Effective probabilistic stopping rules for randomized metaheuristics: GRASP implementations

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 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 probabilistic stopping rules for randomized metaheuristics such as GRASP and VNS. We first show experimentally that the solution values obtained by GRASP fit a Normal distribution. Next, we use this approximation to obtain an online estimation of the number of solutions that might be at least as good as the best known at the time of the current iteration. This estimation is used to implement effective stopping rules based on the trade off between solution quality and the time needed to find a solution that might improve the best found to date. This strategy is illustrated and validated by a computational study reporting results obtained with some GRASP heuristics.

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, 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 very 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 as 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 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 often the absence of effective stopping criteria. Most of their implementations 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 optimal 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.

Bayesian stopping rules proposed in the past were not followed by enough computational results to sufficiently validate their effectiveness or to give evidence of their efficiency. 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 reliability and computational effort.

In Dorea [5] a stochastic algorithm for estimating the global minimum of a function is described and two types of stopping rules are derived. 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. These stopping rules use an estimate of the probability measure of the ϵ-close points to terminate these 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 can also be applied to multistart local search and stratified multistart local search. Numerical results on a standard test set show that these stopping rules can perform as well as Bayesian stopping rules for multistart local search. The authors claimed an improvement on the results in [5].

Orsenigo and Vercellis [15] 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.

In another context, stopping rules have also been discussed in [6, 28]. The statistical estimation of optimal values for combinatorial optimization problems as a way to evaluate the performance of heuristics was also addressed in [16, 25].

We propose effective probabilistic stopping rules for randomized metaheuristics. In the next section, we give a template for a GRASP heuristic and we describe the optimization problems and test instances that have been used in our computational experiments. In Section 3, 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. In Section 4, we first show how this Normal approximation can be used to give an online estimation of the number of solutions that might be at least as good as the currently best known solution. This estimation is used to implement effective stopping rules based on 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 some GRASP implementations. Concluding remarks are made in the last section.

2 GRASP and experimental environment

We consider in what follows a general 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 (which stands for *greedy randomized adaptive search procedures*) [8], is a multi-start 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 its neighborhood until a local minimum is found. The best overall solution is kept as the result; see [18, 21, 19, 20].

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 begin; x* ← x; f* ← f(x); end;
6.      f_k ← f(x);
7.  end;
8.  return x*;
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 (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 [9–11].

Two combinatorial optimization problems have been used in the experiments reported in this paper: the 2-path network design problem and the p-median problem. They are both described below.

Given a connected undirected graph $G = (V, E)$ with non-negative weights associated with its edges, together with a set of formed by K pairs of origin-destination nodes, the 2-path network design problem consists of 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 [4]. The GRASP heuristic that has been used in the computational experiments was firstly presented in [23, 24]. Data of the four instances involved in the experiments are summarized in Table 1.

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

Instance	$\|V\|$	$\|E\|$	K
2pndp50	50	1,225	500
2pndp70	70	2,415	700
2pndp90	90	4,005	900
2pndp200	200	19,900	2000

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 of 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 in location [26] and clustering [17, 27]. The GRASP heuristic that has been used in the computational experiments with the p median problem was firstly presented in [22]. Data of the four instances involved in the experiments are summarized in Table 2.

Table 2. Test instances for 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 Normal approximation for GRASP iterations

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 consists of computing

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

It can be shown that, under the null hypothesis, D follows a chi-square distribution with $k-1$ degrees of freedom. Since the mean and the standard deviation are unknown, they should be estimated from the sample. As a consequence, two degrees of freedom are lost to compensate for that. The null hypothesis that the observations come from the specified distribution cannot be rejected at a level of significance α if D is less than $\chi^2_{[1-\alpha;k-3]}$.

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 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 Section 2. In all experiments, we used $\alpha = 0.1$ and $k = 14$, corresponding to a histogram with the intervals $(-\infty, -3)$, $[-3.0, -2.5)$, $[-2.5, -2.0)$, $[-2.0, -1.5)$, $[-1.5, -1.0)$, $[-1, -0.5)$, $[-0.5, 0.0)$, $[0.0, 0.5)$, $[0.5, 1.0)$, $[1.0, 1.5)$, $[1.5, 2.0)$, $[2.0, 2.5)$, $[2.5, 3.0)$, and $[3.0, \infty)$. For each instance, we illustrate the Normal fittings after $N = 50$, 100, 500, 1000, 5000, and 10000 iterations.

