Probabilistic stopping rules for GRASP heuristics and extensions

Celso C. Ribeiro[1], Isabel Rosseti[1], and Reinaldo C. Souza[2]

[1] Department of Computer Science, Universidade Federal Fluminense, Rua Passo da Pátria 156, Niterói, RJ 24210-240, Brazil.
[2] Department of Electrical Engineering, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, RJ 22453-900, Brazil.
{celso,rosseti}@ic.uff.br, reinaldo@ele.puc-rio.br

Abstract. The main drawback of most metaheuristics is the absence of effective stopping criteria. Most implementations of such algorithms stop after performing a given maximum number of iterations or a given maximum number of consecutive iterations without improvement in the best known solution value, or after the stabilization of the set of elite solutions found along the search. We propose effective probabilistic stopping rules for randomized metaheuristics such as GRASP. We show how the probability density function of the solution values obtained along the iterations of such algorithms can be used to implement stopping rules based on the tradeoff between solution quality and the time needed to find a solution that might improve the best solution found. We show experimentally that, in the particular case of GRASP heuristics, the solution values obtained along its iterations fit a Normal distribution that may be used to give an online estimation of the number of solutions obtained in forthcoming iterations that might be at least as good as the incumbent. This estimation is used to validate the stopping rule based on the tradeoff between solution quality and the time needed to find a solution that might improve the incumbent. The robustness of this strategy is illustrated and validated by a thorough computational study reporting results obtained with GRASP implementations to four different combinatorial optimization problems.

1 Introduction and motivation

Metaheuristics are general high-level procedures that coordinate simple heuristics and rules to find good approximate solutions to computationally difficult combinatorial optimization problems. Among them, we find simulated annealing, tabu search, GRASP, VNS, genetic algorithms, scatter search, ant colonies, and others. They are based on distinct paradigms and offer different mechanisms to escape from locally optimal solutions, contrarily to greedy algorithms or local search methods. Metaheuristics are among the most effective solution strategies for solving combinatorial optimization problems in practice and they have been applied to a large variety of areas and situations. The customization

(or instantiation) of some metaheuristic to a given problem yields a heuristic to the latter.

A number of principles and building blocks blended into different and often innovative strategies are common to different metaheuristics. Randomization plays a very important role in algorithm design. Metaheuristics such as simulated annealing, GRASP, VNS, and genetic algorithms rely on randomization to sample the search space. Randomization can also be used to break ties, so that different trajectories can be followed from the same initial solution in multistart methods or to sample fractions of large neighborhoods.

One particularly important use of randomization appears in the context of greedy randomized algorithms, which are based on the same principle of pure greedy algorithms, but make use of randomization to build different solutions at different runs. Greedy randomized algorithms are used in the construction phase of GRASP heuristics or to create initial solutions to population-based metaheuristics such as genetic algorithms or scatter search.

Randomization is also a major component of metaheuristics such as simulated annealing and VNS, in which a solution in the neighborhood of the current one is randomly generated at each iteration.

The main drawback of most metaheuristics is the absence of effective stopping criteria. Most implementations of such algorithms stop after performing a given maximum number of iterations or a given maximum number of consecutive iterations without improvement in the best known solution value, or after the stabilization of the set of elite solutions found along the search. In some cases, the algorithm may perform an exaggerated and non-necessary number of iterations, when the best solution is quickly found (as it often happens in GRASP implementations). In other situations, the algorithm may stop just before the iteration that could find an optimal solution. Dual bounds may be used to implement quality-based stopping rules, but they are often hard to compute or very far from the optimal values, which make them unusable in both situations.

Although Bayesian stopping rules have already been proposed in the past, they were not followed by too many applications or computational experiments and results. Bartkutė et al. [1, 2] made use of order statistics, keeping the value of the k-th best solution found. A probabilistic criterion is used to infer with some confidence that this value will not change further. The method proposed for estimating the optimal value with an associated confidence interval is implemented for optimality testing and stopping in continuous optimization and in a simulated annealing algorithm for the bin-packing problem. The authors observed that the confidence interval for the minimum value can be estimated with admissible accuracy when the number of iterations is increased.

Boender and Rinnooy Kan [3] observed that the most efficient methods for global optimization are based on starting a local optimization routine from an appropriate subset of uniformly distributed starting points. As the number of local optima is frequently unknown in advance, it is a crucial problem when to stop the sequence of sampling and searching. By viewing a set of observed minima as a sample from a generalized multinomial distribution whose cells correspond to

the local optima of the objective function, they obtain the posterior distribution of the number of local optima and of the relative size of their regions of attraction. This information is used to construct sequential Bayesian stopping rules which find the optimal trade off between solution quality and computational effort.

Dorea [6] described a stochastic algorithm for estimating the global minimum of a function and derived two types of stopping rules. The first is based on the estimation of the region of attraction of the global minimum, while the second is based on the existence of an asymptotic distribution of properly normalized estimators. Hart [12] described sequential stopping rules for several stochastic algorithms that estimate the global minimum of a function. Stopping rules are described for pure random search and stratified random search (which partitions the search domain into a finite set of subdomains, with samples being selected from every subdomain according to a fixed distribution). These stopping rules use an estimation of the probability measure of the ϵ-close points to terminate the algorithms when a specified confidence has been achieved. Numerical results indicate that these stopping rules require fewer samples and are more reliable than the previous stopping rules for these algorithms. They also show that the proposed stopping rules can perform as well as Bayesian stopping rules for multistart local search. The authors claimed an improvement on the results reported in [6].

Orsenigo and Vercellis [17] developed a Bayesian framework for stopping rules aimed at controlling the number of iterations in a GRASP heuristic. Two different prior distributions are proposed and stopping conditions are explicitly derived in analytical form. The authors claimed that the stopping rules lead to an optimal trade-off between accuracy and computational effort, saving from unnecessary iterations and still achieving good approximations.

Stopping rules have also been discussed in [7, 33] in another context. The statistical estimation of optimal values for combinatorial optimization problems as a way to evaluate the performance of heuristics was also addressed in [20, 29].

In this paper, we propose effective probabilistic stopping rules for randomized metaheuristics. In the next section, we show how an estimation of the probability density function of the solution values (obtained by a stochastic local search heuristic) can be used to implement stopping rules based on the tradeoff between solution quality and the time needed to find a solution that might improve the best solution found until the current iteration. In Section 3, we give a template of GRASP heuristics for minimization problems and we describe the test instances of the four combinatorial optimization problems that have been considered in the computational experiments: the 2-path network design problem, the p-median problem, the quadratic assignment problem, and the set k-covering problem. Next, we show experimentally in Section 4 that, in the particular case of GRASP algorithms, the solution values obtained along its iterations fit a Normal distribution. This result is validated by thorough numerical experiments on the four combinatorial optimization problems cited above. This approximation is used in Section 5 to give an online estimation of the number of solutions obtained in forthcoming iterations that might be at least as good as the the best known

solution at the time of the current iteration. This estimation is used to validate the stopping rule based on the tradeoff between solution quality and the time needed to find a solution that might improve the incumbent. The robustness of this strategy is illustrated and validated by a computational study reporting results obtained with the GRASP implementations for the four selected problems. Concluding remarks are made in the last section, together with a discussion of extensions to other heuristics, including more general memory-based methods such as GRASP with path-relinking.

2 Probabilistic stopping rule

We denote by X the random variable associated with the objective function value the local minimum obtained at each GRASP iteration. The probability density function and the cumulative probability distribution of the random variable X are given by $f_X(.)$ and $F_X(.)$, respectively. Let f_k be the solution value obtained at iteration k and $f_1, \ldots, f_k$ be a sample formed by the solution values obtained along the k first iterations. We shall make use later in this paper of the estimated mean and standard deviation of the sample $f_1, \ldots, f_k$, which will be denoted by m^k and S^k, respectively. Furthermore, let $f_X^k(.)$ and $F_X^k(.)$ be estimates of $f_X(.)$ and $F_X(.)$, respectively, obtained after the k first GRASP iterations.

