Martí, R., Lozano, J.A., Mendiburu, A., Hernando, L. (2018). *Multi-start Methods.* In: Martí, R., Pardalos, P., Resende, M. (eds) **Handbook of Heuristics. 155–175,** Springer, Cham. This version of the chapter has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: https://doi.org/10.1007/978-3-319-07124-4_1

Title: Multi-Start Methods and Local Optima

Name: Rafael Martí[1], Jose A. Lozano[2], Alexander Mendiburu[3], Leticia Hernando[2]

Affil./Addr. 1: Departamento de Estadística e Investigación Operativa,
Facultad de Matematicas, Universidad de Valencia,
Dr. Moliner 50, 46100 Burjassot (Valencia), Spain
E-mail: rafael.marti@uv.es

Affil./Addr. 2: Intelligent Systems Group,
Department of Computer Science and Artificial Intelligence,
University of the Basque Country,
Manuel de Lardizabal 1, 20018 Donostia, Gipuzkoa (Spain).
E-mail: {ja.lozano, leticia.hernando}@ehu.es

Affil./Addr. 3: Intelligent Systems Group,
Department of Computer Architecture and Technology,
University of the Basque Country,
Manuel de Lardizabal 1, 20018 Donostia, Gipuzkoa (Spain).
E-mail: alexander.mendiburu@ehu.es

Multi-Start Methods and Local Optima

Summary. Multi-start procedures were originally conceived as a way to exploit a local or neighborhood search procedure, by simply applying it from multiple random initial solutions. Modern multi-start methods usually incorporate a powerful form of diversification in the generation of solutions to help overcome local optimality. Different metaheuristics, such as GRASP or tabu search, have been applied to this end. This survey briefly sketches historical developments that have motivated the field, and then focuses on modern contributions that define the current state-of-the-art. We consider

the two classic categories of multi-start methods according to their domain of application: global optimization and combinatorial optimization. Additionally, we review several methods to estimate the number of local optima in combinatorial problems. The estimation of this number can help to establish the complexity of a given instance, and also to choose the most convenient neighborhood, which is especially interesting in the context of multi-start methods.

Key words: Metaheuristics, multi-start methods, Local optima estimation

Introduction

Heuristic search procedures that aspire to find globally optimal solutions to hard optimization problems usually require some type of diversification to overcome local optimality. One way to achieve diversification is to re-start the procedure from a new solution once a region has been explored. In this chapter we describe the best known multi-start methods for solving optimization problems.

The first multi-start efforts described in the literature, rely on methods used in statistics and calculus as instances of utilizing repeated constructions to produce a preferred candidate, although such methods were not used to solve optimization problems. In our context, early proposals can be found in the domains of heuristic scheduling (Muth and Thompson [34] and Crowston et al [10]) and the traveling salesman problem (Held and Karp [23]). Multi-start global optimization algorithms on the other hand, were introduced in the 1980s for bound constraint optimization problems. The well-known Monte Carlo random re-start approaches, simply evaluate the objective function at randomly generated points (Solis and Wets [39]). The probability of success approaches one as the sample size tends to infinity under very mild assumptions about the objective function. Many algorithms have been proposed that combine the Monte Carlo method with local search procedures (Kan and Timmer [28]).

The re-start mechanism inherent of a multi-start design, has been superimposed on many different search methods. Once a new solution has been generated, a variety of options can be used to improve it, ranging from a simple greedy routine to a complex metaheuristic such as tabu search [20] or GRASP [37]. Note that the former is based on identifying and recording specific types of information (attributes) to exploit in future constructions, while the latter is based on order statistics of sampling and generate unconnected solutions. In Martí et al [31] we can find a detailed description of these two methodologies within the multi-start framework, in the context of combinatorial optimization.

A basic multi-start procedure simply applies procedure `ConstructSolution` multiple times, returning the best solution found over all starts. The constructed solution is typically improved with the `LocaSearch` procedure. These two procedures, also called phases, are alternated until an stopping criterion is satisfied. Then, each global iteration produces a solution (usually a local optima) and the best overall is the algorithm's output. This is illustrated in Algorithm 1.

```
procedure MultiStart
    f* ← ∞;
    while stopping criterion not satisfied do
        Construct feasible solution:
            S ← ConstructSolution;
            S ← LocalSearch(S);
        if f(S) < f* then
            S* ← S;
            f* ← f(S);
        end
    end
    return S*;
```

Algorithm 1: Pseudo-code for multi-start algorithm.

As it is well-known in the heuristic community, the performance of the local search based algorithms, strongly depends on the properties that the neighborhood imposes on the search space. One of the most important properties is the number of local optima. Given an instance and a neighborhood, the estimation of the number of local optima can help to both measure the instance complexity, and to choose the most efficient neighborhood. In this chapter we review and test several methods to estimate the number of local optima in combinatorial optimization problems.

This chapter is focused on studying the different strategies and methods for generating solutions to launch a succession of local searches for global optimum in the context of two domains: global and combinatorial optimization. We therefore organize it as follows. In Section we describe the developments and strategies applied in the context of global (non-linear) optimization. In Section we introduce notation for combinatorial optimization and provide descriptions for solution construction procedures and multi-start algorithms. The difficulty of finding the global optima of a combinatorial problem, is evaluated through the estimation of its number of local optima in Section . Specifically, we review the methods proposed in the literature and then, in Section we describe the associated experiments and apply statistical test to draw significant conclusions. Concluding remarks are drawn in Section .

Global optimization

As mentioned above, many algorithms have been proposed in the 80s that combine the Monte Carlo method with local search procedures [27], being the *multi-level single linkage* the most relevant. In general terms, the probability of success approaches one as the sample size tends to infinity under very mild assumptions about the objective

function. The convergence for random re-start methods is studied in [39], where the probability distribution used to choose the next starting point can depend on how the search evolves. Some extensions of these methods seek to reduce the number of complete local searches that are performed and increase the probability that they start from points close to the global optimum [12].

From a theoretical point of view, Hu et al [26] study the combination of the *gradient algorithm* with random initializations to find a global optimum. Efficacy of parallel processing, choice of the restart probability distribution and number of restarts are studied for both discrete and continuous models. The authors show that the uniform probability is a good choice for restarting procedures.

Hickernell and Yuan [25] present a multi-start algorithm for unconstrained global optimization based on *quasirandom samples*. Quasirandom samples are sets of deterministic points, as opposed to random points, that are evenly distributed over a set. The algorithm applies an inexpensive local search (steepest descent) on a set of quasirandom points to concentrate the sample. The sample is reduced replacing worse points with new quasirandom points. Any point that is retained for a certain number of iterations is used to start an efficient complete local search. The algorithm terminates when no new local minimum is found after several iterations. An experimental comparison shows that the method performs favorably with respect to other global optimization procedures.

Tu and Mayne [41] describe a multi-start with a clustering strategy for constrained optimization problems. It is based on the characteristics of non-linear constrained global optimization problems and extends a strategy previously tested on unconstrained problems. In this study, variations of multi-start with clustering are considered including a simulated annealing procedure for sampling the design domain and a quadratic programming (QP) sub-problem for cluster formation. The strategies are

evaluated by solving 18 non-linear mathematical problems and six engineering design problems. Numerical results show that the solution of a one-step QP sub-problem helps predict possible regions of attraction of local minima and can enhance robustness and effectiveness in identifying local minima without sacrificing efficiency. In comparison with other multi-start techniques, the strategies of this study are superior in terms of the number of local searches performed, the number of minima found and the number of function evaluations required.

