European Journal of Operational Research 199 (2009) 98–110

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

# Adaptive sample size and importance sampling in estimation-based local search for the probabilistic traveling salesman problem

Prasanna Balaprakash *, Mauro Birattari, Thomas Stützle, Marco Dorigo

*IRIDIA, CoDE, Université Libre de Bruxelles, 50, Av. F. Roosevelt, CP 194/6, 1050 Brussels, Belgium*

ARTICLE INFO

*Article history:*
Received 3 September 2007
Accepted 4 November 2008
Available online 3 December 2008

*Keywords:*
Combinatorial optimization
Heuristics
Metaheuristics
Iterative improvement algorithm
Probabilistic traveling salesman problem
Importance sampling

ABSTRACT

The probabilistic traveling salesman problem is a paradigmatic example of a stochastic combinatorial optimization problem. For this problem, recently an estimation-based local search algorithm using delta evaluation has been proposed. In this paper, we adopt two well-known variance reduction procedures in the estimation-based local search algorithm: the first is an adaptive sampling procedure that selects the appropriate size of the sample to be used in Monte Carlo evaluation; the second is a procedure that adopts importance sampling to reduce the variance involved in the cost estimation. We investigate several possible strategies for applying these procedures to the given problem and we identify the most effective one. Experimental results show that a particular heuristic customization of the two procedures increases significantly the effectiveness of the estimation-based local search.



## 1. Introduction

The probabilistic traveling salesman problem (PTSP) (Jaillet, 1985) is a paradigmatic example of a stochastic combinatorial optimization problem. The PTSP models a number of practical problems in the areas of strategic planning, routing, transportation, and scheduling (Bertsimas, 1988). The PTSP is similar to the TSP with the difference that each node has a probability of requiring a visit. The *a priori* optimization approach (Jaillet, 1985; Bertsimas et al., 1990) for the PTSP consists in finding an *a priori* solution that visits all the nodes such that the expected cost of *a posteriori* solutions is minimized: the *a priori* solution must be found prior to knowing which nodes are to be visited; the associated *a posteriori* solution, which is computed *after* knowing which nodes need to be visited, is obtained by visiting the nodes that require being visited in the order prescribed by the *a priori* solution, while skipping the nodes that do not require being visited.

Two classes of techniques for tackling the PTSP by *a priori* optimization have been proposed in the literature: analytical computation and empirical estimation. The former exactly computes the expected cost of the *a posteriori* solutions using a complex analytical development. The latter estimates the expected cost through Monte Carlo simulation.

`2.5-opt-EEs` (Birattari et al., 2008) is an estimation-based local search algorithm for tackling the PTSP. It is an iterative improvement algorithm that starts from some initial solution and then iteratively moves to improved neighbor solutions until a local optimum is found. A particularity of `2.5-opt-EEs` is that the cost of the neighbor solutions are estimated using delta evaluation, a technique that considers only the cost contribution of solution components that are not common between two neighbor solutions. The results from Birattari et al. (2008) show that the performance of `2.5-opt-EEs` depends on the probability associated with the nodes of the given PTSP instance. In particular, for low probabilities, where the coefficient of variation of the PTSP solution cost is high, `2.5-opt-EEs` is less effective.

The goal of this paper is to increase the effectiveness of `2.5-opt-EEs` for PTSP instances with low probabilities by using two procedures that reduce the variance of the cost estimator. The first is an adaptive sampling procedure that selects the appropriate size of the sample with respect to the variance of the cost estimator; the second is a procedure that adopts the importance sampling technique in order to reduce the variance of the cost estimator.

There exists a number of prior publications where adaptive sampling and importance sampling have been studied in the context of stochastic combinatorial optimization. Alkhamis et al. (1999), Gutjahr (2004), Homem-de-Mello (2003), Pichitlamken and Nelson (2003), and Birattari et al. (2006) investigated adaptive sample size procedures that make use of statistical tests to determine the number of samples to be chosen. The adoption of importance sampling to reduce the variance of the cost estimator has been investigated in Gutjahr et al. (2000a) and Gutjahr et al. (2000b). In all these works, the adaptive sample size and the importance sampling techniques have been used in the context

* Corresponding author. Tel.: +32 26503168; fax: +32 26502715.
*E-mail addresses:* pbalapra@ulb.ac.be (P. Balaprakash), mbiro@ulb.ac.be (M. Birattari), stuetzle@ulb.ac.be (T. Stützle), mdorigo@ulb.ac.be (M. Dorigo).

0377-2217/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2008.11.027

of full evaluation, where the cost of each solution is estimated from scratch. This is mainly due to the fact that the usage of delta evaluation is either ignored or not feasible for the given stochastic combinatorial optimization problem. The adoption of adaptive sample size and of importance sampling procedures in delta evaluation has never been investigated. For the PTSP, where delta evaluation is feasible, we expect that the adoption of the two particular techniques will increase the effectiveness of `2.5-opt-EEs`. However, as we show in this paper, the adoption is not trivial and a main contribution of the paper consists in customizing the adaptive sample size and the importance sampling procedures for the delta evaluation applied to the PTSP. In particular, we investigate several ways of applying these procedures in the PTSP delta evaluation and we use a design of experiments approach to identify the most effective one.

The paper is organized as follows: in Section 2, we introduce the proposed approach; in Section 3, we study its performance; and in Section 4, we conclude the paper.

## 2. An estimation-based iterative improvement algorithm for the PTSP

In order to make this section self-contained, we first give a formal description of the PTSP and then we sketch the `2.5-opt-EEs` algorithm; finally, we describe the procedures introduced in this paper.

### 2.1. The probabilistic traveling salesman problem

A PTSP instance is defined through a graph $G = (V, A, C, P)$, with $V = \{1, 2, \ldots, n\}$ being a set of nodes, $A = \{\langle i,j\rangle : i, j \in V, i \neq j\}$ being a set of edges that completely connects the nodes, $C = \{c_{ij} : \langle i,j\rangle \in A\}$ being the set of edge costs, and $P = \{p_i : i \in V\}$ being a set of probabilities where $p_i$ specifies the probability that a node $i$ requires being visited. The events that two distinct nodes $i$ and $j$ require being visited are independent. The stochastic information can be modeled using a random variable $\boldsymbol{\omega}$, which follows an $n$-variate Bernoulli distribution. A realization of $\boldsymbol{\omega}$ is a vector of size $n$ composed of '`1`'s and '`0`'s: the value '`1`' in position $i$ indicates that node $i$ requires being visited, whereas the value '`0`' indicates that it does not. The costs are assumed to be symmetric, that is, for all pairs of nodes $i, j$ we have $c_{ij} = c_{ji}$. A solution to the PTSP is a permutation of the nodes.

The most widely used approach to tackle the PTSP is *a priori* optimization (Jaillet, 1985; Bertsimas et al., 1990). This approach consists of two stages. First, an *a priori* solution—a permutation of the nodes—is determined before the realization of $\boldsymbol{\omega}$ is known. Once the realization is known in the second stage, an *a posteriori* solution is derived from the *a priori* solution by visiting the nodes of the realization in the same order as prescribed by the *a priori* solution and by skipping the nodes that do not require being visited. Fig. 1 shows an example.

The goal in the PTSP is to find an *a priori* solution that produces the minimum expected *a posteriori* solution cost.

On the basis of probability values associated with the nodes, PTSP instances can be classified as follows. If $P = \{p_i = p : i \in V\}$, the PTSP instance is said to be homogeneous, otherwise, if for at least two nodes $i$ and $j$ we have $p_i \neq p_j$, it is said to be heterogeneous.

The usage of effective delta evaluation procedures is of crucial importance for a fast local search for the PTSP. So far, the state-of-the-art iterative improvement algorithms for the PTSP, `2-p-opt` and `1-shift`, have used for the delta evaluation recursive closed-form expressions based on heavy mathematical derivations (Bertsimas, 1988; Chervi, 1988; Bianchi et al., 2005; Bianchi, 2006; Bianchi and Campbell, 2007). Recently, we introduced a more effective algorithm called `2.5-opt-ACs` (Birattari et al., 2008). This algorithm also uses closed-form expressions but adopts the classical TSP neighborhood reduction techniques; this is not possible in `2-p-opt` and `1-shift` since these algorithms require to search the neighborhood in a fixed lexicographic order. `2.5-opt-EEs` (Birattari et al., 2008), the algorithm on which we focus in this paper, makes use of empirical estimation techniques and of the classical TSP neighborhood reduction techniques. The experimental results have shown that this algorithm is much faster than `2.5-opt-ACs`. However, for instances where the probability of visiting nodes is very low, the cost of the solutions obtained by `2.5-opt-EEs` is significantly worse than that of `2.5-opt-ACs`. The methodology that we propose in this paper addresses this issue.

### 2.2. The `2.5-opt-EEs` algorithm

In iterative improvement algorithms for the PTSP, we need to compare two neighbor solutions $x$ and $x'$ to select the one of lower cost. An unbiased estimator of the cost $F(x)$ of a solution $x$ can be computed on the basis of a sample of costs of *a posteriori* solutions obtained from $M$ independent realizations of the random variable $\boldsymbol{\omega}$. Using the method of common random numbers, an unbiased estimator of $F(x')$ can be estimated analogously to $F(x)$ using the same set of $M$ independent realizations of $\boldsymbol{\omega}$. The estimator $\widehat{F}_M(x') - \widehat{F}_M(x)$ of the cost difference is given by

$$\widehat{F}_M(x') - \widehat{F}_M(x) = \frac{1}{M}\sum_{r=1}^{M}(f(x', \omega_r) - f(x, \omega_r)). \quad (1)$$

We implemented iterative improvement algorithms that use this way of estimating cost differences exploiting a neighborhood structure that consists of a node-insertion neighborhood on top of a two-exchange neighborhood structure, that is, the well-known

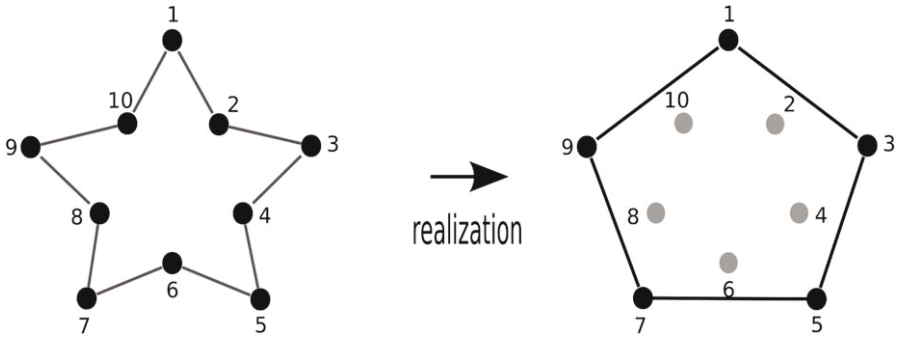


**Fig. 1.** This figure shows an a priori solution for a PTSP instance with 10 nodes. The order in which the nodes are visited in the *a priori* solution is: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1. Let us assume that the nodes 1, 3, 5, 7, and 9 are prescribed to be visited by a realization of $\boldsymbol{\omega}$. The *a posteriori* solution visits the nodes following the *a priori* solution but skipping the nodes 2, 4, 6, 8, and 10.

