GRASP

PAOLA FESTA AND MAURICIO G.C. RESENDE

Abstract. GRASP (Greedy Randomized Adaptive Search Procedures) is a multistart metaheuristic for computing good-quality solutions of combinatorial optimization problems. Each GRASP iteration is usually made up of a construction phase, where a feasible solution is constructed, and a local search phase which starts at the constructed solution and applies iterative improvement until a locally optimal solution is found. Typically, the construction phase of GRASP is a randomized greedy algorithm, but other types of construction procedures have been also proposed. Repeated applications of a construction procedure yields diverse starting solutions for the local search. This report gives an overview of GRASP describing its basic components and enhancements to the basic procedure, including reactive GRASP and intensification strategies.

1. Introduction

Given a finite, or countably infinite, solution set X and a real-valued objective function $f : X \to R$, in any combinatorial optimization problem one seeks a solution $x^* \in X$ with $f(x^*) \leq f(x)$, $\forall\ x \in X$. Several of these problems can be solved in polynomial time, but many of them are computationally intractable, since exact polynomial-time algorithms to solve them are unknown [60]. Furthermore, most real-world problems found in industry and government are either computationally intractable by their nature, or sufficiently large so as to preclude the use of exact algorithms. In such cases, heuristic methods are usually employed to find good, but not necessarily guaranteed optimal solutions. The effectiveness of these methods depends upon their ability to adapt to avoid entrapment at local optima and exploit the basic structure of the problem. Building on these notions, various heuristic search techniques have been developed that have demonstrably improved our ability to obtain good solutions to difficult combinatorial optimization problems. The most promising of such techniques include simulated annealing [80], tabu search [62, 63, 66], genetic algorithms [71], biased random key genetic algorithms [72], scatter search and path-relinking [68], variable neighborhood search [74], and GRASP (Greedy Randomized Adaptive Search Procedures) [43, 44].

GRASP (Greedy Randomized Adaptive Search Procedures) is a multistart metaheuristic for producing good-quality solutions of combinatorial optimization problems. Unlike Ant Colony [39] and Evolutionary Algorithms [19], GRASP is not nature inspired, i.e., it is not inspired by the principles of natural evolution in the sense of nature's capability to evolve living beings to keep them well adapted to

Date: October 2017.

Key words and phrases. Metaheuristics, GRASP, greedy randomized, local search, heuristics.

Parts of this paper were written while the second author was employed at AT&T Labs Research.

Cite as P. Festa and M.G.C. Resende, *GRASP*, in *Handbook of Heuristics*, R. Martí, P.M. Pardalos, and M.G.C. Resende, Eds., pp. 465–488, Springer International Publishing, 2018.

Table 1. Applications of GRASP: operations research problems.

Routing	[14, 18, 23, 31, 83]
Logic	[38, 52, 102, 110, 112]
Covering and partition	[12, 15, 43, 61, 73, 109]
Location	[35, 36, 1, 37, 81, 129]
Minimum Steiner tree	[29, 94, 92, 120]
Optimization in graphs	[2, 17, 45, 84, 101, 114]
Assignment	[4, 42, 59, 88, 95, 103, 104, 111, 125]
Timetabling and scheduling	[5, 7, 11, 10, 24, 41, 46, 47, 85, 90, 121, 122, 123, 124]

Table 2. Applications of GRASP: some industrial applications.

Manufacturing	[6, 21, 22, 27, 82, 98]
Transportation	[14, 20, 42, 126]
Telecommunications	[8, 9, 34, 81, 89, 105, 108, 127]
Graph and map drawing	[53, 54, 87, 91, 114]
Power systems	[25, 26, 40]
Computational biology	[50, 69, 77]
VLSI	[13, 12]

their environment. Instead, GRASP proceeds in iterations and each GRASP iteration is usually made up of a construction phase, where a solution (feasible or even unfeasible) is constructed, and a local search phase that starts at the constructed solution and applies iterative improvement until a locally optimal solution is found. While, in general, the construction phase of GRASP is a randomized greedy algorithm, other types of construction procedures have been proposed. Repeated applications of a construction procedure yields diverse starting solutions for the local search and the best local optimal solution found over all GRASP iterations is returned as final solution.

This report overviews GRASP by describing its basic components along with some among the most fruitful proposed enhancements, including reactive GRASP, intensification strategies, and hybridization with other metaheuristics. The report is organized as follows. Basic components, alternative construction mechanisms and local search characteristics are described in the next Section. Enhancements to the basic procedure, including reactive GRASP and intensification strategies, are discussed in the Section **Enhancements**. Section **Hybridizations** describes several state-of-the-art hybridizations of GRASP with other metaheuristics, while in Section **Automatic tuning** a few techniques are described to automatically tune the typical GRASP parameters. Tables 1 and 2 report a number of GRASP implementations that have appeared in the literature, covering a wide range of applications in several and heterogenous fields. The reader can refer to [57, 55, 56], which contain annotated bibliographies of the GRASP literature from 1989 to 2008.

