Exploiting run time distributions to compare sequential and parallel stochastic local search algorithms

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

[1] Department of Computer Science, Universidade Federal Fluminense,
Rua Passo da Pátria 156, Niterói, RJ 24210-240, Brazil.
[2] Universidad Técnica Federico Santa María, Telematics Group,
Department of Electronic Engineering, Av. España 1680, Valparaíso, Chile
{celso,rosseti}@ic.uff.br, reinaldo.vallejos@usm.cl

Abstract. Run time distributions or time-to-target plots are very useful tools to characterize the running times of stochastic algorithms for combinatorial optimization. We further explore run time distributions and describe a new tool to compare two algorithms based on stochastic local search. For the case where the running times of both algorithms fit exponential distributions, we derive a closed form index that gives the probability that one of them finds a solution at least as good as a given target value in a smaller computation time than the other. This result is extended to the case of general run time distributions and a numerical iterative procedure is described for the computation of the above probability value. Numerical examples illustrate the application of this tool in the comparison of different sequential and parallel algorithms for a number of distinct problems.

1 Motivation

Run time distributions or time-to-target plots display on the ordinate axis the probability that an algorithm will find a solution at least as good as a given target value within a given running time, shown on the abscissa axis. Time-to-target plots were first used by Feo et al. [10]. Run time distributions have been advocated also by Hoos and Stützle [13, 14] as a way to characterize the running times of stochastic algorithms for combinatorial optimization.

It has been observed that in many implementations of local search heuristics for combinatorial optimization problems, such as simulated annealing, genetic algorithms, iterated local search, tabu search, WalkSAT, and GRASP [2, 4, 7, 8, 12, 16, 23, 36, 37, 38], the random variable *time to target value* is exponentially distributed or fits a shifted exponential distribution. Hoos and Stützle [15, 16] conjecture that this is true for all methods for combinatorial optimization based on stochastic local search.

Aiex et al. [3] describe a perl program to create time-to-target plots for measured times that are assumed to fit a shifted exponential distribution, following

closely [2]. Such plots are very useful in the comparison of different algorithms or strategies for solving a given problem and have been widely used as a tool for algorithm design and comparison.

In this work, we further explore run time distributions to evaluate stochastic local search algorithms. We describe a new tool to compare any pair of different stochastic local search algorithms and we use it in the investigation of different sequential and parallel applications. Under the assumption that the running times of the two algorithms follow exponential (or shifted exponential) distributions, we develop in Section 2 a closed form index that gives the probability that one of the algorithms finds a target solution value in a smaller computation time than the other. This result is illustrated by some examples. In Section 3, this result is extended to the case of general run time distributions and a numerical iterative process is described for the computation of the probability that one of the algorithms will find a better solution than the other in the same computation time. Some numerical applications are considered in Section 4, illustrating the comparison of different sequential algorithms for two different problems. Applications in the evaluation of parallel heuristics based on stochastic local search are described in Section 5, giving insightful indications regarding their scalability with the number of processors and trade-offs between running times and solution quality. Section 6 investigates the robustness of the proposed measure, addressing its variation with the hardness of the target solution value. Concluding remarks are made in the last section.

2 Comparing exponential-time algorithms

We assume the existence of two stochastic local search algorithms A_1 and A_2 for the approximate solution of some combinatorial optimization problem. Furthermore, we assume that their running times fit exponential (or shifted exponential) distributions. We denote by X_1 (resp. X_2) the continuous random variable representing the time needed by algorithm A_1 (resp. A_2) to find a solution as good as a given target value:

$$X_1 \mapsto \begin{cases} 0, & \tau < T_1 \\ \lambda_1 e^{-\lambda_1(\tau - T_1)}, & \tau \geq T_1 \end{cases}$$

and

$$X_2 \mapsto \begin{cases} 0, & \tau < T_2 \\ \lambda_2 e^{-\lambda_2(\tau - T_2)}, & \tau \geq T_2 \end{cases}$$

where T_1, λ_1, T_2, and λ_2 are parameters (λ_1 and λ_2 define the shape of each shifted exponential distribution, whereas T_1 and T_2 denote by how much each of them is shifted). The cumulative probability distribution and the probability density function of X_1 are depicted in Figure 1.

Since both algorithms stop when they find a solution at least as good as the target, we may say that algorithm A_1 performs better than A_2 if the former stops before the latter. Therefore, we must evaluate the probability that the random variable X_1 takes a value smaller than or equal to X_2, i.e. we compute

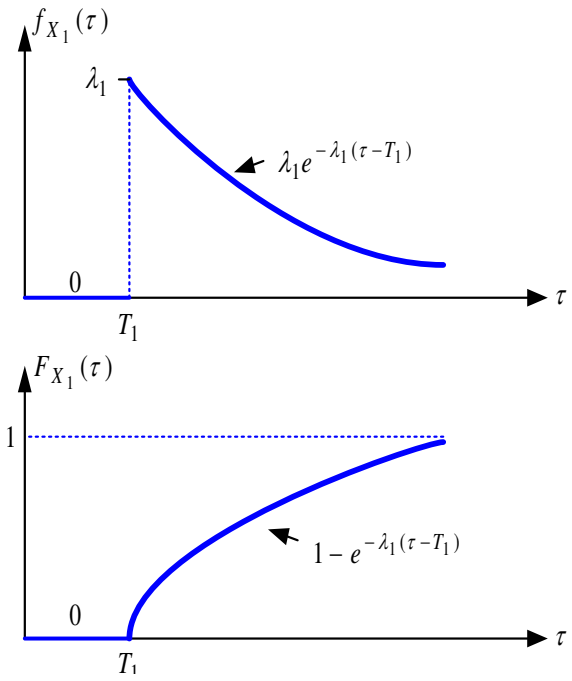


Fig. 1. Probability density function and cumulative probability distribution of the random variable X_1.

$Pr(X_1 \leq X_2)$. Conditioning on the value of X_2 and applying the total probability theorem, we obtain:

$$Pr(X_1 \leq X_2) = \int_{-\infty}^{\infty} Pr(X_1 \leq X_2 | X_2 = \tau) f_{X_2}(\tau) d\tau =$$

$$= \int_{T_2}^{\infty} Pr(X_1 \leq X_2 | X_2 = \tau) \lambda_2 e^{-\lambda_2(\tau - T_2)} d\tau = \int_{T_2}^{\infty} Pr(X_1 \leq \tau) \lambda_2 e^{-\lambda_2(\tau - T_2)} d\tau.$$

Let $\nu = \tau - T_2$. Then, $d\nu = d\tau$ and

$$Pr(X_1 \leq X_2) = \int_{0}^{\infty} Pr[X_1 \leq (\nu + T_2)] \lambda_2 e^{-\lambda_2 \nu} d\nu. \tag{1}$$

Using the formula of cumulative probability function of the random variable X_1 (see Figure 1), we obtain:

$$Pr[X_1 \leq (\upsilon + T_2)] = 1 - e^{-\lambda_1(\upsilon + T_2 - T_1)}. \tag{2}$$

Replacing (2) in (1) and solving the integral, we conclude that:

$$Pr(X_1 \leq X_2) = 1 - e^{-\lambda_1(T_2-T_1)}\frac{\lambda_2}{\lambda_1+\lambda_2}. \quad (3)$$

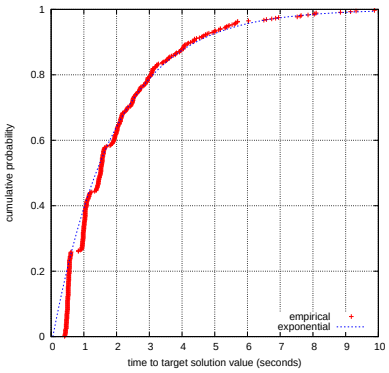
This result can be better interpreted by rewriting expression (3) as:

$$Pr(X_1 \leq X_2) = (1 - e^{-\lambda_1(T_2-T_1)}) + e^{-\lambda_1(T_2-T_1)}\frac{\lambda_1}{\lambda_1+\lambda_2}. \quad (4)$$

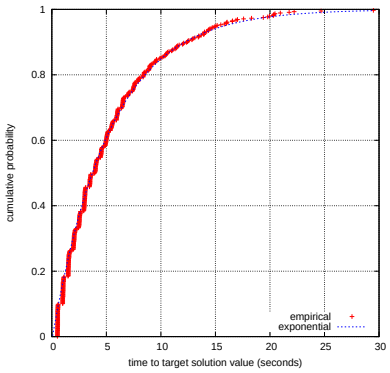
The first term of the right-hand side of equation (4) is the probability that $0 \leq X_1 \leq T_2$, in which case X_1 is clearly less than or equal to X_2. The second term is given by the product of the factors $e^{-\lambda_1(T_2-T_1)}$ and $\lambda_1/(\lambda_1+\lambda_2)$, in which the former corresponds to the probability that $X_1 \geq T_2$ and the latter to the probability that X_1 be less than or equal to X_2, given that $X_1 \geq T_2$.

To illustrate the above result, we consider the comparison of two algorithms described in [11, 35] for solving the server replication for reliable multicast problem. Algorithm A_1 is an implementation of pure GRASP with $\alpha = 0.2$, while algorithm A_2 is a pure GRASP heuristic with $\alpha = 0.9$. GRASP is a multi-start heuristic based on stochastic local search for the approximate solution of combinatorial optimization problems, which may also be hybridized with other metaheuristics; see e.g. [9, 20]. The runs were performed on an Intel Core2 Quad with 2.40 GHz of clock speed and 4 GB of RAM memory. Figure 2 depicts the run time distributions of each algorithm, obtained after 500 runs with different seeds of an instance with $m = 25$ and the target value set at 2830. The plots have been obtained with the perl tool provided in [3], which also computed the parameters of the two distributions: $\lambda_1 = 0.524422349$, $T_1 = 0.36$, $\lambda_2 = 0.190533895$, and $T_2 = 0.51$. Applying expression (3), we get $Pr(X_1 \leq X_2) = 0.684125$. This probability is consistent with Figure 3, in which we superimposed the run time distributions of the two pure GRASP heuristics for the same instance. The plots in this figure show that the pure GRASP with $\alpha = 0.2$ outperforms that with $\alpha = 0.9$, since the run time distribution of the former is much to the left of the run time distribution of the latter.

Aiex et al. [2] have shown experimentally that the time taken by a GRASP heuristic to find a solution at least as good as a given target value is a random variable that fits an exponential distribution. In the case where the setup times are not negligible, it fits a two-parameter shifted exponential distribution. The probability density function of the time-to-target random variable is given by $f(t) = (1/\lambda) \cdot e^{-t/\lambda}$ in the first case or by $f(t) = (1/\lambda) \cdot e^{-(t-\mu)/\lambda}$ in the second, with the parameters $\lambda \in \mathbb{R}^+$ and $\mu \in \mathbb{R}^+$ being associated with the shape and the shift of the exponential function, respectively. Figure 4 illustrates this result, depicting the superimposed empirical and theoretical distributions observed for one of the cases studied along the computational experiments reported in [2], which involved 2,400 runs of GRASP procedures for each of five different problem types: maximum stable set [10, 25], quadratic assignment [19, 26], graph planarization [28, 31], maximum weighted satisfiability [27], and maximum covering [24].