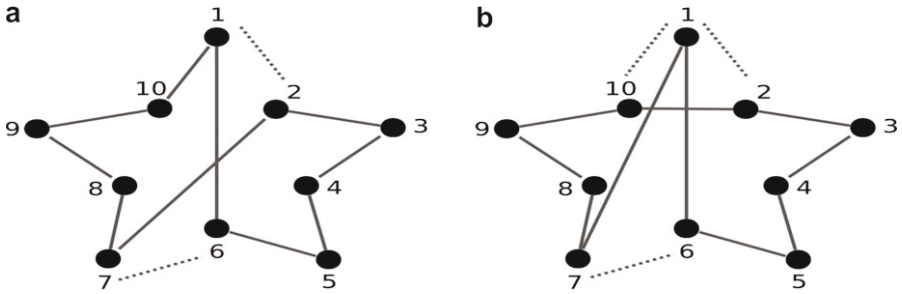


**Fig. 2.** Figure (a) shows a two-exchange move that is obtained by deleting the two edges $\langle 1,2\rangle$ and $\langle 6,7\rangle$ of the solution and by replacing them with $\langle 1,6\rangle$ and $\langle 2,7\rangle$. Figure (b) shows a node-insertion move obtained by deleting node 1 from its current position in the solution and inserting it between nodes 6 and 7.

2.5-exchange neighborhood: when checking for a two-exchange move on any two edges $\langle a,b\rangle$ and $\langle c,d\rangle$, it is also checked whether deleting any one of the nodes of an edge, say for example $a$, and inserting it between nodes $c$ and $d$ results in an improved solution (Bentley, 1992)—see Fig. 2. Given two neighboring *a priori* solutions and a realization $\omega$, the algorithm needs to identify the edges that are not common to the two *a posteriori* solutions, that is, for every edge $\langle i,j\rangle$ that is deleted from $x$, one needs to find the corresponding edge $\langle i^*,j^*\rangle$ that is deleted in the *a posteriori* solution of $x$. This edge is called the *a posteriori edge*. It can efficiently be found as follows. If node $i$ requires being visited, then $i^* = i$; otherwise, $i^*$ is the first predecessor of $i$ in $x$ such that $\omega[i^*] = 1$, that is, the first predecessor for which the realization is one, indicating it requires being visited. If node $j$ requires being visited, then $j^* = j$; otherwise, $j^*$ is the first successor of $j$ such that $\omega[j^*] = 1$. Recall that in a two-exchange move, the edges $\langle a,b\rangle$ and $\langle c,d\rangle$ are deleted from $x$ and replaced by $\langle a,c\rangle$ and $\langle b,d\rangle$. For a given realization $\omega$ and the corresponding *a posteriori edges* $\langle a^*,b^*\rangle$ and $\langle c^*,d^*\rangle$, the cost difference between the two *a posteriori* solutions is given by $c_{a^*,c^*} + c_{b^*,d^*} - c_{a^*,b^*} - c_{c^*,d^*}$. This procedure can be directly extended to node-insertion moves. Furthermore, the algorithm searches the neighborhood using a first-improvement rule and it also exploits the following neighborhood reduction techniques: fixed-radius search, candidate lists, and don't look bits (Bentley, 1992; Johnson and McGeoch, 1997). This algorithm is called `2.5-opt-EEs`. For a more detailed explanation of `2.5-opt-EEs`, we refer the reader to Birattari et al. (2008).

### 2.3. Advanced sampling methods

In this section, we focus on the main contribution of the paper, that is, the customization of the two procedures for the delta evaluation of the PTSP in order to increase the effectiveness of `2.5-opt-EEs`.

#### 2.3.1. Adaptive sampling

For PTSP instances with low probability values, the estimator of the cost of solutions has a very high coefficient of variation. In this case, averaging over a large number of realizations improves the estimation. However, using a large number of realizations for high probability values results in a waste of computation time. To address this issue, we adopt an adaptive sampling procedure that saves computation time by selecting the most appropriate number of realizations for each estimation. This procedure is realized using Student's $t$-test in the following way: given two neighboring *a priori* solutions, the cost difference between their corresponding *a posteriori* solutions is sequentially computed on a number of realizations. As soon as the $t$-test rejects the null hypothesis that the value of the estimated cost difference is equal to zero, the computation is stopped. If no statistical evidence is gathered, then the computation is continued until a maximum number $M$ of realizations is considered, where $M$ is a parameter of the procedure. The sign of the estimated difference determines the solution of lower cost. Note that the significance level of Student's $t$-test is also a parameter of the procedure.

The estimation-based iterative improvement algorithm that adds the adaptive sampling procedure to `2.5-opt-EEs` will be called `2.5-opt-EEas`.

#### 2.3.2. Importance sampling

A difficulty in the adoption of the $t$-test is that for low probability values often the test statistic cannot be computed: since the nodes involved in the cost difference computation may not require being visited in the realizations considered, all cost differences between two *a posteriori* solutions are zero; therefore, the sample mean and the sample variance of the cost difference estimator are zero. Some degenerate cases in which this problem appears are the following. In a two-exchange move that deletes the generic edges $\langle a,b\rangle$ and $\langle c,d\rangle$, and where no node between the nodes $b$ and $c$ (or between $a$ and $d$) requires being visited, the difference between the two *a posteriori* solutions is zero—see Fig. 3a for an illustration. In particular, this case is very frequent when the number of nodes in the tour segment between $b$ and $c$ (or between the tour segment $a$ and $d$) is small. In a node-insertion move, if the insertion node does not require being visited, the cost difference between the two *a posteriori* solutions is zero—see Fig. 3b. A naïve strategy to handle this problem consists in postponing the $t$-test until non-zero sample mean and sample variance are obtained. However, this might increase the number of realizations needed for the cost difference computation. The key idea to address this issue consists in forcing the nodes involved in the cost difference computation to appear frequently in the realizations. More in general, we need to reduce the variance of the cost difference estimator for low probability values. For this purpose, we use the variance reduction technique known as importance sampling (Rubinstein, 1981).

In order to compute the cost difference between two *a posteriori* solutions, importance sampling, instead of using realizations of the given variable $\boldsymbol{\omega}$ parameterized by $P$, considers realizations of another variable $\boldsymbol{\omega}^*$ parameterized by $P^*$; this so-called biased distribution $P^*$ biases the nodes involved in the cost difference computation to occur more frequently. This is achieved by choosing probabilities in $P^*$ larger than the probabilities in $P$. The resulting biased cost difference between two *a posteriori* solutions for the $r$th biased realization $\omega_r^*$ is then corrected for the adoption of the biased distribution: the correction is given by the likelihood ratio $LR_r$ of the original distribution with respect to the biased distribution and it is obtained from the following equation:

$$LR_r = \prod_{i=1}^{n} \frac{(p_i)^{\omega_r^*[i]} \cdot (1-p_i)^{1-\omega_r^*[i]}}{(p_i^*)^{\omega_r^*[i]} \cdot (1-p_i^*)^{1-\omega_r^*[i]}}, \tag{2}$$

where $p_i$ and $p_i^*$ are the original and biased probabilities of a node $i$, respectively. Finally, the unbiased cost difference is obtained as follows:

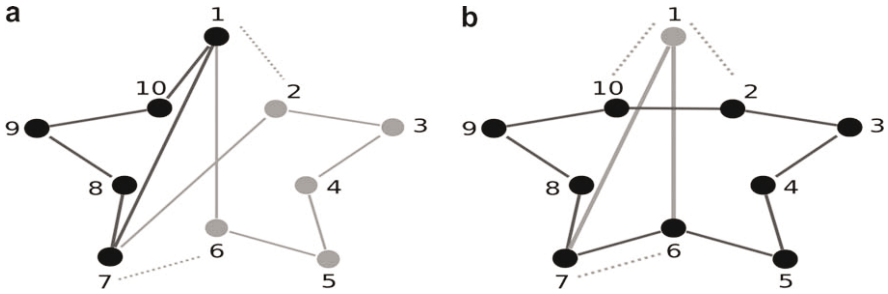


**Fig. 3.** Some degenerate cases that can occur in the evaluation of cost differences. (a) Assume that a realization of $\omega$ prescribes that nodes 1, 7, 8, 9, and 10 are to be visited. In this example, the generic edges $\langle a,b\rangle$ and $\langle c,d\rangle$ considered for the two-exchange move correspond to $\langle 1,2\rangle$ and $\langle 6,7\rangle$. The two-exchange neighbor solutions shown in Fig. 2a lead to the same *a posteriori* solution. The cost difference is therefore zero. (b) Assume that a realization of $\omega$ prescribes that nodes 2, 3, 4, 5, 6, 7, 8, 9, and 10 are to be visited. The node-insertion neighbor solutions shown in Fig. 2b lead to the same *a posteriori* solution. Since the two *a posteriori* solutions are the same, the cost difference is zero.

$$\widehat{F}_M(x') - \widehat{F}_M(x) = \frac{1}{M}\sum_{r=1}^{M} LR_r \cdot (f(x', \omega_r^*) - f(x, \omega_r^*)). \tag{3}$$

The main contribution of this paper consists in adopting importance sampling for the delta evaluation. Recall that in the delta evaluation, given a deleted edge $\langle i,j\rangle$, the algorithm needs to identify the corresponding *a posteriori edge* $\langle i^*,j^*\rangle$. In order to apply importance sampling in the delta evaluation, only the nodes that are involved in finding the *a posteriori edge*, $\langle i^*,j^*\rangle$ are biased. Now, we discuss several ways of realizing this idea.

**Uniform biasing.** This is a simple variant in which all the nodes involved in finding the *a posteriori edge* are biased with a probability $p'$, where $p'$ is a parameter. This variant does not take into account any problem-specific knowledge.

**Geometric biasing.** In this variant, the nodes involved in finding the *a posteriori edge* are biased with probabilities according to a geometric schedule, $\lambda^h \cdot p'$: the node $i$ is biased with probability $p'$, its $h$th predecessor takes the biased probability value of $\lambda^h \cdot p'$. Similarly, the node $j$ is biased with probability $p'$, and its $h$th successor takes the biased probability value of $\lambda^h \cdot p'$. Note that $p'$ and $0 < \lambda < 1$ are parameters of this variant and, similar to the previous variant, this variant does not use any problem-specific knowledge.