Ugray et al [42] propose OptQuest/Multistart or OQMS, a heuristic designed to find global optima for pure and mixed integer nonlinear problems with many constraints and variables, where all problem functions are differentiable with respect to the continuous variables. It uses OptQuest, a commercial implementation of scatter search developed by OptTek Systems, Inc., to provide starting points for any gradient-based local NLP solver. This solver seeks a local solution from a subset of these points, holding discrete variables fixed. Computational results include 155 smooth NLP and MINLP problems, most with both linear and nonlinear constraints, coded in the GAMS modeling language. Some are quite large for global optimization, with over 100 variables and many constraints. Global solutions to almost all problems are found in a small number of local solver calls, often one or two. An improved version of OQMS is proposed in Ugray et al [43] in terms of the filters to apply the NLP solver.

More recently, Kaucic [29] presents a multi-start particle swarm optimization algorithm for the global optimization of a function subject to bound constraints. The procedure consists of three main steps. In the initialization phase, an opposition learning strategy is performed to improve the search efficiency. Then, a variant of the adaptive velocity based on the differential operator enhances the optimization ability of the particles. Finally, a re-initialization strategy based on two diversity measures for the swarm is act in order to avoid premature convergence and stagnation. The algorithm

is evaluated on a set of 100 global optimization test problems. Comparisons with other global optimization methods show its robustness and effectiveness.

Combinatorial optimization

Boese et al [4] analyze relationships among local minima from the perspective of the best local minimum, finding convex structures in the cost surfaces. Based on the results of that study, they propose a multi-start method where starting points for greedy descent are adaptively derived from the best previously found local minima. In the first step, Adaptive Multi-Start (AMS) heuristics generate r random starting solutions and run a greedy descent method from each one to determine a set of corresponding random local minima. In the second step, *adaptive starting solutions* are constructed based on the local minima obtained so far and improved with a greedy descent method. This improvement is applied several times from each adaptive starting solution to yield corresponding *adaptive local minima*. The authors test this method for the traveling salesman problem and obtain significant speedups over previous multi-start implementations. Hagen and Kahng [22] apply this method for the iterative partitioning problem.

Moreno et al [33] propose a stopping rule for the multi-start method based on a statistical study of the number of iterations needed to find the global optimum. The authors introduce two random variables that together provide a way of estimating the number of global iterations needed to find the global optima: the number of initial solutions generated and the number of objective function evaluations performed to find the global optima. From these measures, the probability that the incumbent solution is the global optimum is evaluated via a normal approximation. Thus, at each global iteration, this value is computed and if it is greater than a fixed threshold, the algorithm stops, otherwise a new solution is generated. The authors illustrate the method using the median p-hub problem.

One of the most well known Multi-start methods is the Greedy Adaptive Search Procedures (GRASP), which was introduced by Feo and Resende Feo and Resende [17]. It was first used to solve set covering problems (Feo and Resende [16]). Each GRASP iteration consists of constructing a trial solution and then applying a local search procedure to find a local optimum (i.e., the final solution for that iteration). The construction step is an adaptive and iterative process guided by a greedy evaluation function. It is iterative because the initial solution is built considering one element at a time. It is greedy because the addition of each element is guided by a greedy function. It is adaptive because the element chosen at any iteration in a construction is a function of previously chosen elements. (That is, the method is adaptive in the sense of updating relevant information from one construction step to the next.). At each stage, the next element to be added to the solution is randomly selected from a candidate list of high quality elements according to the evaluation function. Once a solution has been obtained, it is typically improved by a local search procedure. The improvement phase performs a sequence of moves towards a local optimum solution, which becomes the output of a complete GRASP iteration.

Hagen and Kahng [22] implement an adaptive multi start method for a VLSI partitioning optimization problem where the objective is to minimize the number of signals sent between components. The method consists of two phases. It first generates a set of random starting points and performs the iterative (local search), thus determining a set of local minimum solutions. Then it constructs adaptive starting points derived from the best local minimum solutions found so far. The authors add a preprocessing cluster module to reduce the size of the problem. The resulting Clustering Adaptive Multi Start (CAMS) method is fast and stable, and improves upon previous partitioning results reported in the literature.

Fleurent and Glover [18] propose some adaptive memory search principles to enhance multi-start approaches. The authors introduce a template of a constructive version of Tabu Search based on both, a set of elite solutions and the intensification strategies based on identifying *strongly determined and consistent variables* according to the following definitions:

- *Strongly determined variables* are those whose values cannot be changed without significantly eroding the objective function value or disrupting the values of other variables.
- A *consistent variable* is defined as one that receives a particular value in a significant portion of good solutions.

The authors propose the inclusion of memory structures within the multi-start framework as it is done with tabu search. Computational experiments for the quadratic assignment problem show that these methods improve significantly over previous multi-start methods like GRASP and random restart that do not incorporate memory-based strategies.

Patterson et al [35] introduce a multi-start framework called *Adaptive Reasoning Techniques* (ART), based on memory structures. The authors implement the short term and long term memory functions, proposed in the Tabu Search framework, to solve the Capacitated Minimum Spanning Tree Problem. ART is an iterative, constructive solution procedure that implements learning methodologies on top of memory structures. ART derives its success from being able to learn about, and modify the behavior of a primary greedy heuristic. The greedy heuristic is executed repeatedly, and for each new execution, constraints that prohibit certain solution elements from being considered by the greedy heuristic are introduced in a probabilistic fashion. The active constraints are held in a short-term memory, while a long-term memory holds information regarding the constraints that were in the active memory for the best set of solutions.

Glover [19] approaches the multi-start framework from a different perspective. The author views multi-start methods as an extreme version of the *strategic oscillation* approach. Strategic oscillation is a mechanism used in tabu search to allow the process to visit solutions around a "critical boundary," by approaching such a boundary from both sides.

Braysy et al [5] propose a multi-start local search heuristic for the vehicle routing problem with time windows. The objective in this problem is to design least cost routes for a fleet of identical capacitated vehicles to service geographically scattered customers within pre-specified service time windows. The suggested method uses a two-phase approach. In the first phase, a fast construction heuristic is used to generate several initial solutions. Then, injection trees, an extension of the well-known ejection chain approach [20] are used to reduce the number of routes. In the second phase, two new improvement heuristics, based on CROSS-exchanges [40] are applied for distance minimization. The best solution identified by the algorithm is post-optimized using a threshold accepting post-processor with both intra-route and inter-route improvement heuristics. The resulting hybrid method is shown to be fast, cost-effective, and highly competitive.

Mezmaz et al [32] hybridize the multi-start framework with a model in which several evolutionary algorithms run simultaneously and cooperate to compute better solutions (called *island model*). They propose a solution method in the context of multi-objective optimization on a computational grid. The authors point out that although the combination of these two models usually provides very effective parallel algorithms, experiments on large-size problem instances are often stopped before convergence is achieved. The full exploitation of the cooperation model needs a large amount of computational resources and the management of the fault tolerance issue.

Under the template of a typical multi-start metaheuristic, Lan and DePuy [30] propose Meta-RaPS (Meta-heuristic for Randomized Priority Search), in which several randomization methods and memory mechanisms are present. With the Set Covering Problem (SCP) as the application problem, it is found that these randomization and memory-based methods work well for Meta-RaPS.

Beausoleil et al [3] consider a multi-objective combinatorial optimization problem called Extended Knapsack Problem. By applying multi-start search and path-relinking their solving method rapidly guides the search toward the most balanced zone of the Pareto-optimal front (the zone in which all the objectives are equally important).

Essafi et al [15] propose a multi-start ant based heuristics for a machine line balancing problem. The proposed procedure is a constructive algorithm that assigns operations sequentially to stations. The algorithm builds a feasible solution step by step according to a greedy function that computes the contribution of each unassigned operation to the partial solution under construction based on operational time and weights. The selection of the operation to be added is performed with a roulette wheel mechanism based on the typical ant probability distribution (pheromones of previous assignments). The proposed heuristic is applied to solve a real industry problem.