2. Basic components

Given a finite solution set X and a real-valued objective function $f : X \to R$ to be minimized, a basic GRASP metaheuristic [43, 44] is a multi-start or iterative method, in which each iteration consists of two phases: construction of a solution and local search.

```
algorithm GRASP(f(·), g(·), MaxIterations, Seed)
    x_best:=∅; f(x_best):=+∞;
    for k = 1, 2, ..., MaxIterations→
        x:=ConstructGreedyRandomizedSolution(Seed, g(·));
        if (x not feasible) then
            x:=repair(x);
        endif
        x:=LocalSearch(x, f(·));
        if (f(x) < f(x_best)) then
            x_best:=x;
        endif
    endfor;
    return(x_best);
end GRASP
```

Figure 1. Pseudo-code of a basic GRASP for a minimization problem.

The construction phase builds a solution x, usually feasible, but optionally also infeasible. If x is not feasible, a repair procedure may be invoked to obtain feasibility. Once a solution x is obtained, its neighborhood is investigated by the local search until a local minimum is found. The best overall solution is kept as the result. An extensive survey of the literature is given in [57]. The pseudo-code in Figure 1 illustrates the main blocks of a GRASP procedure for minimization, in which `MaxIterations` iterations are performed and `Seed` is used as the initial seed for the pseudorandom number generator.

2.1. **Construction phase.** Starting from an empty solution, a complete solution is iteratively constructed in the construction phase, one element at a time (see Figure 2). The basic GRASP construction phase is similar to the semi-greedy heuristic proposed independently by [76]. At each construction iteration, the choice of the next element to be added is determined by ordering all candidate elements (i.e. those that can be added to the solution) in a candidate list C with respect to a greedy function $g : C \rightarrow R$. This function measures the (myopic) benefit of selecting each element. The heuristic is adaptive because the benefits associated with every element are updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous element. The probabilistic component of a GRASP is characterized by randomly choosing one of the best candidates in the list, but not necessarily the top candidate. The list of best candidates is called the *restricted candidate list* (RCL). This choice technique allows for different solutions to be obtained at each GRASP iteration, but does not necessarily compromise the power of the adaptive greedy component of the method.

The most used technique to build the RCL applies the $\min\max-\alpha$ percentage rules, which will be explained in the following.
At any GRASP iteration, let g_{min} and g_{max} be the smallest and the largest incremental costs, respectively, i.e.

$$g_{min} = \min_{i \in C} g(i), \qquad g_{max} = \max_{i \in C} g(i). \tag{1}$$

```
procedure ConstructGreedyRandomizedSolution(Seed, g(·))
    x:=∅;
    Sort the candidate elements i according to their incremental costs g(i);
    while (x is not a complete solution)→
        RCL:=MakeRCL();
        v:=SelectIndex(RCL, Seed);
        x := x ∪ {v};
        Resort remaining candidate elements j according to g(j);
    endwhile;
    return(x);
end ConstructGreedyRandomizedSolution;
```

FIGURE 2. Basic GRASP construction phase pseudo-code.

```
procedure ConstructGreedyRandomizedSolution(Seed, α, k, g(·))
    x:=∅;
    Initialize the candidate set C by all elements;
    Evaluate the incremental cost g(i) for all i ∈ C;
    while (|C| > 0)→
        g_min := min_{i∈C} g(i); g_max := max_{i∈C} g(i);
        if (CB RCL is used) then
            Sort candidate elements i ∈ C according to g(i);
            RCL:=C[1 ··· k];
        else RCL:={i ∈ C | g(i) ≤ g_min + α(g_max − g_min)};
        endif;
        v:=SelectIndex(RCL, Seed);
        x := x ∪ {v};
        Update the candidate set C;
        Reevaluate the incremental costs g(i) for all i ∈ C;
    endwhile;
    return(x);
end ConstructGreedyRandomizedSolution;
```

FIGURE 3. Refined pseudo-code of the GRASP construction phase.

The RCL is made up of elements $i \in C$ with the best (i.e., the smallest) incremental costs $g(i)$. There are two main mechanisms to build this list: a *cardinality-based* (CB) and a *value-based* (VB) mechanism. In the CB case, the RCL is made up of the k elements with the best incremental costs, where k is a parameter. In the VB case, the RCL is associated with a parameter $\alpha \in [0, 1]$ and a threshold value $\mu = g_{min} + \alpha(g_{max} - g_{min})$. All candidate elements i whose incremental cost $g(i)$ is no greater than the threshold value are inserted into the RCL, i.e. $g(i) \in [g_{min}, \mu]$. Note that, the case $\alpha = 0$ corresponds to a pure greedy algorithm, while $\alpha = 1$ is equivalent to a random construction. The pseudo-code in Figure 3 is a refinement of the greedy randomized construction pseudo-code shown in Figure 2.

Prais and Ribeiro in [106, 107] observed the behavior of GRASP and the quality of the GRASP output solutions for different RCL construction mechanisms, based on different strategies for the variation of the value of the parameter α:

(a) α is randomly chosen from a uniform discrete probability distribution;
(b) α is randomly chosen from a decreasing non-uniform discrete probability distribution;
(c) fixed value of α, close to the purely greedy choice.