**Strong greedy biasing.** This variant uses $k$-exchange specific knowledge to bias the nodes. In the case of a two-exchange move, the cost difference computation involves four distinct nodes. If these four nodes require being visited in all the realizations, then there is no need for finding the predecessor and the successor of the starting nodes $i$ and $j$, respectively. This variant is designed for biasing only those four nodes and their biased probability values are set to a same value $p'$. In the same way, for a node-insertion move only the five nodes that are involved in the cost difference computation are biased to appear in a given realization. Out of the five nodes, the biased probability of the insertion node is set to $p''$ and the biased probability values of the other four nodes are set to a same value $p'$. The reason for choosing a different value for the insertion node is that the appearance of the insertion node is more crucial than that of the other nodes, as illustrated in Fig. 3b. Note that $p'$ and $p''$ are parameters of this variant.

**Weak greedy biasing.** This variant differs from the strong greedy biasing with respect to the biasing scheme in the node-insertion move: the insertion node is biased with a value $p''$ and the other four nodes are not biased. This variant is designed for the following purpose: by comparing this variant with the previous one, we can study the necessity for biasing the four nodes with $p'$ in the node-insertion move.

**Heuristic biasing.** This variant is similar to the weak greedy biasing except the fact that in the two-exchange move the nodes involved in finding the *a posteriori edge* are biased. As illustrated in Fig. 3, in a two-exchange move, the nodes in the shorter tour segment (either between $b$ and $c$ or between $a$ and $d$) are more important than other nodes for the cost difference computation. Therefore, the nodes in the shorter segment are biased with a probability $p'$, if the number of nodes in the shorter segment is less than $min_{is}$% of $n$, the number of nodes in the PTSP instance; $min_{is}$ is a parameter of the procedure.

For PTSP instances with very low probability values, the computational results of the state-of-the-art iterative improvement algorithms show that the two-exchange neighborhood relation is not very effective (Bianchi, 2006; Birattari et al., 2008; Birattari et al., 2007): often the improvements obtained with a two-exchange move are rather small and therefore the time needed to reach a local optimum is high. The estimation-based local search might suffer from the aforementioned problem when all the nodes in the shorter segment are biased. In order to address this issue, the importance sampling in the two-exchange move is used only occasionally: instead of biasing all the nodes in the shorter segment, only a certain number of nodes, determined by another parameter $u$, are biased in the shorter segment. This parameter is used in the following way: let us assume that $[b, b+1, b+2, \ldots, c+2, c+1, c]$ is the shorter segment; let us denote the number of nodes in this segment as *seg*; this variant biases $u$% of *seg* nodes, on each side of this segment. To give a concrete example, let us assume that *seg* is 50. If $u$ is set to 2, then the nodes that are biased are $b$ and $c$ (2% of 50, that is, 1 node on each side of the segment is biased); if $u$ is set to 4, then the nodes that are biased are $b$, $b+1$ and $c$, $c+1$ (4% of 50, that is, 2 nodes on each side of the segment are biased). In the same way, the nodes are biased if $[a, a+1, a+2, \ldots, d+2, d+1, d]$ is the shorter segment. The usage of $min_{is}$ and $u$ is illustrated in Fig. 4. For the node-insertion move, only the insertion node is biased with a value $p''$. The parameters of this variant are $p'$, $p''$, $min_{is}$, and $u$.

### 2.3.3. General remarks on the importance sampling variants

The computation of the cost difference between two *a posteriori* solutions using any of the importance sampling variants proceeds as follows: given a deleted edge $\langle i,j\rangle$, the nodes are biased with probabilities according to a selected variant. A biased realization is sampled with respect to the biased probabilities, on which the *a posteriori edge* $\langle i^*,j^*\rangle$ and the biased cost difference between two *a posteriori* solutions are obtained. The overall likelihood ratio for the biased realization is obtained by the product of the likelihood ratio of each biased node $l$, which is used in finding the *a posteriori edge* $\langle i^*,j^*\rangle$. Note that the likelihood ratio of each biased node $l$ is given by

$$LR_r^l = \frac{(p_l)^{\omega_r^*[l]} \cdot (1-p_l)^{1-\omega_r^*[l]}}{(p_l^*)^{\omega_r^*[l]} \cdot (1-p_l^*)^{1-\omega_r^*[l]}}, \tag{4}$$

where $p_l$ and $p_l^*$ are the original and the biased probability of node $l$ and $\omega_r^*[l]$ is sampled with the biased probability $p_l^*$. The unbiased

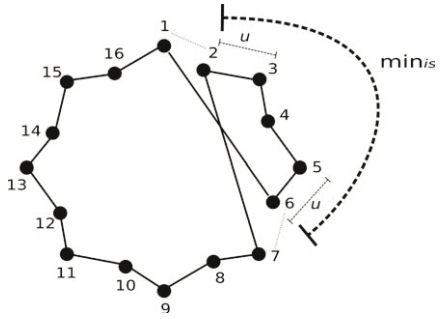


**Fig. 4.** This figure illustrates heuristic biasing. In this example, a two-exchange move is obtained by deleting the two edges $\langle 1,2\rangle$ and $\langle 6,7\rangle$ and by replacing them with $\langle 1,6\rangle$ and $\langle 2,7\rangle$. Assume that the parameter $min_{is}$ is set to 50. Since the number of nodes in the segment $[2,\ldots,6]$ is less than 50% of 16, that is eight, importance sampling is used to bias the nodes. However, instead of biasing all the nodes between 2 and 6, only a certain number of nodes that are close to them are biased. We assume $u$ to be 40; therefore 40% of 5, that is, two nodes are biased on each side of the segment. The nodes that are biased are 2, 3, 5, and 6.

cost difference between two *a posteriori* solutions is simply given by the product of the overall likelihood ratio and the biased cost difference between two *a posteriori* solutions.

The values of the biased probabilities of the aforementioned importance sampling variants are crucial for the variance reduction. In particular, if those values are inappropriate, then the adoption of importance sampling variants will increase the variance of the cost estimator. We address this issue using a parameter tuning algorithm in Section 3.

We denote `2.5-opt-EEais` the algorithm that adds to `2.5-opt-EEas` any of the described importance sampling variants.

### 2.4. Implementation-specific details

In order to implement `2.5-opt-EEais` efficiently, we use the same data structure as that of `2.5-opt-EEs`, which is composed of a doubly circularly linked list and some auxiliary arrays as described in Birattari et al. (2008). In `2.5-opt-EEais` with strong greedy, weak greedy and heuristic biasing, for each node three realization arrays, $\omega$, $\omega'$, and $\omega''$ are stored, each of size $M$, indexed from 1 to $M$. Element $r$ of a realization array is either `1` or `0` indicating whether node $i$ requires being visited or not in a realization and it is obtained as follows: first a random number between 0 and 1 is generated; if this number is less than or equal to $p_i$, $p'_i$, or $p''_i$, node $i$ requires being visited in realization $\omega_r$, $\omega'_r$, or $\omega''_r$, respectively. In `2.5-opt-EEais` with uniform biasing, each node has two realization arrays, $\omega$, $\omega'$, each of size $M$. In `2.5-opt-EEais` with geometric biasing, given $\lambda$ and $p'$, it is possible to compute the set of all possible biased probability values that a node can take. Therefore, each node has a set of biased realizations, where each of them is sampled with respect to a possible biased probability. In all the variants, when the biased probability value of a node is less than the original probability, the former is set to the latter. Given $p_i$ and the set of all biased probability values of a node $i$, the likelihood ratio is pre-computed and stored when the algorithm starts.

`2.5-opt-EEais` uses a same set of realizations for all iterative improvement steps. In the context of the PTSP, this strategy is more effective than changing realizations for each improvement or for each comparison (Birattari et al., 2008). However, for each two-exchange and node-insertion move, the realizations are selected randomly from this set until the $t$-test rejects the null hypothesis.

The following techniques are used to speed up the computations involved in the $t$-test: the critical values of Student's $t$-distribution are pre-computed and stored in a lookup table; the sample mean and the sample variance of the cost difference estimator are computed recursively.

The implementation of `2.5-opt-EEas` is identical to `2.5-opt-EEais` except for the fact that the importance sampling procedure and the pre-computations required for `2.5-opt-EEais` are excluded.

## 3. Experimental analysis

In this section, we present the experimental setting considered and the empirical results. Our goal is to show that the integration of the adaptive sample size and the importance sampling procedures into the estimation-based local search increases significantly its effectiveness.

### 3.1. Experimental setup

We generate TSP instances with the DIMACS instance generator (Johnson et al., 2001) from which the PTSP instances are obtained by associating a probability value to each node. We use uniform and clustered instances of 1000 nodes: in the former, the nodes are distributed uniformly and in the latter the nodes are arranged in a number of clusters, both in a $10^6 \times 10^6$ square. We use two classes of instances: homogeneous and heterogeneous PTSP instances. For the homogeneous instances, we consider probability values starting from 0.050 to 0.200 with an increment of 0.025 and from 0.3 to 0.9 with an increment of 0.1. The probability values in the heterogeneous instances are generated using a beta distribution as described in Bianchi (2006): each instance is characterized by two parameter values: mean probability $p_m$ and a percentage of maximum variance $per_{m_v}$. The two parameter values have the following meaning: if an instance is generated with mean probability $p_m$ of 0.05 and percentage of maximum variance $per_{m_v}$ of 50, then the mean of the probability values is approximately equal to 0.05 and the variance of the probability values is approximately equal to $50\% \cdot 0.05(1-0.05)$. For the sake of convenience, we denote a heterogeneous instance with $p = 0.100(50)$ when it has the mean probability of 0.100 and percentage of maximum variance of 50. We consider the values for $p_m$ from 0.050 to 0.200 with an increment of 0.025 and from 0.3 to 0.5 with an increment of 0.1; for each value of $p_m$, we consider three values for $per_{m_v}$ in $\{16,50,83\}$. We generate 50 instances for each combination of $p_m$ and $per_{m_v}$.

For PTSP instances of size 1000, the algorithms `2.5-opt-ACs`, `2-p-opt`, and `1-shift` suffer from numerical problems such as overflow and underflow for high and low probability values (Birattari et al., 2008). Therefore, these algorithms use MPFR (Fousse et al., 2007), a state-of-the-art library for arbitrary precision arithmetics, to overcome the numerical problems. Note that algorithms based on empirical estimation do not suffer from this problem. Due to space limitations, we highlight only the results obtained for probability values up to 0.200. The trends of the results obtained for the higher probability values are very similar; we refer the reader to Balaprakash et al. (2007b) and Balaprakash et al. (2008) for the complete set of results.

All algorithms are implemented in `C` and the source code is compiled with `gcc`, version 3.3. Experiments are carried out on AMD Opteron™244 1.75 GHz processors with 1 MB L2-Cache and 2 GB RAM, running under Rocks Cluster GNU/Linux.

The nearest-neighbor heuristic is used to generate initial solutions. The candidate list is set to size 40 and is constructed with the quadrant nearest-neighbor strategy (Penky and Miller, 1994; Johnson and McGeoch, 1997). Each iterative improvement algorithm is run until it reaches a local optimum.