Table 3 reports on 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 heuristic fit very close a Normal distribution.

To further illustrate that this close fitting is maintained when the number of iterations increase, we present in Table 4 the main statistics for each instance and for increasing values of the number $N = 50$, 100, 500, 1000, 5000, and 10000 of iterations: mean, standard deviation, skewness (η_3), and kurtosis (η_4). The skewness and the kurtosis are computed as follows [7]:

$$\eta_3 = \frac{\sqrt{N} \cdot \sum_{i=1}^{N} (f_i - m)^3}{[\sum_{i=1}^{N} (f_i - m)^2]^{3/2}} \qquad \text{and} \qquad \eta_4 = \frac{N \cdot \sum_{i=1}^{N} (f_i - m)^4}{[\sum_{i=1}^{N} (f_i - m)^2]^2}.$$

Table 3. Chi-square test for 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

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 value consistently converges very quickly to a steady-state value when 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.

Table 4. 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

Figure 3 displays the Normal distributions fitted for the three first instances for each number of iterations. Together with the above statistics, 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 5 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 heuristic for this problem also fit very close a Normal distribution.

Table 5. Chi-square test for 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

Table 6 gives the same 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 iterations. As for the previous problem, we notice that the mean value consistently converges very quickly to a steady-state value when the number of iterations increases. Furthermore, the mean 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 4 displays the Normal distributions fitted for the three first instances for each number of iterations. Once again, these results illustrate the robustness of the Normal fittings to the solution values obtained along the iterations of the GRASP heuristic for the p-median problem.

Similar experiments have been performed for other problems and test instances, such as the quadratic assignment and the set k-covering problems, with results of the same caliber. We conclude this section by observing that the null hypothesis cannot be rejected with 90% of confidence. Therefore, we may approximate the solution values obtained by a GRASP heuristic by a Normal distribution that can be progressively fitted and improved as more iterations are performed. This approximation will be used in the next section to establish and validate a probabilistic stopping rule for GRASP heuristics.

4 Probabilistic stopping rule

We show in this section that the Normal distribution fitted to the solution values obtained along the GRASP iterations can be used to give an online estimation of the number of solutions that might be at least as good as the best known solution at the time of the current iteration. This estimation is used to implement an effective stopping rule based on the time needed to find a solution that might improve the incumbent. The robustness of the proposed strategy is illustrated and validated by a computational study reporting the results obtained.

We denote by X the random variable representing the value of the local minimum obtained at each iteration. We recall that $f_1, \ldots, f_k$ is a sample formed by the solution values obtained along the k first iterations. Let m^k and S^k be, respectively, the estimated mean and standard deviation of $f_1, \ldots, f_k$. As already established, we assume that X fits 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 $F_X^k(.)$.

Table 6. 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

Let UB^k be the value of the best solution found along the k first iterations. Therefore, the probability of finding a solution value smaller than or equal to UB^k in the next iteration can be estimated by $F_X^k(UB^k) = \int_{-\infty}^{UB^k} f_X^k(\tau)d\tau$. This estimation is periodically updated or whenever the best solution value improves.

We propose the following stopping rule: for any given threshold β, stop the GRASP iterations whenever $F_X^k(UB^k) \leq \beta$. In other words, the iterations will be interrupted whenever the probability of finding a solution at least as good as the current best becomes less than or equal to β.

To assess the effectiveness of this stopping rule, we have devised and performed the following experiment for each problem and test instance considered in Section 3. For each value of the threshold $\beta = 10^{-3}, 10^{-4}$, and 10^{-5}, we run the GRASP heuristic until $F_X^k(UB^k)$ becomes less than or equal to β. Let us denote by $\overline{k}$ the iteration counter when this condition is met and by $\overline{UB}$ the best known solution at this time. At this point, we may estimate by $\hat{N}^{\leq} = \lfloor N \cdot F_X^{\overline{k}}(\overline{UB}) \rfloor$ the number of solutions whose value will be at least as good as $\overline{UB}$ if N additional iterations are performed. We empirically set $N = 1,000,000$. Next, we perform N additional iterations and we count the number $N^{\leq}$ of solutions whose value is less than or equal to $F_X^{\overline{k}}(\overline{UB})$.

