760.94	0.00%
P-n55-k7	588.56	107.67	592.92	0.74%	590.45	0.32%	452.80	593.10	0.77%	588.56	0.00%
P-n55-k10	745.70	67.39	747.91	0.30%	744.16	-0.21%	208.90	752.36	0.89%	745.70	0.00%
P-n60-k10	804.24	81.31	805.05	0.10%	804.60	0.05%	296.10	810.22	0.74%	804.24	0.00%
Max.		179.61		1.62%		1.28%	603.80		1.89%		0.38%
Min.		1.90		-0.28%		-0.29%	9.00		0.00%		0.00%
Avg.		50.19		0.31%		0.14%	268.66		0.33%		0.01%
Std. Dev.		36.86		0.43%		0.29%	132.73		0.40%		0.06%

Online Resources 3 Multi-space sampling heuristic: detailed results for the 40 instances by Christiansen and Lysgaard running with T=5,000

Mendoza J. E. and Villegas J. G. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Under review since Sep. 28, 2011.
Corresponding Author: Jorge E. Mendoza, Université Catholique de l'Ouest, Angers (France). Email: jorge.mendoza@uco.fr

Instance	BKS	Multi-space sampling heuristic running with T=10,000					Goodson et al.				
		CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best E[C]	Gap best E[C]	CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best	Gap
A-n32-k5	**853.6***	46.93	854.82	0.14%	853.60	0.00%	199.80	853.60	0.00%	853.60	0.00%
A-n33-k5	**704.2***	43.46	709.44	0.74%	709.40	0.74%	178.20	705.91	0.24%	704.20	0.00%
A-n33-k6	**793.9***	34.40	794.05	0.02%	793.90	0.00%	141.10	793.95	0.01%	793.90	0.00%
A-n39-k5	**869.18***	77.42	874.84	0.65%	869.45	0.03%	257.60	869.58	0.05%	869.18	0.00%
A-n39-k6	**876.6***	75.85	876.60	0.00%	876.60	0.00%	239.90	883.44	0.78%	876.60	0.00%
A-n45-k7	**1264.83***	93.71	1,270.94	0.48%	1,266.57	0.14%	216.00	1288.70	1.89%	1264.99	0.01%
E-n22-k4	**411.57***	11.93	411.67	0.02%	411.57	0.00%	104.10	411.57	0.00%	411.57	0.00%
E-n33-k4	**850.27***	69.40	854.91	0.55%	854.74	0.53%	371.70	851.24	0.11%	850.27	0.00%
P-n16-k8	**512.82***	3.29	512.82	0.00%	512.82	0.00%	9.00	512.82	0.00%	512.82	0.00%
P-n19-k2	**224.06***	22.18	224.06	0.00%	224.06	0.00%	234.80	224.06	0.00%	224.06	0.00%
P-n20-k2	**233.05***	26.93	234.36	0.56%	233.05	0.00%	269.60	233.05	0.00%	233.05	0.00%
P-n21-k2	**218.96***	32.32	218.96	0.00%	218.96	0.00%	332.70	218.96	0.00%	218.96	0.00%
P-n22-k2	**231.26***	43.84	231.26	0.00%	231.26	0.00%	352.30	231.26	0.00%	231.26	0.00%
P-n22-k8	**681.06***	7.30	681.06	0.00%	681.06	0.00%	28.60	681.06	0.00%	681.06	0.00%
P-n23-k8	**619.52***	7.78	619.61	0.01%	619.53	0.00%	21.10	619.57	0.01%	619.53	0.00%
P-n40-k5	**472.5***	81.68	472.50	0.00%	472.50	0.00%	367.20	472.50	0.00%	472.50	0.00%
P-n51-k10	**809.7***	79.14	810.01	0.04%	809.71	0.00%	159.90	816.95	0.90%	812.74	0.38%