(a) Pure GRASP with $\alpha = 0.2$



(b) Pure GRASP with $\alpha = 0.9$

Fig. 2. Run time distributions of an instance of the server replication for reliable multicast problem with $m = 25$ and the target value set at 2830.

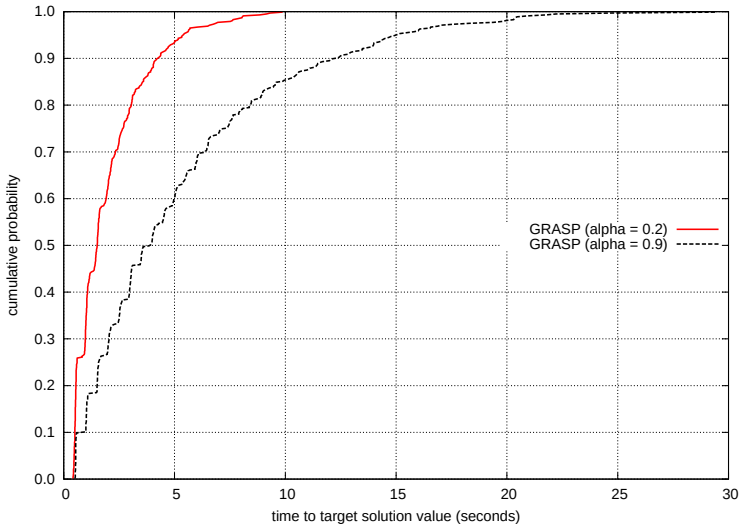


Fig. 3. Superimposed run time distributions of pure GRASP with $\alpha = 0.2$ and pure GRASP with $\alpha = 0.9$.

However, if path-relinking is applied as an intensification step at the end of each GRASP iteration [5, 29, 33], then the iterations are no longer independent and the memoryless characteristic of GRASP is destroyed. This also happens in the case of cooperative parallel implementations of GRASP. Consequently, the time-to-target random variable may not fit an exponential distribution in such situations.

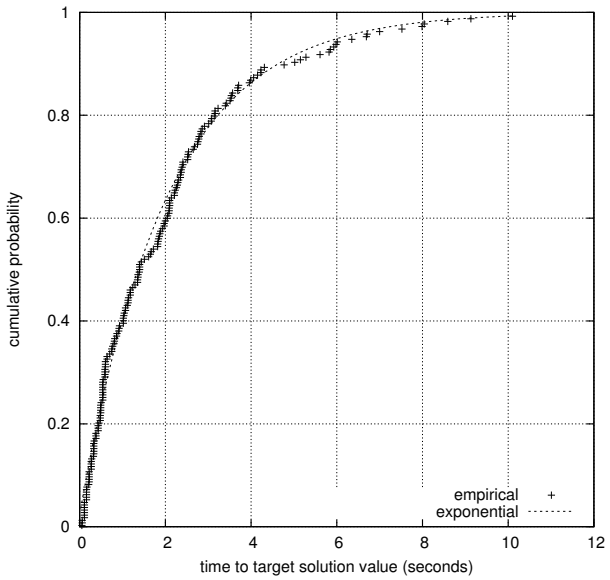


Fig. 4. Superimposed empirical run time distribution and best exponential fit.

This claim is illustrated by two implementations of GRASP with bidirectional path-relinking. The first is an application to the 2-path network design problem [33]. The run time distribution and the corresponding quantile-quantile plot for an instance with 80 nodes and 800 origin-destination pairs are depicted in Figure 5. The second is an application to the three-index assignment problem [1]. Run time distributions and the corresponding quantile-quantile plots for Balas and Saltzman problems 22.1 (target value set to 8) and 24.1 (target value set to 7) are shown in Figures 6 and 7, respectively. For all these cases, we observe that points steadily deviate by more than one standard deviation from the estimate for the upper quantiles in the quantile-quantile plots (i.e., many points associated with large computation times fall outside the plus or minus one standard deviation bounds). Therefore, we may say that these run time distributions are not exponential.

If the running times do not fit exponential (or two-parameter shifted exponential) distributions, then the result established by expression (3) does not hold. Algorithms in this situation cannot be compared by this approach. The next section extends this approach to general run time distributions.

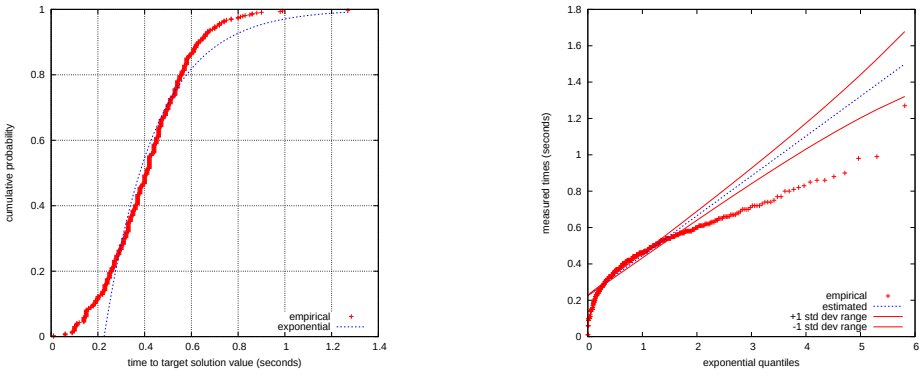


Fig. 5. Run time distribution and quantile-quantile plot for GRASP with bidirectional path-relinking of an instance of the 2-path network design problem with 80 nodes and 800 origin-destination pairs, with target set to 588.

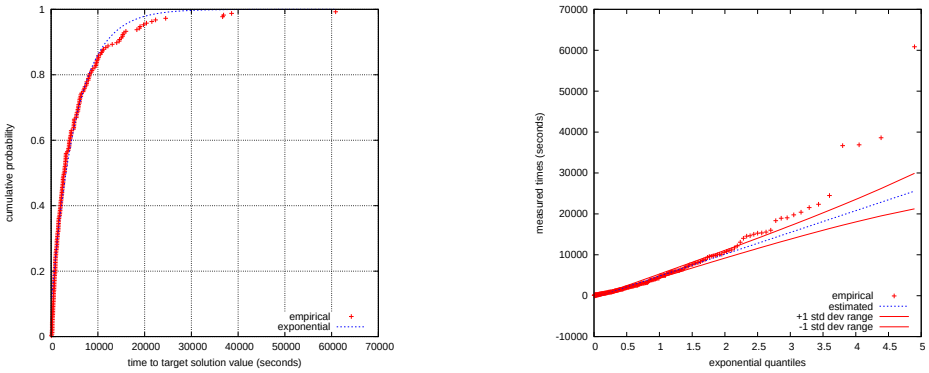


Fig. 6. Run time distribution and quantile-quantile plot for GRASP with bidirectional path-relinking on Balas and Saltzman problem 22.1, with the target value set to 8.

3 General run time distributions

Let X_1 and X_2 be two continuous random variables, with cumulative probability distributions $F_{X_1}(\tau)$ and $F_{X_2}(\tau)$ and probability density functions $f_{X_1}(\tau)$ and $f_{X_2}(\tau)$, respectively. Then,

$$Pr(X_1 \leq X_2) = \int_{-\infty}^{\infty} Pr(X_1 \leq \tau) f_{X_2}(\tau) d\tau = \int_{0}^{\infty} Pr(X_1 \leq \tau) f_{X_2}(\tau) d\tau,$$

since $f_{X_1}(\tau) = f_{X_2}(\tau) = 0$ for any $\tau < 0$. For an arbitrary small real number ε, the above expression can be rewritten as

$$Pr(X_1 \leq X_2) = \sum_{i=0}^{\infty} \int_{i\varepsilon}^{(i+1)\varepsilon} Pr(X_1 \leq \tau) f_{X_2}(\tau) d\tau. \tag{5}$$

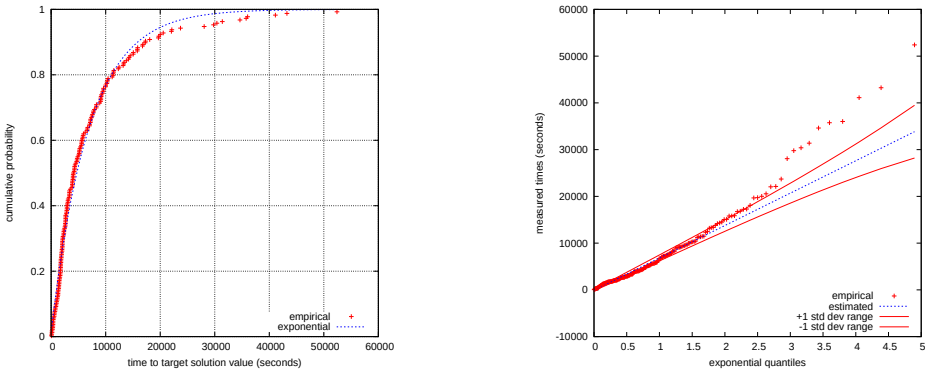


Fig. 7. Run time distribution and quantile-quantile plot for GRASP with bidirectional path-relinking on Balas and Saltzman problem 24.1, with the target value set to 7.