The computational results displayed in Table 7 show that $\hat{N}^{\leq} = \lfloor N \cdot F_X^{\overline{k}}(\overline{UB}) \rfloor$ is a good estimation for the number $N^{\leq}$ of solutions that might be found after N additional iterations whose value is less than or equal to the best value at the time the algorithm would stop for each threshold value β. The probability $F_X^k(UB^k)$ may be used to estimate the number of iterations that must be performed by the algorithm to find a new solution 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 bound the computation time when the probability of finding improving solutions becomes very small.

The pseudo-code in Figure 2 extends the previous template of a GRASP procedure for minimization, implementing the termination rule based on stopping the GRASP iterations whenever the probability $F_X^k(UB^k)$ of improving the best known solution value gets smaller than or equal to β. Lines 8 and 9 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 10. The probability of finding a solution whose value is better than the currently best known solution value is computed in line 11 and used in the stopping criterion implemented in line 12.

The threshold β used to implement the stopping criterion may either be a fixed parameter or iteratively computed. In the last case, it will be computed considering the probability of finding an improving solution (or, alternatively, the estimated number of iterations to find an improving solution) and the average computation time per iteration.

We also notice that 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 probabilistically improve the best solution found by each percent point.

5 Concluding remarks

The main drawback of most metaheuristics is often the absence of effective stopping criteria. Most of their implementations stop after performing a given maximum number of iterations or a given maximum number of consecutive iterations without improvement in the best solution value, or after the stabilization of a population of solutions or of a set of elite solutions found along the search. 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.

Bayesian stopping rules proposed in the past were not followed by enough computational results to sufficiently validate their effectiveness or to give evidence of their efficiency. In this paper, we proposed effective probabilistic stopping rules for randomized metaheuristics.

We first showed experimentally that the solution values obtained by a GRASP heuristic fit a Normal distribution. Next, we used the above Normal approxima-

Table 7. Stopping criterion vs. estimated and counted number of solutions at least as good as the incumbent after $N = 1,000,000$ additional iterations.

Problem	Instance	Threshold β	Probability $F_X^{\overline{k}}(\overline{UB})$	Estimation $\hat{N}^{\leq}$	Count $N^{\leq}$
2-path	2pndp50	10^{-3}	0.000701657	701	738
		10^{-4}	0.000001326	1	0
		10^{-5}	0.000001326	1	0
	2pndp70	10^{-3}	0.000655383	655	465
		10^{-4}	0.000036147	36	26
		10^{-5}	0.000005363	5	4
	2pndp90	10^{-3}	0.000322033	322	190
		10^{-4}	0.000014878	14	7
		10^{-5}	0.000001265	1	0
	2pndp200	10^{-3}	0.000525545	525	503
		10^{-4}	0.000098792	98	95
		10^{-5}	0.000000853	0	1
p-median	pmed10	10^{-3}	0.000181323	181	47
		10^{-4}	0.000088594	88	16
		10^{-5}	0.000007667	7	0
	pmed15	10^{-3}	0.000331692	331	123
		10^{-4}	0.000028636	28	7
		10^{-5}	0.000005236	5	0
	pmed25	10^{-3}	0.000293215	293	211
		10^{-4}	0.000053319	53	31
		10^{-5}	0.000008891	8	3
	pmed30	10^{-3}	0.000569064	569	310
		10^{-4}	0.000028080	28	8
		10^{-5}	0.000000790	0	0

tion to estimate the probability of finding a solution at least as good as the currently best known solution at any iteration. With this probability, we have been able 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.

We proposed a stopping rule based on the trade off between this estimation and the time needed to find a solution that might improve the current best one. GRASP iterations will be interrupted whenever the probability of finding a solution at least as good as the current best becomes smaller than or equal a certain threshold.

The robustness of this strategy was illustrated and validated by a computational study reporting results obtained with GRASP implementations for two combinatorial optimization problems. Similar results already obtained for other problems, such as the quadratic assignment and the set k-covering problems, will be reported elsewhere in an extended version of this work.

Since the average time consumed by each GRASP iteration is known, another promising avenue of research consists in investigating stopping rules based

```
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 begin; x* ← x; f* ← f(x); end;
7.      k ← k + 1;
8.      f_k ← f(x);
9.      UB^k ← f*;
10.     Update the average m^k and the standard deviation S^k of f_1, ..., f_k;
11.     Compute the estimate F_X^k(UB^k) = F_X^k(f*) = ∫_{-∞}^{f*} f_X^k(τ)dτ;
12. until F_X^k(f*) < β;
13. return x*;
end.
```

Fig. 2. Template of a GRASP heuristic for minimization with the probabilistic stopping criterion.

on estimating the amount of time needed to probabilistically improve the best solution found by each percent point. We notice that the approach proposed in this paper can be extended and applied not only to GRASP, but also to other metaheuristics that rely on randomization to sample the search space.