P-n55-k15	**1068.05***	88.75	1,068.05	0.00%	1,068.05	0.00%	117.00	1072.17	0.39%	1068.05	0.00%
P-n60-k15	**1085.49***	112.74	1,085.49	0.00%	1,085.49	0.00%	134.80	1098.11	1.16%	1087.41	0.18%
A-n34-k5	826.87	45.58	826.87	0.00%	826.87	0.00%	236.40	827.26	0.05%	826.87	0.00%
A-n36-k5	858.71	72.50	870.25	1.34%	858.71	0.00%	276.10	859.48	0.09%	858.71	0.00%
A-n37-k5	708.34	92.78	709.51	0.16%	709.51	0.16%	386.90	709.67	0.19%	708.34	0.00%
A-n37-k6	1030.73	48.01	1,030.97	0.02%	1,030.73	0.00%	205.50	1031.74	0.10%	1030.73	0.00%
A-n38-k5	775.14	64.28	777.09	0.25%	776.50	0.17%	313.40	775.25	0.01%	775.14	0.00%
A-n44-k6	1025.48	108.29	1,026.45	0.09%	1,025.54	0.01%	281.40	1029.72	0.41%	1025.48	0.00%
A-n45-k6	1026.73	94.83	1,026.87	0.01%	1,026.87	0.01%	301.40	1027.92	0.12%	1026.73	0.00%
A-n46-k7	1002.22	99.32	1,002.29	0.01%	1,002.22	0.00%	314.10	1003.19	0.10%	1002.22	0.00%
A-n48-k7	1187.14	141.43	1,203.00	1.34%	1,200.90	1.16%	292.00	1188.06	0.08%	1187.14	0.00%
A-n53-k7	1124.27	186.19	1,129.76	0.49%	1,127.55	0.29%	468.80	1129.36	0.45%	1124.27	0.00%
A-n54-k7	1287.07	182.30	1,293.00	0.46%	1,290.18	0.24%	409.40	1292.29	0.41%	1287.07	0.00%
A-n55-k9	1179.11	125.42	1,182.50	0.29%	1,179.11	0.00%	265.50	1184.77	0.48%	1179.11	0.00%
A-n60-k9	1529.82	314.09	1,529.83	0.00%	1,529.82	0.00%	393.70	1535.35	0.36%	1529.82	0.00%
E-n51-k5	552.26	244.34	552.64	0.07%	552.26	0.00%	586.00	552.81	0.10%	552.26	0.00%
P-n45-k5	533.52	156.92	534.52	0.19%	534.52	0.19%	603.80	537.13	0.68%	533.52	0.00%
P-n50-k7	582.37	133.82	587.19	0.83%	585.61	0.56%	343.00	584.95	0.44%	582.37	0.00%
P-n50-k8	669.81	91.70	669.28	-0.08%	669.23	-0.09%	225.30	674.11	0.64%	669.81	0.00%
P-n50-k10	760.94	77.62	758.78	-0.28%	758.76	-0.29%	150.60	764.12	0.42%	760.94	0.00%
P-n55-k7	588.56	213.34	591.24	0.46%	589.64	0.18%	452.80	593.10	0.77%	588.56	0.00%
P-n55-k10	745.70	128.50	746.74	0.14%	744.16	-0.21%	208.90	752.36	0.89%	745.70	0.00%
P-n60-k10	804.24	156.78	804.42	0.02%	803.60	-0.08%	296.10	810.22	0.74%	804.24	0.00%
Max.		314.09		1.34%		1.16%	603.80		1.89%		0.38%
Min.		3.29		-0.28%		-0.29%	9.00		0.00%		0.00%
Avg.		93.43		0.23%		0.09%	268.66		0.33%		0.01%
Std. Dev.		67.55		0.36%		0.26%	132.73		0.40%		0.06%

Online Resources 4 Multi-space sampling heuristic: detailed results for the 40 instances by Christiansen and Lysgaard running with T=10,000

Mendoza J. E. and Villegas J. G. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Under review since Sep. 28, 2011.
Corresponding Author: Jorge E. Mendoza, Université Catholique de l'Ouest, Angers (France). Email: jorge.mendoza@uco.fr