The authors incorporated these three strategies into the GRASP procedures developed for four optimization problems: (1) matrix decomposition for traffic assignment in communication satellite [108]; (2) set covering [43]; (3) weighted MAX-SAT [112, 113]; and (4) graph planarization [114, 117]. The resulting heuristics were tested on a subset of state-of-the-art instances for each type of problem. The total number of iterations performed was fixed at 10,000. The observed conclusions can be summarized as follows. Strategy (c) presented the shortest average computation times for three out the four problem types. It was also the one with the least variability in the constructed solutions and, consequently, found the best solution the fewest times. Strategy (a) presented a high number of hits and this behavior also illustrates the effectiveness of strategies based on the variation of the RCL parameter.

In [106, 108], Prais and Ribeiro also tested GRASP with a further RCL construction mechanism, in which the parameter α is self-adjusted and its value is periodically modified according to the quality of the obtained solutions. This extension of the basic GRASP is called *Reactive GRASP* and will be described in detail in the next Section devoted to the description of the enhancements to the basic GRASP.

```
procedure LocalSearch(x, f(·))
    Let N(x) be the neighborhood of x;
    H:={y ∈ N(x) | f(y) < f(x)};
    while (|H| > 0)→
        x:=Select(H);
        H:={y ∈ N(x) | f(y) < f(x)};
    endwhile
    return(x);
end LocalSearch
```

FIGURE 4. Pseudo-code of a generic local search procedure.

2.2. **Local search phase.** As is the case for many deterministic methods, the solutions generated by a GRASP construction are not guaranteed to be locally optimal with respect to simple neighborhood definitions. Hence, it is almost always beneficial to apply a local search to attempt to improve each constructed solution. A local search algorithm works in an iterative fashion by successively replacing the current solution by a better solution in the neighborhood of the current solution. It terminates when no better solution is found in the neighborhood. The *neighborhood structure* N for a problem relates a solution s of the problem to a subset of solutions $N(s)$. A solution s is said to be *locally optimal* if there is no better solution in $N(s)$ with respect to the objective function value. The key to success for a local search algorithm consists of the suitable choice of a neighborhood structure, efficient neighborhood search techniques, and the starting solution. Figure 4 illustrates the pseudo-code of a generic local search procedure for a minimization problem.

In a GRASP framework, a local search starts from an initial solution $x_0 \in X$ and iteratively generates a sequence of improving solutions $x_1, \ldots, x_M$, where $M =$ `MaxIterations`. At the k-th iteration, $k = 1, \ldots, M$, x_k is locally optimal respect to the neighborhood $N(x_{k-1})$ since $N(x_{k-1})$ is searched for an improving solution x_k such that $f(x_k) < f(x_{k-1})$. If such a solution is found, it is made the current solution. Otherwise, the search ends with x_{k-1} as a local optimum.

The effectiveness of local search depends on several factors, such as the neighborhood structure, the function to be minimized, and the starting solution. It has been experimentally shown that randomly generated solutions are of poor quality on average. On the the other hand, greedy algorithms usually produce solutions of better quality than those of randomly generated solutions. Therefore, using greedy solutions as starting points for local search in a multi-start procedure will usually lead to good, though, most often, suboptimal solutions. This is because the amount of variability in greedy solutions is small and it is less likely that a greedy starting solution will be in the basin of attraction of a global optimum than a random solution. A greedy randomized construction as the one embedded in GRASP adds variability to the greedy algorithm.

In [115], besides analyzing the quality of the solution obtained by varying between randomness and greediness in the VB mechanisms of the GRASP construction procedure, Resende and Ribeiro also analyzed the quality of the solutions output of the local search starting from solutions obtained by applying VB mechanisms with different values for the α parameter. As result of this analysis, the variance of the overall solution diversity, final solution quality, and running time increased with the variance of the solution values obtained in the construction phase. Moreover, it emerged that it is unlikely that GRASP finds an optimal solution if the average solution value is low, even if there is a large variance in the overall solution values, such as is the case for $\alpha = 0$. On the other hand, if there is little variance in the overall solution values, it is also unlikely that GRASP finds an optimal solution, even if the average solution is high, as is the case for $\alpha = 1$. Good solutions are usually obtained in the presence of relatively high average solution values and of a relatively large variance, such as is the case for $\alpha = 0.8$.

3. Enhancements

To improve the performance of the basic GRASP framework, most efforts have focused on construction mechanisms. Since Mockus et al. [97] pointed out that GRASP with a fixed nonzero RCL parameter α is not asymptotically convergent to a global optimum[1], several remedies have been proposed to get around this problem. They include Reactive GRASP, cost perturbations in place of randomized selection, bias functions, memory and learning, and local search on partially constructed solutions.

3.1. **Reactive GRASP.** The results of the study conducted in [106, 108] involving variation of the value of the RCL parameter α motivated the proposal of the extension of the basic GRASP called Reactive GRASP. Prais and Ribeiro in [108] have shown that using a single fixed value for the value of RCL parameter α very often hinders finding a high-quality solution, which eventually could be found if

[1]During construction, a fixed RCL parameter may rule out a candidate that is present in all optimal solutions.

another value was used. Moreover, one drawback of the basic GRASP is the lack of *learning* from the history of solutions found in previous iterations. The basic algorithm discards information about any solution encountered that does not improve the incumbent. Instead, it is worth to use information gathered from good solutions leading to *memory-based* procedures. Information about the quality of previously generated solutions can influence the construction phase, by modifying the selection probabilities associated with each element of the RCL.