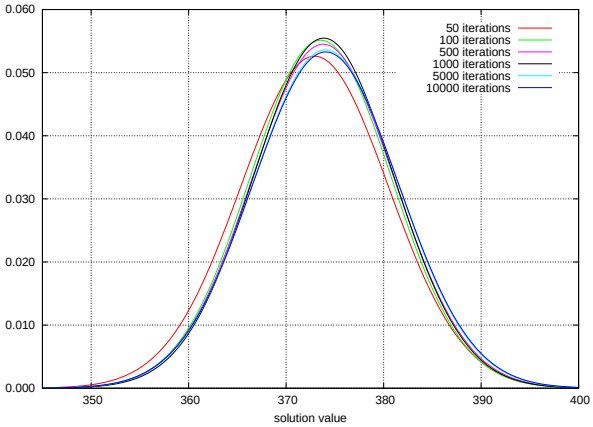
Acknowledgments: The authors are grateful to M.G.C. Resende and R. Werneck for making available their GRASP code for solving the p-median problem.

References

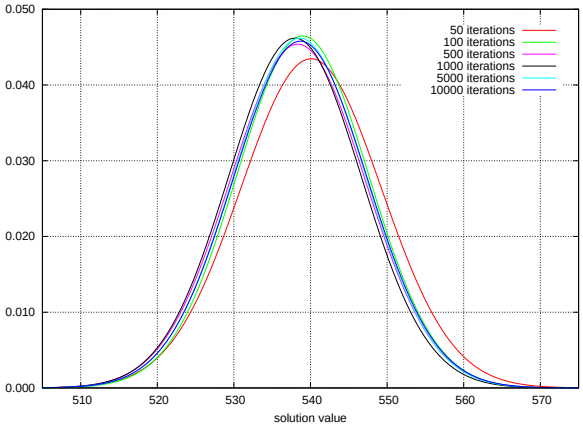
1. V. Bartkutė, G. Felinskas, and L. Sakalauskas. Optimality testing in stochastic and heuristic algorithms. Technical report, Vilnius Gediminas Technical University, 2006. 4–10.
2. V. Bartkutė and L. Sakalauskas. Statistical inferences for termination of markov type random search algorithms. *Journal 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. G. Dahl and B. Johannessen. The 2-path network problem. *Networks*, 43:190–199, 2004.
5. C. Dorea. Stopping rules for a random optimization method. *SIAM Journal on Control and Optimization*, 28:841–850, 1990.
6. 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.
7. M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions*. Wiley, New York, 3rd edition, 2000.

8. T.A. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.
9. P. Festa and M.G.C. Resende. GRASP: An annotated bibliography. In C.C. Ribeiro and P. Hansen, editors, *Essays and surveys in metaheuristics*, pages 325–367. Kluwer Academic Publishers, 2002.
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. C. Orsenigo and C. Vercellis. Bayesian stopping rules for greedy randomized procedures. *Journal of Global Optimization*, 36:365–377, 2006.
16. R.L. Rardin R. and Uzsoy. Experimental evaluation of heuristic optimization algorithms: A tutorial. *Journal of Heuristics*, 7:261–304, 2001.
17. M.R. Rao. Cluster analysis and mathematical programming. *Journal of the American Statistical Association*, 66:622–626, 1971.
18. M.G.C. Resende and C.C. Ribeiro. GRASP. In E.K. Burke and G. Kendall, editors, *Search Methodologies*. Springer, 2nd edition. To appear.
19. M.G.C. Resende and C.C. Ribeiro. A GRASP with path-relinking for private virtual circuit routing. *Networks*, 41:104–114, 2003.
20. 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.
21. 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.
22. M.G.C. Resende and R.F. Werneck. A hybrid heuristc for the p-median problem. *Journal of Heuristics*, 10:59–88, 2004.
23. 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.
24. C.C. Ribeiro and I. Rosseti. Efficient parallel cooperative implementations of GRASP heuristics. *Parallel Computing*, 33:21–35, 2007.
25. 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.
26. B.C. Tansel, R.L. Francis, and T.J. Lowe. Location on networks: A survey. *Management Science*, 29:482–511, 1983.
27. H.D. Vinod. Integer programming and the theory of groups. *Journal of the American Statistical Association*, 64:506–519, 1969.
28. S. Voss, A. Fink, and C. Duin. Looking ahead with the Pilot method. *Annals of Operations Research*, 136:285–302, 2005.