Instance	BKS	Multi-space sampling heuristic running with T=20,000					Goodson et al.				
		CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best E[C]	Gap best E[C]	CPU (s)	Avg. E[C]	Gap Avg. E[C]	Best	Gap
A-n32-k5	**853.6***	83.80	854.13	0.06%	853.60	0.00%	199.80	853.60	0.00%	853.60	0.00%
A-n33-k5	**704.2***	81.30	707.99	0.54%	704.20	0.00%	178.20	705.91	0.24%	704.20	0.00%
A-n33-k6	**793.9***	62.94	794.03	0.02%	793.90	0.00%	141.10	793.95	0.01%	793.90	0.00%
A-n39-k5	**869.18***	145.48	872.41	0.37%	869.45	0.03%	257.60	869.58	0.05%	869.18	0.00%
A-n39-k6	**876.6***	143.78	876.60	0.00%	876.60	0.00%	239.90	883.44	0.78%	876.60	0.00%
A-n45-k7	**1264.83***	173.75	1,268.63	0.30%	1,266.57	0.14%	216.00	1288.70	1.89%	1264.99	0.01%
E-n22-k4	**411.57***	22.10	411.57	0.00%	411.57	0.00%	104.10	411.57	0.00%	411.57	0.00%
E-n33-k4	**850.27***	134.35	854.64	0.51%	853.04	0.33%	371.70	851.24	0.11%	850.27	0.00%
P-n16-k8	**512.82***	5.91	512.82	0.00%	512.82	0.00%	9.00	512.82	0.00%	512.82	0.00%
P-n19-k2	**224.06***	40.29	224.06	0.00%	224.06	0.00%	234.80	224.06	0.00%	224.06	0.00%
P-n20-k2	**233.05***	46.08	233.59	0.23%	233.05	0.00%	269.60	233.05	0.00%	233.05	0.00%
P-n21-k2	**218.96***	61.36	218.96	0.00%	218.96	0.00%	332.70	218.96	0.00%	218.96	0.00%
P-n22-k2	**231.26***	81.34	231.26	0.00%	231.26	0.00%	352.30	231.26	0.00%	231.26	0.00%
P-n22-k8	**681.06***	14.19	681.06	0.00%	681.06	0.00%	28.60	681.06	0.00%	681.06	0.00%
P-n23-k8	**619.52***	14.98	619.59	0.01%	619.53	0.00%	21.10	619.57	0.01%	619.53	0.00%
P-n40-k5	**472.5***	155.27	472.50	0.00%	472.50	0.00%	367.20	472.50	0.00%	472.50	0.00%
P-n51-k10	**809.7***	151.07	810.01	0.04%	809.71	0.00%	159.90	816.95	0.90%	812.74	0.38%
P-n55-k15	**1068.05***	174.07	1,068.05	0.00%	1,068.05	0.00%	117.00	1072.17	0.39%	1068.05	0.00%
P-n60-k15	**1085.49***	220.70	1,085.49	0.00%	1,085.49	0.00%	134.80	1098.11	1.16%	1087.41	0.18%
A-n34-k5	826.87	84.24	826.87	0.00%	826.87	0.00%	236.40	827.26	0.05%	826.87	0.00%