We show in this section that $f_X^k(.)$ and $F_X^k(.)$ can be used to give an online estimation of the number of solutions obtained in forthcoming iterations that might be at least as good as the best known solution at the time of the current iteration. This estimation will used to implement the stopping rules based on the time needed to find a solution that might improve the incumbent.

Let UB^k be the value of the best solution found along the k first iterations of the heuristic. Therefore, the probability of finding a solution value smaller than or equal to UB^k in the next iteration can be estimated by

$$F_X^k(UB^k) = \int_{-\infty}^{UB^k} f_X^k(\tau) d\tau. \tag{1}$$

For sake of computational efficiency, the value of $F_X^k(UB^k)$ may be recomputed periodically or whenever the value of the best known solution improves, and not at every iteration of the heuristic.

For any given threshold β, the GRASP iterations can be interrupted when $F_X^k(UB^k)$ becomes smaller than or equal to β, i.e., as soon as the probability of finding in the next iteration a solution at least as good as the current best becomes smaller than or equal to the threshold β. Therefore, the probability value $F_X^k(UB^k)$ may be used to estimate the number of iterations that must be performed by the algorithm to find a new solution that is at least as good as the currently best one. Since the user is able to account for the average time taken by each GRASP iteration, the threshold defining the stopping criterion can either be fixed or determined online, so as to limit the computation time when the probability of finding improving solutions becomes very small.

This strategy will be validated in the next section for GRASP implementations.

3 GRASP and experimental environment

In this section, we give a template for a GRASP heuristic and we describe the optimization problems and test instances that have been used in all computational experiments reported in this paper.

3.1 A template for GRASP

We consider in what follows the combinatorial optimization problem of minimizing $f(x)$ over all solutions $x \in F$, which is defined by a ground set $E = \{e_1, \ldots, e_n\}$, a set of feasible solutions $F \subseteq 2^E$, and an objective function $f : 2^E \to \mathbb{R}$. The ground set E, the objective function $f(.)$, and the constraints defining the set of feasible solutions F are defined and specific for each problem. We seek an optimal solution $x^* \in F$ such that $f(x^*) \leq f(x),\ \forall x \in F$.

GRASP [9] is a multistart metaheuristic, in which each iteration consists of two phases: construction and local search. The construction phase builds a feasible solution. The local search phase investigates the neighborhood of the latter, until a local minimum is found. The best overall solution is kept as the result; see [10, 11, 21–24] for surveys on GRASP and its extensions and applications.

The pseudo-code in Figure 1 gives a template illustrating the main blocks of a GRASP procedure for minimization, in which `MaxIterations` iterations are performed and `Seed` is used as the initial seed for the pseudo-random number generator.

```
procedure GRASP(MaxIterations, Seed)
1.   Set f* ← ∞;
2.   for k = 1, ..., MaxIterations do
3.        x ← GreedyRandomizedAlgorithm(Seed);
4.        x ← LocalSearch(x);
5.        if f(x) < f* then
6.             x* ← x;
7.             f* ← f(x);
8.        end;
9.   end;
10.  return x*, f*;
end.
```

Fig. 1. Template of a GRASP heuristic for minimization.

An especially appealing characteristic of GRASP is the ease with which it can be implemented. Few parameters need to be set and tuned, and therefore

development can focus on implementing efficient data structures to assure quick iterations. Basic implementations of GRASP rely exclusively on two parameters: the stopping criterion (which is usually set as a predefined number of iterations) and the parameter used to limit the size of the restricted candidate list within the greedy randomized algorithm used by the construction phase. In spite of its simplicity and ease of implementation, GRASP is a very effective metaheuristic and produces the best known solutions for many problems, see [10, 11] for extensive surveys of applications of GRASP.

The four combinatorial optimization problems and the test instances used in our computational experiments are reported below.

3.2 The 2-path network design problem

Given a connected undirected graph $G = (V, E)$ with non-negative weights associated with its edges, together with a set formed by K pairs of origin-destination nodes, the 2-path network design problem consists in finding a minimum weighted subset of edges containing a path formed by at most two edges between every origin-destination pair. Applications can be found in the design of communication networks, in which paths with few edges are sought to enforce high reliability and small delays. Its decision version was proved to be NP-complete by Dahl and Johannessen [5]. The GRASP heuristic that has been used in the computational experiments with the 2-path network design problem was originally presented in [26, 27]. The main characteristics of the four instances involved in the experiments are summarized in Table 1.

Table 1. Test instances of the 2-path network design problem.

Instance	$\|V\|$	$\|E\|$	K
2pndp50	50	1225	500
2pndp70	70	2415	700
2pndp90	90	4005	900
2pndp200	200	19900	2000

3.3 The *p*-median problem

Given a set F of m potential facilities, a set U of n customers, a distance function $d : U \times F \to \mathbb{R}$, and a constant $p \leq m$, the p-median problem consists in determining which p facilities to open so as to minimize the sum of the distances from each costumer to its closest open facility. It is a well-known NP-hard problem [14], with numerous applications to location [30] and clustering [19, 32] problems. The GRASP heuristic that has been used in the computational experiments with the p-median problem was originally presented in [25]. The main characteristics of the four instances involved in the experiments are summarized in Table 2.

Table 2. Test instances of the p-median problem.

Instance	m	n	p
pmed10	200	800	67
pmed15	300	1800	100
pmed25	500	5000	167
pmed30	600	7200	200

3.4 The quadratic assignment problem

Given n facilities and n locations represented, respectively, by the sets $F = \{f, \ldots, f_n\}$ and $L = \{\ell_1, \ldots, \ell_n\}$, the quadratic assignment problem proposed by Koopmans and Beckman [15] consists in determining to which location each facility must be assigned. Let $A^{n\times n} = (a_{ij})$ be a matrix where each of its entries $a_{ij} \in \mathbb{R}^+$ represents the flow between facilities f_i and f_j. Let $B^{n\times n} = (b_{ij})$ be a matrix where each of its entries $b_{ij} \in \mathbb{R}^+$ represents the distance between locations ℓ_i and ℓ_j. Let $p : \{1, \ldots, n\} \to \{1, \ldots, n\}$ be an assignment with cost $c(p) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} b_{p(i),p(j)}$. We seek a permutation vector $p \in \Pi_n$ that minimizes the assignment cost $c(p)$, where Π_n stands for the set of all permutations of $\{1, \ldots, n\}$. The quadratic assignment problem is well known to be strongly NP-hard [28]. The GRASP heuristic that has been used in the computational experiments with the quadratic problem was originally presented in [16]. The main characteristics of the four instances involved in the experiments are summarized in Table 3.

Table 3. Test instances of the quadratic assignment problem.

Instance	n
tai30a	30
tai35a	35
tai40a	40
tai50a	50

3.5 The set *k*-covering problem

Given a set $I = \{1, \ldots, m\}$ of objects, let $\{P_1, \ldots, P_n\}$ be a collection of subsets of I, with a non-negative cost c_j associated with each subset P_j, for $j = 1, \ldots, n$. A subset $\hat{J} \subseteq J = \{1, \ldots, n\}$ is a *cover* of I if $\cup_{j\in\hat{J}} P_j = I$. The cost of a cover $\hat{J}$ is $\sum_{j\in\hat{J}} c_j$. The *set covering problem* consists in finding a minimum cost cover J^*. The *set multi-covering problem* is a generalization of the set covering problem, in which each object $i \in I$ must be covered by at least $\ell_i \in \mathbb{Z}_+$ elements of $\{P_1, \ldots, P_n\}$. A special case of the set multi-covering problem arises when

$\ell_i = k$, for all $i \in I$. Following [31], we refer to this problem as the *set k-covering problem*. The problem finds applications in the design of communication networks and in computational biology. The GRASP heuristic that has been used in the computational experiments with the quadratic problem was originally presented in [18]. The main characteristics of the four instances involved in the experiments are summarized in Table 4.