Dhouib et al [11] propose a multi-start adaptive threshold accepting algorithm (MS-TA) to find multiple Pareto-optimal solutions for continuous optimization problems. Threshold accepting methods (TAs) are deterministic and faster variants of the well-known simulated annealing algorithms, in which every new move is accepted if it is not much worse than the old one. A multi-start technique is applied in this paper to the TA algorithm to allow more diversifications.

Villegas et al [44] propose two hybrid algorithms for the single truck and trailer routing problem. The first one is based on GRASP and variable neighborhood de-

scent (VND), while the second one is and evolutionary local search (ELS). In the first one, large tours are constructed with a randomized nearest neighbor method with a restricted candidate list that ignores capacity constraints and trailer-point selection. VND is applied to improve these initial solutions obtained with GRASP. In the second one, a multi-start evolutionary search is applied starting from an initial solution (giant tour). The best solution found is strongly perturbed to obtain different solutions from which the search is re-started. The perturbation is managed by a mutation operator. The results of the computational experiments on a set of 32 randomly generated instances also unveil the robustness of the proposed metaheuristics, all of them achieving gaps to best known solutions of less than 1% even in the worst case. Among the proposed methods,the multi-start evolutionary local search is more accurate, faster, and scales better than the GRASP/VND.

Estimating the number of local optima

As we have previously seen, multi-start algorithms depend on a neighborhood defined in the search space. Furthermore, the same multi-start algorithm can produce different results in the same instance when is used with two different neighborhoods. Although there are several characteristics of a neighborhood that influence the behavior of a multi-start algorithm, probably, the most relevant is the number of local optima it generates.

The knowledge about the number of local optima that a neighborhood generates in an instance of a Combinatorial Optimization Problem (COP) can have a high impact on the choice of the multi-start algorithm used to solve the instance. On the first hand, different neighborhoods can generate a dramatically different number of local optima. Of course, as a general rule, the higher the size of the neighborhood the lower the number of local optima. However, given two neighborhoods of the same size complexity

(number of local solutions in the same order of magnitude), the one that generates a lower number of local optima will always be preferred.

Given an instance of a COP and a neighborhood, the exact calculation of the number of local optima is impractical, except for extremely low dimensions ($n \leq 14$). Furthermore, this exact calculation requires, in most of the cases, the exhaustive inspection of every solution in the search space, what makes this approach usefulness. Therefore, we have to resort in statistical estimation methods in order to have an approximation of the number of local optima of a landscape. However, even if we were able to exactly calculate the number of local optima, we still would have to face another challenge, that is, how to represent such number. When the dimension of a problem is high (for instance, $n \geq 200$) and the size of the search space is exponential in n, the number of local optima are usually so high that can not be accurately represented in the computer. The alternative choice to represent the proportion of solutions of the search space that are local optima deals with similar issues as this number is too small to be accurately represented. Therefore, the estimation of the number of local optima of an instance of a COP is only plausible for moderate values of the problem size.

In spite of the useful information that the knowledge of the number of local optima can provide in order to choose the best algorithm for solving a COP instance, there have not been many proposals in the literature. This can probably be due to the difficulty of the estimation problem. These proposals are mainly divided into three groups (a good review and a comparison of several proposals can be found in [24]). A first group of proposals try to find expected values on the number of local optima, departing from the hypothesis that the instances have been generated uniformly at random [21; 1; 2]. In a second group we mainly include proposals that have been developed in the metaheuristics community. Finally, a third group includes a set of

proposals that were not designed with the objective of estimating the number of local optima, but were adapted to this problem.

In order to estimate the number of local optima, all the methods share some steps of the process. All of them obtain uniformly at random a sample of size M from the search space. Departing from these solutions and applying a greedy local search algorithm, they finally obtain a sample of local optima (notice that in this sample some local optima could be repeated several times). In addition to that, the concept of basin of attraction is presented in all the estimation methods. Basically, a solution belongs to the basin of attraction of a local optimum if, after applying a greedy local search departing from that solution, it finishes in the corresponding local optimum. Notice that, in case of injective functions the basins of attraction represent a partition of the search space.

Methods proposed in the metaheuristics arena

Most of the methods proposed in the metaheuristics field are mainly due to [13; 14; 36]. Fundamentally, they developed two kinds of methods: a first group where they use confidence intervals to provide lower bounds (with high probability) for the number of local optima and a second group using bias-correcting methods.

In the first group, by assuming that the sizes of the basins of attraction of all the local optima are the same, they manage to get lower bounds by means of several methods:

- *Fist Repetition Time*: A random solution is taken from the search space and a greedy local search algorithm is applied to reach a local optimum. This process continues until a local optimum is repeated for the first time. This number of initial solutions needed until a local optimum is seen twice, is the value used to make the estimation. Note that in this method the sample size is not fixed.

- *Maximal First Repetition Time*: This method is similar to the previous one, except that the sample size is fixed beforehand and the maximal first repetition time (the maximum number of solutions between the first reoccurrence of a local optimum) is used to carry out the estimation.
- *Schnabel Census*: In order to make the estimation, the authors take into account in this case the probability that r different local optima are discovered from a sample of size M.

The main drawback of the previous methods is that they are strongly biased by the assumption that all the local optima have the same size of basin of attraction. In order to overcome this bias, a couple of very well-known methods in the statistical literature are proposed: Jackknife and Bootstrap. These are non-parametric methods that, departing from a biased estimation, try to correct the bias:

- *Jackknife method*: It consists in calculating M new estimations by leaving each time one of the current solutions out from the original sample. These new M estimations are used to modify the original estimation decreasing its bias.
- *Bootstrap method*: This resampling technique consists in obtaining several new samples from the original sample. Each sample has the same size as the original one and is obtained by uniformly at random sampling solutions with replacement from the original sample. On the contrary to the previous method, it assumes a probabilistic model in the basins of attraction as we need to make a new estimation for each re-sampling.

Methods proposed in the field of statistics

The community working in metaheuristics quickly realized that the methods used by biologists and ecologists to calculate the number of species in a population could be easily adapted to the problem of calculating the number of local optima. Particularly

four nonparametric methods [7; 9; 8], were adapted to estimate the number of local optima of instances of COPs [24]. Although they are nonparametric methods, they are based on particular sampling models. An important consideration is that all these methods assume an infinite population, in our case an infinite number of local solutions. This assumption does not impose a big constraint because of the large cardinality of the search space of the COPs:

- *Chao1984 method*: It is based on multinomial sampling. The estimator is the result of adding to the number of local optima obtained from the sample a term that depends only on the number of local optima seen once and twice in the sample.
- *Chao&Lee methods*: These two methods are also based on multinomial sampling. They include the idea of sample coverage (sum of the proportions of the basins of attraction of the local optima observed in the sample). They distinguish between the local optima observed many times in the sample, and those observed a few number of times in the sample. So that the estimators are the sum of the number of local optima observed many times, and some terms that depend on the sample coverage and the number of local optima found few times, trying to compensate for the local optima that have small basins of attraction.
- *Chao&Bunge method*: It is based on a mixed Poisson sampling model, and is closely related to the previous estimators. This method bases the estimation of unobserved local optima on a formula that depends on the expected proportion of duplicates in the sample. It also makes the distinction between the local optima observed many times in the sample, and those observed a few number of times in the sample.

Experiments

The accuracy of the different estimators presented in the previous section has been tested on instances of three different common problems: Permutation Flowshop Scheduling Problem (PFSP), Linear Ordering Problem (LOP) and Quadratic Assignment Problem (QAP). Using these datasets we can test the methods over a wide set of instances with different characteristics. We work with instances for which we already know the number of local optima for different neighborhoods, which allows us to evaluate the accuracy of the different estimations. We report a comparison of the different methods, giving recommendations of the methods that provide the best estimations. Based on the results found in [24], we remove for the analysis the three first methods presented in the previous section, that is: First Repetition Time, Maximal First Repetition Time and Schnabel Census.