In this paragraph, we describe Reactive GRASP, the first enhancement that incorporates a learning mechanism in the memoryless construction phase of the basic GRASP. In Reactive GRASP, the value of the RCL parameter α is selected in each iteration from a discrete set of possible values with a probability that depends on the solution values found along the previous iterations. One way to accomplish this is to use the rule proposed in [108]. Let $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_m\}$ be the set of possible values for α. At the first GRASP iteration, all m values have the same probability to be selected, i.e.

$$p_i = \frac{1}{m}, \qquad i = 1, 2, \ldots, m. \tag{2}$$

At any subsequent iteration, let $\hat{z}$ be the incumbent solution and let A_i be the average value of all solutions found using $\alpha = \alpha_i$, $i = 1, \ldots, m$. The selection probabilities are periodically reevaluated as follows:

$$p_i = \frac{q_i}{\sum_{j=1}^{m} q_j}, \tag{3}$$

where $q_i = \hat{z}/A_i$, $i = 1, \ldots, m$. If values of $\alpha = \alpha_i$ $(i \in \{1, \ldots, m\})$ lead to the best solutions on average, then the value of q_i is increased and larger values of q_i correspond to more suitable values for α. The probabilities associated with these more appropriate values will then increase when they are reevaluated.

Due to greater diversification and less reliance on parameter tuning, Reactive GRASP has lead to improvements over the basic GRASP in terms of robustness and solution quality. In fact, it has been successfully applied in power system transmission network planning [25] and in a capacitated location problem [37].

3.2. **Cost perturbations.** Another step toward an improved and alternative solution construction mechanism is to allow *cost perturbations*. The idea is to introduce some "noise" in the original costs in a fashion that resembles the noising method of Charon and Hudry [32, 33]. Cost perturbations are effective in all cases when the construction algorithm is not very sensitive to randomization, as for example in the case of the Steiner problem in graphs. To solve this problem, the hybrid GRASP procedure proposed by Ribeiro et al. in [120] used as one of the main building blocks of the construction phase the shortest-path heuristic of Takahashi and Matsuyama [128].

Another situation where cost perturbations can be effective is when there is no greedy algorithm available for the problem to be solved, as for example in the case of the prize-collecting Steiner tree problem. To solve this problem, the hybrid GRASP procedure proposed by Canuto et al. in [30] used the primal-dual algorithm of Goemans and Williamson [70] to build initial solutions using perturbed costs. More in details, in [30], at each iteration a new solution for the prize-collecting Steiner tree problem is built using node prizes updated by a perturbation function, according to the structure of the current solution. Two different prize perturbation schemes are used to enforce search diversification, as described in the following.

***Perturbation by eliminations*:**: The primal-dual algorithm used in the construction phase is driven to build a new solution without some of the nodes appearing in the solution constructed in the previous iteration. This is done by changing to zero the prizes of some persistent nodes, which appeared in the last solution built and remained at the end of the local search. A parameter μ controls the fraction of the persistent nodes whose prizes are temporarily set to zero;

***Perturbation by prize changes*:**: Similarly to the noising method of Charon and Hudry [32, 33], some noise is introduced into the node prizes, resulting in a change of the objective function as well. For each node i, a perturbation factor $\beta(i)$ is randomly generated in the interval $[1-a, 1+a]$, where a is an implementation parameter. The original prize $\pi(i)$ associated with node i is temporarily changed to $\pi(i) = \pi(i) \cdot \beta(i)$.

Experimental results have shown that embedding a strategy of costs perturbation into a GRASP framework improves the best overall results. The hybrid GRASP with path-relinking proposed for the Steiner problem in graphs by Ribeiro et al. in [120] uses this cost perturbation strategy and is among the most effective heuristics currently available. Path-relinking will be described in detail in the subsequent section devoted to the description of hybrid GRASP with other heuristic frameworks.

3.3. **Bias functions.** Another construction mechanism was proposed by Bresina [28]. Once built the RCL, instead of choosing with equal probability one candidate among the RCL elements, Bresina introduced a family of probability distributions to bias the selection toward some particular candidates. A bias function is based on a rank $r(x)$ assigned to each candidate x according to its greedy function value and is evaluated only for the elements in RCL. Several different bias functions were introduced:

i. random bias: $\text{bias}(r(x)) = 1$;
ii. linear bias: $\text{bias}(r(x)) = 1/r(x)$;
iii. log bias: $\text{bias}(r(x)) = \log^{-1}[r(x)+1]$;
iv. exponential bias: $\text{bias}(r(x)) = e^{-r}$;
v. polynomial bias of order n: $\text{bias}(r(x)) = r^{-n}$.

Let $\text{bias}(r(x))$ be one of the bias functions defined above. Once these values have been evaluated for all elements of the RCL, the probability p_x of selecting element x is

$$p_x = \frac{\text{bias}(r(x))}{\sum_{y \in RCL} \text{bias}(r(y))}. \tag{4}$$

A successful application of Bresina's bias function can be found in [24], where experimental results show that, although valid on all candidates, the evaluation of bias functions may be restricted only to the elements of the RCL.