Since $Pr(X_1 \leq i\varepsilon) \leq Pr(X_1 \leq \tau) \leq Pr(X_1 \leq (i+1)\varepsilon)$ for $i\varepsilon \leq \tau \leq (i+1)\varepsilon$, replacing $Pr(X_1 \leq \tau)$ by $Pr(X_1 \leq i\varepsilon)$ and by $Pr(X_1 \leq (i+1)\varepsilon)$ in (5) leads to

$$\sum_{i=0}^{\infty} F_{X_1}(i\varepsilon) \int_{i\varepsilon}^{(i+1)\varepsilon} f_{X_2}(\tau)d\tau \leq Pr(X_1 \leq X_2) \leq \sum_{i=0}^{\infty} F_{X_1}((i+1)\varepsilon) \int_{i\varepsilon}^{(i+1)\varepsilon} f_{X_2}(\tau)d\tau.$$

Let $L(\varepsilon)$ and $R(\varepsilon)$ be the value of the left and right hand sides of the above expression, respectively, with $\Delta(\varepsilon) = R(\varepsilon) - L(\varepsilon)$ being the difference between the upper and lower bounds of $Pr(X_1 \leq X_2)$. Then,

$$\Delta(\varepsilon) = \sum_{i=0}^{\infty} [F_{X_1}((i+1)\varepsilon) - F_{X_1}(i\varepsilon)] \int_{i\varepsilon}^{(i+1)\varepsilon} f_{X_2}(\tau)d\tau. \tag{6}$$

Let $\delta = \max_{\tau \geq 0}\{f_{X_1}(\tau)\}$. Since $|F_{X_1}((i+1)\varepsilon) - F_{X_1}(i\varepsilon)| \leq \delta\varepsilon$ for $i \geq 0$, expression (6) turns out to be

$$\Delta(\varepsilon) \leq \sum_{i=0}^{\infty} \delta\varepsilon \int_{i\varepsilon}^{(i+1)\varepsilon} f_{X_2}(\tau)d\tau = \delta\varepsilon \int_{0}^{\infty} f_{X_2}(\tau)d\tau = \delta\varepsilon.$$

Consequently,

$$\Delta(\varepsilon) \leq \delta\varepsilon, \tag{7}$$

i.e., the difference $\Delta(\varepsilon)$ between the upper and lower bounds of $Pr(X_1 \leq X_2)$ (or the absolute error in the integration) is smaller than or equal to $\delta\varepsilon$. Therefore, this difference can be made as small as desired by choosing a sufficiently small value for ε.

In order to numerically evaluate a good approximation to $Pr(X_1 \leq X_2)$, we select the appropriate value of ε such that the resulting approximation error $\Delta(\varepsilon)$ is sufficiently small. Next, we compute $L(\varepsilon)$ and $R(\varepsilon)$ to obtain the approximation

$$Pr(X_1 \leq X_2) \approx \frac{L(\varepsilon) + R(\varepsilon)}{2}. \tag{8}$$

In practice, the probability distributions are unknown. Instead of them, all information available is a sufficiently large number N_1 (resp. N_2) of observations of the random variable X_1 (resp. X_2). Since the value of $\delta = \max_{\tau \geq 0}\{f_{X_1}(\tau)\}$ is also unknown beforehand, the appropriate value of ε cannot be estimated. Then, we proceed iteratively as follows.

Let $t_1(j)$ (resp. $t_2(j)$) be the value of the j-th smallest observation of the random variable X_1 (resp. X_2), for $j = 1, \ldots, N_1$ (resp. N_2). We set the bounds $a = \min\{t_1(1), t_2(1)\}$ and $b = \max\{t_1(N_1), t_2(N_2)\}$ and choose an arbitrary number h of integration intervals to compute an initial value $\varepsilon = (b-a)/h$ for each integration interval. For sufficiently small values of the integration interval ε, the probability density function $f_{X_1}(\tau)$ in the interval $[i\varepsilon, (i+1)\varepsilon]$ can be approximated by $\hat{f}_{X_1}(\tau) = (\hat{F}_{X_1}((i+1)\varepsilon) - \hat{F}_{X_1}(i\varepsilon))/\varepsilon$, where

$$\hat{F}_{X_1}(i\varepsilon) = |\{t_1(j), j = 1, \ldots, N_1 : t_1(j) \leq i\varepsilon\}|. \tag{9}$$

The same approximations hold for random variable X_2.

Finally, the value of $Pr(X_1 \leq X_2)$ can be computed as in expression (8), using the estimates $\hat{f}_{X_1}(\tau)$ and $\hat{f}_{X_2}(\tau)$ in the computation of $L(\varepsilon)$ and $R(\varepsilon)$. If the approximation error $\Delta(\varepsilon) = R(\varepsilon) - L(\varepsilon)$ is sufficiently small, then the procedure stops. Otherwise, the value of ε is halved and the above steps are repeated until convergence.

4 Numerical applications to sequential algorithms

We illustrate the application of the tool described in the previous section in the comparison of pairs of stochastic local search algorithms (running on the same instance) for three different test problems: server replication for reliable multicast, routing and wavelength assignment, and 2-path network design.

4.1 DM-D5 and GRASP algorithms for server replication

Multicast communication consists of simultaneously delivering the same information to many receivers, from single or multiple sources. Network services specially designed to multicast are needed. The scheme used in current multicast services create a delivery tree, whose root represents the sender, leaves represent the receivers, and internal nodes represent network routers or relaying servers. Transmission is performed by creating copies of the data at split points of the tree. An important issue regarding multicast communication is how to provide a reliable service, ensuring the delivery of all packets from the sender to receivers. A successful technique to provide a reliable multicast service is the server replication approach, in which data is replicated at some of the multicast-capable relaying hosts (also called replicated or repair servers) and each of them is responsible for the retransmission of packets to receivers in its group. The problem consists of selecting the best subset of the multicast-capable relaying hosts to act as replicated servers in a multicast scenario. It is a particular case of the p-medians problem, which has been proven to be NP-hard [18].

DM-GRASP is a hybrid version of the GRASP metaheuristic that incorporates a data-mining process [35]. Its basic principle consists of mining for patterns found in good-quality solutions to guide the construction of new solutions, leading to a more effective exploration of the solution space. We compare two different stochastic local search heuristics for the server replication problem: algorithm A_1 is an implementation of the DM-D5 version [11] of DM-GRASP, in which the mining algorithm is periodically applied, while A_2 is a pure GRASP heuristic. We present illustrative results for two instances using the same network scenario, with $m = 25$ and $m = 50$ replication servers.

Each algorithm was run 200 times with different seeds. The target was set at 2,818.925 (the best known solution value is 2,805.89) for the instance with $m = 25$ and at 2,299.07 (the best known solution value is 2,279.84) for that with $m = 50$. Figures 8 and 9 depict run time distributions and quantile-quantile plots for DM-D5, for the instances with $m = 25$ and $m = 50$, respectively. Running times of DM-D5 did not fit exponential distributions for any of the instances. GRASP running times were exponential for both.

The empirical run time distributions of DM-D5 and GRASP are superimposed in Figure 10. Algorithm DM-D5 outperformed GRASP, since the run-time distribution of the first is slightly to the left of that of the second for the instance with $m = 25$, and much more clearly for $m = 50$. Consistently, the computations show that $Pr(X_1 \leq X_2) = 0.619763$ (with $L(\varepsilon) = 0.619450$, $R(\varepsilon) = 0.620075$, $\Delta(\varepsilon) = 0.000620$, and $\varepsilon = 0.009552$) and $Pr(X_1 \leq X_2) = 0.854113$ (with $L(\varepsilon) = 0.853800$, $R(\varepsilon) = 0.854425$, $\Delta(\varepsilon) = 0.000625$, and $\varepsilon = 0.427722$) for the instances with $m = 25$ and $m = 50$, respectively.

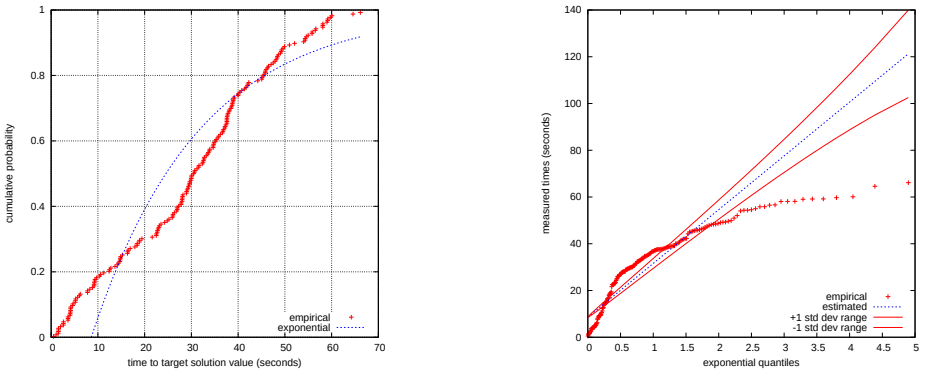


Fig. 8. Run time distribution and quantile-quantile plot for DM-D5 algorithm on the instance with $m = 25$ and the target value set at 2,818.925.

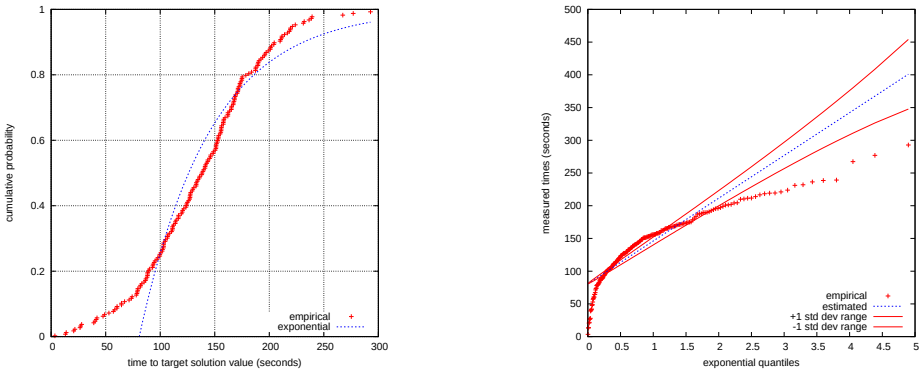


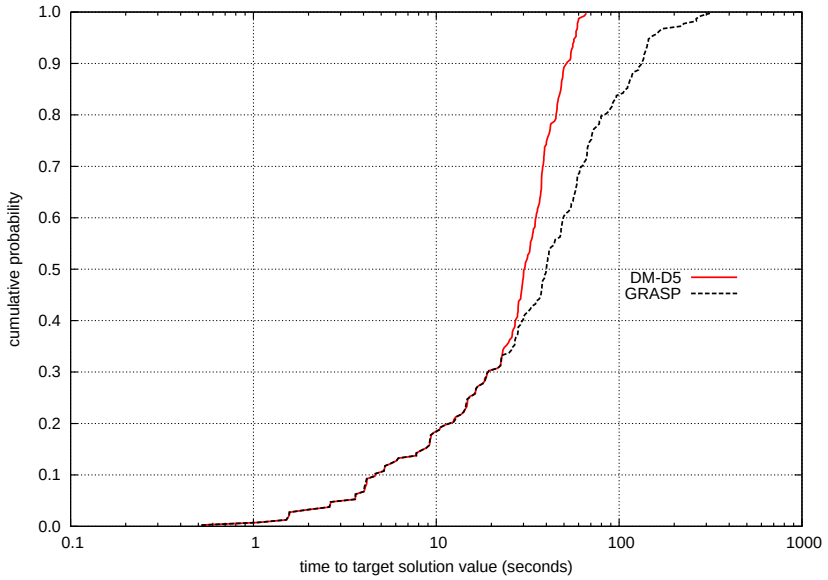
Fig. 9. Run time distribution and quantile-quantile plot for DM-D5 algorithm on the instance with $m = 50$ and the target value set at 2,299.07.