Experimental design

The instances used in the experiments are taken from three well-known benchmarks. We work with 5 instances of the PFSP obtained from the Taillards benchmark, 5 instances of the LOP taken from the xLOLIB benchmark [38] and 5 instances of the QAP chosen from the QAPLIB.

The Flowshop Scheduling Problem can be stated as follows: there are b jobs to be scheduled in c machines. A job consists of c operations and the j-th operation of each job must be processed on machine j for a specific processing time without interruption. We consider that the jobs are processed in the same order on different machines. The objective of the PFSP is to find the order in which the jobs have to be scheduled on the machines, minimizing the total flow time.

In the LOP, given a matrix $B = [b_{ij}]_{nxn}$ of numerical entries, we have to find a simultaneous permutation of the rows and columns of B, such that the sum of the

entries above the main diagonal is maximized (or equivalently, the sum of the entries below the main diagonal is minimized).

The QAP is the problem of allocating a set of facilities to a set of locations, with a cost function associated to the distance and the flow between the facilities. The objective is to assign each facility to a location such that the total cost is minimized. Specifically, we are given two $n \times n$ input matrices with real values $\mathbf{H} = [h_{ij}]$ and $\mathbf{D} = [d_{kl}]$, where h_{ij} is the flow between facility i and facility j and d_{kl} is the distance between location k and location l.

For the three problems, we work with instances of permutation size $n = 13$, so the search space is of size $|\Omega| = 13! \approx 6.23 \cdot 10^9$. In the case of the PFSP we consider instances with 13 jobs and 5 machines. Notice that, first, we need to evaluate all the solutions of the search space to know in advance the exact number of local optima. Therefore, working with higher permutation sizes becomes unaffordable.

The local optima of the instances have been calculated according two commonly used operators: the 2-exchange and the insert neighborhoods. Denoting by $\pi = \pi_1\pi_2...\pi_n$ a solution (permutation of size n) of the search space, the 2-exchange neighborhood considers that two solutions are neighbors if one is generated by swapping two elements of the other:

$$N_S(\pi_1\pi_2...\pi_n) = \left\{(\pi'_1\pi'_2...\pi'_n) \mid \pi'_k = \pi_k, \forall k \neq i,j, \pi'_i = \pi_j, \pi'_j = \pi_i, i \neq j\right\}.$$

Two solutions are neighbors under the insert neighborhood (N_I) if one is the result of moving an element of the other one to a different place:

$$\begin{aligned} N_I(\pi_1\pi_2...\pi_n) = &\left\{(\pi'_1\pi'_2...\pi'_n) \mid \pi'_k = \pi_k, \forall k < i \text{ and } \forall k > j, \pi'_k = \pi_{k+1}, \forall i \leq k < j, \pi'_j = \pi_i\right\} \\ &\cup \left\{(\pi'_1\pi'_2...\pi'_n) \mid \pi'_k = \pi_k, \forall k < i \text{ and } \forall k > j, \pi'_i = \pi_j, \pi'_k = \pi_{k-1}, \forall i < k \leq j\right\}. \end{aligned}$$

The different methods for estimating the number of local optima are applied to the 5 instances of each of the considered problems (PFSP, LOP and QAP) using both neighborhoods (N_S and N_I). So, a total of 30 different landscapes are considered. The

reason why we have considered these two neighborhoods is that they provoke different situations for the estimations obtained with the different methods. As the Insert neighborhood explores at each step more solutions than the 2-exchange neighborhood, the number of local optima obtained when using the first neighborhood is probabilistically lower than when assuming the second one. However, as we will see, due to the intrinsic nature of the QAP, given an instance of this problem the number of local optima when using the Insert neighborhood is much higher than when considering the 2-exchange neighborhood.

The estimation methods *Jackknife*, *Bootstrap*, *Chao1984*, *Chao&Lee1*, *Chao&Lee2* and *Chao&Bunge* are applied to each of the landscapes 10 times, and the average estimation value of the 10 repetitions is reported. We apply the methods for different sample sizes: $M = 100, 500, 1000, 5000$.

Results

Our aim is to compare the accuracy of the different methods and their relation to the sample size. We analyze the results according the type of problem and the neighborhood used, and study the effect of the sample size on the methods using different sample sizes. As we realized that in real life we have to face problems of such high dimensions that they have a huge number of local optima, and therefore, the sample we are able to deal with is usually tiny compared to the number of local optima, we are interested in finding methods that do not need such a large sample size to provide a good estimation.

We report in the Table 1 the average estimations (of the 10 repetitions) obtained with the different methods and the different sample sizes for all the instances of the three problems when using both neighborhoods, as well as we indicate the real number

of local optima of each instance. We denote with v the real number of local optima of the instances with the corresponding neighborhood.

We observe in the Table 1 that the method that provides the best results more times is *Chao&Lee2*. In fact, in general, the estimations given by those methods that come from the field of statistics, provide better results than *Bootstrap* and *Jackknife*, above all, when the number of local optima is considerably high.

We carry out a statistical analysis to compare the estimations obtained for the different methods. We consider three different scenarios for comparison. In the first scenario considered, the estimations are grouped in three sets according to the type of problem: PFSP, LOP or QAP. The second scenario considers two different sets that contain the estimations of the instances when using the neighborhoods N_S or N_I. In the last scenario the estimations are grouped in four sets, according to the parameter $M = 100, 500, 1000, 5000$. A nonparametric Friedman's test with level of significance $\alpha = 0.05$ is used to test if there are statistical significant differences among the estimations provided by the 6 methods in the different scenarios. It provides a ranking of the methods while also giving an average rank value for each method. As we always find statistical differences in all the cases, we proceed with a post-hoc test which carries out all pairwise comparisons. Particularly, we use the Holm's procedure fixing the level of significance to $\alpha = 0.05$.

Table 2 shows the ranking obtained for the methods with the Friedman's test in the first scenario, that is, for the instances of the PFSP (first pair of columns), LOP (the pair of columns in the middle), and QAP (last pair of columns). The lower the rank, the worse the performance of the method is. So, the methods are ordered from best to worst. Therefore, the best methods in the three groups are: *Chao&Lee2*, *Chao&Lee1* and *Chao1984*. However, pairwise significant differences are not found between *Chao&Lee1* and *Chao1984*, *Chao1984* and *Bootstrap*, and *Bootstrap* and *Chao&Bunge*, for the

Table 1. Mean of the estimations obtained for all the instances of the PFSP, LOP, and QAP, under the 2-exchange and the Insert neighborhoods.