Table 4. Test instances of the set k-covering problem.

Instance	m	n	k
scp42	200	1000	2
scp47	200	1000	2
scp55	200	2000	2
scpa2	300	3000	2

4 Normal approximation for GRASP iterations

In this section, we assume that the solution values obtained by a GRASP procedure fit a Normal distribution. This hypothesis is validated experimentally for all problems and test instances described in the previous section.

Let $f_1, \ldots, f_N$ be a sample formed by all solution values obtained along N GRASP iterations. We assume that the null (H_0) and alternative (H_1) hypotheses are:

H_0: the sample $f_1, \ldots, f_N$ follows a Normal distribution; and
H_1: the sample $f_1, \ldots, f_N$ does not follow a Normal distribution.

The chi-square test is the most commonly used to determine if a given set of observations fits a specified distribution. It is very general and can be used to fit both discrete or continuous distributions [13].

First, a histogram of the sample data is estimated. Next, the observed frequencies are compared with those obtained from the specified density function. If the histogram is formed by k cells, let o_i and e_i be the observed and expected frequencies for the i-th cell, with $i = 1, \ldots, k$. The test starts by computing

$$D = \sum_{i=1}^{k} \frac{(o_i - e_i)^2}{e_i}. \tag{2}$$

If the null hypothesis holds, then D follows a chi-square distribution with $k-1$ degrees of freedom. Since the mean and the standard deviation is estimated from the sample, then two degrees of freedom are lost to compensate for that. The null hypothesis cannot be rejected at a level of significance α if D is less than the value tabulated for $\chi^2_{[1-\alpha;k-3]}$.

Let m and S be, respectively, the average and the standard deviation of the sample $f_1, \ldots, f_N$. A normalized sample $f'_i = (f_i - m)/S$ is obtained by subtracting the average m from each value f_i and dividing the result by the standard deviation S, for $i = 1, \ldots, N$. Then, the null hypothesis stating that the original sample fits a Normal distribution with mean m and standard deviation S is equivalent to compare the normalized sample with the $N(0, 1)$ distribution.

We show below that the solution values obtained along N GRASP iterations fit a Normal distribution, for all problems and test instances presented in Sections 3.2 to 3.5. In all experiments, we used $\alpha = 0.1$ and $k = 14$, corresponding to a histogram with the intervals $(-\infty, -3.0)$, $[-3.0, -2.5)$, $[-2.5, -2.0)$, $[-2.0, -1.5)$, $[-1.5, -1.0)$, $[-1, -0.5)$, $[-0.5, 0.0)$, $[0.0, 0.5)$, $[0.5, 1.0)$, $[1.0, 1.5)$, $[1.5, 2.0)$, $[2.0, 2.5)$, $[2.5, 3.0)$, and $[3.0, \infty)$.

For each instance, we illustrate the Normal fittings after $N = 50$, 100, 500, 1000, 5000, and 10000 GRASP iterations.

Table 5 reports the application of the chi-square test to the four instances of the 2-path network design problem after $N = 50$ iterations. We observe that already after as few as 50 iterations, the solution values obtained by the GRASP heuristic fit very close a Normal distribution.

Table 5. Chi-square test for $1 - \alpha = 90\%$ confidence level: 2-path network design problem.

Instance	Iterations	D	$\chi^2_{[1-\alpha;k-3]}$
2pndp50	50	0.398049	17.275000
2pndp70	50	0.119183	17.275000
2pndp90	50	0.174208	17.275000
2pndp200	50	0.414327	17.275000

To further illustrate that this close fitting is maintained when the number of iterations increase, we present in Table 6 the main statistics for each instance and for increasing values of the number $N = 50$, 100, 500, 1000, 5000, and 10000 of GRASP iterations: mean, standard deviation, skewness (η_3), and kurtosis (η_4). The skewness and the kurtosis are computed as follows [8]:

$$\eta_3 = \frac{\sqrt{N} \cdot \sum_{i=1}^{N} (f_i - m)^3}{[\sum_{i=1}^{N} (f_i - m)^2]^{3/2}} \tag{3}$$

$$\eta_4 = \frac{N \cdot \sum_{i=1}^{N} (f_i - m)^4}{[\sum_{i=1}^{N} (f_i - m)^2]^2}. \tag{4}$$

The skewness measures the symmetry of the original data, while the kurtosis measures the shape of the fitted distribution. Ideally, they should be equal to 0 and 3, respectively, in the case of a perfect Normal fitting. We first notice that the mean is either stable or converges very quickly to a steady-state value when

Table 6. Statistics for Normal fittings: 2-path network design problem.

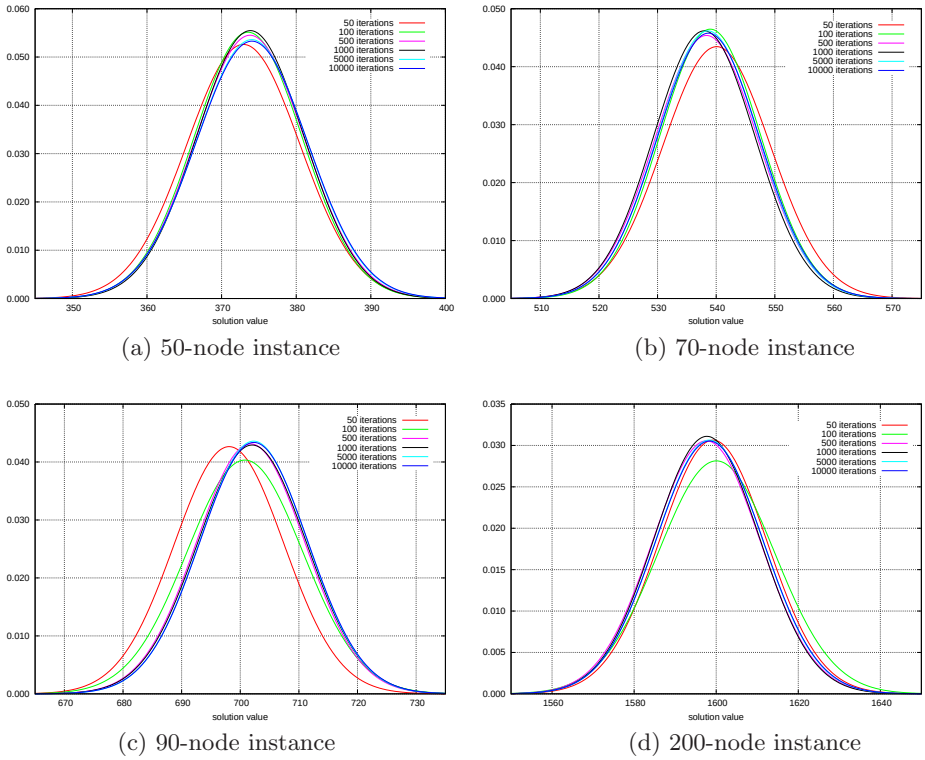
Instance	Iterations	Mean	Std. dev.	Skewness	Kurtosis
	50	372.920000	7.583772	0.060352	3.065799
	100	373.550000	7.235157	-0.082404	2.897830
2pndp50	500	373.802000	7.318661	-0.002923	2.942312
	1000	373.854000	7.192127	0.044952	3.007478
	5000	374.031400	7.442044	0.019068	3.065486
	10000	374.063500	7.487167	-0.010021	3.068129
	50	540.080000	9.180065	0.411839	2.775086
	100	538.990000	8.584282	0.314778	2.821599
2pndp70	500	538.334000	8.789451	0.184305	3.146800
	1000	537.967000	8.637703	0.099512	3.007691
	5000	538.576600	8.638989	0.076935	3.016206
	10000	538.675600	8.713436	0.062057	2.969389
	50	698.100000	9.353609	-0.020075	2.932646
	100	700.790000	9.891709	-0.197567	2.612179
2pndp90	500	701.766000	9.248310	-0.035663	2.883188
	1000	702.023000	9.293141	-0.120806	2.753207
	5000	702.281000	9.149319	0.059303	2.896096
	10000	702.332600	9.196813	0.022076	2.938744
	50	1599.240000	13.019309	0.690802	3.311439
	100	1600.060000	14.179436	0.393329	2.685849
2pndp200	500	1597.626000	13.052744	0.157841	3.008731
	1000	1597.727000	12.828035	0.083604	3.009355
	5000	1598.313200	13.017984	0.057133	3.002759
	10000	1598.366100	13.066900	0.008450	3.019011

the number of iterations increases. Furthermore, the mean after 50 iterations is already very close to that of the Normal fitting after 10000 iterations. The skewness values are consistently very close to 0, while the measured kurtosis of the sample is always close to 3.