We have also investigated the convergence of the proposed measure with the sample size (i.e., with the number of independent runs of each algorithm). Convergence with the sample size is illustrated next for the same $m = 25$ instance of the server replication problem, with the same target 2,818.925 already used in the previous experiment. Once again, algorithm A_1 is the DM-D5 version of DM-GRASP and algorithm A_2 is the pure GRASP heuristic. The estimation of $Pr(X_1 \leq X_2)$ is computed for $N = 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 2000, 3000, 4000$, and 5000 independent runs of each algorithm. Table 1 shows the results obtained, which are also displayed in Figure 11. We notice that the estimation of $Pr(X_1 \leq X_2)$ stabilizes as the sample size N increases. Estimates obtained with 2,000 independent runs are already very reasonable.

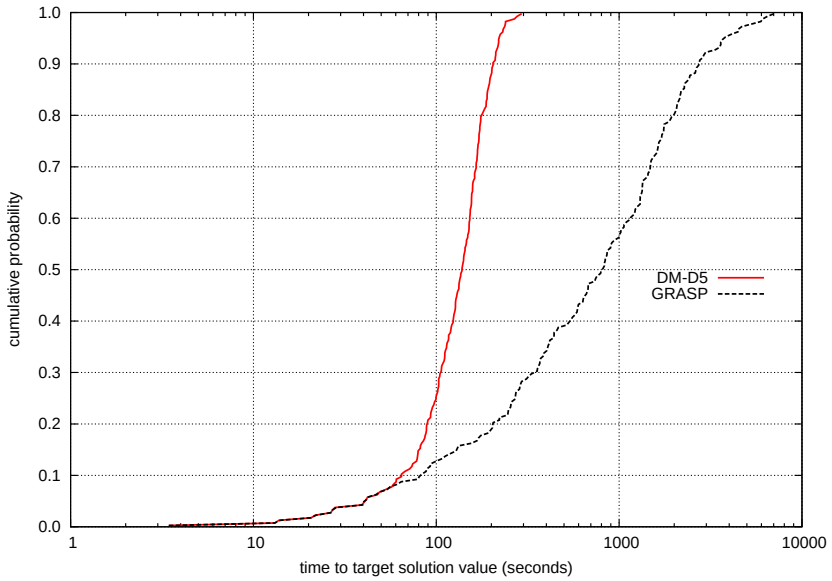
4.2 Multistart and tabu search algorithms for routing and wavelength assignment

A point-to-point connection between two endnodes of an optical network is called a lightpath. Two lightpaths may use the same wavelength, provided they do not share any common link. The routing and wavelength assignment problem is that of routing a set of lightpaths and assigning a wavelength to each of them, minimizing the number of wavelengths needed. Noronha and Ribeiro [22] proposed a decomposition heuristic for solving this problem. First, a set of possible routes is precomputed for each lightpath. Next, one of the precomputed routes and a wavelength are assigned to each lightpath by a tabu search heuristic solving an instance of the partition coloring problem.

We compare this decomposition strategy based on the tabu search heuristic with the multistart greedy heuristic of Manohar et al. [21]. Two networks are used for benchmarking. The first has 27 nodes representing the capital cities in Brazil, with 70 links connecting them. There are 702 lightpaths to be routed. Instance [17] Finland is formed by 31 nodes and 51 links, with 930 lightpaths to be routed.



(a) $m = 25$ with target 2,818.925



(b) $m = 50$ with target 2,299.07

Fig. 10. Superimposed run time distributions of DM-D5 and GRASP: (a) $Pr(X_1 \leq X_2) = 0.619763$, and (b) $Pr(X_1 \leq X_2) = 0.854113$.

Table 1. Convergence of the estimation of $Pr(X_1 \leq X_2)$ with the sample size for the $m = 25$ instance of the server replication problem.

N	$L(\varepsilon)$	$Pr(X_1 \leq X_2)$	$R(\varepsilon)$	$\Delta(\varepsilon)$	ε
100	0.655900	0.656200	0.656500	0.000600	0.032379
200	0.622950	0.623350	0.623750	0.000800	0.038558
300	0.613344	0.613783	0.614222	0.000878	0.038558
400	0.606919	0.607347	0.607775	0.000856	0.038558
500	0.602144	0.602548	0.602952	0.000808	0.038558
600	0.596964	0.597368	0.597772	0.000808	0.038558
700	0.591041	0.591440	0.591839	0.000798	0.038558
800	0.593197	0.593603	0.594009	0.000812	0.042070
900	0.593326	0.593719	0.594113	0.000788	0.042070
1000	0.594849	0.595242	0.595634	0.000785	0.042070
2000	0.588913	0.589317	0.589720	0.000807	0.047694
3000	0.583720	0.584158	0.584596	0.000875	0.047694
4000	0.582479	0.582912	0.583345	0.000866	0.047694
5000	0.584070	0.584511	0.584953	0.000882	0.050604

Each algorithm was run 200 times with different seeds. The target was set at 24 (the best known solution value is 24) for instance Brazil and at 50 (the best known solution value is 47) for instance Finland. Algorithm A_1 is the multistart heuristic, while A_2 is the tabu search decomposition scheme. The multistart running times fit exponential distributions for both instances. Figures 12 and

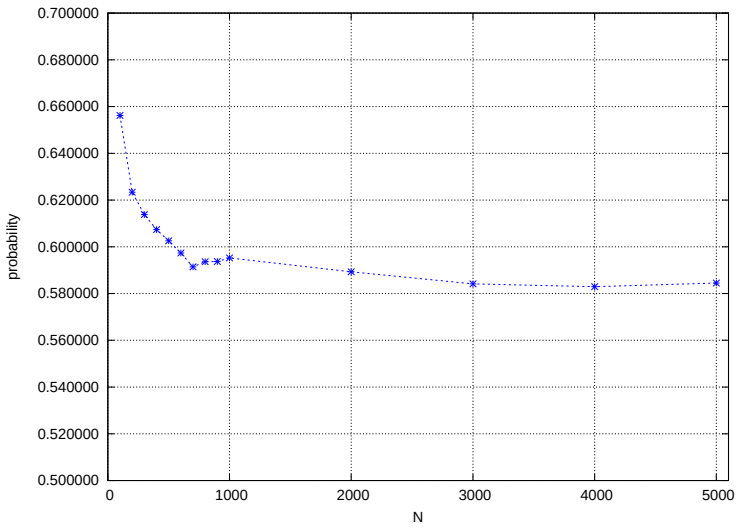


Fig. 11. Convergence of the estimation of $Pr(X_1 \leq X_2)$ with the sample size for the $m = 25$ instance of the server replication problem.

13 display run time distributions and quantile-quantile plots for instances Brazil and Finland, respectively.

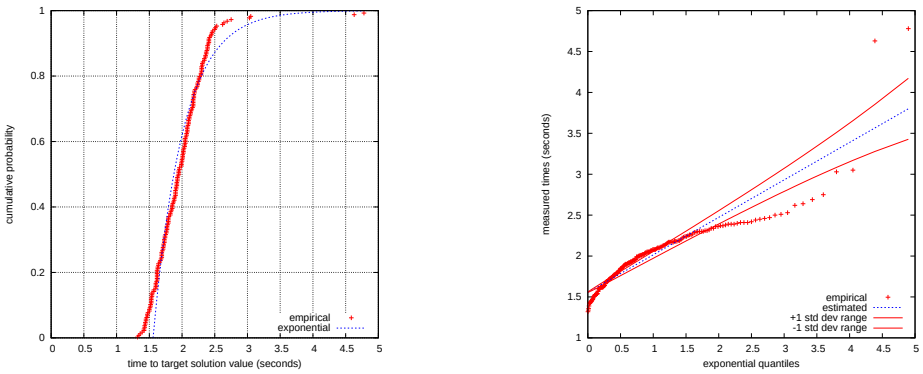


Fig. 12. Run time distribution and quantile-quantile plot for tabu search on Brazil instance with the target value set at 24.

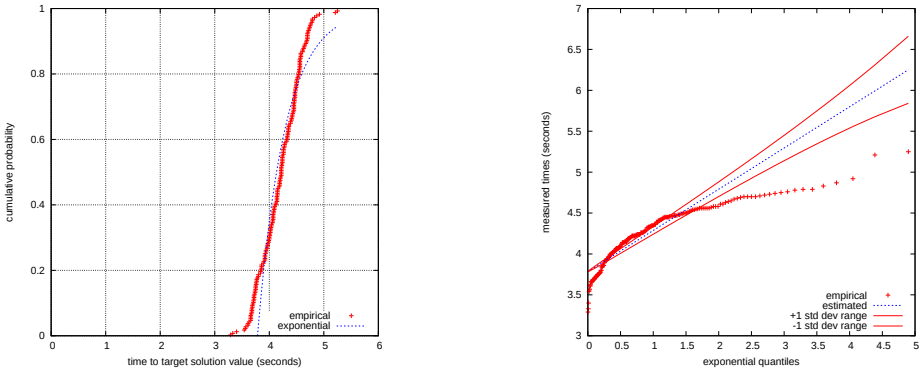


Fig. 13. Run time distribution and quantile-quantile plot for tabu search on Finland instance with the target value set at 50.

The empirical run time distributions of the decomposition and multistart strategies are superimposed in Figure 14. The direct comparison of the two approaches shows that decomposition clearly outperformed the multistart strategy

for instance Brazil, since $Pr(X_1 \leq X_2) = 0.13$ in this case (with $L(\varepsilon) = 0.129650$, $R(\varepsilon) = 0.130350$, $\Delta(\varepsilon) = 0.000700$, and $\varepsilon = 0.008163$). However, the situation changes for instance Finland. Although both algorithms have similar performances, multistart is slightly better with respect to the measure proposed in this work, since $Pr(X_1 \leq X_2) = 0.536787$ (with $L(\varepsilon) = 0.536525$, $R(\varepsilon) = 0.537050$, $\Delta(\varepsilon) = 0.000525$, and $\varepsilon = 0.008804$).