		Method	M=100	M=500	M=1000	M=5000	M=100	M=500	M=1000	M=5000	M=100	M=500	M=1000	M=5000	M=100	M=500	M=1000	M=5000	M=100	M=500	M=1000	M=5000
			Instance 1. $v = 2386$				Instance 2. $v = 8194$				Instance 3. $v = 1997$				Instance 4. $v = 5119$				Instance 5. $v = 192$			
PFSP	2-exchange	*Jackknife*	158.20	517.30	744.80	1371.20	185.50	709.50	1139.10	2733.60	142.70	397.50	590.40	1098.50	167.40	575.20	886.60	1920.50	77.10	131.80	153.00	**183.10**
		Bootstrap	155.90	562.80	862.00	1786.30	173.20	706.50	1199.00	3281.70	143.60	452.90	688.60	1428.00	161.40	609.20	982.40	2376.60	90.70	170.80	**208.80**	240.10
		Chao1984	400.80	692.80	874.70	1405.30	1148.60	1438.50	1751.30	3105.50	331.70	517.30	774.30	1153.30	544.30	843.40	1161.60	2122.60	94.00	136.20	151.60	178.30
		Chao&Lee1	397.30	725.70	932.00	1401.90	1196.50	1544.50	1884.70	3223.90	340.50	546.50	769.10	1146.60	510.90	889.20	1242.60	2194.20	90.10	133.30	147.70	175.00
		Chao&Lee2	**482.40**	**917.90**	1185.30	1565.30	**1302.50**	**2132.40**	**2605.80**	**4020.70**	**519.70**	**737.80**	1030.10	1301.30	587.60	1145.70	1679.00	2689.10	108.80	146.70	155.00	177.60
		Chao&Bunge	236.50	822.80	**2304.40**	**1901.50**	494.40	390.50	660.30	14054.20	204.60	3789.80	**2399.70**	**1695.90**	**793.20**	**2103.80**	**1887.90**	**6446.60**	**258.10**	**173.30**	162.10	178.60
			Instance 1. $v = 134$				Instance 2. $v = 923$				Instance 3. $v = 506$				Instance 4. $v = 190$				Instance 5. $v = 14$			
	Insert	*Jackknife*	54.30	95.40	104.00	173.40	125.10	317.30	435.90	**955.40**	87.50	208.80	265.10	**520.40**	72.10	127.80	150.60	300.30	12.70	13.30	**14.20**	**14.00**
		Bootstrap	66.60	**124.30**	**141.40**	231.20	129.00	374.60	535.90	1253.50	99.40	252.10	339.90	703.20	85.30	163.00	**199.70**	415.00	16.00	14.50	15.50	14.30
		Chao1984	62.40	100.10	103.20	170.60	256.60	389.90	482.80	997.40	138.40	253.80	277.20	536.60	94.60	138.10	154.10	290.90	11.60	12.90	13.10	**14.00**
		Chao&Lee1	65.80	95.00	100.20	**165.80**	247.70	408.90	488.20	978.10	128.30	263.50	285.70	522.30	83.50	133.90	147.10	**287.10**	12.30	13.40	**14.20**	**14.00**
		Chao&Lee2	**84.00**	103.90	104.60	169.60	**371.80**	545.50	583.90	1092.90	**178.10**	350.40	331.20	570.80	**101.40**	152.20	157.00	296.60	12.50	13.70	14.70	**14.00**
		Chao&Bunge	250.80	121.30	109.20	171.80	112.20	**1202.90**	**1108.50**	1341.40	74.70	**367.70**	**498.50**	652.70	281.00	**212.60**	169.80	303.80	**12.80**	**14.10**	15.10	**14.00**
			Instance 1. $v = 9969$				Instance 2. $v = 3355$				Instance 3. $v = 4732$				Instance 4. $v = 3227$				Instance 5. $v = 6810$			
LOP	2-exchange	*Jackknife*	183.00	732.10	1185.50	3256.30	176.20	642.40	980.70	1946.70	172.80	615.90	921.00	2068.10	136.30	394.10	587.80	1274.40	185.00	746.60	1248.20	2995.20
		Bootstrap	171.30	723.00	1237.20	3754.40	167.50	662.00	1073.30	2458.60	164.80	638.90	1019.00	2539.90	138.50	443.90	685.60	1607.50	172.70	734.40	1291.00	3578.20
		Chao1984	923.50	1633.00	1939.20	4073.00	810.70	1001.10	1225.40	2007.00	842.70	1013.50	1211.00	2321.60	259.90	562.90	765.70	1428.50	1884.60	1523.20	1970.00	3297.30
		Chao&Lee1	935.30	1612.50	2043.00	4580.20	740.40	1003.10	1319.30	2033.40	813.00	1070.80	1299.90	2369.00	257.40	615.10	792.10	1463.30	1757.50	1589.10	2157.70	3485.10
		Chao&Lee2	**987.30**	**2104.70**	**2860.80**	**6487.20**	**904.80**	1185.50	1677.20	2308.20	**1068.70**	**1450.40**	**1744.20**	**2914.70**	**328.00**	**935.00**	1088.80	1785.30	**1966.40**	**1960.20**	2868.70	**4263.30**
		Chao&Bunge	349.80	399.90	680.10	2040.20	144.30	**3188.10**	**2642.10**	**2950.40**	156.90	27501.80	558.90	7544.60	121.70	714.80	**2946.60**	**4654.60**	666.80	14991.50	**7635.90**	10894.20
			Instance 1. $v = 1113$				Instance 2. $v = 60$				Instance 3. $v = 12$				Instance 4. $v = 9$				Instance 5. $v = 67$			
	Insert	*Jackknife*	124.40	335.20	472.70	1093.50	51.90	**60.50**	61.50	137.10	9.60	12.40	12.60	19.30	7.80	9.40	**9.00**	20.60	49.10	63.80	**60.90**	108.90
		Bootstrap	128.80	391.80	578.00	1431.90	66.90	82.00	77.70	169.70	**11.80**	13.40	14.40	20.20	9.70	13.00	10.20	23.00	**64.00**	86.00	78.70	128.30
		Chao1984	224.00	396.30	528.70	**1121.70**	53.20	57.70	60.40	133.70	8.80	**11.70**	**12.10**	**18.20**	7.20	**9.00**	8.90	**20.20**	51.00	**65.40**	59.90	107.30
		Chao&Lee1	235.60	445.10	556.90	1137.30	52.30	57.60	59.60	**133.40**	9.10	13.20	12.60	20.10	**9.20**	9.20	8.90	20.30	47.80	60.40	58.70	**106.20**
		Chao&Lee2	351.70	595.80	**693.60**	1285.00	**57.30**	57.70	**59.70**	133.60	9.30	14.20	12.60	22.30	10.40	9.40	8.90	20.40	51.30	61.30	59.10	106.60
		Chao&Bunge	**1111.70**	**859.80**	8551.50	1630.70	67.70	57.70	**59.70**	133.60	9.20	14.80	12.40	19.00	8.20	9.40	8.90	20.50	57.00	61.80	59.10	106.80
			Instance 1. $v = 18720$				Instance 2. $v = 3472$				Instance 3. $v = 62$				Instance 4. $v = 173568$				Instance 5. $v = 563$			
QAP	2-exchange	*Jackknife*	197.10	958.80	1811.60	6594.90	171.70	598.50	910.60	1889.80	27.40	42.30	48.80	57.40	198.70	992.00	1965.20	9280.00	94.40	204.50	264.00	368.00
		Bootstrap	180.00	873.60	1678.60	6740.80	164.80	628.60	1009.20	2369.00	33.00	**54.30**	**66.00**	73.10	180.90	891.60	1771.50	8532.00	105.20	254.10	338.80	**494.60**
		Chao1984	2000.70	9847.80	8007.20	9818.00	609.40	870.40	1191.10	2009.20	39.90	48.70	47.50	**58.00**	580.10	48561.00	49112.70	51951.40	139.00	227.10	286.40	373.90
		Chao&Lee1	**2025.00**	9914.80	7973.80	9750.90	616.20	934.40	1247.50	2026.20	32.30	44.10	49.50	54.90	**585.00**	**48783.40**	**49601.10**	51848.80	157.80	224.70	278.60	351.90
		Chao&Lee2	**2025.00**	**10323.30**	**8296.00**	10683.60	**679.90**	1188.50	1652.60	2351.40	39.80	50.90	53.30	56.00	**585.00**	**48783.40**	**49601.10**	53432.60	257.00	266.60	320.90	365.80
		Chao&Bunge	1050.00	3752.20	128363.10	**17782.40**	260.10	**1330.50**	**4564.80**	**3386.60**	**55.20**	130.20	66.70	56.50	340.00	24529.10	25050.50	**174841.50**	**261.30**	**521.50**	**457.40**	378.10
			Instance 1. $v = 4615326$				Instance 2. $v = 1501175$				Instance 3. $v = 579275$				Instance 4. $v = 6712090$				Instance 5. $v = 83240$			
	Insert	*Jackknife*	198.70	995.70	1991.60	9821.40	198.70	995.30	1988.20	9795.60	**199.00**	987.00	1947.40	9108.50	**199.00**	997.70	1997.30	9955.30	193.70	884.40	1616.80	5935.70
		Bootstrap	180.80	892.70	1787.00	8851.40	180.80	892.50	1785.20	8837.20	181.00	889.90	1758.20	8408.30	181.00	894.50	1791.90	8929.10	179.00	829.50	1548.00	6141.30
		Chao1984	580.10	93351.20	210869.40	226925.80	580.10	45585.50	196680.90	199572.40	100.00	43655.70	33049.70	54343.30	100.00	25200.50	225000.60	938528.00	2902.10	3923.60	4892.70	10856.90
		Chao&Lee1	**585.00**	**93575.00**	**211316.70**	226706.30	**585.00**	41316.70	179342.60	199911.00	100.00	38924.50	32904.70	58780.20	100.00	**25275.00**	**225250.00**	933604.60	**2950.10**	3925.40	5521.50	12057.10
		Chao&Lee2	**585.00**	**93575.00**	**211316.70**	**245043.10**	**585.00**	**45980.20**	**198268.50**	**213030.90**	100.00	**45088.10**	**36647.70**	**83214.20**	100.00	**25275.00**	**225250.00**	**976729.70**	**2950.10**	**4803.90**	**8319.40**	**18835.30**
		Chao&Bunge	340.00	46925.00	105933.30	26271.90	340.00	16916.30	74087.70	34635.20	100.00	12960.90	8107.80	4677.70	100.00	12850.00	113000.00	304803.40	1499.90	458.70	857.40	3354.70