A-n36-k5	858.71	128.27	865.59	0.80%	858.71	0.00%	276.10	859.48	0.09%	858.71	0.00%
A-n37-k5	708.34	179.40	709.51	0.16%	709.51	0.16%	386.90	709.67	0.19%	708.34	0.00%
A-n37-k6	1030.73	90.15	1,030.73	0.00%	1,030.73	0.00%	205.50	1031.74	0.10%	1030.73	0.00%
A-n38-k5	775.14	118.63	776.33	0.15%	775.13	0.00%	313.40	775.25	0.01%	775.14	0.00%
A-n44-k6	1025.48	197.09	1,025.95	0.05%	1,025.54	0.01%	281.40	1029.72	0.41%	1025.48	0.00%
A-n45-k6	1026.73	173.74	1,026.87	0.01%	1,026.87	0.01%	301.40	1027.92	0.12%	1026.73	0.00%
A-n46-k7	1002.22	192.27	1,002.22	0.00%	1,002.22	0.00%	314.10	1003.19	0.10%	1002.22	0.00%
A-n48-k7	1187.14	281.82	1,200.96	1.16%	1,198.07	0.92%	292.00	1188.06	0.08%	1187.14	0.00%
A-n53-k7	1124.27	346.62	1,128.09	0.34%	1,125.78	0.13%	468.80	1129.36	0.45%	1124.27	0.00%
A-n54-k7	1287.07	352.10	1,291.50	0.34%	1,287.82	0.06%	409.40	1292.29	0.41%	1287.07	0.00%
A-n55-k9	1179.11	238.76	1,180.84	0.15%	1,179.11	0.00%	265.50	1184.77	0.48%	1179.11	0.00%
A-n60-k9	1529.82	782.77	1,529.83	0.00%	1,529.82	0.00%	393.70	1535.35	0.36%	1529.82	0.00%
E-n51-k5	552.26	451.47	552.42	0.03%	552.26	0.00%	586.00	552.81	0.10%	552.26	0.00%
P-n45-k5	533.52	283.84	534.52	0.19%	534.52	0.19%	603.80	537.13	0.68%	533.52	0.00%
P-n50-k7	582.37	248.88	586.36	0.69%	585.05	0.46%	343.00	584.95	0.44%	582.37	0.00%
P-n50-k8	669.81	176.39	669.23	-0.09%	669.23	-0.09%	225.30	674.11	0.64%	669.81	0.00%
P-n50-k10	760.94	145.89	758.76	-0.29%	758.76	-0.29%	150.60	764.12	0.42%	760.94	0.00%
P-n55-k7	588.56	426.56	590.80	0.38%	589.64	0.18%	452.80	593.10	0.77%	588.56	0.00%
P-n55-k10	745.70	230.37	745.24	-0.06%	743.04	-0.36%	208.90	752.36	0.89%	745.70	0.00%
P-n60-k10	804.24	289.00	804.42	0.02%	803.60	-0.08%	296.10	810.22	0.74%	804.24	0.00%
Max.		782.77		1.16%		0.92%	603.80		1.89%		0.38%
Min.		5.91		-0.29%		-0.36%	9.00		0.00%		0.00%
Avg.		180.78		0.15%		0.05%	268.66		0.33%		0.01%
Std. Dev.		146.41		0.28%		0.19%	132.73		0.40%		0.06%