In `2.5-opt-EEas` and `2.5-opt-EEais`, the minimum number of realizations used in the adaptive sampling procedure before applying the $t$-test is set to five. The null hypothesis is

rejected at a significance level of 0.05. If the test statistic cannot be computed after five realizations, then the cost difference computation is stopped and the algorithm considers the next neighbor solution. The maximum number $M$ of realizations is set to one thousand.

For a PTSP instance of size $n$, given an *a priori* solution $x$, the exact cost $F(x)$ of $x$ can be computed using a closed-form expression given in Jaillet (1985) and Bertsimas (1988). We use this formula for the post-evaluation of the best-so-far solutions found by each algorithm according to its evaluation procedure.

In addition to tables, we visualize the results using runtime development plots. These plots show how the cost of solutions develops over computation time. We report one such plot for each probability level under consideration and the results are averaged over 50 instances.

For the homogeneous PTSP with $p \geqslant 0.1$, `2.5-opt-ACs` has already been shown to be more effective than `1-shift` and `2-p-opt` (Birattari et al., 2008; Birattari et al., 2007). We carried out some preliminary experiments to verify that the same tendency holds also for low probability values, that is, for $p < 0.1$. In these experiments, `2.5-opt-ACs` outperforms both `1-shift` and `2-p-opt` and, therefore, we take `2.5-opt-ACs` as a yardstick for measuring the effectiveness of the proposed algorithms. The computational results for $p < 0.1$ are given in Balaprakash et al. (2007b).

### 3.2. Parameter tuning

Finding appropriate values for the parameters—in particular the biased probability values—of the importance sampling variants adopted in `2.5-opt-EEais` is crucial for the variance reduction. For this purpose, we use a parameter tuning algorithm, `Iterative F-Race` (Balaprakash et al., 2007a), to identify suitable values for the parameters of each importance sampling variant. We tune each importance sampling variant separately for the homogeneous and the heterogeneous clustered instances of size 1000. We use 210 instances (7 levels of probability $\times$ 30 instances) for the homogeneous case and 210 instances (7 levels of probability $\times$ 3 levels of percentage of maximum variance $\times$ 10 instances) for the heterogeneous case. Table 1 shows, for each importance sampling variant, the range of each parameter given to the tuning algorithm and the selected value.

From the parameter values of strong greedy biasing, weak greedy biasing, and heuristic biasing, we can observe the following trend: low biased probability values are appropriate for the nodes involved in the two-exchange moves, whereas high biased probability values are selected for the nodes involved in the node-insertion moves. In particular, under our experimental setting, the appropriate ranges for the biased probability values, $p'$, in two-exchange moves for homogeneous and heterogeneous PTSP instances are [0.11,0.2] and [0.07,0.12], respectively. This low range of values is due to the ineffectiveness of the two-exchange neighborhood relation for low probability values as discussed in Section 2.4. We also made some tests in which importance sampling is completely disabled for two-exchange moves. The results show that indeed the usage of importance sampling in two-exchange moves results in solution costs that were slightly better than the ones in which importance sampling was completely disabled. The biased probability values for the nodes involved in the node-insertion moves are relatively high. In the current experimental setting, the appropriate range for the biased probability values, $p''$, are [0.57,0.79] for both homogeneous and heterogeneous PTSP instances. For what concerns uniform and geometric biasing, which do not use a separate biased probability value for two-exchange and node-insertion moves, the tuning algorithm tries to find a good biased probability that is suitable for both types of moves. Eventually, this results in values that are higher than those for the two-exchange moves and lower than those for the node-insertion moves in the greedy and heuristic biasing.

**Table 1**
Parameter values considered for tuning the importance sampling variants in `2.5-opt-EEais` and the values selected by `Iterative/F-Race`.

| Variant | Parameter | Range | Selected value | |
|---|---|---|---|---|
| | | | Homogeneous | Heterogeneous |
| Uniform biasing | $p'$ | [0.0, 1.0] | 0.23 | 0.08 |
| Geometric biasing | $p'$ | [0.0, 1.0] | 0.39 | 0.18 |
| | $h$ | [0.0, 1.0] | 0.25 | 0.12 |
| Strong greedy biasing | $p'$ | [0.0, 1.0] | 0.20 | 0.12 |
| | $p''$ | [0.0, 1.0] | 0.76 | 0.60 |
| Weak greedy biasing | $p'$ | [0.0, 1.0] | 0.24 | 0.08 |
| | $p''$ | [0.0, 1.0] | 0.79 | 0.64 |
| Heuristic biasing | $p'$ | [0.0, 1.0] | 0.11 | 0.07 |
| | $p''$ | [0.0, 1.0] | 0.60 | 0.57 |
| | $min_{is}$ | [0.0, 20.0] | 0.55 | 1.30 |
| | $u$ | [0, 100] | 72.00 | 10.00 |

### 3.3. A study on the parameters of *2.5-opt-EEais*

In this section, we study the impact of the parameters of `2.5-opt-EEais` on solution quality and computation time. The main aim of this analysis is to identify an appropriate significance level for the adaptive sample size procedure and to determine the most promising importance sampling variant. Moreover, we also study the robustness of `2.5-opt-EEais` with respect to instance size and the way in which the nodes are distributed in the instances. For this purpose, we use the analysis of variance (ANOVA) technique. In ANOVA terminology, the parameters of `2.5-opt-EEais` are called factors and the solution quality and computation time are called response variables. In this analysis, we use `2.5-opt-EEs-l000` (`2.5-opt-EEs` that uses 1000 realizations without adaptive sample size and importance sampling) as a reference algorithm: we study the solution quality of a given algorithm as the percentage deviation from the cost of `2.5-opt-EEs-l000`. The computation time of a given algorithm is normalized with respect to the computation time of `2.5-opt-EEs-l000`. We perform an ANOVA analysis for each of the response variables. In order to apply ANOVA, it is necessary to check three main assumptions on the distribution of the response variables, namely, normality, homogeneity of variance, and independence of residuals. Since grouping over all probability levels results in violation of the assumptions, the ANOVA analysis is done for each level. Even in this setting, there are a few probability levels for which the ANOVA assumptions are violated. In such cases, we use the non-parametric Wilcoxon rank sum test to verify the results of the ANOVA analysis. For the complete ANOVA results, we refer the reader to Balaprakash et al. (2008); here, we highlight some main results of the analysis.

#### 3.3.1. Importance sampling variants

In this analysis, we study the effect of importance sampling variants in `2.5-opt-EEais` on the solution quality and computation time. The results are shown as box plots in Fig. 5. Note that the significance level in the adaptive sample size procedure is set to 0.05. For what concerns the homogeneous instances, the $F$-ratio and the $p$-values from the ANOVA table show that importance sampling has a significant impact on the solution quality for probability levels less than 0.150. The $p$-values from the pairwise $t$-test indicate that strong greedy, weak greedy and heuristic biasing are significantly better than the uniform and the geometric biasing.

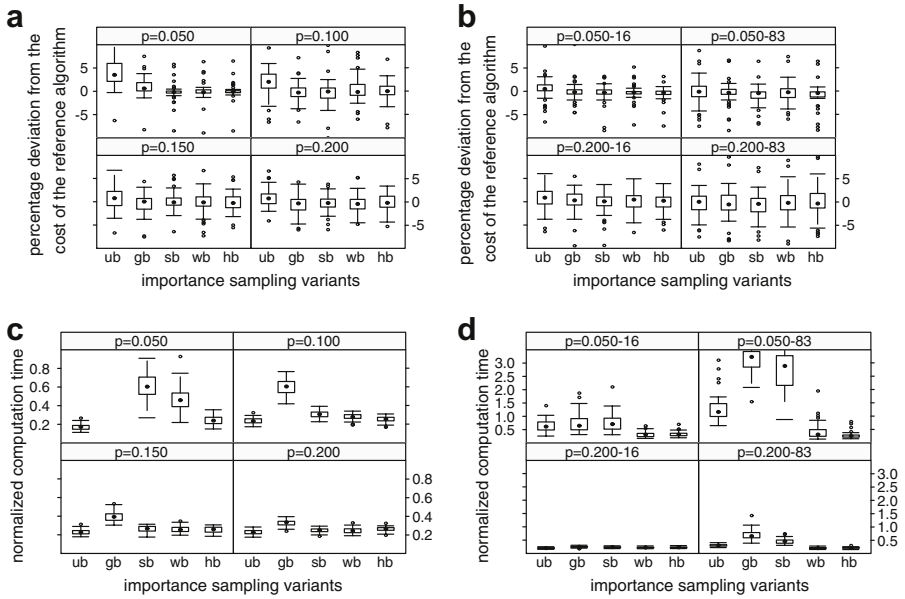


**Fig. 5.** Analysis of different importance sampling variants on clustered PTSP instances of size 1000. Plots (a) and (b) show the cost of the solutions of `2.5-opt-EEais` with uniform biasing (ub), geometric biasing (gb), strong greedy biasing (sb), weak greedy (wb), and heuristic biasing (hb) as the percentage deviation from the cost of `2.5-opt-EEs-1000` on homogeneous and heterogeneous PTSP instances. Plots (c) and (d) show the normalized computation time of `2.5-opt-EEais` with different importance sampling variants, where the normalization is done with respect to the computation time of `2.5-opt-EEs-1000` for homogeneous and heterogeneous instances, respectively.

However, there is no significant difference among strong greedy, weak greedy and heuristic biasing. For what concerns the heterogeneous instances, the *F*-ratio and the *p*-values from the ANOVA table show that there is no significant difference among the different importance sampling variants. Nevertheless, the heuristic biasing obtains local optima whose average is slightly better than that of the other variants.

Concerning the computation time, the *F*-ratio and the *p*-values from the ANOVA table show the following general trend for the probability values less than 0.150: the heuristic biasing and the week greedy biasing are significantly faster than other variants for both homogeneous and heterogeneous instances. Also note that, although there is no significant difference between heuristic biasing and week greedy biasing, the computation time of the former is slightly lower than that of the latter.

Taking into account both solution quality and computation time, we select the heuristic biasing as the most promising variant and we use it as a basis for the following analyses.

### 3.3.2. Significance level

In this analysis, we study the effect of the significance level used in the adaptive sample size procedure on the solution quality and on the computation time. We use `2.5-opt-EEais` with heuristic biasing for the experimental analysis. We consider four significance levels: [0.01, 0.02, 0.05, 0.10]. The *F*-ratio and the *p*-values from the ANOVA table show that the significance level does not have a significant impact on the solution quality; however, it has a significant impact on the computation time. The results on the computation time are shown as box plots in Fig. 6: at significance levels 0.01 and 0.02, the algorithm needs more realizations, thus more computation time, to reject the null hypothesis at each step than for significance level 0.05. Nevertheless, the computation time of `2.5-opt-EEais` at significance level 0.10 is higher than for other significance levels for low probability values. This can be attributed to the fact that the estimates of the cost differences are less precise for low probability values and as a consequence the algorithm with significance level 0.10 incorrectly moves to a number of non-improving neighbor solutions before reaching a local optimum. However, for high probability levels, where the variance of the cost estimate is low, the computation time of `2.5-opt-EEais` at significance level 0.10 is lower than for other significance levels. Taking into account both low and high probability values, we can see that the significance level 0.05 is appropriate for `2.5-opt-EEais`.