PFSP instances, and between *Bootstrap* and *Chao&Bunge*, *Bootstrap* and *Jackknife*, and *Jackknife* and *Chao&Bunge*, for the LOP and the QAP instances.

In Table 3, the ranking for the methods is shown, but this time for the second scenario, that is, when grouping the estimations for the instances using the 2-exchange neighborhood N_S (the first pair of columns), and the Insert neighborhood N_I (the last pair of columns). In the first case the Holm's procedure states that significant differences exist among each pair of methods. Nevertheless, for the second neighborhood significant differences are not found between *Chao&Lee1* and *Chao1984*, and between *Chao&Bunge* and *Bootstrap*. From this table we can observe that, also in this scenario the best estimations are provided by *Chao&Lee2*, *Chao&Lee1* and *Chao1984*.

Table 2. Average rankings of the methods according to the type of problem

PFSP		LOP		QAP	
Method	Ranking	Method	Ranking	Method	Ranking
Chao&Lee2	4.85	*Chao&Lee2*	4.83	*Chao&Lee2*	4.77
Chao&Lee1	3.74	*Chao&Lee1*	4.08	*Chao&Lee1*	3.97
Chao1984	3.51	*Chao1984*	3.78	*Chao1984*	3.60
Bootstrap	3.37	*Chao&Bunge*	2.82	*Jackknife*	2.97
Chao&Bunge	3.05	*Jackknife*	2.78	*Bootstrap*	2.88
Jackknife	2.47	*Bootstrap*	2.69	*Chao&Bunge*	2.81

Finally, Table 4 shows the ranking obtained for the methods when the estimations are separated according to the sample size used by the methods $M = 100, 500, 1000, 5000$. Again, we find that the three best methods are *Chao&Lee2*, *Chao&Lee1* and *Chao1984*. Significant differences are not found between *Chao&Lee1* and *Chao1984*, and between *Bootstrap* and *Jackknife*, when using $M = 100$. For $M = 500$ there are not significant differences among Bootrstrap, *Chao&Bunge* and *Jackknife*, while significant differences are not found between *Bootstrap* and *Chao&Bunge* when $M = 1000$. Finally, when taking $M = 5000$, there are not significant differences among *Chao&Lee1*, *Chao1984* and *Chao&Bunge*, and between *Jackknife* and *Bootstrap*.

Table 3. Average rankings of the methods according to the neighborhood.

N_S		N_I	
Method	Ranking	Method	Ranking
Chao&Lee2	5.22	*Chao&Lee2*	4.41
Chao&Lee1	3.94	*Chao&Lee1*	3.92
Chao1984	3.61	*Chao1984*	3.66
Bootstrap	3.12	*Jackknife*	3.26
Chao&Bunge	2.88	*Chao&Bunge*	2.91
Jackknife	2.23	*Bootstrap*	2.84

Table 4. Average rankings of the methods according to the sample size

M=100		M=500		M=1000		M=5000	
Method	Ranking	Method	Ranking	Method	Ranking	Method	Ranking
Chao&Lee2	4.72	*Chao&Lee2*	5.01	*Chao&Lee2*	5.01	*Chao&Lee2*	4.54
Chao&Lee1	3.87	*Chao&Lee1*	4.06	*Chao&Lee1*	3.97	*Chao&Lee1*	3.83
Chao1984	3.65	*Chao1984*	3.63	*Chao1984*	3.57	*Chao1984*	3.67
Bootstrap	3.19	*Bootstrap*	2.93	*Bootstrap*	3.00	*Chao&Bunge*	3.48
Jackknife	3.12	*Chao&Bunge*	2.69	*Chao&Bunge*	2.93	*Bootstrap*	2.81
Chao&Bunge	2.46	*Jackknife*	2.67	*Jackknife*	2.52	*Jackknife*	2.67

Although the statistical analysis gives a global picture of the performance of the methods, and we have concluded that *Chao&Lee2* provides the best solutions with significant differences between this method and the remaining methods, it is also relevant to consider some aspects that are not reflected in the hypothesis tests. The closeness of the estimations provided by the methods to the value we want to estimate is the most important factor. Obviously, there are methods that estimate better than others, but it does not mean that the estimations provided by the best methods are close enough to the real value. In order to check if the methods provide useful estimations, and so as

to add more information to that deduced from Table 1, the average errors of the estimations with respect to the real number of local optima are calculated. Tables 5, 6 and 7 show the average relative errors and the standard deviations (in brackets) grouped by the neighborhood used and the sample size, for all the instances of the PFSP, LOP and QAP, respectively.

A general conclusion deduced from Tables 5, 6 and 7, is that, in general terms, the estimations improve as the sample size grows, and they are worse as the number of local optima increases. Specifically, as we saw in the Table 1, the instances of the LOP with the Insert neighborhood, had a low number of local optima, and we observe in Table 6 that the average error is lower than in the rest of the cases. Moreover, the instances of the QAP with the Insert neighborhood, had a high number of local optima, and the errors obtained for those instances (in Table 7) are higher than for the rest. It is remarkable that for sample size $M = 100$ and $M = 500$ the estimations are far from the real value. However, there is one method which behavior is different and does not follow the general lines. This is the *Chao&Bunge* method. This estimation method does not necessarily improve as the sample size grows, and the standard deviation presented in most of the cases is really high comparing with other methods. We consider that *Chao&Bunge* is a very unstable method. The instability of this method is a consequence of the variability on the estimation of the parameter that represents the expected proportion of duplicates in the sample [24]. This estimation is unreliable when we have a sample where many local optima are seen only once, but there is a small number of local optima seen a low (but higher than one) number of times. These particularities are commonly found when the sample size is small with respect to the number of local optima, or even when the variance of the sizes of the attraction basins of the local optima is high.