Figure 2 displays the Normal distribution fitted for each instance and for each number of iterations. Together with the statistics reported above, these plots illustrate the robustness of the Normal fittings to the solution values obtained along the iterations of the GRASP heuristic for the 2-path network design problem.

Table 7 reports the application of the chi-square test to the four instances of the p-median problem after $N = 50$ iterations. As before, we observe that already after as few as 50 iterations the solution values obtained by the GRASP heuristic for this problem also fit very close a Normal distribution.

Table 8 gives the main statistics for each instance of the p-median problem and for increasing values of the number $N = 50$, 100, 500, 1000, 5000, and 10000 of GRASP iterations: mean, standard deviation, skewness, and kurtosis. As for the previous problem, we notice that the mean value converges or oscillates very slightly when the number of iterations increases. Furthermore, the mean



(a) 50-node instance

(b) 70-node instance

(c) 90-node instance

(d) 200-node instance

Fig. 2. Normal distributions: fitted probability density functions for the 2-path network design problem.

after 50 iterations is already very close to that of the Normal fitting after 10000 iterations. Once again, the skewness values are consistently very close to 0, while the measured kurtosis of the sample is always close to 3.

Figure 3 displays the Normal distribution fitted for each instance and for each number of iterations. Once again, the statistics and plots in these tables and figure illustrate the robustness of the Normal fittings to the solution values obtained along the iterations of the GRASP heuristic for the p-median problem.

Results obtained with the GRASP heuristic for the quadratic assignment problem are reported in Tables 9 and 10 and in Figure 4. The same statistics and plots provided for the previous problems lead to similar findings: they illustrate the robustness of the Normal fittings to the solution values obtained along the iterations of the GRASP heuristic for the quadratic assignment problem.

Finally, we report in Tables 11 and 12 and in Figure 5 the results obtained with the GRASP heuristic for the set k-covering problem. The same statistics and plots already given to the other problems show that also for the set k-

Table 7. Chi-square test for $1 - \alpha = 90\%$ confidence level: p-median problem.

Instance	Iterations	D	$\chi^2_{[1-\alpha;k-3]}$
pmed10	50	0.196116	17.275000
pmed15	50	0.167526	17.275000
pmed25	50	0.249443	17.275000
pmed30	50	0.160131	17.275000

covering problem the Normal fittings to the solution values obtained along the iterations of the GRASP heuristic are very robust.

We conclude this section by observing that the null hypothesis cannot be rejected with $1 - \alpha = 90\%$ of confidence. Therefore, we may approximate the solution values obtained along N iterations of a GRASP heuristic by a Normal distribution that can be progressively fitted and improved as more iterations are performed and more information is available. This approximation will be used in the next section to validate the probabilistic stopping rule proposed in Section 2 for some GRASP heuristics.

5 Validation of the probabilistic stopping rule

We recall from Section 2 that X is the random variable representing the values of the objective function associated with the local minima obtained by each GRASP iteration. The sample $f_1, \ldots, f_k$ is formed by the solution values obtained along the k first iterations of the heuristic, whose estimated mean and standard deviation are m^k and S^k, respectively. Denoting by $UB^k = \min\{f_1, \ldots, f_k\}$ the value of the best solution found along the k first iterations of the heuristic, the probability of finding a solution value smaller than or equal to UB^k in the next iteration can then be estimated by equation (1):

$$F_X^k(UB^k) = \int_{-\infty}^{UB^k} f_X^k(\tau) d\tau.$$

We have shown in the previous section that, in the case of implementations of the GRASP metaheuristic, the random variable X can be approximated by a Normal distribution $N(m^k, S^k)$ with average m^k and standard deviation S^k, whose probability density function and cumulative probability distribution are, respectively, $f_X^k(.)$ and $F_X^k(.)$.

However, we observe that although the Normal $N(m^k, S^k)$ already gives a good approximation to the random variable X, this estimation may be further improved. First, we suppose that $\underline{\ell}$ and $\overline{u}$ are lower and upper bounds, respectively, of the value obtained by the heuristic at each of its iterations. For all minimization test problems considered in this work, trivial bounds can be obtained, for example, by setting $\underline{\ell} = 0$ and $\overline{u}$ equal to the sum of all positive costs, as if the corresponding variables were set to 1 and the others to 0. Better, and

Table 8. Statistics for Normal fittings: p-median problem.

Instance	Iterations	Mean	Std. dev.	Skewness	Kurtosis
	50	1622.020000	57.844097	-0.179163	3.255009
	100	1620.890000	59.932611	-0.364414	3.304588
pmed10	500	1620.332000	63.484721	0.111186	3.142248
$p = 67$	1000	1619.075000	64.402076	0.074091	2.964164
	5000	1617.875200	63.499795	0.043152	2.951273
	10000	1618.415400	63.415181	0.087909	2.955408
	50	2170.500000	58.880642	-0.041262	1.949923
	100	2168.450000	65.313609	0.270892	2.693553
pmed15	500	2173.060000	65.881958	0.202400	2.828056
$p = 100$	1000	2173.484000	65.590272	0.129234	2.784433
	5000	2174.860000	64.639604	0.086450	2.940204
	10000	2175.651600	65.101495	0.096328	2.954639
	50	2277.780000	54.782220	0.330959	3.028905
	100	2279.610000	58.034799	0.360133	3.466265
pmed25	500	2271.546000	56.029848	0.219415	3.311486
$p = 167$	1000	2274.182000	56.915366	0.081878	3.068963
	5000	2276.305200	56.985195	-0.041096	3.108109
	10000	2277.151600	57.583524	-0.041570	3.073374
	50	2434.660000	57.809899	-0.130383	2.961249
	100	2446.560000	57.292464	-0.259531	2.667470
pmed30	500	2444.638000	56.109134	-0.189935	2.691882
$p = 200$	1000	2441.465000	57.265005	-0.053183	2.858399
	5000	2441.340400	54.941836	-0.013377	3.054188
	10000	2441.277700	54.978827	0.006407	3.066879

easily computable bounds reported in Table 13, can be obtained as follows for each problem instance:

- 2-path network design problem: set $\underline{\ell} = 0$ and $\overline{u}$ to the sum over all K demand pairs of the longest 2-path between each origin-destination pair.
- p-median problem: set $\underline{\ell}$ to the sum over all customers of the distance from each customer to its closest facility. Similarly, set $\overline{u}$ to the sum over all customers of the distance from each customer to its most distant facility.
- Quadratic assignment problem: bounds for the quadratic assignment problem have been collected from [4].
- Set k-covering problem: set $\underline{\ell}$ to the optimal value of the linear relaxation, whose value for each test instance is available in [18]. To compute $\overline{u}$, create a list associated with each row of the constraint matrix, formed by the k largest costs of the variables that cover this row. Next, build the set of variables formed by the union of the m individual lists and set $\overline{u}$ to the sum of the costs of these variables.

Since there may be no solution whose solution value is smaller than $\underline{\ell}$ or greater than $\overline{u}$, the Normal approximation of the random variable X can be

Table 9. Chi-square test for $1 - \alpha = 90\%$ confidence level: quadratic assignment problem.