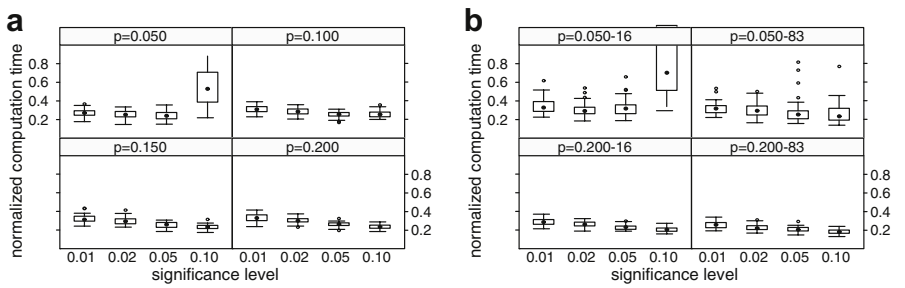


**Fig. 6.** Analysis of different significance levels on clustered PTSP instances of size 1000. Plots (a) and (b) show the normalized computation time of `2.5-opt-EEais` with different significance levels, where the normalization is done with respect to the computation time of `2.5-opt-EEs-1000` for homogeneous and heterogeneous instances, respectively.

**Table 2**
Experimental results on variance reduction by heuristic biasing on clustered instances of size 1000. The algorithm is allowed to explore 1000 solutions. The table gives, for each probability level, the average of the variances computed for 1000 delta estimations with and without importance sampling. The last column shows the percentage reduction of the cost estimator variance when using heuristic biasing (hb).

| $p_m$ | $per_{m_v}$ | Sample size | Average of 1000 cost estimator variances | | Reduction in % |
|---|---|---|---|---|---|
| | | | Without hb | With hb | |
| *Node-insertion move* | | | | | |
| 0.050 | 00 | 10 | 5.24e+08 | 1.09e+07 | 97.92 |
| | | 100 | 9.51e+06 | 4.48e+05 | 95.29 |
| | | 1000 | 6.94e+05 | 3.82e+04 | 94.50 |
| | 16 | 10 | 5.98e+08 | 6.71e+07 | 88.78 |
| | | 100 | 2.38e+07 | 3.54e+06 | 85.13 |
| | | 1000 | 3.86e+05 | 7.23e+04 | 81.25 |
| | 50 | 10 | 2.29e+08 | 7.27e+07 | 68.22 |
| | | 100 | 2.89e+07 | 7.80e+06 | 72.99 |
| | | 1000 | 2.07e+06 | 4.86e+05 | 76.55 |
| | 83 | 10 | 7.20e+07 | 4.13e+07 | 42.60 |
| | | 100 | 5.90e+06 | 3.90e+06 | 33.86 |
| | | 1000 | 1.57e+05 | 9.94e+04 | 36.81 |
| 0.200 | 00 | 10 | 9.90e+07 | 2.31e+07 | 76.72 |
| | | 100 | 1.24e+07 | 3.87e+06 | 68.71 |
| | | 1000 | 1.14e+06 | 3.35e+05 | 70.50 |
| | 16 | 10 | 1.29e+08 | 6.43e+07 | 50.29 |
| | | 100 | 1.08e+07 | 6.36e+06 | 41.25 |
| | | 1000 | 1.05e+06 | 5.90e+05 | 44.08 |
| | 50 | 10 | 1.02e+08 | 8.03e+07 | 21.04 |
| | | 100 | 2.78e+06 | 2.09e+06 | 24.87 |
| | | 1000 | 2.22e+05 | 1.46e+05 | 34.07 |
| | 83 | 10 | 7.51e+07 | 6.47e+07 | 13.85 |
| | | 100 | 5.43e+06 | 4.48e+06 | 17.45 |
| | | 1000 | 3.83e+05 | 3.29e+05 | 14.15 |

#### 3.3.3. Instance size and distribution of nodes

In this analysis, we study the robustness of `2.5-opt-EEais` that adopts heuristic biasing with respect to instance size and nodes distribution. Note that the significance level in the adaptive sample size procedure is set to 0.05. We consider three levels [100,300,1000] for instance size and two levels for nodes distribution, namely, uniform and clustered. The *F*-ratio and the *p*-values from the ANOVA table show that these factors do not have any significant impact on the relative solution quality and computation time. Note that the instance size will always have a significant impact on the absolute solution cost and computation time; however, recall that in this analysis, we always study the relative solution quality with respect to `2.5-opt-EEs-1000`.

Even though the parameter tuning for `2.5-opt-EEais` with heuristic biasing is performed only on the clustered instances of size 1000, it achieves good solutions also for other levels of instance size. This is mainly attributed to the ineffectiveness of two-exchange moves for small instances: $min_{is}$ and $u$ are the two parameters of `2.5-opt-EEais` that depend on the instance size; under the given parameter setting, for instance sizes 100 and 300, importance sampling is completely disabled for two-exchange moves; from the results it seems that the usage of importance sampling in two-exchange moves is not crucial for small instances.

### 3.4. Experiments to assess variance reduction

In this section, we study the magnitude of reduction in variance obtained using heuristic biasing. For this purpose, we analyze the variance under two settings: `2.5-opt-EEs` that adopts a fixed sample size without importance sampling and `2.5-opt-EEs`

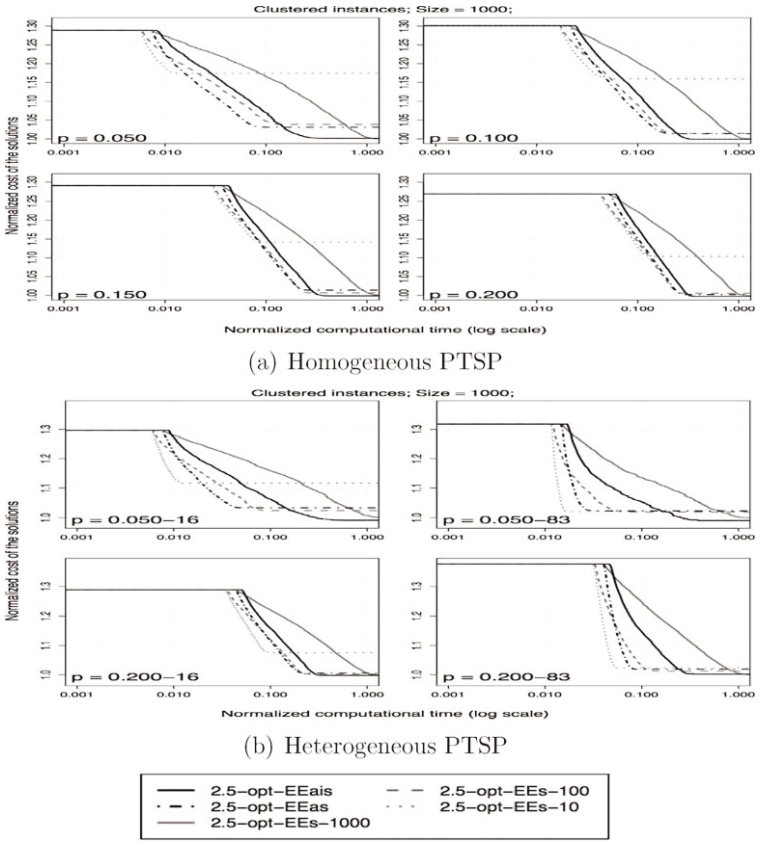


**Fig. 7.** Experimental results on clustered PTSP instances of size 1000. The plots represent the cost of the solutions obtained by `2.5-opt-EEas`, `2.5-opt-EEais`, `2.5-opt-EEs-10`, and `2.5-opt-EEs-100` normalized by those obtained by `2.5-opt-EEs-1000`. Each algorithm is stopped when it reaches a local optimum. The normalization is done on an instance by instance basis for 50 instances; the normalized solution cost and the computation time are then aggregated.

that adopts a fixed sample size and heuristic biasing. We consider three sample size: 10, 100, and 1000. The two algorithms are then allowed to explore 1000 solutions. The variance of the cost difference estimator for each estimation in two-exchange and node-insertion moves is recorded. The results for node-insertion moves are shown in Table 2. Since the computational results for two-exchange moves show that the reduction in variance is rather small and less than 1%, we do not list the values in a table.

The magnitude of reduction in variance is very high for the node-insertion moves, in particular for low probability values: on average, the variance of the cost estimator when using heuristic biasing is about 97% to 14% less than when not using importance sampling. Another important observation is that a reduction in variance is achieved by increasing the size of the sample; however, the percentage reduction does not follow a same trend. For example, consider the case at $p_m = 0.200$ and $per_{m_v} = 0$: the average variance is reduced from 2.31e+07 to 3.35e+06 by increasing the realizations from 10 to 1000; however, the percentage reduction does not show a strictly decreasing trend as it goes from 76.72 to 68.71, and finally 70.50.

### 3.5. Experiments on estimation-based algorithms

In this section, we study the performance of `2.5-opt-EEas` and `2.5-opt-EEais` by comparing their solution cost and computation time to `2.5-opt-EEs`. In the case of `2.5-opt-EEs`, we consider samples of size 10, 100, and 1000; we denote these algorithms by `2.5-opt-EEs-10`, `2.5-opt-EEs-100`, and `2.5-opt-EEs-1000`, respectively. Note that these algorithms do not use the adaptive sample size and the importance sampling procedures. The results of the comparison of the five algorithms are given in Fig. 7, where `2.5-opt-EEs-1000` is taken as a reference. Table 3 shows the absolute values.

The computational results show that `2.5-opt-EEais` is more effective than the other algorithms—in particular, for low probability levels. For what concerns the comparison of `2.5-opt-EEais` and `2.5-opt-EEas`, the results show that the adoption of importance sampling allows the former to achieve high quality solutions for very low probability levels, that is, for $p$ and $p_m < 0.2$: the average cost of the local optima obtained by `2.5-opt-EEais` is between 1% and 3% less than that of `2.5-opt-EEas`. The observed differences are significant in a statistical sense. The poor solution cost of `2.5-opt-EEas` can be mainly attributed to the following reason: since this algorithm considers the next neighbor solution when the test statistic cannot be computed after five realizations (see Section 2.3.2), it rejects moves which are likely to be accepted. However, the adoption of importance sampling in `2.5-opt-EEais` reduces the effect of this problem. For high probability levels, the average cost of the solutions and the computation time of `2.5-opt-EEais` are comparable to those of `2.5-opt-EEas`.

Concerning the comparison of `2.5-opt-EEais` and `2.5-opt-EEs-1000`, on average they have very similar costs. However, the advantage of `2.5-opt-EEais` is the computation time: `2.5-opt-EEais` is faster than `2.5-opt-EEs-1000` approximately by a factor of four.