Table 5. Average relative errors and standard deviations (in brackets) of the estimations provided by the different methods for the instances of the PFSP, according to the neighborhood and the sample size M.

		M=100	***M=500***	***M=1000***	***M=5000***
2-exchange	*Jackknife*	0.88 (0.14)	0.74 (0.22)	0.66 (0.24)	0.44 (0.22)
	Bootstrap	0.87 (0.17)	0.69 (0.30)	0.61 (0.28)	0.38 (0.15)
	Chao1984	0.79 (0.15)	0.68 (0.21)	0.60 (0.22)	0.42 (0.20)
	Chao&Lee1	0.79 (0.14)	0.67 (0.20)	0.60 (0.20)	0.42 (0.19)
	Chao&Lee2	**0.74 (0.18)**	**0.60 (0.20)**	**0.51 (0.19)**	0.35 (0.16)
	Chao&Bunge	0.89 (0.26)	1.06 (2.23)	0.84 (1.12)	**0.28 (0.38)**
Insert	*Jackknife*	0.60 (0.28)	0.39 (0.22)	0.30 (0.18)	0.19 (0.23)
	Bootstrap	0.58 (0.26)	0.28 (0.23)	0.21 (0.16)	0.54 (0.40)
	Chao1984	0.53 (0.23)	0.34 (0.19)	0.28 (0.17)	0.19 (0.21)
	Chao&Lee1	0.54 (0.24)	0.34 (0.18)	0.29 (0.15)	**0.17 (0.19)**
	Chao&Lee2	**0.45 (0.22)**	**0.25 (0.15)**	0.24 (0.14)	0.23 (0.20)
	Chao&Bunge	0.92 (1.40)	0.50 (0.78)	**0.20 (0.18)**	0.32 (0.24)

As a conclusion from this analysis, we highly recommend the use of the *Chao&Lee2* method to estimate the number of local optima. According to the hypothesis test, this estimation method provides the best results. Although in the statistical analysis is not reflected, the *Chao&Bunge* method gives also good estimations in a high number of occasions, however, its instability provokes that we do not know if we should or should not trust it. Therefore, we recommend to execute both methods independently. If the results provided are close, *Chao&Bunge* is usually the choice, otherwise, select *Chao&Lee2*.

Table 6. Average relative errors and standard deviations (in brackets) of the estimations provided by the different methods for the instances of the LOP, according to the neighborhood and the sample size M.

		M=100	***M=500***	***M=1000***	***M=5000***
2-exchange	*Jackknife*	0.96 (0.01)	0.87 (0.04)	0.81 (0.06)	0.56 (0.08)
	Bootstrap	0.97 (0.01)	0.87 (0.04)	0.79 (0.06)	0.47 (0.12)
	Chao1984	0.83 (0.14)	0.79 (0.06)	0.73 (0.06)	0.52 (0.07)
	Chao&Lee1	0.84 (0.12)	0.78 (0.05)	0.71 (0.07)	0.49 (0.06)
	Chao&Lee2	**0.80 (0.15)**	**0.71 (0.07)**	**0.62 (0.09)**	**0.37 (0.05)**
	Chao&Bunge	0.95 (0.06)	2.38 (8.21)	1.12 (1.44)	0.51 (0.47)
Insert	*Jackknife*	0.33 (0.30)	0.18 (0.27)	0.16 (0.22)	0.77 (0.49)
	Bootstrap	**0.28 (0.32)**	0.37 (0.21)	0.26 (0.17)	1.05 (0.61)
	Chao1984	0.35 (0.25)	**0.18 (0.24)**	0.14 (0.20)	**0.72 (0.47)**
	Chao&Lee1	0.35 (0.28)	0.19 (0.23)	0.14 (0.19)	0.75 (0.48)
	Chao&Lee2	0.33 (0.33)	**0.18 (0.24)**	**0.12 (0.15)**	0.82 (0.51)
	Chao&Bunge	0.49 (0.74)	0.31 (0.70)	1.39 (4.95)	0.83 (0.37)

Conclusions

The objective of this study has been to extend and advance the knowledge associated to implementing multi-start procedures. Unlike other well-known methods, it has not yet become widely implemented and tested as a metaheuristic itself for solving complex optimization problems.

We have reviewed different methods for estimating the number of local optima of instances of combinatorial optimization problems. We have compared some methods in the optimization field with methods previously used for estimating the number of species in a population in the field of statistics. The methods have been applied to instances of three different problems (PFSP, LOP and QAP), under two neighborhoods

Table 7. Average relative errors and standard deviations (in brackets) of the estimations provided by the different methods for the instances of the QAP, according to the neighborhood and the sample size M.

		M=100	***M=500***	***M=1000***	***M=5000***
2-exchange	*Jackknife*	0.87 (0.17)	0.75 (0.25)	0.67 (0.28)	0.50 (0.29)
	Bootstrap	0.84 (0.20)	0.69 (0.33)	0.62 (0.34)	0.44 (0.32)
	Chao1984	0.77 (0.24)	0.57 (0.21)	0.53 (0.19)	0.41 (0.19)
	Chao&Lee1	0.78 (0.20)	0.56 (0.21)	0.53 (0.19)	0.42 (0.19)
	Chao&Lee2	**0.73 (0.25)**	**0.52 (0.22)**	**0.49 (0.19)**	0.38 (0.19)
	Chao&Bunge	0.91 (0.40)	0.87 (1.41)	1.89 (5.45)	**0.27 (0.39)**
Insert	*Jackknife*	1.00 (0.00)	1.00 (0.00)	1.00 (0.01)	0.98 (0.03)
	Bootstrap	1.00 (0.00)	1.00 (0.00)	1.00 (0.01)	0.98 (0.03)
	Chao1984	**0.99 (0.02)**	**0.96 (0.04)**	**0.93 (0.06)**	0.89 (0.04)
	Chao&Lee1	**0.99 (0.02)**	0.97 (0.03)	0.94 (0.05)	0.89 (0.04)
	Chao&Lee2	**0.99 (0.02)**	**0.96 (0.04)**	**0.93 (0.06)**	**0.86 (0.06)**
	Chao&Bunge	1.00 (0.01)	0.99 (0.01)	0.98 (0.03)	0.98 (0.03)

(2-exchange and Insert). The main conclusions observed in the three scenarios are that, in general, the higher the sample used by the methods, the more precise the estimations, and the higher the number of local optima, the worse the estimations provided. Based on the results observed through the experiments, we provide the following rules of thumb: we recommend using *ChaoBunge* and *Chao&Lee2*. Due to the instability observed for *ChaoBunge*, both methods should be executed independently, and compare both results. If the results provided are considerably far, *ChaoBunge* is probably giving an unreliable estimation and we should select *Chao&Lee2*. However, if analyzing the sample we realize that each (or most) of the initial solutions reach different local optima, none of the previous methods can be applied. In this case, we can base our estimator on the proportion of local optima over the sample [6; 21].

Acknowledgments

The first author was partially supported by grants TIN2009-07516 and TIN2012-35632 of *Ministerio de Ciencia e Innovación* of Spain. The third author was partially supported by grants 308687/2010-8 and 483243/2010-8 of CNPq, *Conselho Nacional de Desenvolvimento Científico e Tecnológico* of Brazil and by grants E-26/110.552/2010 and E-26/102.954/2011 of FAPERJ, *Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro*, Brazil. This work has been partially supported by the Saiotek and Research Groups 2013-2018 (IT- 609-13) programs (Basque Government), TIN2013-41272P (Spanish Ministry of Science and Innovation), COMBIOMED network in computational biomedicine (Carlos III Health Institute).