Reactive GRASP has been the first and very simple attempt to enhance the basic GRASP in order to save and use history from previous iterations. Another very simple attempt is due to Fleurent and Glover [59] who proposed improved constructive multistart strategies that besides defining a special bias function also maintains a pool of *elite solutions* to be used in the construction phase. To become an elite solution, a solution must be either better than the best member of the pool, or better than its worst member and sufficiently different from the other solutions in the pool, in order to preserve not only solution quality but also the diversity of solutions. Fleurent and Glover defined: 1) a *strongly determined variable* as one that cannot be changed without eroding the objective or changing significantly other variables; 2) a *consistent variable* as one that receives a particular value in a large portion of the elite solution set, and 3) for each solution component i, a measure $I(i)$ of its strongly determined and consistent features that becomes larger as i appears more often in the pool of elite solutions. The intensity function $I(i)$ is used in the construction phase as follows. Recall that $g(i)$ is the greedy function, i.e. the incremental cost associated with the insertion of element i into the solution under construction. Let $K(i) = F(g(i), I(i))$ be a function of the greedy and the intensification functions. The idea of Fleurent and Glover is to define a special bias function that depends on $K(\cdot)$. In fact, the intensification scheme biases selection from the RCL to those elements i with a high value of $K(i)$ by setting its selection probability to be

$$p_i = \frac{K(i)}{\sum_{y \in RCL} K(y)}. \tag{5}$$

They suggested $K(i) = \lambda g(i) + I(i)$, with $K(i)$ varying with time by changing the value of λ, e.g. initially λ may be set to a large value that is decreased when diversification is called for. Rules and procedures for changing the value of λ are given by Fleurent and Glover [59] and Binato et al. [24].

3.4. **POP in construction.** The intuitive idea behind the *Proximate Optimality Principle* (POP) is that “good solutions at one level (stage of the algorithm) are likely to be found ’close’ to good solutions at an adjacent level”. Given the combinatorial character of the problem to be solved, Fleurent and Glover [59] proposed a GRASP for the quadratic assignment problem that applies local search not only at the end of each construction phase, but also during the construction itself on a subset of components of the solution under construction. This further application of local search aims to “iron-out” from the current solution its “bad”components. Nevertheless, experimental investigation conducted in the literature has shown that applying the POP idea at each construction iteration is excessively running time consuming. One possibility to implement the idea in a more efficient way is to apply local search during a few points in the construction phase and not during each construction iteration. In Binato et al. [24], local search is applied after 40% and 80% of the construction moves have been taken, as well as at the end of the construction phase.

4. Hybridizations

As enhancements to its basic framework, different hybridizations of GRASP with several other metaheuristics have been studied and proposed in the literature. In this section, some of them are surveyed and briefly described.

Laguna and Gonzalez-Velarde in 1991 [85] have first studied hybridization of GRASP with tabu search. Later, in 1999 Delmaire et al. [37] proposed a Reactive GRASP whose local search have been strengthened by tabu search. In particular, they have proposed two approaches. In the first, GRASP is applied as a powerful diversification strategy in the context of a tabu search procedure. The second approach is an implementation of the Reactive GRASP algorithm, in which the local search phase is strengthened by tabu search. Results reported for the capacitated location problem show that the hybrid approaches perform better than the pure methods previously used.

GRASP has been used also in conjunction with genetic algorithms. Basically, the feasible solution found by using a GRASP construction phase has been used as initial population by a genetic algorithm, as for example in [16] and in [3], where a greedy genetic algorithm is proposed for the quadratic assignment problem.

Another interesting hybridization of GRASP involves VNS (Variable Neighborhood Search) and Variable Neighborhood Descent (VND) proposed by Hansen and Mladenović [75, 96]. Almost all randomization effort in the basic GRASP algorithm involves the construction phase, while local search stops at the first local optimum. On the other hand, strategies such as VNS and VND rely almost entirely on the randomization of the local search to escape from local optima. With respect to this issue, probabilistic strategies such as GRASP and VNS may be considered as complementary and potentially capable of leading to effective hybrid methods.

VNS is based on the exploration of a dynamic neighborhood model. Contrary to other metaheuristics based on local search methods, VNS allows changes of the neighborhood structure along the search. It explores increasingly distant neighborhoods of the current best found solution x. Each step has three major phases: neighbor generation, local search, and jump. Let N_k, $k = 1, \dots, k_{max}$ be a set of pre-defined neighborhood structures and let $N_k(x)$ be the set of solutions in the kth-order neighborhood of a solution x. In the first phase, a neighbor $x' \in N_k(x)$ of the current solution is applied. Next, a solution x'' is obtained by applying local search to x'. Finally, the current solution jumps from x to x'' in case the latter improved the former. Otherwise, the order of the neighborhood is increased by one and the above steps are repeated until some stopping condition is satisfied.