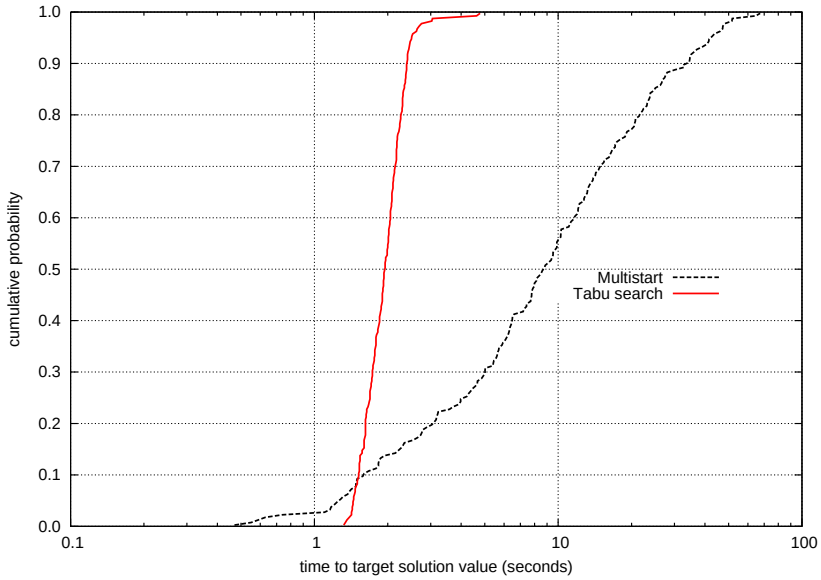
As done for the server replication problem in Section 4.1, we have also investigated the convergence of the proposed measure with the sample size (i.e., with the number of independent runs of each algorithm). Convergence with the sample size is illustrated next for the Finland instance of the routing and wavelength assignment problem, with the target set at 49. Once again, algorithm A_1 is the multistart heuristic and algorithm A_2 is the tabu search decomposition scheme. The estimation of $Pr(X_1 \leq X_2)$ is computed for N = 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 2000, 3000, 4000, and 5000 independent runs of each algorithm. Table 2 shows the results obtained, which are also displayed in Figure 15. We notice that the estimation of $Pr(X_1 \leq X_2)$ stabilizes as the sample size N increases. Estimates obtained with 1,000 independent runs are already very reasonable.

Table 2. Convergence of the estimation of $Pr(X_1 \leq X_2)$ with the sample size for the Finland instance of the routing and wavelength assignment problem.

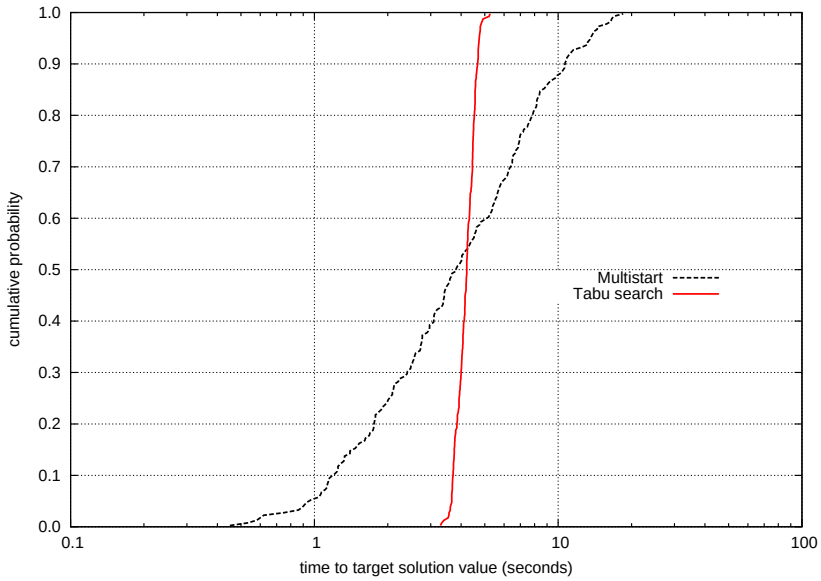
N	$L(\varepsilon)$	$Pr(X_1 \leq X_2)$	$R(\varepsilon)$	$\Delta(\varepsilon)$	ε
100	0.000001	0.000200	0.000400	0.000400	1.964844
200	0.000100	0.004875	0.009650	0.009550	0.000480
300	0.006556	0.012961	0.019367	0.012811	0.000959
400	0.007363	0.013390	0.019425	0.012063	0.000959
500	0.007928	0.014694	0.021460	0.013532	0.000610
600	0.006622	0.013069	0.019517	0.012894	0.000610
700	0.005722	0.011261	0.016800	0.011078	0.000610
800	0.005033	0.011667	0.018302	0.013269	0.000610
900	0.004556	0.010461	0.016367	0.011811	0.000610
1000	0.004100	0.009425	0.014750	0.010650	0.000610
2000	0.006049	0.011580	0.017112	0.011063	0.000610
3000	0.007802	0.014395	0.020987	0.013185	0.000610
4000	0.007408	0.013698	0.019988	0.012580	0.000610
5000	0.006791	0.013090	0.019389	0.012598	0.000623

4.3 GRASP algorithms for 2-path network design

Given a connected undirected graph with non-negative weights associated with its edges, together with a set 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.



(a) Brazil instance with target 24



(b) Finland instance with target 50

Fig. 14. Superimposed run time distributions of multistart and tabu search: (a) $Pr(X_1 \leq X_2) = 0.13$, and (b) $Pr(X_1 \leq X_2) = 0.536787$.

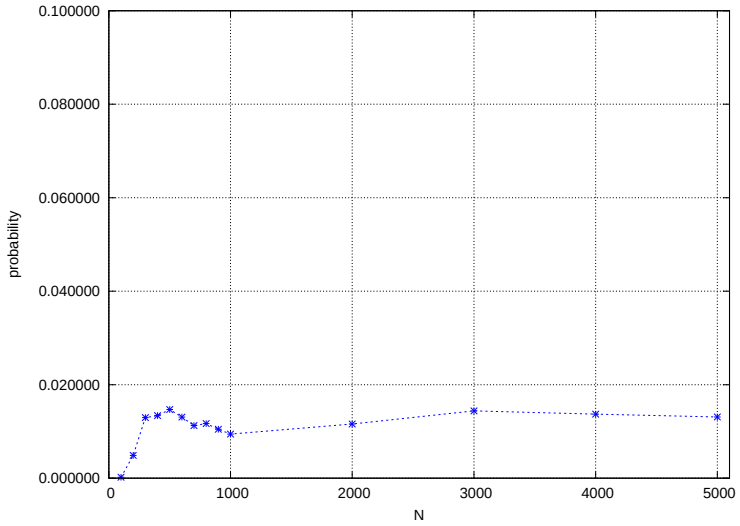


Fig. 15. Convergence of the estimation of $Pr(X_1 \leq X_2)$ with the sample size for the Finland instance of the routing and wavelength assignment problem.

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 [6].

Instance with 90 nodes. We first compare four heuristics [32, 33] for approximately solving this problem. The first is a pure GRASP algorithm (algorithm A_1). The others integrate different path-relinking strategies for search intensification at the end of each GRASP iteration: forward (algorithm A_2), bidirectional (algorithm A_3), and backward (algorithm A_4) [30, 29].

Each algorithm was run 500 independent times on a benchmark instance with 90 nodes and 900 origin-destination pairs, with the target value set at 673 (the best known solution value is 639). The run time distributions and quantile-quantile plots for the different versions of GRASP with path-relinking are illustrated in Figures 16 to 18.

The empirical run time distributions of the four algorithms are superimposed in Figure 19. Algorithm A_2 (as well as A_3 and A_4) performs much better than A_1, since $Pr(X_2 \leq X_1) = 0.986604$ (with $L(\varepsilon) = 0.986212$, $R(\varepsilon) = 0.986996$, $\Delta(\varepsilon) = 0.000784$, and $\varepsilon = 0.029528$). Algorithm A_3 outperforms A_2, as illustrated by the fact that $Pr(X_3 \leq X_2) = 0.636000$ (with $L(\varepsilon) = 0.630024$, $R(\varepsilon) = 0.641976$, $\Delta(\varepsilon) = 0.011952$, and $\varepsilon = 1.354218 \times 10^{-6}$). Finally, we observe that algorithms A_3 and A_4 behave very similarly, although A_4 performs slightly better for this instance with respect to the measure proposed in this work, since $Pr(X_4 \leq X_3) = 0.536014$ (with $L(\varepsilon) = 0.528560$, $R(\varepsilon) = 0.543468$, $\Delta(\varepsilon) = 0.014908$, and $\varepsilon = 1.001358 \times 10^{-6}$).

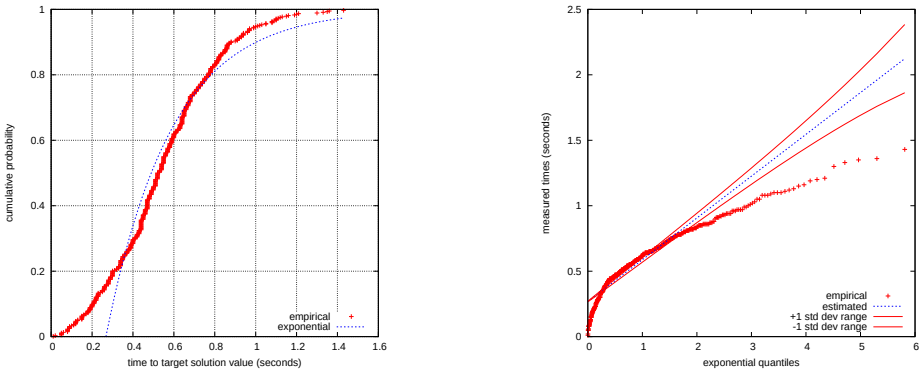


Fig. 16. Run time distribution and quantile-quantile plot for GRASP with forward path-relinking on 90-node instance with the target value set at 673.

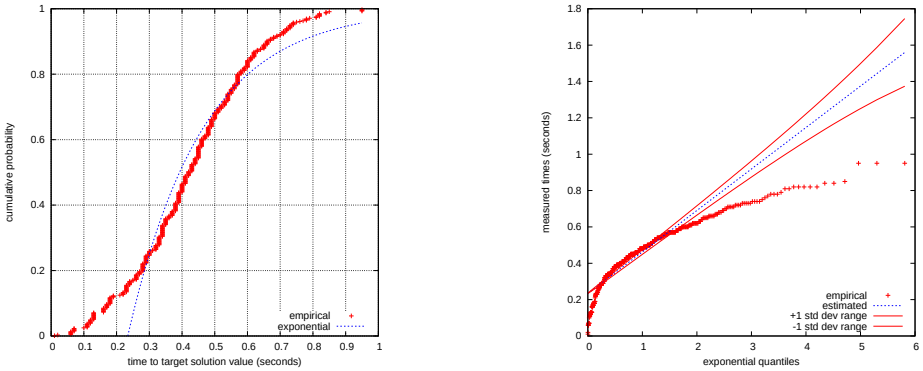


Fig. 17. Run time distribution and quantile-quantile plot for GRASP with bidirectional path-relinking on 90-node instance with the target value set at 673.