References

1. Albrecht A, Lane P, Steinhofel K (2008) Combinatorial landscape analysis for k-SAT instances. In: Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on, pp 2498 –2504
2. Albrecht A, Lane P, Steinhofel K (2010) Analysis of Local Search Landscapes for k-SAT Instances. Mathematics in Computer Science 3(4):465–488
3. Beausoleil R, Baldoquin G, Montejo R (2008) A multi-start and path relinking methods to deal with multiobjective knapsack problems. Annals of Operations Research 157:105–133
4. Boese K, Kahng A, Muddu S (1994) A new adaptive multi-start technique for combinatorial global optimisation. Operations Research Letters 16:103–113
5. Braysy O, Hasle G, Dullaert W (2004) A multi-start local search algorithm for the vehicle routing problem with time windows. European J of Operational Research 159:586–605
6. Caruana R, Mullin M (1999) Estimating the Number of Local Minima in Big, Nasty Search Spaces. In: In Proceedings of IJCAI-99 Workshop on Statistical Machine Learning for Large-Scale Optimization
7. Chao A (1984) Nonparametric Estimation of the Number of Classes in a Population. Scandinavian Journal of Statistics 11(4):265–270

8. Chao A, Bunge J (2002) Estimating the Number of Species in a Stochastic Abundance Model. Biometrics 58(3):531–539
9. Chao A, Lee SM (1992) Estimating the Number of Classes via Sample Coverage. Journal of the American Statistical Association 87(417):210–217
10. Crowston WB, Glover F, Thompson GL, Trawick JD (1963) Probabilistic and parametric learning combinations of local job shop scheduling rules. Tech. Rep. 117, Carnegie-Mellon University, Pittsburgh
11. Dhouib S, Kharrat A, Chabchoub H (2010) A multi-start threshold accepting algorithm for multiple objective continuous optimization problems. International J for Numerical Methods in Engineering 83:1498–1517
12. DQ DM, Meewella C (1988) A non-clustering multistart algorithm for global optimization. In: Bensoussan, Lions (eds) Analysis and Optimization of Systems, Lect Notes in Control Inf. Sci, Springer, pp 111–117
13. Eremeev AV, Reeves CR (2002) Non-parametric Estimation of Properties of Combinatorial Landscapes. In: Cagnoni S, Gottlieb J, Hart E, Middendorf M, Raidl G (eds) Applications of Evolutionary Computing, Lecture Notes in Computer Science, vol 2279, Springer Berlin / Heidelberg, pp 31–40
14. Eremeev AV, Reeves CR (2003) On Confidence Intervals for the Number of Local Optima. In: In Proceedings of EvoWorkshops 2003, pp 224–235
15. Essafi M, Delorme X, Dolgui A (2010) Balancing lines with CNC machines: A multi-start and based heuristic. CIRP J of Manufacturing Science and Technology 2:176–182
16. Feo T, Resende M (1989) A probabilistic heuristic for a computationally difficult set covering problem. Operations Research Letters 8:67–71
17. Feo T, Resende M (1995) Greedy randomized adaptive search procedures. J of Global Optimization 6:109–133
18. Fleurent C, Glover F (1999) Improved constructive multi-start strategies for the quadratic assignment problem using adaptive memory. INFORMS J on Computing 11:198–204
19. Glover F (2000) Multi-start and strategic oscillation methods - Principles to exploit adaptive memory. In: Laguna M, Gonzalez-Velarde J (eds) Computing tools for modeling optimization and simulation, Kluwer Academic Publishers, pp 1–25
20. Glover F, Laguna M (1997) Tabu search. Kluwer Academic Publishers

21. Grundel D, Krokhmal P, Oliveira C, Pardalos P (2007) On the Number of Local Minima for the Multidimensional Assignment Problem. Journal of Combinatorial Optimization 13:1–18
22. Hagen L, Kahng A (1997) Combining problem reduction and adaptive multi-start: A new technique for superior iterative partitioning. IEEE Trans on CAD 16:709–717
23. Held M, Karp R (1970) The traveling-salesman problem and minimum spanning trees. Operations Research 18:1138–1162
24. Hernando L, Mendiburu A, Lozano JA (2013) An evaluation of methods for estimating the number of local optima in combinatorial optimization problems. Evolutionary Computation 21(4):625–658
25. Hickernell F, Yuan Y (1997) A simple multistart algorithm for global optimization. OR Trans 1:1–11
26. Hu X, Shonkwiler R, Spruill M (1994) Random restarts in global optimization. Georgia Institute of technology 1:1–10
27. Kan AR, Timmer G (1987) Stochastic global optimization methods (part ii): Multi level methods. Mathematical Programming 39:57–78
28. Kan AR, Timmer G (1998) Global optimization. In: Kan R, Todds (eds) Handbooks in operations research and management science, North Holland, pp 631–662
29. Kaucic M (2013) A multi-start opposition-based particle swarm optimization algorithm with adaptive velocity for bound constrained global optimization. Journal of Global Optimization 55:165–188
30. Lan G, DePuy G (2006) On the effectiveness of incorporating randomness and memory into a multi-start metaheuristic with application to the set covering problem. Computers and Industrial Engineering 51:362–374
31. Martí R, Resende M, Ribeiro C (2013) Multi-start methods for combinatorial optimization. European Journal of Operational Research 226 (1):1–8
32. Mezmaz M, Melab N, Talbi E (2006) Using the multi-start and island models for parallel multi-objective optimization on the computational grid. In: Second IEEE International Conference on e-Science and Grid Computing
33. Moreno J, Mladenovic N, Moreno-Vega J (1995) An statistical analysis of strategies for multistart heuristic searches for p-facility location-allocation problems. In: Eighth Meeting of the EWG on Locational Analysis Lambrecht
34. Muth JF, Thompson GL (1963) Industrial Scheduling. Prentice-Hall

35. Patterson R, Pirkul H, Rolland E (1999) Adaptive reasoning technique for the capacitated minimum spanning tree problem. J of Heuristics 5:159–180

36. Reeves C, Aupetit-Bélaidouni M (2004) Estimating the Number of Solutions for SAT problems. In: Yao X, Burke E, Lozano J, Smith J, Merelo-Guervós J, Bullinaria J, Rowe J, Tino P, Kabán A, Schwefel HP (eds) Parallel Problem Solving from Nature - PPSN VIII, Lecture Notes in Computer Science, vol 3242, Springer Berlin / Heidelberg, pp 101–110

37. Resende M, Ribeiro C (2010) Greedy randomized adaptive search procedures: Advances and applications. In: Gendreau M, Potvin JY (eds) Handbook of Metaheuristics, 2nd edn, Springer, pp 293–319

38. Schiavinotto T, Stützle T (2004) The linear ordering problem: instances, search space analysis and algorithms. Journal of Mathematical Modelling and Algorithms

39. Solis F, Wets R (1981) Minimization by random search techniques. Math Oper Res 6:19–30

40. Taillard E, Badeau P, Gendreau M, Guertin F, Potvin J (1997) A tabu search heuristic of the vehicle routing problem with time windows. Transportation Science 31:170–186

41. Tu W, Mayne R (2002) An approach to multi-start clustering for global optimization with non-linear constraints. Int J Numer Methods Eng 53:2253–2269

42. Ugray Z, Lasdon L, Plummer J, Glover F, Kelly J, Martí R (2007) Scatter search and local nlp solvers: A multistart framework for global optimization. INFORMS Journal on Computing 19(3):328–340

43. Ugray Z, Lasdon L, Plummer J, Bussieck M (2009) Dynamic filters and randomized drivers for the multi-start global optimization algorithm msnlp. Optimization Methods & Software 24:4–5

44. Villegas J, Prins C, Prodhon C, Medaglia A, Velasco N (2010) GRASP/VND and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots. Engineering Applications of Artificial Intelligence 23:780–794