A first attempt in the direction of integrating VNS into GRASP was done by Martins et al. [93]. The construction phase of their hybrid heuristic for the Steiner problem in graphs follows the greedy randomized strategy of GRASP, while the local search phase makes use of two different neighborhood structures as a VND strategy. Their heuristic was later improved by Ribeiro, Uchoa, and Werneck [119], one of the key components of the new algorithm being another strategy for the exploration of different neighborhoods. Ribeiro and Souza [118] also combined GRASP with VND in a hybrid heuristic for the degree-constrained minimum spanning tree problem. Festa et al. [54] studied different variants and combinations of GRASP and VNS for the MAX-CUT problem, finding and improving some of the solutions that at the time were the best known solutions for some open instances from the literature. In [48], the authors studied several hybridization of GRASP, including VNS, for the far from most string problem.

At last, we devote the remainder of this section to the combination of the basic GRASP with path-relinking. Path-relinking was originally proposed by Glover [64] as an intensification strategy exploring trajectories connecting elite solutions obtained by tabu search or scatter search [65, 67, 68]. It can be traced back to the pioneering work of Kernighan and Lin [79]. Starting from one or more elite solutions, paths in the solution space leading towards other guiding elite solutions are generated and explored in the search for better solutions. This is accomplished by selecting moves that introduce attributes contained in the guiding solutions. At each iteration, all moves that incorporate attributes of the guiding solution are analyzed and the move that best improves (or least deteriorates) the initial solution is chosen.

The first proposal of a hybrid GRASP with path-relinking was in 1999 due to Laguna and Martí [86]. It was followed by several extensions, improvements, and successful applications [4, 30, 52, 54, 58, 48, 49]. Path-relinking is applied to a pair of solutions $\mathbf{x}$ and $\mathbf{y}$, where one can be the solution obtained from the current GRASP iteration, and the other is a solution from an elite set of solutions. $\mathbf{x}$ is called the *initial solution* and $\mathbf{y}$ the *guiding solution*. The set $\mathcal{E}$ of elite solutions has usually a fixed size that does not exceed `MaxElite`. Given the pair $\mathbf{x}, \mathbf{y}$, their common elements are kept constant, and the space of solutions spanned by these elements is searched with the objective of finding a better solution. The size of the solution space grows exponentially with the the distance between the *initial* and *guiding* solutions and therefore only a small part of the space is explored by path-relinking. The procedure starts by computing the symmetric difference $\Delta(\mathbf{x}, \mathbf{y})$ between the two solutions, i.e. the set of moves needed to reach $\mathbf{y}$ (target solution) from $\mathbf{x}$ (initial solution). A path of solutions is generated linking $\mathbf{x}$ and $\mathbf{y}$. The best solution x^* in this path is returned by the algorithm. Since there is no guarantee that x^* is locally optimal, often local search is applied, starting from x^*, and the resulting locally optimal solution is returned.

Let us denote the set of solutions spanned by the common elements of the n-vectors $\mathbf{x}$ and $\mathbf{y}$ as

$$S(\mathbf{x}, \mathbf{y}) := \{w \text{ feasible} \mid w_i = x_i = y_i, i \notin \Delta(\mathbf{x}, \mathbf{y})\} \setminus \{\mathbf{x}, \mathbf{y}\}. \quad (6)$$

Clearly, $|S(\mathbf{x}, \mathbf{y})| = 2^{n-d(\mathbf{x},\mathbf{y})} - 2$, where $d(\mathbf{x}, \mathbf{y}) = |\Delta(\mathbf{x}, \mathbf{y})|$. The underlying assumption of path-relinking is that there exist good-quality solutions in $S(\mathbf{x}, \mathbf{y})$, since this space consists of all solutions which contain the common elements of two good solutions $\mathbf{x}$ and $\mathbf{y}$. Since the size of this space is exponentially large, a greedy search is usually performed where a path of solutions

$$\mathbf{x} = \mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{d(\mathbf{x},\mathbf{y})}, \mathbf{x}_{d(\mathbf{x},\mathbf{y})+1} = \mathbf{y}, \quad (7)$$

is built, such that $d(\mathbf{x}_i, \mathbf{x}_{i+1}) = 1, \;\; i = 0, \ldots, d(\mathbf{x}, \mathbf{y})$, and the best solution from this path is chosen. Note that, since both $\mathbf{x}$ and $\mathbf{y}$ are, by construction, local optima in some neighborhood $N(\cdot)$[2], then in order for $S(\mathbf{x}, \mathbf{y})$ to contain solutions which are not contained in the neighborhoods of $\mathbf{x}$ or $\mathbf{y}$, $\mathbf{x}$ and $\mathbf{y}$ must be sufficiently distant from each other.

Figure 5 illustrates the pseudo-code of the path-relinking procedure applied to the pair of solutions $\mathbf{x}$ (starting solution) and $\mathbf{y}$ (target solution). In line 1, an initial solution $\mathbf{x}$ is select at random among the elite set elements and usually it differs sufficiently from the guiding solution $\mathbf{y}$. The loop in lines 6 through 14 computes a path of solutions $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{d(\mathbf{x},\mathbf{y})-2}$, local search is applied in line 15, and the solution x^* with the best objective function value is returned in line 16. This is achieved by advancing one solution at a time in a greedy manner. At each iteration, the procedure examines all moves $m \in \Delta(x, \mathbf{y})$ from the current solution x and selects the one which results in the least cost solution (line 7), i.e. the one