As for the problems considered in Sections 4.1 and 4.2, we have also investigated the convergence of the proposed measure with the sample size (i.e., with the number of independent runs of each algorithm). Convergence with the sample size is illustrated next for the 90-node instance of the 2-path network

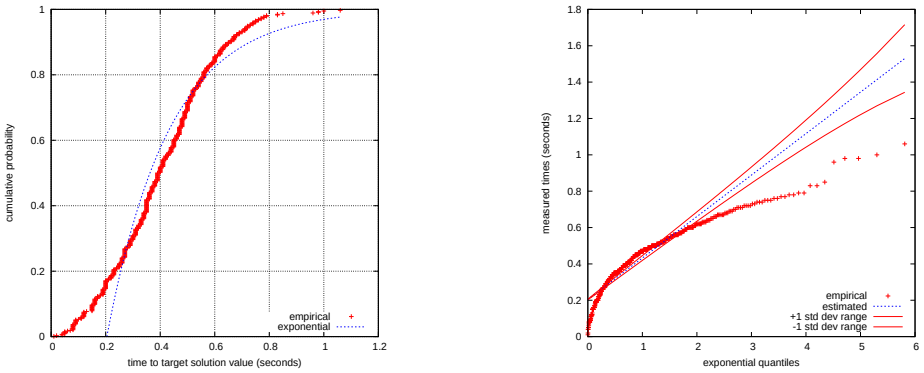


Fig. 18. Run time distribution and quantile-quantile plot for GRASP with backward path-relinking on 90-node instance with the target value set at 673.

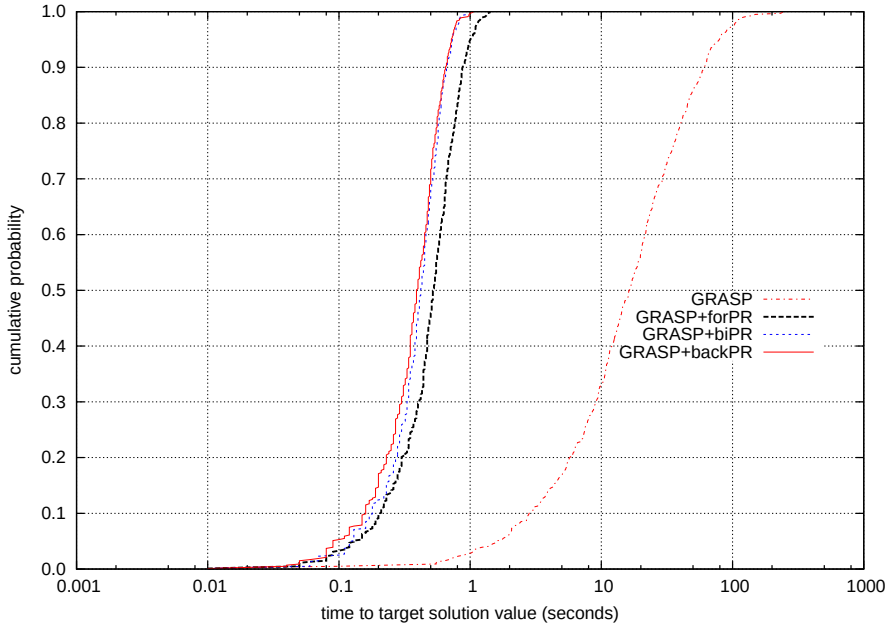


Fig. 19. Superimposed run time distributions of pure GRASP and three versions of GRASP with path-relinking.

design problem, with the same target 673 already used. We recall that algorithm A_1 is the GRASP with backward path-relinking heuristic, while algorithm A_2 is the GRASP with bidirectional path-relinking heuristic. The estimation of $Pr(X_1 \leq X_2)$ is computed for $N = 100$, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 2000, 3000, 4000, and 5000 independent runs of each algorithm. Table 3

Table 3. Convergence of the estimation of $Pr(X_1 \leq X_2)$ with the sample size for the 90-node instance of the 2-path network design problem.

N	$L(\varepsilon)$	$Pr(X_1 \leq X_2)$	$R(\varepsilon)$	$\Delta(\varepsilon)$	ε
100	0.553300	0.559150	0.565000	0.011700	4.387188×10^{-7}
200	0.553250	0.553850	0.554450	0.001199	4.501629×10^{-7}
300	0.551578	0.557483	0.563389	0.011811	4.501629×10^{-7}
400	0.545244	0.551241	0.557238	0.011994	4.730511×10^{-7}
500	0.546604	0.552420	0.558236	0.011632	5.035686×10^{-7}
600	0.538867	0.544749	0.550631	0.011764	5.073833×10^{-7}
700	0.536320	0.542181	0.548041	0.011720	5.073833×10^{-7}
800	0.537533	0.543298	0.549064	0.011531	5.073833×10^{-7}
900	0.533912	0.539671	0.545430	0.011517	5.073833×10^{-7}
1000	0.531595	0.537388	0.543180	0.011585	5.073833×10^{-7}
2000	0.528224	0.533959	0.539698	0.011469	5.722427×10^{-7}
3000	0.530421	0.536128	0.541835	0.011414	6.027603×10^{-7}
4000	0.532695	0.538364	0.544033	0.011338	6.027603×10^{-7}
5000	0.530954	0.536566	0.542178	0.011225	6.027603×10^{-7}

shows the results obtained, which are also displayed in Figure 20. We notice that the estimation of $Pr(X_1 \leq X_2)$ stabilizes as the sample size N increases. Estimates obtained with 1,000 independent runs are already very reasonable.

Instance with 80 nodes. We now compare five different GRASP heuristics [32, 33] for the 2-path network design problem, with and without path-relinking, for solving an instance with 80 nodes and 800 origin-destination pairs, with the target value set at 588 (the best known solution value is 577). In this example, the first algorithm is a pure GRASP heuristic (algorithm A_1). The other heuristics integrate different path-relinking strategies for search intensification at the end of each GRASP iteration: forward (algorithm A_2), bidirectional (algorithm A_3), backward (algorithm A_4), and mixed (algorithm A_5) [30, 29]. As before, each version was run 500 independent times.

The empirical run time distributions of the five algorithms are superimposed in Figure 21. Algorithm A_2 (as well as A_3, A_4, and A_5) performs much better than A_1, since $Pr(X_2 \leq X_1) = 0.970652$ (with $L(\varepsilon) = 0.970288$, $R(\varepsilon) = 0.971016$, $\Delta(\varepsilon) = 0.000728$, and $\varepsilon = 0.014257$). Algorithm A_3 outperforms A_2, as illustrated by the fact that $Pr(X_3 \leq X_2) = 0.617278$ (with $L(\varepsilon) = 0.610808$, $R(\varepsilon) = 0.623748$, $\Delta(\varepsilon) = 0.012940$, and $\varepsilon = 1.220703 \times 10^{-6}$). Algorithm A_4 performs slightly better than A_3 for this instance, since $Pr(X_4 \leq X_3) = 0.537578$ (with $L(\varepsilon) = 0.529404$, $R(\varepsilon) = 0.545752$, $\Delta(\varepsilon) = 0.016348$, and $\Delta(\varepsilon) = 1.201630 \times 10^{-6}$). Algorithms A_5 and A_4 also behave very similarly, but A_5 is slightly better for this instance since $Pr(X_5 \leq X_4) = 0.556352$ (with $L(\varepsilon) = 0.547912$, $R(\varepsilon) = 0.564792$, $\Delta(\varepsilon) = 0.016880$, and $\varepsilon = 1.001358 \times 10^{-6}$).

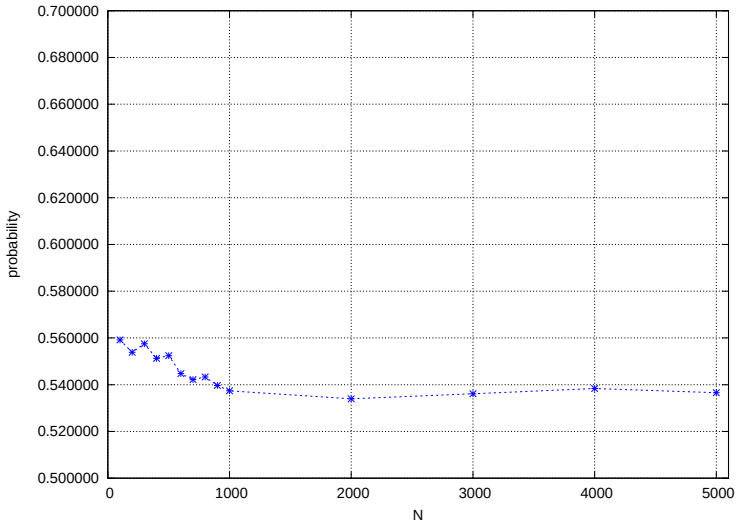


Fig. 20. Convergence of the estimation of $Pr(X_1 \leq X_2)$ with the sample size for the 90-node instance of the 2-path network design problem.

5 Comparing and evaluating parallel algorithms

In this section, we describe the use of the tool proposed in this work to evaluate and compare parallel implementations of stochastic local search algorithms. Once again, the 2-path network design problem is used to illustrate the application.

Figures 22 and 23 superimpose the run time distributions of cooperative and independent parallel implementations of GRASP with bidirectional path-relinking for the same problem, respectively, on 2, 4, 8, 16, and 32 processors, for an instance with 100 nodes and 1000 origin-destination pairs, with the target value set at 683. Each algorithm was run 200 times. We denote by A_1^k (resp. A_2^k) the cooperative (resp. independent) parallel implementation running on $k = 2, 4, 8, 16, 32$ processors.

Table 4 displays the probability that the cooperative parallel implementation performs better than the independent implementation on 2, 4, 8, 16, and 32 processors. We observe that the independent implementation performs better than the cooperative implementation on two processors. In this case, the cooperative implementation does not benefit from the existence of two processors, since only one of them performs iterations, while the other acts as the master. However, as the number of processors increases from two to 32, the cooperative implementation performs progressively better than the independent implementation, since more processors are devoted to perform GRASP iterations. The performance measure proposed in this work is clearly consistent with the relative behavior of the two parallel versions for any number of processors. Furthermore, it illus-

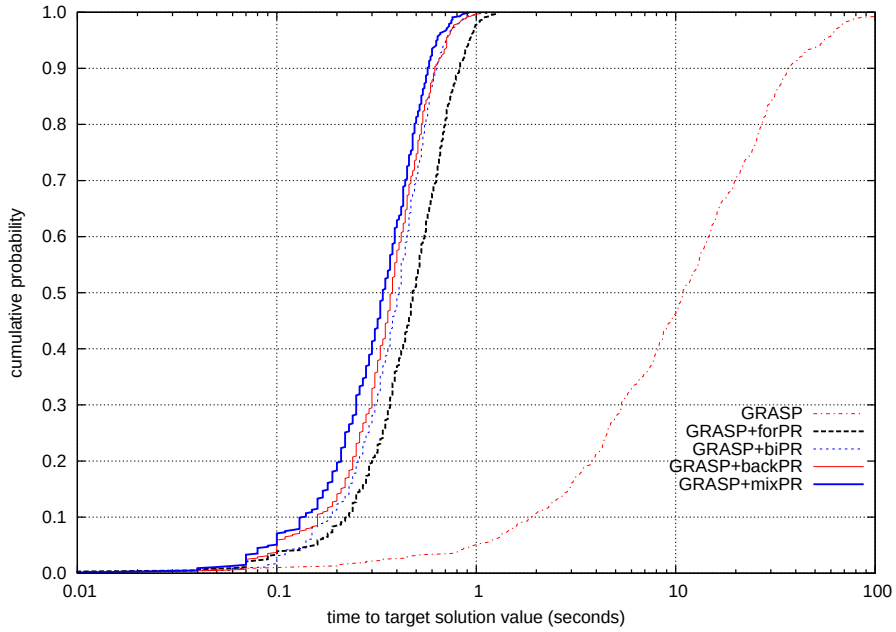


Fig. 21. Superimposed empirical run time distributions of pure GRASP and four versions of GRASP with path-relinking.

trates that the cooperative implementation becomes progressively better than the independent implementation when the number of processors increase.