Instance	Iterations	D	$\chi^2_{[1-\alpha;k-3]}$
tai30a	50	0.127260	17.275000
tai35a	50	0.213226	17.275000
tai40a	50	0.080164	17.275000
tai50a	50	0.075752	17.275000

Table 10. Statistics for Normal fittings: quadratic assignment problem.

Instance	Iterations	Mean	Std. dev.	Skewness	Kurtosis
	50	1907129.960000	15106.752548	-0.068782	2.562099
	100	1906149.760000	16779.060456	0.112965	3.028193
tai30a	500	1907924.412000	17663.997163	-0.005122	3.071314
	1000	1908292.204000	17241.785219	-0.058100	2.982074
	5000	1907542.144400	17484.852454	0.077001	2.978316
	10000	1907411.800800	17354.183037	0.044985	2.982363
	50	2544227.480000	24293.234765	0.260849	2.906127
	100	2541730.980000	21782.204670	0.374843	3.131055
tai35a	500	2541151.156000	20167.926106	0.098408	2.990821
	1000	2541735.064000	20809.432271	0.079094	3.073285
	5000	2541625.512800	20952.352020	0.057649	3.069945
	10000	2541104.138000	21191.460956	0.055055	3.089498
	50	3289069.880000	23456.147422	-0.043084	2.503491
	100	3287766.120000	25032.774604	-0.162022	2.395749
tai40a	500	3291146.756000	24690.208400	0.058502	2.745715
	1000	3290388.372000	23935.199864	0.020905	2.800432
	5000	3290638.646000	24404.060084	-0.006498	2.942513
	10000	3290795.196400	24493.025438	0.032977	2.980864
	50	5172456.560000	32336.075962	-0.080097	2.622378
	100	5175961.000000	31507.868139	-0.010372	2.451724
tai45a	500	5175471.476000	32557.814838	0.100482	2.705749
	1000	5175322.794000	32090.270871	0.027317	2.919791
	5000	5175315.985600	31907.168093	0.045656	2.968253
	10000	5174955.537200	31883.827203	0.048446	2.981676

Table 11. Chi-square test for $1 - \alpha = 90\%$ confidence level: set k-covering problem.

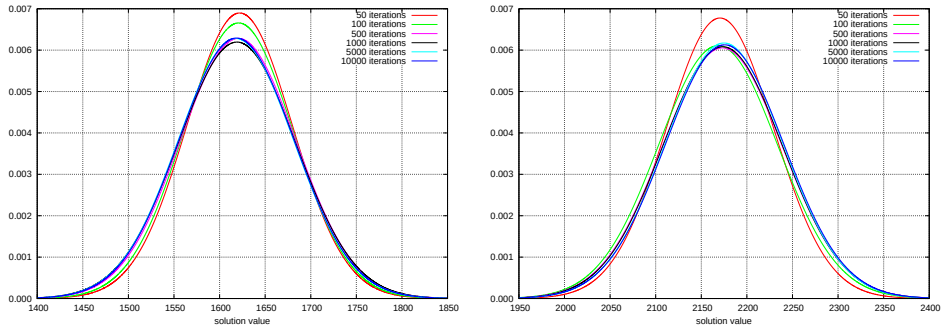
Instance	Iterations	D	$\chi^2_{[1-\alpha;k-3]}$
scp42	50	0.119939	17.275000
scp47	50	0.147765	17.275000
scp55	50	0.164476	17.275000
scpa2	50	0.092947	17.275000

Table 12. Statistics for Normal fittings: set k-covering problem.

Instance	Iterations	Mean	Std. dev.	Skewness	Kurtosis
scp42	50	1692.200000	122.108968	0.346549	2.485267
	100	1707.790000	138.210513	0.747575	3.727116
	500	1682.012000	129.047681	0.453641	3.395710
	1000	1677.603000	127.209156	0.424774	3.437712
	5000	1678.960800	129.853048	0.481598	3.395114
	10000	1678.848600	130.216475	0.478711	3.328128
scp47	50	1596.560000	133.676050	0.551858	2.722159
	100	1604.100000	132.584200	0.589321	3.260413
	500	1610.160000	135.050740	0.577331	3.677277
	1000	1603.936000	134.173380	0.528879	3.560794
	5000	1598.799600	133.743778	0.424347	3.147632
	10000	1600.043100	134.398671	0.428076	3.160244
scp55	50	1105.160000	149.749439	0.139493	2.671918
	100	1115.010000	154.429304	0.585166	4.036000
	500	1146.800000	157.817350	0.299096	3.059246
	1000	1146.450000	155.945348	0.332401	3.045766
	5000	1151.254200	164.425966	0.384420	3.099880
	10000	1154.463700	164.456147	0.397244	3.144651
scpa2	50	1383.340000	169.200072	0.092324	2.373417
	100	1395.780000	174.312282	0.085142	2.453907
	500	1392.900000	184.297428	0.070337	2.920209
	1000	1395.541000	183.278641	0.210396	2.873949
	5000	1398.725200	185.724296	0.326508	3.088006
	10000	1400.956100	186.330231	0.335011	3.087160

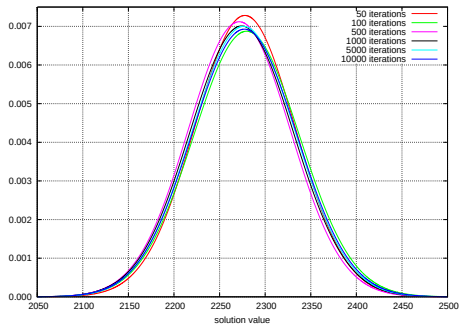
Table 13. Lower and upper bounds.

Problem	Instance	$\underline{\ell}$	$\overline{u}$
2-path	2pndp50	0	6244
	2pndp70	0	10353
	2pndp90	246	14621
	2pndp200	0	37314
p-median	pmed14	2968	5898
	pmed15	1729	9791
	pmed25	1828	16477
	pmed30	1989	19826
QAP	tai30a	1706855	8596620
	tai35a	2216627	11803330
	tai40a	2843274	15469120
	tai50a	4390920	24275700
set k-covering	scp42	1205	37132
	scp47	1115	36570
	scp55	550	38972
	scpa2	560	58550

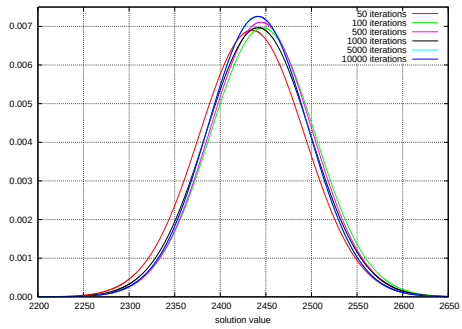


(a) Instance pmed10 with $p = 67$

(b) Instance pmed15 with $p = 100$



(c) Instance pmed25 with $p = 167$



(d) Instance pmed30 with $p = 200$

Fig. 3. Normal distributions: fitted probability density functions for the p-median problem.

improved to a doubly-truncated Normal distribution at $x = \underline{\ell}$ in the left and at $x = \overline{u}$ in the right. Let

$$f_X^k(x) = 1/(S^k\sqrt{2\pi}) \cdot e^{\frac{-(x-m^k)^2}{2S^{k\,2}}} \tag{5}$$

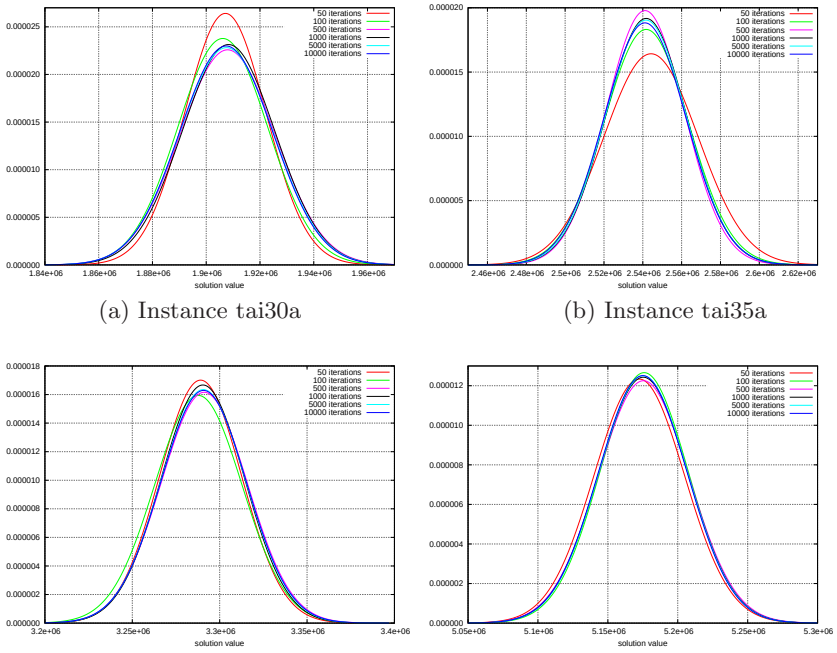
be the probability density function of the Normal approximation $N(m^k, S^k)$. The probability density function $\hat{f}_X^k(x)$ of the truncated Normal approximation is given below, depending on the existence of each of the lower and upper bounds.