Regarding the comparison of `2.5-opt-EEais` and `2.5-opt-EEs-100`, for low probability levels the average cost of the solutions obtained by the former is between 1% and 3% lower than that of `2.5-opt-EEs-100`. This clearly shows that the adoption of 100 realizations is not sufficient for these probability levels. Note that these differences are significant according to the paired $t$-test. On the other hand, for high probability levels, the two algorithms are comparable to one another with respect to solution quality and computation time.

Although faster, `2.5-opt-EEs-10` achieves a very poor solution quality: the average cost of the solutions obtained by `2.5-opt-EEs-10` is between 17% and 2% higher than that of `2.5-opt-EEais`.

The experimental results for $p > 0.5$ are reported in Balaprakash et al. (2007b). These results show that the algorithms achieve equivalent results with respect to solution quality. Moreover, the results of `2.5-opt-EEs-10` show that a sample size of 10 is sufficient to tackle instances with $p > 0.5$. For what concerns computation time, `2.5-opt-EEais` and `2.5-opt-EEs-100` are comparable to `2.5-opt-EEs-10`. However, `2.5-opt-EEais` is faster than `2.5-opt-EEs-1000` by a factor of three. Note that `2.5-opt-EEais` and `2.5-opt-EEas` are essentially the same for these probability levels.

Taking into account both the computation time and the cost of the solutions obtained, we can see that `2.5-opt-EEais` emerges as a clear winner among the considered estimation-based algorithms.

**Table 3**
Experimental results for `2.5-opt-EEas`, `2.5-opt-EEais`, `2.5-opt-EEs-10`, `2.5-opt-EEs-100`, and `2.5-opt-EEs-1000` on clustered instances of size 1000. Each algorithm is allowed to run until it reaches a local optimum. The table gives, for each probability level, the mean and the standard deviation (s.d.) of the final solution cost and of the computation time in seconds over 50 instances.

| | Algorithm | Solution cost | | Computation time | |
|---|---|---|---|---|---|
| | | Mean | s.d. | Mean | s.d. |
| *Homogeneous PTSP* | | | | | |
| $p = 0.050$ | 2.5-opt-EEais | 4,020,433 | 437,996 | 11.100 | 1.794 |
| | 2.5-opt-EEas | 4,137,855 | 430,945 | 2.970 | 0.497 |
| | 2.5-opt-EEs-1000 | 4,014,200 | 455,410 | 41.104 | 6.821 |
| | 2.5-opt-EEs-100 | 4,168,788 | 434,760 | 4.309 | 0.713 |
| | 2.5-opt-EEs-10 | 4,713,400 | 491,452 | 0.482 | 0.035 |
| $p = 0.100$ | 2.5-opt-EEais | 5,103,869 | 508,867 | 4.119 | 0.451 |
| | 2.5-opt-EEas | 5,179,648 | 486,450 | 2.230 | 0.249 |
| | 2.5-opt-EEs-1000 | 5,108,555 | 503,921 | 14.096 | 1.972 |
| | 2.5-opt-EEs-100 | 5,183,844 | 470,288 | 2.629 | 0.306 |
| | 2.5-opt-EEs-10 | 5,922,935 | 509,627 | 0.591 | 0.053 |
| $p = 0.150$ | 2.5-opt-EEais | 5,959,120 | 496,566 | 2.495 | 0.301 |
| | 2.5-opt-EEas | 6,050,183 | 505,229 | 1.702 | 0.161 |
| | 2.5-opt-EEs-1000 | 5,966,002 | 479,174 | 8.104 | 1.172 |
| | 2.5-opt-EEs-100 | 6,007,125 | 500,754 | 1.827 | 0.187 |
| | 2.5-opt-EEs-10 | 6,808,184 | 555,047 | 0.648 | 0.045 |
| $p = 0.200$ | 2.5-opt-EEais | 6,701,562 | 545,366 | 1.776 | 0.147 |
| | 2.5-opt-EEas | 6,734,587 | 558,760 | 1.407 | 0.120 |
| | 2.5-opt-EEs-1000 | 6,720,197 | 543,464 | 5.596 | 0.574 |
| | 2.5-opt-EEs-100 | 6,758,117 | 563,812 | 1.349 | 0.114 |
| | 2.5-opt-EEs-10 | 7,416,077 | 612,763 | 0.661 | 0.053 |
| *Heterogeneous PTSP* | | | | | |
| $p = 0.050(16)$ | 2.5-opt-EEais | 3,949,356 | 409,824 | 13.494 | 3.379 |
| | 2.5-opt-EEas | 4,119,370 | 461,810 | 1.473 | 0.207 |
| | 2.5-opt-EEs-1000 | 3,984,725 | 441,295 | 38.069 | 7.455 |
| | 2.5-opt-EEs-100 | 4,082,128 | 435,499 | 2.806 | 0.405 |
| | 2.5-opt-EEs-10 | 4,449,540 | 447,232 | 0.418 | 0.028 |
| $p = 0.050(83)$ | 2.5-opt-EEais | 3,876,139 | 483,153 | 6.251 | 2.457 |
| | 2.5-opt-EEas | 4,005,424 | 550,905 | 0.511 | 0.054 |
| | 2.5-opt-EEs-1000 | 3,914,341 | 522,037 | 19.476 | 3.409 |
| | 2.5-opt-EEs-100 | 3,990,014 | 519,289 | 1.033 | 0.163 |
| | 2.5-opt-EEs-10 | 3,996,970 | 531,437 | 0.305 | 0.011 |
| $p = 0.200(16)$ | 2.5-opt-EEais | 6,589,342 | 537,399 | 1.910 | 0.203 |
| | 2.5-opt-EEas | 6,641,879 | 547,318 | 1.283 | 0.118 |
| | 2.5-opt-EEs-1000 | 6,601,665 | 537,940 | 6.674 | 0.890 |
| | 2.5-opt-EEs-100 | 6,620,425 | 553,190 | 1.447 | 0.114 |
| | 2.5-opt-EEs-10 | 7,100,032 | 569,439 | 0.579 | 0.049 |
| $p = 0.200(83)$ | 2.5-opt-EEais | 6,244,761 | 605,180 | 1.991 | 0.284 |
| | 2.5-opt-EEas | 6,348,075 | 607,690 | 0.546 | 0.039 |
| | 2.5-opt-EEs-1000 | 6,230,550 | 602,679 | 7.217 | 1.128 |
| | 2.5-opt-EEs-100 | 6,306,303 | 604,002 | 0.811 | 0.090 |
| | 2.5-opt-EEs-10 | 6,375,420 | 613,433 | 0.367 | 0.020 |

### 3.6. Comparison with the analytical computation algorithm

In this section, we compare 2.5-opt-EEais with heuristic biasing to 2.5-opt-ACs. For this purpose, we generate 50 new instances for each probability level. The rationale behind the adoption of a new set of instances is the following: 2.5-opt-EEais and 2.5-opt-ACs are selected as winners from a set of five and three algorithms, respectively, where all of them are evaluated on a same set of instances. Basing the comparison of 2.5-opt-EEais and 2.5-opt-ACs on the same set of instances might possibly introduce a bias in favor of 2.5-opt-EEais. This issue is known as over-tuning; we refer the reader to Birattari (2004) for further discussion.

The computational results given in Fig. 8 show that 2.5-opt-EEais is very competitive. Regarding the time required to reach local optima, irrespective of the probability levels, 2.5-opt-EEais is approximately two orders and three orders of magnitude faster than 2.5-opt-ACs, for homogeneous and heterogeneous instances, respectively. This very large speed difference in the heterogeneous case—approximately one order of magnitude more than the difference in speed between the algorithms for the homogeneous case—can be attributed to the computational overhead involved in the adoption of the arbitrary precision arithmetics. We refer the reader to Balaprakash et al. (2007b) for the absolute values.

The average cost of local optima obtained by 2.5-opt-EEais is comparable to the one of 2.5-opt-ACs. In Table 4, we report the observed relative difference between the cost of the local optima obtained by the two algorithms and a 95% confidence bound on this relative difference. This bound is obtained through a two sided paired *t*-test. Table 4 confirms that, concerning the average cost of the local optima found, 2.5-opt-EEais is essentially equivalent to 2.5-opt-ACs. Nevertheless, with 95% confidence, under the current experimental setting, we can state that, should ever the average cost obtained by 2.5-opt-EEais be higher than the one obtained by 2.5-opt-ACs, the difference would be at most 1.2% and 2.1% for the homogeneous and the heterogeneous instances, respectively.

### 3.7. Experiments with iterated local search

In this section, we study the behavior of 2.5-opt-EEais and 2.5-opt-EEs-lOO integrated into iterated local search (ILS) (Lourenço et al., 2002), a metaheuristic on which many state-of-the-art algorithms for the TSP are based (Hoos and Stützle, 2005). We denote the two algorithms ILS-2.5-opt-EEais and ILS-2.5-opt-EEs-lOO. The goal is to find the most effective algorithm for tackling the PTSP. It is interesting to note that, for a given computation time, the two algorithms behave differently. ILS-2.5-opt-EEais finds at each iteration a high quality local optimum because of its use of the 2.5-opt-EEais local search; however, this is obtained at the expense of performing relatively few iterations. On the contrary, ILS-2.5-opt-EEs-lOO, which uses the faster though less effective 2.5-opt-EEs-lOO local search, produces at each iteration lower quality local optima but performs more iterations than ILS-2.5-opt-EEais.