Table 4. Comparing cooperative (algorithm A_1) and independent (algorithm A_2) parallel implementations.

Processors (k)	$Pr(X_1^k \leq X_2^k)$
2	0.309784
4	0.597253
8	0.766806
16	0.860864
32	0.944938

Table 5 displays the probability that each of the two parallel implementations performs better on 2^{j+1} than on 2^j processors, for $j = 1, 2, 3, 4$. Both implementations scale appropriately as the number of processors grows. Once again, we may see that the performance measure appropriately describes the relative behavior of the two parallel strategies and gives a good insight on how the parallel algorithms scale with the number of processors. It provides numerical evidence to

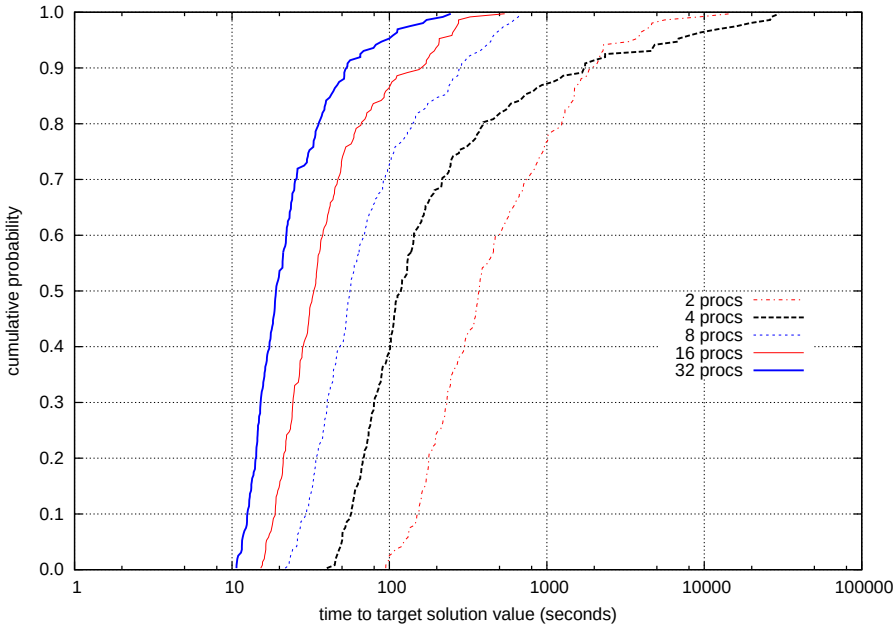


Fig. 22. Superimposed empirical run time distributions of cooperative parallel GRASP with bidirectional path-relinking running on 2, 4, 8, 16, and 32 processors.

evaluate the trade-offs between computation times and the number of processors in parallel implementations.

Table 5. Comparing the parallel implementations on 2^{j+1} (algorithm A_1) and 2^j (algorithm A_2) processors, for $j = 1, 2, 3, 4$.

Processors (a)	Processors (b)	$Pr(X_1^a \leq X_1^b)$	$Pr(X_2^a \leq X_2^b)$
4	2	0.766235	0.651790
8	4	0.753904	0.685108
16	8	0.724398	0.715556
32	16	0.747531	0.669660

6 Variation with the hardness of the target value

In the last experiment, we investigate the behavior of the measure $Pr(X_1 \leq X_2)$ as the hardness of the target value changes. Numerical results are illustrated for the same instance of the 2-path network design problem with 80 nodes and 800 origin-destination pairs already considered in Section 4.3. Algorithm A_1 is

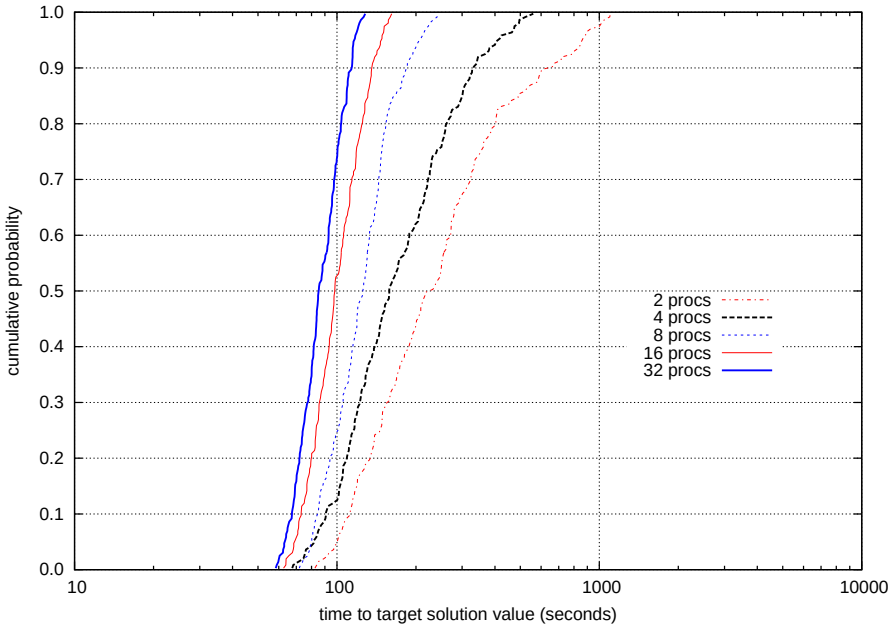


Fig. 23. Superimposed empirical run time distributions of independent parallel GRASP with bidirectional path-relinking running on 2, 4, 8, 16, and 32 processors.

the GRASP with bidirectional path-relinking heuristic, while algorithm A_2 is a pure GRASP heuristic. Figure 24 displays the estimation of $Pr(X_1 \leq X_2)$ as the target ranges from 577 to 735. $N = 1,000$ independent runs have been performed for each target value. Both algorithms behave similarly for easy (i.e., large) targets. The GRASP with bidirectional path-relinking heuristic performs progressively better for hard (i.e., small) targets, with $Pr(X_1 \leq X_2) \to 1$ when the target decreases and approaches the optimal value. However, a first look at this plot gives the wrong idea that $Pr(X_1 \leq X_2)$ seems to behave erratically for medium-range target values between 600 and 640. To better understand what happens in this range, the upper part of Figure 25 displays $Pr(X_1 \leq X_2)$ for this restricted range of target values. The lower part of this figure shows the average running time over all 1,000 runs performed for the same target. We first observe that the average running time of the pure GRASP heuristic decays faster as the target approaches 615 from above. Contrarily, the average running time of the GRASP with bidirectional path-relinking heuristic increases quicker when the target approaches 615 from below. Consequently, $\Pr(X_1 \leq X_2)$ attains a minimum when the target is equal to 615, showing that its behavior is not monotonic as one could possibly expect from a superficial analysis.

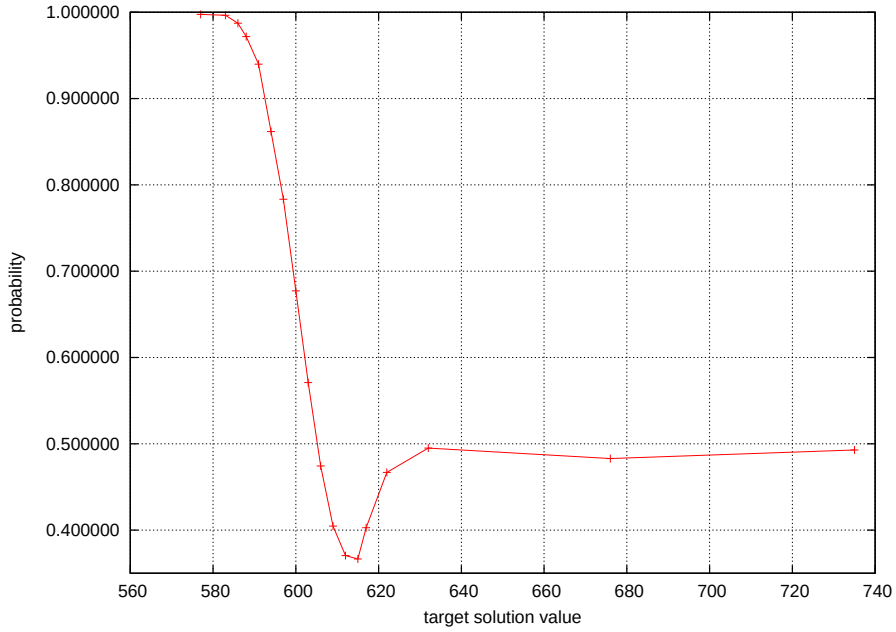


Fig. 24. Variation of $Pr(X_1 \leq X_2)$ with the target hardness.

7 Concluding remarks

Run time distributions are very useful tools to characterize the running times of stochastic algorithms for combinatorial optimization. In this work, we extended previous tools for plotting and evaluating run time distributions.

Under the assumption that running times of two stochastic local search algorithms follow exponential distributions, we derived a closed form index to compute the probability that one of them finds a target solution value in a smaller computation time than the other. A numerical iterative procedure was described for the computation of such index in the case of general run time distributions.

This new tool and the resulting probability index revealed themselves as very promising and provide a new, additional measure for comparing the performance of stochastic local search algorithms or different versions of the same algorithm. They can also be used for setting the best parameters of a given algorithm, by providing an strategy for comparing the resulting implementations. Numerical applications to different algorithm paradigms, problem types, and test instances illustrated the applicability of the tool.

In another context, this tool was also used in the evaluation of parallel implementations of local search algorithms. It made it possible to provide insightful analysis involving the trade-offs between computation time and solution quality and the scalability of parallel implementations when the number of available processors varies.