– If there exists only a lower bound, then we have a left-truncated Normal at $x = \underline{\ell}$:

$$\hat{f}_X^k(x) = 1/(1 - F_X^k(\underline{\ell})) \cdot f_X^k(x), \quad \underline{\ell} \leq x. \tag{6}$$

– If there exists only an upper bound, then we have a right-truncated Normal at $x = \overline{u}$:

$$\hat{f}_X^k(x) = 1/F_X^k(\overline{u}) \cdot f_X^k(x), \quad x \leq \underline{u}. \tag{7}$$



(a) Instance tai30a (b) Instance tai35a

(c) Instance tai40a (d) Instance tai50a

Fig. 4. Normal distributions: quadratic assignment problem.

– If there exist both lower and upper bounds, then we have a doubly-truncated Normal at $x = \underline{\ell}$ in the left and at $x = \overline{u}$ in the right:

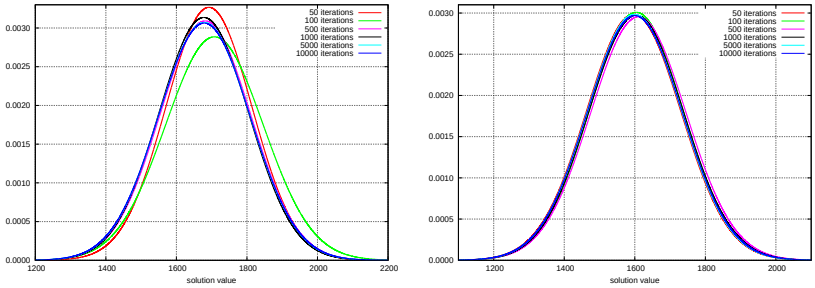
$$\hat{f}_X^k(x) = 1/[(1 - F_X^k(\underline{\ell})) \cdot F_X^k(\overline{u})] \cdot f_X^k(x), \quad \underline{\ell} \leq x \leq \underline{u}. \tag{8}$$

The three cases above are illustrated in Figure 6, where $\underline{\ell} = -1.5$ and $\overline{u} = 1.5$, which depicts the probability density functions of the truncated Normal distributions. We remark that the bounds $\underline{\ell}$ and $\overline{u}$ can eventually be further elaborated and improved. Furthermore, we notice that the tighter the bounds are, the better will be the doubly-truncated Normal approximation.

Therefore, if UB^k denotes the value of the best solution found along the k first iterations of the heuristic, the probability of finding a solution value smaller than or equal to UB^k in the next iteration can be better estimated by using the cumulative probability function of the truncated Normal distribution, which is given by

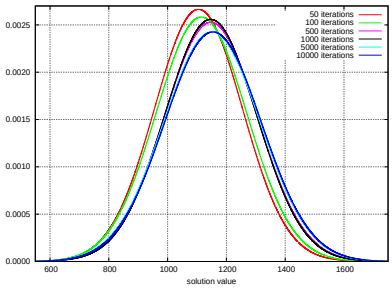
$$\hat{F}_X^k(UB^k) = \int_{\underline{\ell}}^{UB^k} \hat{f}_X^k(\tau) d\tau. \tag{9}$$

We consider the stopping rule proposed in Section 2: for any given threshold β, stop the iterations of the heuristic whenever $\hat{F}_X^k(UB^k) \leq \beta$. The iterations will be interrupted as soon as the probability of finding in the next iteration a

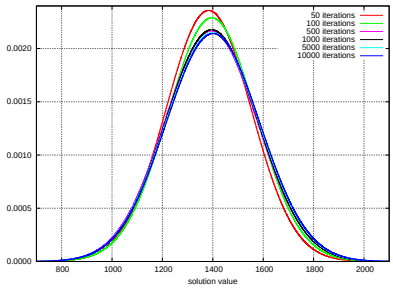


(a) Instance scp42

(b) Instance scp47



(c) Instance scp55



(d) Instance scpa2

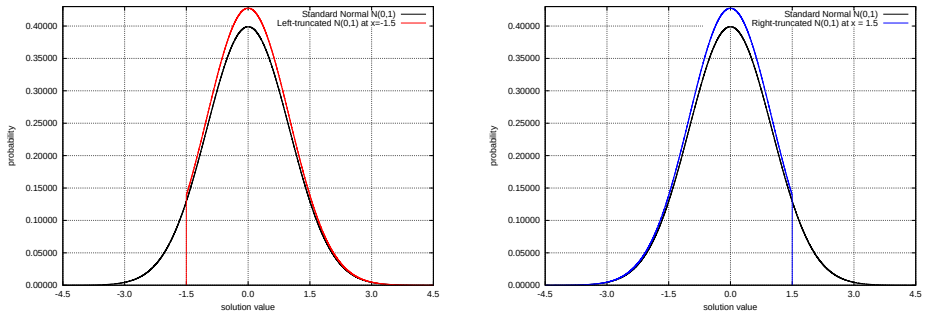
Fig. 5. Normal distributions: set k-covering problem.

solution at least as good as the current best becomes smaller than or equal to the threshold β.

This strategy will be validated for some GRASP implementations already considered in Section 4, using the fact that the probability $\hat{F}_X^k(UB^k)$ can be used to give an online estimation of the number of solutions obtained in forthcoming iterations that might be at least as good as the best known solution at the current iteration.

To validate and assess the effectiveness of the proposed stopping rule based on the estimation of $F_X^k(UB^k)$, we have devised and performed the following experiment for each problem and test instance already considered in Section 4. For each value of the threshold β, we run the GRASP heuristic until $\hat{F}_X^k(UB^k)$ becomes less than or equal to β. Let us denote by $\overline{k}$ the iteration counter where this condition is met and by $\overline{UB}$ the best known solution value at this time. At this point, we may estimate by $\hat{N}^{\leq} = \lfloor N \cdot \hat{F}_X^{\overline{k}}(\overline{UB}) \rfloor$ the number of solutions whose value will be at least as good as $\overline{UB}$ if N additional GRASP iterations are performed. We empirically set $N = 1000000$. Next, we perform N additional iterations and we count the number $N^{\leq}$ of solutions whose value is smaller than or equal to $\hat{F}_X^{\overline{k}}(\overline{UB})$.

(a) Left-truncated Normal (0,1) at $\underline{\ell} = -1.5$ (b) Right-truncated Normal (0,1) at $\overline{u} = 1.5$



(c) Doubly-truncated Normal (0,1) at $\underline{\ell} = -1.5$ in the left and $\overline{u} = 1.5$ in the right

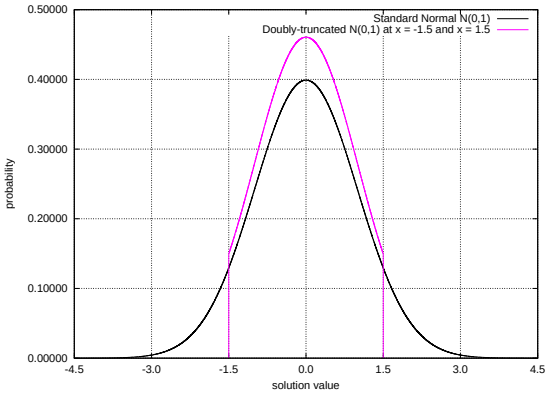


Fig. 6. Probability density functions of the truncated Normal distributions.