(a) 50-node instance



(b) 70-node instance



(c) 90-node instance

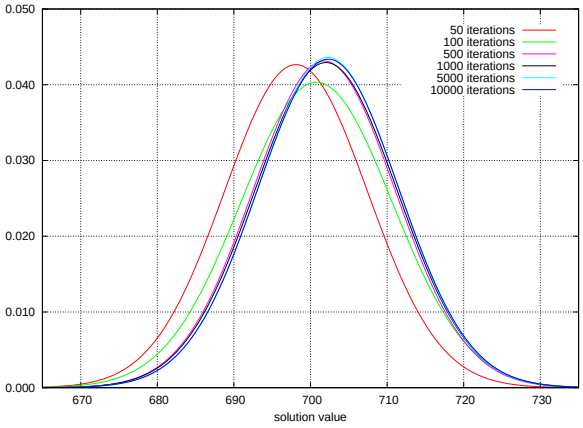
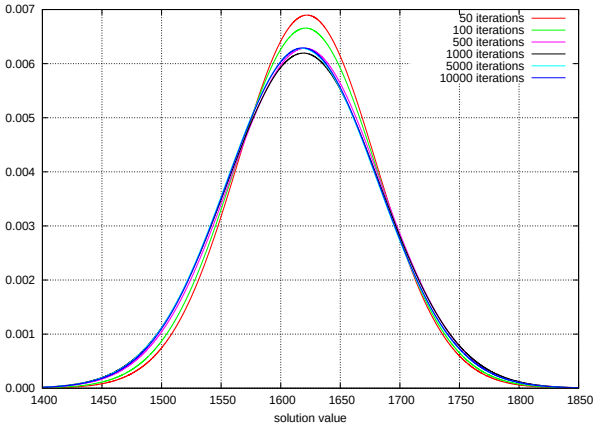
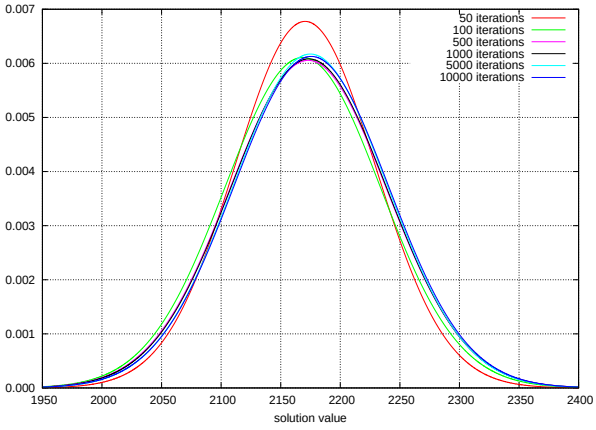


Fig. 3. Fitted probability density functions for the 2-path network design problem.

(a) Instance pmed10 with $p = 67$



(b) Instance pmed15 with $p = 100$



(c) Instance pmed25 with $p = 167$

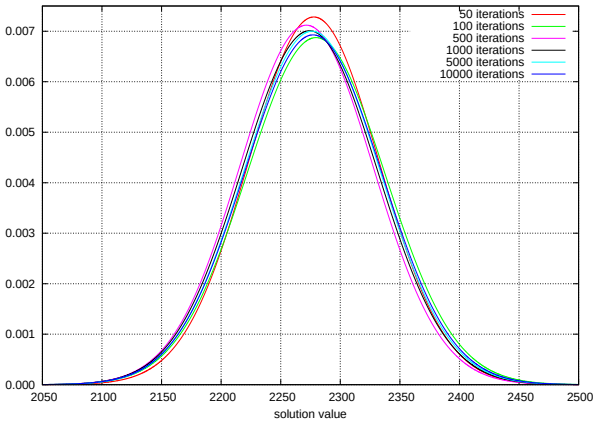


Fig. 4. Fitted probability density functions for the p-median problem.