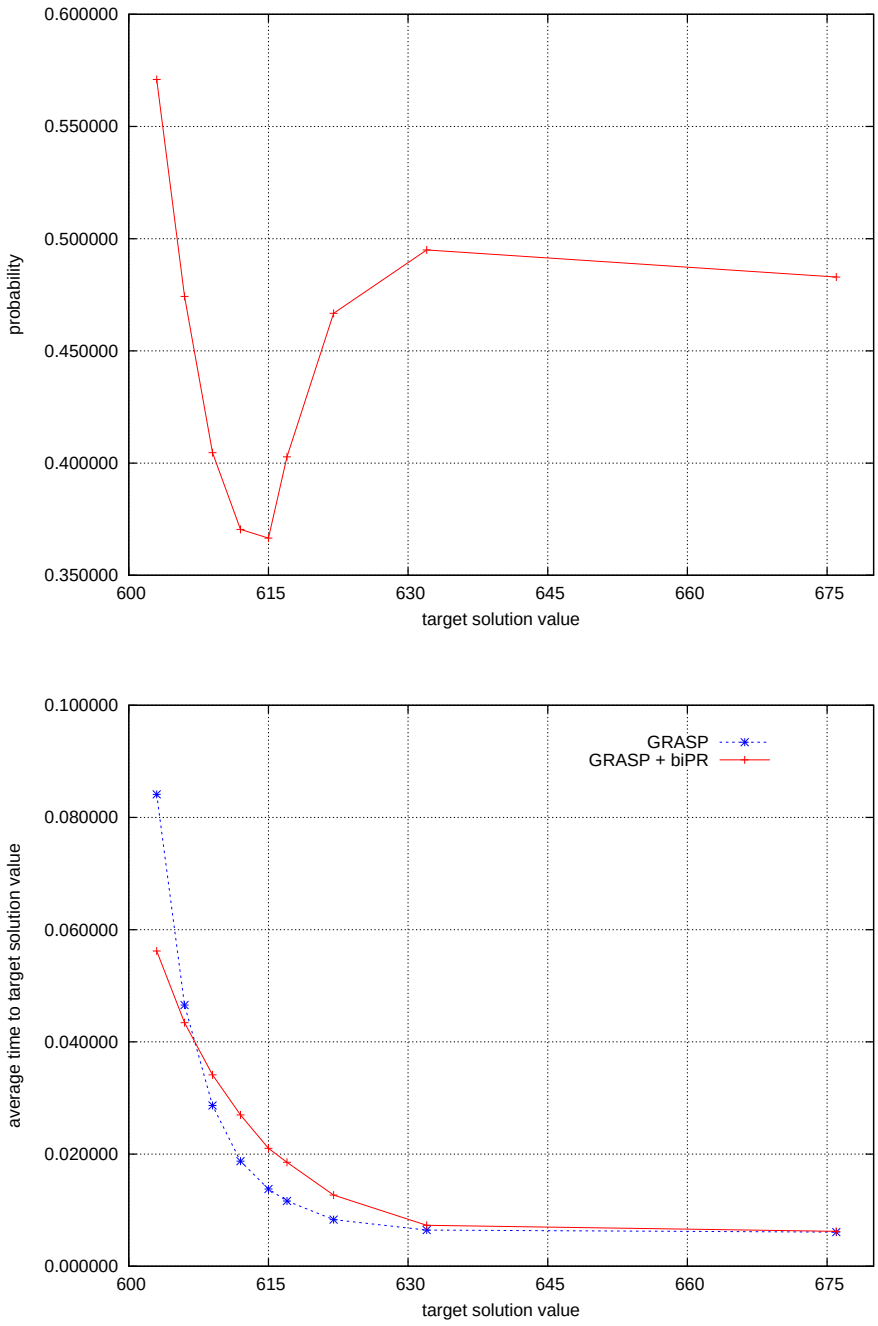


Fig. 25. Analysis of the variation in the probability estimation with the target hardness.

Acknowledgments. This paper is an extended version of that originally titled "On the use of run time distributions to evaluate and compare sequential and parallel stochastic local search algorithms" [34], which received the "Best Paper Presentation Award" among all papers presented at the conference "Engineering Stochastic Local Search Algorithms" held in Brussels from September 3 to 4, 2009.

References

[1] R.M. Aiex, P.M. Pardalos, M.G.C. Resende, and G. Toraldo. GRASP with path relinking for three-index assignment. *INFORMS Journal on Computing*, 17:224–247, 2005.

[2] R.M. Aiex, M.G.C. Resende, and C.C. Ribeiro. Probability distribution of solution time in GRASP: An experimental investigation. *Journal of Heuristics*, 8:343–373, 2002.

[3] R.M. Aiex, M.G.C. Resende, and C.C. Ribeiro. TTTPLOTS: A perl program to create time-to-target plots. *Optimization Letters*, 1:355–366, 2007.

[4] R. Battiti and G. Tecchiolli. Parallel biased search for combinatorial optimization: Genetic algorithms and TABU. *Microprocessors and Microsystems*, 16:351–367, 1992.

[5] S.A. Canuto, M.G.C. Resende, and C.C. Ribeiro. Local search with perturbations for the prize-collecting Steiner tree problem in graphs. *Networks*, 38:50–58, 2001.

[6] G. Dahl and B. Johannessen. The 2-path network problem. *Networks*, 43:190–199, 2004.

[7] N. Dodd. Slow annealing versus multiple fast annealing runs: An empirical investigation. *Parallel Computing*, 16:269–272, 1990.

[8] H.M.M. Ten Eikelder, M.G.A. Verhoeven, T.W.M. Vossen, and E.H.L. Aarts. A probabilistic analysis of local search. In I.H. Osman and J.P. Kelly, editors, *Metaheuristics: Theory and Applications*, pages 605–618. Kluwer, 1996.

[9] T.A. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

[10] T.A. Feo, M.G.C. Resende, and S.H. Smith. A greedy randomized adaptive search procedure for maximum independent set. *Operations Research*, 42:860–878, 1994.

[11] E. Fonseca, R. Fuchsuber, L.F.M. Santos, A. Plastino, and S.L. Martins. Exploring the hybrid metaheuristic DM-GRASP for efficient server replication for reliable multicast. In *International Conference on Metaheuristics and Nature Inspired Computing*, page 44, Hammamet, 2008.

[12] H.H. Hoos. On the run-time behaviour of stochastic local search algorithms for SAT. In *Proc. AAAI-99*, pages 661–666. MIT Press, 1999.

[13] H.H. Hoos and T. Stützle. Evaluation of Las Vegas algorithms - Pitfalls and remedies. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 238–245, 1998.

[14] H.H. Hoos and T. Stützle. On the empirical evaluation of Las Vegas algorithms - Position paper. Technical report, Computer Science Department, University of British Columbia, 1998.

[15] H.H. Hoos and T. Stützle. Some surprising regularities in the behaviour of stochastic local search. *Lecture Notes in Computer Science*, 1520:470, 1998.

[16] H.H. Hoos and T. Stützle. Towards a characterisation of the behaviour of stochastic local search algorithms for SAT. *Artificial Intelligence*, 112:213–232, 1999.

[17] E. Hyytiã and J. Virtamo. Wavelength assignment and routing in WDM networks. In *Nordic Teletraffic Seminar 14*, pages 31–40, 1998.

[18] O. Kariv and S.L. Hakimi. An algorithmic approach to network location problems ii: The p-medians. *SIAM Journal of Applied Mathematics*, 37:513–538, 1979.

[19] Y. Li, P.M. Pardalos, and M.G.C. Resende. A greedy randomized adaptive search procedure for the quadratic assignment problem. In P.M. Pardalos and H. Wolkowicz, editors, *Quadratic Assignment and Related Problems*, volume 16 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, pages 237–261. American Mathematical Society, 1994.

[20] A.P. Lucena, C.C. Ribeiro, and A.C. Santos. A hybrid heuristic for the diameter constrained minimum spanning tree problem. *Journal of Global Optimization*, 46:363–381, 2010.

[21] P. Manohar, D. Manjunath, and R.K. Shevgaonkar. Routing and wavelength assignment in optical networks from edge disjoint path algorithms. *IEEE Communications Letters*, 5:211–213, 2002.

[22] T.F. Noronha and C.C. Ribeiro. Routing and wavelength assignment by partition coloring. *European Journal of Operational Research*, 171:797–810, 2006.

[23] L.J. Osborne and B.E. Gillett. A comparison of two simulated annealing algorithms applied to the directed Steiner problem on networks. *ORSA Journal on Computing*, 3:213–225, 1991.

[24] M.G.C. Resende. Computing approximate solutions of the maximum covering problem using GRASP. *Journal of Heuristics*, 4:161–171, 1998.

[25] M.G.C. Resende, T.A. Feo, and S.H. Smith. Algorithm 787: Fortran subroutines for approximate solution of maximum independent set problems using GRASP. *ACM Trans. Math. Software*, 24:386–394, 1998.

[26] M.G.C. Resende, P.M. Pardalos, and Y. Li. Algorithm 754: Fortran subroutines for approximate solution of dense quadratic assignment problems using GRASP. *ACM Transactions on Mathematical Software*, 22:104–118, 1996.

[27] M.G.C. Resende, L.S. Pitsoulis, and P.M. Pardalos. Fortran subroutines for computing approximate solutions of MAX-SAT problems using GRASP. *Discrete Applied Mathematics*, 100:95–113, 2000.

[28] M.G.C. Resende and C.C. Ribeiro. A GRASP for graph planarization. *Networks*, 29:173–189, 1997.

[29] 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.

[30] M.G.C. Resende and C.C. Ribeiro. Greedy randomized adaptive search procedures: Advances and applications. In M. Gendreau and J.-Y. Potvin, editors, *Handbook of Metaheuristics*, pages 283–319. Springer, 2nd edition, 2010.

[31] C.C. Ribeiro and M.G.C. Resende. Algorithm 797: Fortran subroutines for approximate solution of graph planarization problems using GRASP. *ACM Transactions on Mathematical Software*, 25:342–352, 1999.

[32] 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.

[33] C.C. Ribeiro and I. Rosseti. Efficient parallel cooperative implementations of GRASP heuristics. *Parallel Computing*, 33:21–35, 2007.

[34] C.C. Ribeiro, I. Rosseti, and R. Vallejos. On the use of run time distributions to evaluate and compare stochastic local search algorithms. In T. Stützle, M. Biratari, and H.H. Hoos, editors, *Engineering Stochastic Local Search Algorithms*, volume 5752 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 2009.

[35] L.F. Santos, S.L. Martins, and A. Plastino. Applications of the DM-GRASP heuristic: A survey. *International Transactions in Operational Research*, 15:387–416, 2008.

[36] B. Selman, H.A. Kautz, and B. Cohen. Noise strategies for improving local search. In *Proceedings of the AAAI-94*, pages 337–343. MIT Press, 1994.

[37] E.D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17:443–455, 1991.

[38] M.G.A. Verhoeven and E.H.L. Aarts. Parallel local search. *Journal of Heuristics*, 1:43–66, 1995.