The computational results displayed in Tables 14 and 15 show that $\hat{N}^{\leq}$ is a good estimation for the number $N^{\leq}$ of solutions found after N additional iterations whose value is smaller than or equal to the best solution value at the time the algorithm would stop for each threshold value β. Using the threshold $\beta = 10^{-1}$ is not appropriate, since at this point we are usually still very far from the optimal value and $\hat{F}_X^{\overline{k}}(\overline{UB})$ does not give a good estimate of the probability of finding a solution at least as good as the best known at this time. We also observe that for both the quadratic assignment and the set k-covering problems, whose results are depicted in Table 15, it has not been possible to reach a solution satisfying the threshold $\beta = 10^{-4}$ for any of their instances.

Therefore, the probability $\hat{F}_X^{\overline{k}}(UB^k)$ may be used to estimate the number of iterations that must be performed by the algorithm to find a new solution at least as good as the currently best one. The threshold β used to implement the stopping criterion may either be fixed a priori as a parameter or iteratively computed. In the last case, since the user is able to account for the average time taken by each GRASP iteration, this threshold can be determined online

Table 14. 2-path network design and p-median problems: stopping criterion vs. estimated and counted number of solutions at least as good as the incumbent after $N = 1000000$ additional iterations.

Problem	Instance	Threshold	Iteration	Probability Estimation		Count
		β	$\overline{k}$	$\hat{F}_X^{\overline{k}}(\overline{UB})$	$\hat{N}^{\leq}$	$N^{\leq}$
2pndp	2pndp50	10^{-1}	3	0.079046	79046	1843
		10^{-2}	25	0.009970	9970	1843
		10^{-3}	318	0.000757	757	738
		10^{-4}	4778	0.000001	1	0
		10^{-5}	4778	0.000001	1	0
	2pndp70	10^{-1}	3	0.078669	78669	148028
		10^{-2}	102	0.008923	8923	9537
		10^{-3}	1870	0.000643	643	465
		10^{-4}	32771	0.000036	36	26
		10^{-5}	49633	0.000005	5	4
	2pndp90	10^{-1}	4	0.085933	85933	2066
		10^{-2}	41	0.009257	9257	2066
		10^{-3}	722	0.000326	326	190
		10^{-4}	5209	0.000015	15	7
		10^{-5}	270618	0.000001	1	0
	2pndp200	10^{-1}	23	0.028989	28989	32151
		10^{-2}	232	0.001821	1821	1539
		10^{-3}	556	0.000566	566	503
		10^{-4}	5377	0.000100	100	95
		10^{-5}	77448	0.000001	1	1
p-median	pmed14	10^{-1}	4	0.060647	60647	79535
		10^{-2}	21	0.008542	8542	7507
		10^{-3}	608	0.000786	787	215
		10^{-4}	217169	6.93×10^{-5}	69	5
		10^{-5}	437422	5.55×10^{-6}	6	0
	pmed15	10^{-1}	5	0.069694	69694	117054
		10^{-2}	56	0.009214	9214	16968
		10^{-3}	3533	0.000626	626	311
		10^{-4}	10264	6.36×10^{-5}	63	26
		10^{-5}	235853	9.99×10^{-6}	10	3
	pmed25	10^{-1}	3	0.089011	89011	12428
		10^{-2}	34	0.009309	9309	4176
		10^{-3}	1060	0.000998	998	1232
		10^{-4}	2760	2.82×10^{-5}	28	38
		10^{-5}	81382	4.84×10^{-6}	5	4
	pmed30	10^{-1}	4	0.089941	89941	120598
		10^{-2}	40	0.004635	4635	1426
		10^{-3}	320	0.000992	992	1133
		10^{-4}	29142	2.86×10^{-6}	3	1
		10^{-5}	29142	2.86×10^{-6}	3	1

Table 15. Quadratic assignment and set k-covering problems: stopping criterion vs. estimated and counted number of solutions at least as good as the incumbent after $N = 1000000$ additional iterations.

Problem	Instance	Threshold β	Iteration $\overline{k}$	Probability $\hat{F}_X^{\overline{k}}(\overline{UB})$	Estimation $\hat{N}^{\leq}$	Count $N^{\leq}$
Quadratic assignment	tai30a	10^{-1}	4	0.045151	45151	94107
		10^{-2}	82	0.003773	3773	3759
		10^{-3}	755	0.000996	996	1031
	tai35a	10^{-1}	10	0.098051	98051	39090
		10^{-2}	57	0.009754	9754	5389
		10^{-3}	1440	0.000999	1000	1152
	tai40a	10^{-1}	4	0.064517	64517	15748
		10^{-2}	21	0.009094	9094	15748
		10^{-3}	1212	0.000121	121	111
	tai50a	10^{-1}	3	0.078783	78783	281757
		10^{-2}	35	0.009633	9633	10214
		10^{-3}	1737	0.000477	477	373
set k-covering	scp42	10^{-1}	6	0.090414	90414	111059
		10^{-2}	78	0.009679	9679	5944
		10^{-3}	283340	0.000457	457	0
	scp47	10^{-1}	7	0.088826	88826	125995
		10^{-2}	142	0.008171	8171	1897
		10^{-3}	58788	0.000359	359	0
	scp55	10^{-1}	4	0.058515	58515	36119
		10^{-2}	221	0.004957	4957	604
		10^{-3}	137239	0.000999	1000	7
	scpa2	10^{-1}	3	0.078973	78973	160036
		10^{-2}	155	0.009952	9952	8496
		10^{-3}	359	0.000199	199	0

so as to limit the computation time when the probability of finding improving solutions becomes very small and the time needed to find improving solutions could become very large.

The pseudo-code in Figure 7 extends the previous template of a GRASP procedure for minimization, implementing the termination rule based on stopping the GRASP iterations whenever the probability $\hat{F}_X^k(UB^k)$ of improving the best known solution value gets smaller than or equal to β. Lines 11 and 12 update the sample $f_1, \ldots, f_k$ and the best known solution value $UB^k = f^*$ at each iteration k. The mean m^k and the standard deviation s^k of the fitted Normal distribution in iteration k are estimated in line 13. The probability of finding a solution whose value is better than the currently best known solution value is computed in line 14 and used in the stopping criterion implemented in line 15.

Another promising avenue of research consists in investigating stopping rules based on estimating the number of iterations needed to improve the value of the best solution found by different amounts. Figure 8 displays the results obtained

```
procedure GRASP(β, Seed)
1.  Set f* ← ∞;
2.  Set k ← 0;
3.  repeat
4.      x ← GreedyRandomizedAlgorithm(Seed);
5.      x ← LocalSearch(x);
6.      if f(x) < f* then
7.          x* ← x;
8.          f* ← f(x);
9.      end;
10.     k ← k + 1;
11.     f_k ← f(x);
12.     UB^k ← f*;
13.     Update the average m^k and the standard deviation S^k of f_1, ..., f_k;
14.     Compute the estimate F̂_X^k(f*) = F̂_X^k(UB^k) = ∫_{-∞}^{f*} f̂_X^k(τ)dτ;
15. until F̂_X^k(f*) < β;
16. return x*, f*;
end.
```

Fig. 7. Template of a GRASP heuristic for minimization with the probabilistic stopping criterion.

for instance 2pndp90 of the 2-path network design problem with the threshold β set at 10^{-3}. For each percent improvement that is sought in the objective function, the figure plots the expected additional number of iterations needed to find a solution that improves the best known solution value by this amount. For instance, we may see that the expected number of iterations needed to improve by 0.5% the best solution value found at termination is 12131. If one seeks a percent improvement of 1%, then the expected number of additional iterations to be performed increases to 54153.