Online Resources 5 Multi-space sampling heuristic: detailed results for the 40 instances by Christiansen and Lysgaard running with T=20,000

Mendoza J. E. and Villegas J. G. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Under review since Sep. 28, 2011.
Corresponding Author: Jorge E. Mendoza, Université Catholique de l'Ouest, Angers (France). Email: jorge.mendoza@uco.fr

Instance	Multi-space sampling heuristic running with T=10,000							
	Avg. E[C] Sampling	*Avg. E[C] Mapping*	*Avg. Improvement*	*Avg. Total CPU*	*Avg. CPU Sampling*	*Percentage*	*Avg. CPU Mapping*	*Percentage*
A-n34-k5	847.43	826.87	2.49%	45.58	25.18	55.25%	20.40	44.75%
A-n36-k5	905.28	870.25	4.03%	72.50	32.50	44.83%	40.00	55.17%
A-n37-k5	723.95	709.51	2.04%	92.78	36.60	39.44%	56.19	60.56%
A-n37-k6	1,052.05	1,030.97	2.04%	48.01	29.81	62.09%	18.20	37.91%
A-n38-k5	783.32	777.09	0.80%	64.28	34.41	53.53%	29.87	46.47%
A-n44-k6	1,061.33	1,026.45	3.40%	108.29	51.41	47.47%	56.89	52.53%
A-n45-k6	1,054.23	1,026.87	2.66%	94.83	52.93	55.81%	41.91	44.19%
A-n46-k7	1,049.21	1,002.29	4.68%	99.32	55.76	56.14%	43.56	43.86%
A-n48-k7	1,266.07	1,203.00	5.24%	141.43	62.42	44.13%	79.01	55.87%
A-n53-k7	1,184.22	1,129.76	4.82%	186.19	84.73	45.51%	101.46	54.49%
A-n54-k7	1,338.94	1,293.00	3.55%	182.30	94.22	51.68%	88.08	48.32%
A-n55-k9	1,227.34	1,182.50	3.79%	125.42	89.12	71.06%	36.29	28.94%
A-n60-k9	1,599.47	1,529.83	4.55%	314.09	117.13	37.29%	196.96	62.71%
E-n51-k5	580.02	552.64	4.96%	244.34	88.33	36.15%	156.02	63.85%
P-n45-k5	552.78	534.52	3.42%	156.92	61.06	38.91%	95.86	61.09%
P-n50-k7	618.65	587.19	5.36%	133.82	71.34	53.31%	62.48	46.69%
P-n50-k8	707.67	669.28	5.74%	91.70	65.21	71.12%	26.49	28.88%
P-n50-k10	791.41	758.78	4.30%	77.62	62.15	80.07%	15.47	19.93%
P-n55-k7	627.62	591.24	6.15%	213.34	99.88	46.82%	113.47	53.18%
P-n55-k10	787.05	746.74	5.40%	128.50	85.99	66.92%	42.51	33.08%
P-n60-k10	848.82	804.42	5.52%	156.78	108.56	69.24%	48.22	30.76%
A-n32-k5	877.75	854.82	2.68%	46.93	23.60	50.29%	23.33	49.71%
A-n33-k5	722.58	709.44	1.85%	43.46	23.71	54.57%	19.75	45.43%
A-n33-k6	813.30	794.05	2.42%	34.40	21.81	63.40%	12.59	36.60%
A-n39-k5	929.61	874.84	6.26%	77.42	39.18	50.60%	38.24	49.40%
A-n39-k6	897.31	876.60	2.36%	75.85	37.48	49.41%	38.37	50.59%
A-n45-k7	1,341.96	1,270.94	5.59%	93.71	51.52	54.98%	42.19	45.02%
E-n22-k4	418.27	411.67	1.60%	11.93	8.98	75.29%	2.95	24.71%
E-n33-k4	888.33	854.91	3.91%	69.40	30.27	43.62%	39.13	56.38%
P-n16-k8	514.73	512.82	0.37%	3.29	3.24	98.68%	0.04	1.32%
P-n19-k2	224.06	224.06	0.00%	22.18	10.05	45.30%	12.13	54.70%
P-n20-k2	235.84	234.36	0.63%	26.93	11.76	43.68%	15.17	56.32%
P-n21-k2	220.44	218.96	0.67%	32.32	13.51	41.80%	18.81	58.20%
P-n22-k2	231.68	231.26	0.18%	43.84	14.87	33.92%	28.97	66.08%
P-n22-k8	687.94	681.06	1.01%	7.30	6.81	93.27%	0.49	6.73%
P-n23-k8	630.49	619.61	1.76%	7.78	7.32	94.09%	0.46	5.91%
P-n40-k5	486.82	472.50	3.03%	81.68	43.31	53.03%	38.36	46.97%
P-n51-k10	851.15	810.01	5.08%	79.14	65.45	82.70%	13.69	17.30%
P-n55-k15	1,103.21	1,068.05	3.29%	88.75	84.77	95.52%	3.98	4.48%
P-n60-k15	1,134.00	1,085.49	4.47%	112.74	105.78	93.83%	6.96	6.17%
Max.			6.26%	314.09	117.13	98.68%	196.96	66.08%
Min.			0.00%	3.29	3.24	33.92%	0.04	1.32%
Avg.			3.30%	93.43	50.30	58.62%	43.12	41.38%
Std. Dev.			1.82%	67.55	32.59	18.25%	42.16	18.25%

Online Resources 6 Multi-space sampling heuristic: objective function and CPU time at the end of each phase running with T=10,000. Average values over 10 runs for each of the 40 instances by Christiansen and Lysgaard.

Mendoza J. E. and Villegas J. G. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Under review since Sep. 28, 2011.
Corresponding Author: Jorge E. Mendoza, Université Catholique de l'Ouest, Angers (France). Email: jorge.mendoza@uco.fr