[2] Where the same metric $d(\mathbf{x}, \mathbf{y})$ is usually used.

algorithm `Path-relinking`$(f(\cdot), \mathbf{x}, \mathcal{E})$
1 Choose, at random, a pool solution $\mathbf{y} \in \mathcal{E}$ to relink with $\mathbf{x}$;
2 Compute symmetric difference $\Delta(\mathbf{x}, \mathbf{y})$;
3 $f^* := \min\{f(\mathbf{x}), f(\mathbf{y})\}$;
4 $x^* := \arg\min\{f(\mathbf{x}), f(\mathbf{y})\}$;
5 $x := \mathbf{x}$;
6 **while** $(\Delta(x, \mathbf{y}) \neq \emptyset) \rightarrow$
7 $m^* := \arg\min\{f(x \oplus m) \mid m \in \Delta(x, \mathbf{y})\}$;
8 $\Delta(x \oplus m^*, \mathbf{y}) := \Delta(x, \mathbf{y}) \setminus \{m^*\}$;
9 $x := x \oplus m^*$;
10 **if** $(f(x) < f^*)$ **then**
11 $f^* := f(x)$;
12 $x^* := x$;
13 **endif**;
14 **endwhile**;
15 $x^* :=$ `LocalSearch`$(x^*, f(\cdot))$;
16 **return** (x^*);
end `Path-relinking`

FIGURE 5. Pseudo-code of a generic path-relinking for a minimization problem.

which minimizes $f(x \oplus m)$, where $x \oplus m$ is the solution resulting from applying move m to solution x. The best move m^* is made, producing solution $x \oplus m^*$ (line 9). The set of available moves is updated (line 8). If necessary, the best solution x^* is updated (lines 10–13). The procedure terminates when $\mathbf{y}$ is reached, i.e. when $\Delta(x, \mathbf{y}) = \emptyset$, returning the best solution found.

We now describe a possible way to hybridize with path-relinking the basic GRASP described in Section **Basic components**. The integration of the path-relinking procedure with the basic GRASP is shown in Figure 6. The pool $\mathcal{E}$ of elite solutions is initially empty, and until it reaches its maximum size no path relinking takes place. After a solution $\mathbf{x}$ is found by GRASP, it is passed to the path-relinking procedure to generate another solution. The procedure `AddToElite`$(\mathcal{E}, x_p)$ attempts to add to the elite set of solutions the solution that was just found. Since we wish to maintain a pool of good but diverse solutions, each solution obtained by path-relinking is considered as a candidate to be inserted into the pool if it is sufficiently different from every other solution currently in the pool. If the pool already has `MaxElite` solutions and the candidate is better than the worst of them, then a simple strategy is to have the former replace the latter. Another strategy, which tends to increase the diversity of the pool, is to replace the pool element most similar to the candidate among all pool elements with cost worse than the candidate's.

More formally, in several papers, a solution x_p is added to the elite set $\mathcal{E}$ if either one of the following conditions holds:

(1) $f(x_p) < \min\{f(\mathbf{w}) : \mathbf{w} \in \mathcal{E}\}$,
(2) $\min\{f(\mathbf{w}) : \mathbf{w} \in \mathcal{E}\} \leq f(x_p) < \max\{f(\mathbf{w}) : \mathbf{w} \in \mathcal{E}\}$ and $d(x_p, \mathbf{w}) > \beta n, \ \forall \mathbf{w} \in \mathcal{E}$, where β is a parameter between 0 and 1 and n is the number of decision variables.

```
procedure GRASP+PR(f(·), g(·), MaxIterations, Seed, MaxElite)
    x_best:=∅; f(x_best):=+∞; ℰ := ∅
    for k = 1, 2, ..., MaxIterations→
        x:=ConstructGreedyRandomizedSolution(Seed, g(·));
        if (x not feasible) then
            x:=repair(x);
        endif
        x:=LocalSearch(x, f(·));
        if (k ≤MaxElite) then
            ℰ := ℰ ∪ {x};
            if (f(x) < f(x_best)) then
                x_best:=x;
            endif
        else
            x_p:=Path-relinking(f(·), x, ℰ);
            AddToElite(ℰ, x_p);
            if (f(x_p) < f(x_best)) then
                x_best:=x_p;
            endif
        endif
    endfor;
    return(x_best);
end GRASP+PR
```

Figure 6. Pseudo-code of a basic GRASP with path-relinking heuristic for a minimization problem.

If x_p satisfies either of the above, it then replaces an elite solution $\mathbf{z}$ no better than x_p and most similar to x_p, i.e. $\mathbf{z} = \text{argmin}\{d(x_p, \mathbf{w}) : \mathbf{w} \in \mathcal{E} \text{ such that } f(\mathbf{w}) \geq f(x_p)\}$.

Figure 6 shows the simplest way to combine GRASP with path-relinking, which is applied as an intensification strategy to each local optimum obtained after the GRASP local search phase.

More generally, two basic strategies can be used:

i. path-relinking is applied as a post-optimization step to all pairs of elite solutions;
ii. path-relinking is applied as an intensification strategy to each local optimum obtained after the local search phase.

Applying path-relinking as an intensification strategy to each local optimum (strategy ii.) seems to be more effective than simply using it as a post-optimization step [116].