6 Concluding remarks

The main drawback of most metaheuristics is the absence of effective stopping criteria. Most implementations of such algorithms stop after performing a given maximum number of iterations or a given maximum number of consecutive iterations without improvement in the best known solution value, or after the stabilization of a population of elite solutions. In some cases, the algorithm may perform an exaggerated and non-necessary number of iterations. In other situations, the algorithm may stop just before the iteration that could find a better, or even optimal, solution.

In this paper, we proposed effective probabilistic stopping rules for randomized metaheuristics. We first showed experimentally that the solution values obtained by a randomized heuristic such as GRASP fit a Normal distribution. Next, we used this Normal approximation to estimate the probability of finding

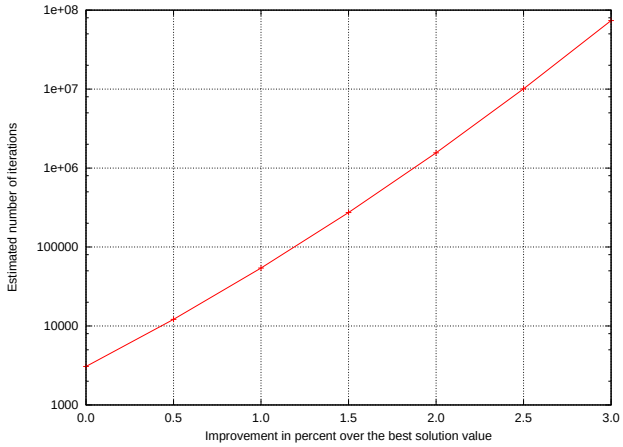


Fig. 8. Estimated number of additional iterations needed to improve the best solution value.

in the next iteration a solution at least as good as the currently best known solution. This probability also gives an estimate of the expected number of iterations that must be performed by the algorithm to find a new solution at least as good as the currently best one. The stopping rule is based on the trade-off between solution quality and the time (or the number of additional iterations) needed to improve the best known solution.

Since the average time consumed by each GRASP iteration is known, another promising avenue of research consists in investigating stopping rules based on estimating the amount of time needed to improve the best solution value for some amount and evaluating the trade-off between these two quantities. Figure 8 illustrated this issue, displaying the expected additional number of iterations needed to improve the best solution value by different percent values.

The robustness of the proposed strategy was illustrated and validated by a thorough computational study reporting results obtained with GRASP implementations to four combinatorial optimization problems. We are investigating extensions and applications of the proposed approach to other metaheuristics that rely on randomization to sample the search space, such as simulated annealing, VNS, genetic algorithms and, in particular, to implementations of GRASP with path-relinking.

References

1. V. Bartkutė, G. Felinskas, and L. Sakalauskas. Optimality testing in stochastic and heuristic algorithms. Technical report, Vilnius Gediminas Technical University, 2006.

2. V. Bartkutė and L. Sakalauskas. Statistical inferences for termination of markov type random search algorithms. *Journal of Optimization Theory and Applications*, 141:475–493, 2009.
3. C.G.E. Boender and A.H.G. Rinnooy Kan. Bayesian stopping rules for multistart global optimization methods. *Mathematical Programming*, 37:59–80, 1987.
4. R.E. Burkard, E. Çela, S.E. Karisch, and F. Rendl. QAPLIB - A quadratic assignment problem library, 2011. Online reference at http://www.seas.upenn.edu/qaplib/, last visited on January 20, 2013.
5. G. Dahl and B. Johannessen. The 2-path network problem. *Networks*, 43:190–199, 2004.
6. C. Dorea. Stopping rules for a random optimization method. *SIAM Journal on Control and Optimization*, 28:841–850, 1990.
7. C. Duin and S. Voss. The Pilot method: A strategy for heuristic repetition with application to the Steiner problem in graphs. *Networks*, 34:181–191, 1999.
8. M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions*. Wiley, New York, 3rd edition, 2000.
9. T.A. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.
10. P. Festa and M.G.C. Resende. An annotated bibliography of GRASP, Part I: Algorithms. *International Transactions in Operational Research*, 16:1–24, 2009.
11. P. Festa and M.G.C. Resende. An annotated bibliography of GRASP, Part II: Applications. *International Transactions in Operational Research*, 16:131–172, 2009.
12. W.E. Hart. Sequential stopping rules for random optimization methods with applications to multistart local search. *SIAM Journal on Optimization*, 9:270–290, 1998.
13. R. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley, New York, 1991.
14. O. Kariv and L. Hakimi. An algorithmic approach to nework location problems, Part II: The p-medians. *SIAM Journal of Applied Mathematics*, 37:539–560, 1979.
15. T.C. Koopmans and M.J. Beckmann. Assignment problems and the location of economic activities. *Econometrica*, 25:53–76, 1957.
16. C.A. Oliveira, P.M. Pardalos, and M.G.C. Resende. GRASP with path-relinking for the quadratic assignment problem. *Lecture Notes in Computer Science*, 3059:356–368, 2004.
17. C. Orsenigo and C. Vercellis. Bayesian stopping rules for greedy randomized procedures. *Journal of Global Optimization*, 36:365–377, 2006.
18. L.S. Pessôa, M.G.C. Resende, and C.C. Ribeiro. A hybrid Lagrangean heuristic with GRASP and path-relinking for set k-covering. *Computers & Operations Research*, to appear. Accepted in 2012.
19. M.R. Rao. Cluster analysis and mathematical programming. *Journal of the American Statistical Association*, 66:622–626, 1971.
20. R.L. Rardin, R., and Uzsoy. Experimental evaluation of heuristic optimization algorithms: A tutorial. *Journal of Heuristics*, 7:261–304, 2001.
21. M.G.C. Resende and C.C. Ribeiro. A GRASP with path-relinking for private virtual circuit routing. *Networks*, 41:104–114, 2003.
22. M.G.C. Resende and C.C. Ribeiro. GRASP with path-relinking: Recent advances and applications. In T. Ibaraki, K. Nonobe, and M. Yagiura, editors, *Metaheuristics: Progress as Real Problem Solvers*, pages 29–63. Springer, 2005.

23. M.G.C. Resende and C.C. Ribeiro. Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. In M. Gendreau and J.-Y. Potvin, editors, *Handbook of Metaheuristics*, pages 283–319. Springer, 2nd edition, 2010.
24. M.G.C. Resende and C.C. Ribeiro. GRASP. In E.K. Burke and G. Kendall, editors, *Search Methodologies*. Springer, 2nd edition, to appear. Accepted in 2011.
25. M.G.C. Resende and R.F. Werneck. A hybrid heuristc for the p-median problem. *Journal of Heuristics*, 10:59–88, 2004.
26. C.C. Ribeiro and I. Rosseti. A parallel GRASP heuristic for the 2-path network design problem. *Lecture Notes in Computer Science*, 2400:922–926, 2002.
27. C.C. Ribeiro and I. Rosseti. Efficient parallel cooperative implementations of GRASP heuristics. *Parallel Computing*, 33:21–35, 2007.
28. S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the Association of Computing Machinery*, 23:555–565, 1976.
29. F.S. Serifoglu and G. Ulusoy. Multiprocessor task scheduling in multistage hybrid flow-shops: A genetic algorithm approach. *Journal of the Operational Research Society*, 55:504–512, 2004.
30. B.C. Tansel, R.L. Francis, and T.J. Lowe. Location on networks: A survey. *Management Science*, 29:482–511, 1983.
31. V.V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Berlin, 2004.
32. H.D. Vinod. Integer programming and the theory of groups. *Journal of the American Statistical Association*, 64:506–519, 1969.
33. S. Voss, A. Fink, and C. Duin. Looking ahead with the Pilot method. *Annals of Operations Research*, 136:285–302, 2005.