We implemented standard ILS algorithms in which new starting solutions for the subsequent local search are generated by perturbing the incumbent local optimum $s^*$. For the perturbation, we adopt a hybrid scheme that consists in first performing two random dou-

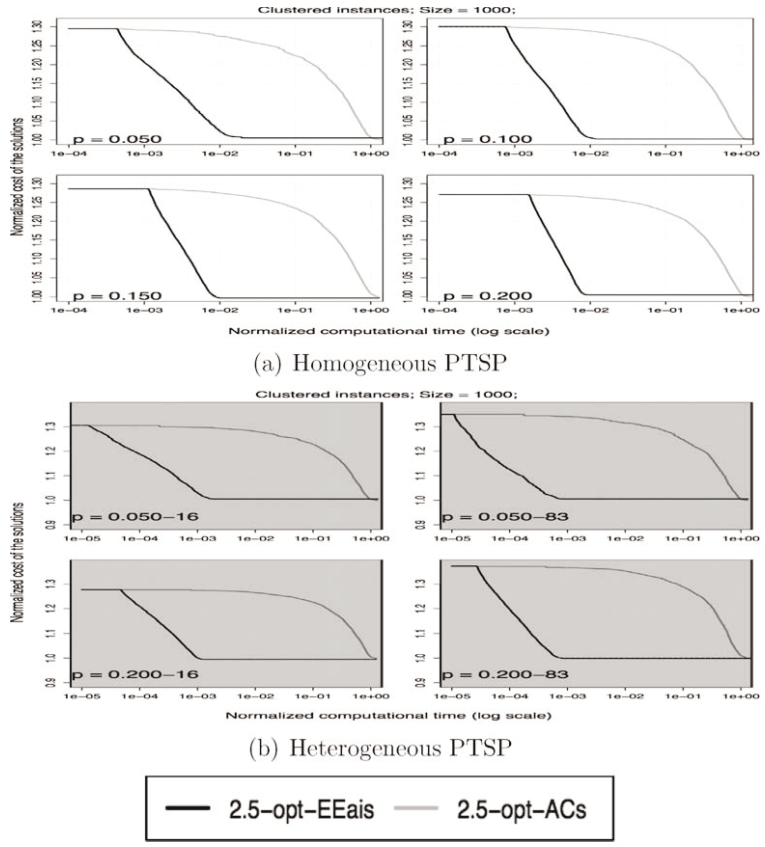


**Fig. 8.** Experimental results on clustered PTSP instances of size 1000. The plots represent the cost of the solutions obtained by 2.5-opt-EEais normalized by those obtained by 2.5-opt-ACs. Each algorithm is stopped when it reaches a local optimum. For the heterogeneous case, 2.5-opt-ACs uses a library for arbitrary precision arithmetics. To emphasize this fact, the backgrounds of plots are gray. The normalization is done on an instance by instance basis for 50 instances; the normalized solution cost and the computation time are then aggregated.

**Table 4**
Comparison of the average cost obtained by 2.5-opt-EEais and by 2.5-opt-ACs on clustered instances of size 1000. For each level of probability, the table reports the observed relative difference and a 95% confidence interval (CI) obtained through the *t*-test on the relative difference. Concerning the relative difference, if the value is positive, 2.5-opt-EEais obtained an average cost that is larger than the one obtained by the other algorithm considered; if it is negative, 2.5-opt-EEais reached solutions of lower average cost. In both cases, a value is typeset in boldface if it is significantly different from zero according to the *t*-test at a confidence of 95%.

| *p* | 2.5-opt-EEais vs. 2.5-opt-ACs | |
|---|---|---|
| | Difference | 95% CI |
| *Homogeneous PTSP* | | |
| 0.050 | +0.546% | [−0.157,+1.248]% |
| 0.075 | +0.232% | [−0.675,+1.139]% |
| 0.100 | +0.284% | [−0.645,+1.214]% |
| 0.125 | −0.333% | [−1.122,+0.456]% |
| 0.150 | −0.327% | [−1.132,+0.478]% |
| 0.200 | +0.422% | [−0.386,+1.231]% |
| *Heterogeneous PTSP* | | |
| 0.050(16) | +0.472% | [−0.243,+1.187]% |
| 0.050(50) | +**0.812**% | [+0.147,+1.478]% |
| 0.050(83) | +0.390% | [−0.648,+1.428]% |
| 0.075(16) | +0.998% | [−0.081,+2.077]% |
| 0.075(50) | +0.191% | [−0.372,+0.754]% |
| 0.075(83) | +0.780% | [−0.128,+1.688]% |
| 0.100(16) | −0.037% | [−0.868,+0.795]% |
| 0.100(50) | −0.199% | [−1.091,+0.693]% |
| 0.100(83) | +0.352% | [−0.494,+1.199]% |
| 0.200(16) | −0.514% | [−1.394,+0.366]% |
| 0.200(50) | −**1.052**% | [−1.781,−0.323]% |
| 0.200(83) | −0.086% | [−0.800,+0.629]% |

ble-bridge moves and then changing the position of *ps*% of the nodes, where *ps* is a parameter. A change of the position is done by picking uniformly at random *ps*% of nodes, removing them from the tour and then re-inserting them again according to the farthest insertion heuristic. In our experiments, the parameter *ps* is set to 10. From the solution obtained after the perturbation, a new local search is started. If the newly identified local optimum has a lower cost than $s^*$, it is accepted as the new incumbent solution.

We include two more algorithms in the analysis: an ILS algorithm built on top of 2.5-opt-EEs that uses a sample size schedule proposed by Gutjahr (2004). In this schedule, the number of realizations is increased on the basis of the iteration counter. We denote this algorithm as ILS-2.5-opt-EEs-sss, where sss stands for sample size schedule. The second algorithm is ILS-2.5-opt-EEs-1000, which adopts 2.5-opt-EEs-1000; this algorithm is used as a reference algorithm.

In ILS-2.5-opt-EEais, the acceptance criterion compares two local optima by using the *t*-test with a maximum of 1000 realizations, which are unchanged throughout all the iterations. In ILS-2.5-opt-EEs-100 and ILS-2.5-opt-EEs-1000, the acceptance criterion compares two local optima based on 100 and 1000 realizations, respectively. In ILS-2.5-opt-EEs-sss, the sample size schedule determines the number of realizations for comparing the two local optima.

The stopping criterion for the considered algorithms is set to 100 seconds. We use 50 instances for each probability level. The results on clustered instances with 1000 nodes are given in Fig. 9 and Table 5.

From the computational results, we can see that the ILS algorithm that uses 2.5-opt-EEais for each iteration is very effective. The average cost of the solutions obtained by ILS-2.5-opt-EEais is between 4% and 0.7% (homogeneous case), 2% and 0.8% (heterogeneous case), lower than ILS-2.5-opt-

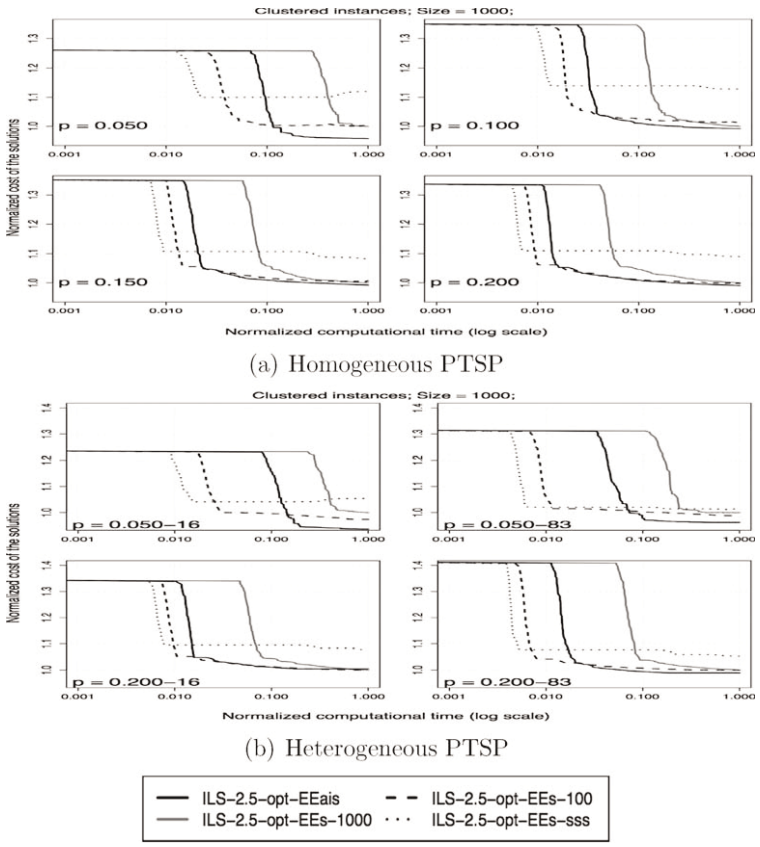


**Fig. 9.** Experimental results on clustered PTSP instances of size 1000. The plots represent the cost of the solutions obtained by ILS-2.5-opt-EEais, ILS-2.5-opt-EEs-100, and ILS-2.5-opt-EEs-sss normalized by the one obtained by ILS-2.5-opt-EEs-1000. The normalization is done on an instance by instance basis for 50 instances; the normalized solution cost and the computation time are then aggregated.

**Table 5**
Experimental results for ILS-2.5-opt-EEais, ILS-2.5-opt-EEs-l00, ILS-2.5-opt-EEs-l000, and ILS-2.5-opt-EEs-sss on clustered instances of size 1000. The table gives mean and standard deviation (s.d.) of final solution cost and computation time in seconds. The results are given for 50 instances at each probability level. Each algorithm is allowed to run for 100 seconds.

| | Algorithm | Solution cost | |
|---|---|---|---|
| | | Mean | s.d. |
| *Homogeneous PTSP* | | | |
| $p = 0.050$ | ILS-2.5-opt-EEais | 3,929,167 | 391,155 |
| | ILS-2.5-opt-EEs-l000 | 4,101,042 | 525,937 |
| | ILS-2.5-opt-EEs-l00 | 4,124,123 | 402,049 |
| | ILS-2.5-opt-EEs-sss | 4,591,082 | 621,302 |
| $p = 0.100$ | ILS-2.5-opt-EEais | 4,882,334 | 416,657 |
| | ILS-2.5-opt-EEs-l000 | 4,919,269 | 420,385 |
| | ILS-2.5-opt-EEs-l00 | 4,983,033 | 424,849 |
| | ILS-2.5-opt-EEs-sss | 5,550,812 | 776,072 |
| $p = 0.150$ | ILS-2.5-opt-EEais | 5,645,487 | 438,838 |
| | ILS-2.5-opt-EEs-l000 | 5,689,150 | 446,088 |
| | ILS-2.5-opt-EEs-l00 | 5,724,625 | 450,742 |
| | ILS-2.5-opt-EEs-sss | 6,148,684 | 748,337 |
| $p = 0.200$ | ILS-2.5-opt-EEais | 6,306,761 | 461,408 |
| | ILS-2.5-opt-EEs-l000 | 6,365,820 | 469,658 |
| | ILS-2.5-opt-EEs-l00 | 6,356,017 | 457,958 |
| | ILS-2.5-opt-EEs-sss | 6,939,857 | 997,612 |
| *Heterogeneous PTSP* | | | |
| $p = 0.050(16)$ | ILS-2.5-opt-EEais | 3,913,474 | 356,073 |
| | ILS-2.5-opt-EEs-l000 | 4,177,331 | 658,040 |
| | ILS-2.5-opt-EEs-l00 | 4,026,444 | 350,354 |
| | ILS-2.5-opt-EEs-sss | 4,333,535 | 531,310 |
| $p = 0.050(83)$ | ILS-2.5-opt-EEais | 3,829,801 | 404,242 |
| | ILS-2.5-opt-EEs-l000 | 3,962,324 | 522,985 |
| | ILS-2.5-opt-EEs-l00 | 3,892,286 | 361,572 |
| | ILS-2.5-opt-EEs-sss | 4,006,539 | 452,628 |
| $p = 0.200(16)$ | ILS-2.5-opt-EEais | 6,317,297 | 404,523 |
| | ILS-2.5-opt-EEs-l000 | 6,290,327 | 387,254 |
| | ILS-2.5-opt-EEs-l00 | 6,270,120 | 385,406 |
| | ILS-2.5-opt-EEs-sss | 6,766,163 | 784,251 |
| $p = 0.200(83)$ | ILS-2.5-opt-EEais | 5,933,131 | 375,063 |
| | ILS-2.5-opt-EEs-l000 | 5,999,728 | 494,003 |
| | ILS-2.5-opt-EEs-l00 | 5,980,738 | 382,930 |
| | ILS-2.5-opt-EEs-sss | 6,312,369 | 709,112 |

EEs-l00. Note that the observed differences between the algorithms are statistically significant according to a *t*-test, at a significance level of 0.05. For what concerns the comparison of ILS-2.5-opt-EEais with the reference algorithm ILS-2.5-opt-EEs-l000, the average solution cost of the former is between 4% and 0.09% (homogeneous case), 6% and 0.1% (heterogeneous case), lower than ILS-2.5-opt-EEs-l000. There is only one exception to this general trend: for $p_m = 0.200$ and $per_{m_v} = 16$, ILS-2.5-opt-EEs-l00 and ILS-2.5-opt-EEs-l000 obtain average solution costs which are 0.7% and 0.4% lower than that of ILS-2.5-opt-EEais, respectively.

An interesting observation concerning the comparison of ILS-2.5-opt-EEs-l00 and ILS-2.5-opt-EEs-l000 is that the average cost reached by the former is either better than or comparable to the latter. This is due to the fact that the use of 100 realizations instead of 1000 allows ILS-2.5-opt-EEs-l00 to perform more iterations than ILS-2.5-opt-EEs-l000, which in turn results in solutions of higher quality.

For what concerns the performance of ILS-2.5-opt-EEs-sss, the average solution cost is rather poor and significantly worse than all the other algorithms. This can be attributed to the fact that the particular sample size schedule is designed for a metaheuristic that is allowed to run for a relatively long computation time without an effective local search.

For the instances with high probability values ($p > 0.5$), the average cost obtained by ILS-2.5-opt-EEais is comparable to the one obtained by ILS-2.5-opt-EEs-l00 and ILS-2.5-opt-EEs-l000.

Finally, we study the behavior of 2.5-opt-EEais with heuristic biasing and 2.5-opt-ACs integrated into ILS, namely, ILS-2.5-opt-EEais and ILS-2.5-opt-ACs. Since 2.5-opt-ACs is rather slow when compared to 2.5-opt-EEais, we use the following stopping criterion: ILS-2.5-opt-EEais is run until it performs 15 perturbations and the time needed for completion is recorded. The time limit for ILS-2.5-opt-ACs is then set to 100 times the time taken by ILS-2.5-opt-EEais. The computational results obtained on the homogeneous and the heterogeneous instances show that ILS-2.5-opt-EEais is very effective with respect to both solution quality and computation time. In spite of the fact that ILS-2.5-opt-EEais is allowed to run for a computation time that is two orders of magnitude less than the one of ILS-2.5-opt-ACs, the average cost of the solutions obtained by the former is between 0.4–8% and 0.8–26% lower than that of the latter, for the homogeneous and heterogeneous cases, respectively. We refer the reader to Balaprakash et al. (2007b) for the complete results.

## 4. Conclusion and future work

In this paper, we integrated two widely known variance reduction techniques, adaptive sample size and importance sampling, into an estimation-based local search to tackle the PTSP. We investigated several ways of using the two procedures in the PTSP delta evaluation. In particular, we customized the two procedures by taking into account problem-specific knowledge. Moreover, we showed that an offline parameter tuning algorithm can be used effectively for finding the biased probability distribution of the importance sampling procedure for the PTSP. The computational results show that the heuristic customization of the adaptive sample size and the importance sampling procedures allows the estimation-based local search to achieve high quality solutions for the PTSP instances with high and low probability values in a relatively short computation time. In virtue of this speed advantage, we have obtained a new state-of-the-art iterative improvement algorithm for the PTSP.

Further research will be devoted to assess the behavior of the proposed algorithm when used as an embedded heuristic in other metaheuristics such as ant colony optimization and memetic algorithms. From an application perspective, the estimation-based local search will be extended to solve more complex problems such as stochastic vehicle routing problems.

## Acknowledgements

The authors thank Leonora Bianchi for providing the source code of 2-p-opt and 1-shift. This research has been supported by COMP$^2$SYS, an Early Stage Trainig project funded by the European Commission within the Marie Curie Actions program (MEST-CT-2004-505079), and by ANTS and META-X, which are ARC projects funded by the French Community of Belgium. The authors acknowledge support from the fund for scientific research F.R.S.-FNRS of the French Community of Belgium.

## References

Alkhamis, T.M., Ahmed, M.A., Tuan, V.K., 1999. Simulated annealing for discrete optimization with estimation. European Journal of Operational Research 116 (3), 530–544.

Balaprakash, P., Birattari, M., Stützle, T., 2007a. Improvement strategies for the F-Race algorithm: Sampling design and iterative refinement. In: Bartz-Beielstein, T., Blesa, M., Blum, C., Naujoks, B., Roli, A., Rudolph, G., Sampels, M. (Eds.),

Hybrid Metaheuristics of LNCS, vol. 4771. Springer-Verlag, Berlin, Germany, pp. 113–127.
Balaprakash, P., Birattari, M., Stützle, T., Dorigo, M., 2007b. Adaptive sample size and importance sampling in estimation-based local search for the probabilistic traveling salesman problem: A complete analysis. Technical Report TR/IRIDIA/2007-015, IRIDIA, Université Libre de Bruxelles, Brussels, Belgium. <http://iridia.ulb.ac.be/IridiaTrSeries/IridiaTr2007-015r002.pdf>.
Balaprakash, P., Birattari, M., Stützle, T., Dorigo, M., 2008. Extended empirical analysis of adaptive sample size and importance sampling in estimation-based local search for the probabilistic traveling salesman problem. IRIDIA Supplementary Page. <http://iridia.ulb.ac.be/supp/IridiaSupp2008-010/>.
Bentley, J.L., 1992. Fast algorithms for geometric traveling salesman problems. ORSA Journal on Computing 4 (4), 387–411.
Bertsimas, D., 1988. Probabilistic combinatorial optimization problems. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
Bertsimas, D., Jaillet, P., Odoni, A., 1990. A priori optimization. Operations Research 38 (6), 1019–1033.
Bianchi, L., 2006. Ant colony optimization and local search for the probabilistic traveling salesman problem: A case study in stochastic combinatorial optimization. Ph.D. Thesis, Université Libre de Bruxelles, Brussels, Belgium.
Bianchi, L., Campbell, A., 2007. Extension of the 2-p-opt and 1-shift algorithms to the heterogeneous probabilistic traveling salesman problem. European Journal of Operational Research 176 (1), 131–144.
Bianchi, L., Knowles, J., Bowler, N., 2005. Local search for the probabilistic traveling salesman problem: Correction to the 2-p-opt and 1-shift algorithms. European Journal of Operational Research 162, 206–219.
Birattari, M., 2004. The problem of tuning metaheuristics as seen from a machine learning perspective. Ph.D. Thesis, Université Libre de Bruxelles, Brussels, Belgium.
Birattari, M., Balaprakash, P., Dorigo, M., 2006. The ACO/F-RACE algorithm for combinatorial optimization under uncertainty. In: Doerner, K.F., Gendreau, M., Greistorfer, P., Gutjahr, W.J., Hartl, R.F., Reimann, M. (Eds.), Metaheuristics – Progress in Complex Systems Optimization Operations Research/Computer Science Interfaces Series. Springer-Verlag, Berlin, Germany, pp. 189–203.
Birattari, M., Balaprakash, P., Stützle, T., Dorigo, M., 2007. Extended empirical analysis of estimation-based local search for stochastic combinatorial optimization. IRIDIA Supplementary Page. <http://iridia.ulb.ac.be/supp/IridiaSupp2007-001/>.
Birattari, M., Balaprakash, P., Stützle, T., Dorigo, M., 2008. Estimation-based local search for stochastic combinatorial optimization using delta evaluations: A case study in the probabilistic traveling salesman problem. INFORMS Journal on Computing 20 (4), 644–658.
Chervi, P., 1988. A computational approach to probabilistic vehicle routing problems. Master's Thesis, Massachusetts Institute of Technology, Cambridge, MA.
Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., Zimmermann, P., 2007. MPFR: A multiple-precision binary floating-point library with correct rounding. ACM Transactions on Mathematical Software 33 (2), 1–15. <http://www.mpfr.org/>.
Gutjahr, W.J., 2004. S-ACO: An ant based approach to combinatorial optimization under uncertainity. In: Dorigo, M., Birattari, M., Blum, C., Gambardella, L.M., Mondada, F., Stützle, T. (Eds.), Ant Colony Optimization and Swarm Intelligence, Fifth International Workshop, ANTS 2004, of LNCS, vol. 3172. Springer-Verlag, Berlin, Germany, pp. 238–249.
Gutjahr, W.J., Strauss, C., Toth, M., 2000a. Crashing of stochastic activities by sampling and optimization. Business Process Management Journal 12, 125–135.
Gutjahr, W.J., Strauss, C., Wagner, E., 2000b. A stochastic branch-and-bound approach to activity crashing in project management. INFORMS Journal on Computing 12, 125–135.
Homem-de-Mello, T., 2003. Variable-sample methods for stochastic optimization. ACM Transactions on Modeling and Computer Simulation 13 (2), 108–133.
Hoos, H., Stützle, T., 2005. Stochastic Local Search: Foundations and Applications. Morgan Kaufmann, San Francisco, CA.
Jaillet, P., 1985. Probabilistic traveling salesman problems. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA.
Johnson, D.S., McGeoch, L.A., 1997. The travelling salesman problem: A case study in local optimization. In: Aarts, E.H.L., Lenstra, J.K. (Eds.), Local Search in Combinatorial Optimization. John Wiley and Sons, Chichester, UK, pp. 215–310.
Johnson, D.S., McGeoch, L.A., Rego, C., Glover, F., 2001. In: Eighth DIMACS Implementation Challenge. <http://www.research.att.com/dsj/chtsp/>.
Lourenço, H.R., Martin, O., Stützle, T., 2002. Iterated local search. In: Glover, F., Kochenberger, G. (Eds.), Handbook of Metaheuristics of International Series in Operation Research and Management Science, vol. 57. Kluwer Academic Publishers, Norwell, MA, pp. 321–353.
Penky, J.F., Miller, D.L., 1994. A staged primal–dual algorithm for finding a minimum cost perfect two-matching in an undirected graph. ORSA Journal on Computing 6 (1), 68–81.
Pichitlamken, J., Nelson, B.L., 2003. A combined procedure for optimization via simulation. ACM Transactions on Modeling and Computer Simulation 13 (2), 55–179.
Rubinstein, R.Y., 1981. Simulation and the Monte Carlo Method. John Wiley and Sons Inc., New York, NY.