5. Automatic tuning

An annoying drawback of heuristics is the large number of parameters that need to be tuned for good performance. The tuning phase can take several hundreds of experiments and can be a labor intensive activity. Moreover, the performance of a heuristic depends on the instance being solved, so a tuned set of parameters obtained for one instance may not result in a good performing heuristic for another instance. When documenting a heuristic, a description of the tuning process is often left out and therefore it is often difficult to reproduce computational results. These are some of the factors that point to the need for an algorithmic approach to parameter tuning. Unfortunately, still nowadays very few attempts have been made in the design of efficient automatic tuning procedures for GRASP. In [51] and [99], the authors proposed a scheme for automatic tuning of GRASP with evolutionary path-relinking heuristics and tested for the generalized quadratic assignment problem, and for the set covering, the maximum cut, and the node capacitated graph partitioning, respectively. Similarly to IRACE (Iterated Race for Automatic Algorithm Configuration) [100] and ParamILS [78], both the proposed scheme consist of two phases. In the first phase, a biased random-key genetic algorithm searches the space of parameters for a set of values that results in a good performance of the heuristic. In the second phase, the GRASP+PR heuristic is run using the parameters found in the first phase. For all the optimization problems selected, the authors conducted rigorous experiments whose results showed that the two-phase approach is a robust hybrid heuristic.

6. Conclusions

The goal of this report was to provide an overview of GRASP describing its basic components and enhancements to the basic procedure, including reactive GRASP, intensification strategies, and its hybridizations with different metaheuristics.

References

[1] S. Abdinnour-Helm and S.W. Hadley. Tabu search based heuristics for multi-floor facility layout. *International J. of Production Research*, 38:365–383, 2000.
[2] J. Abello, P.M. Pardalos, and M.G.C. Resende. On maximum clique problems in very large graphs. In J. Abello and J. Vitter, editors, *External memory algorithms and visualization*, volume 50 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, pages 199–130. American Mathematical Society, 1999.
[3] R.K. Ahuja, J.B. Orlin, and A. Tiwari. A greedy genetic algorithm for the quadratic assignment problem. *Computers and Operations Research*, 27:917–934, 2000.
[4] R. Aiex, M.G.C. Resende, P.M. Pardalos, and G. Toraldo. GRASP with path relinking for three-index assignment. *INFORMS J. on Computing*, 17(2):224–247, 2005.
[5] R.M. Aiex, S. Binato, and M.G.C. Resende. Parallel GRASP with path-relinking for job shop scheduling. *Parallel Computing*, 29:393–430, 2003.
[6] R. Alvarez-Valdes, F. Parreño, and J.M. Tamarit. A GRASP algorithm for constrained two-dimensional non-guillotine cutting problems. *J. of the Operational Research Society*, 56(4):414–425, 2005.
[7] R. Alvarez-Valdes, F. Parreño, and J.M. Tamarit. Reactive GRASP for the strip-packing problem. *Computers & Operations Research*, 35(4):1065–1083, 2008.
[8] E. Amaldi, A. Capone, and F. Malucelli. Planning umts base station location: Optimization models with power control and algorithms. *IEEE Transactions on Wireless Communications*, 2(5):939–952, 2003.
[9] D.V. Andrade and M.G.C. Resende. A GRASP for PBX telephone migration scheduling. In *Eighth INFORMS Telecommunication Conference*, April 2006.
[10] D.V. Andrade and M.G.C. Resende. GRASP with path-relinking for network migration scheduling. In *Proceedings of the International Network Optimization Conference (INOC 2007)*, 2007.
[11] C. Andres, C. Miralles, and R. Pastor. Balancing and scheduling tasks in assembly lines with sequence-dependent setup times. *European J. of Operational Research*, 187(3):1212–1223, 2008.
[12] S. Areibi and A. Vannelli. A GRASP clustering technique for circuit partitioning. In J. Gu and P.M. Pardalos, editors, *Satisfiability problems*, volume 35 of *DIMACS Series on Discrete*

Mathematics and Theoretical Computer Science, pages 711–724. American Mathematical Society, 1997.
[13] S.M. Areibi. GRASP: An effective constructive technique for VLSI circuit partitioning. In *Proc. IEEE Canadian Conference on Electrical & Computer Engineering (CCECE'99)*, May 1999.
[14] M.F. Argüello, J.F. Bard, and G. Yu. A GRASP for aircraft routing in response to groundings and delays. *J. of Combinatorial Optimization*, 1:211–228, 1997.
[15] M.F. Argüello, T.A. Feo, and O. Goldschmidt. Randomized methods for the number partitioning problem. *Computers & Operations Research*, 23(2):103–111, 1996.
[16] M. Armony, J.G. Klincewicz, H. Luss, and M.B. Rosenwein. Design of stacked self-healing rings using a genetic algorithm. *J. of Heuristics*, 6:85–105, 2000.
[17] J.E.C. Arroyo, P.S. Vieira, and D.S. Vianna. A GRASP algorithm for the multi-criteria minimum spanning tree problem. *Annals of Operations Research*, 159:125–133, 2008.
[18] J.B. Atkinson. A greedy randomised search heuristic for time-constrained vehicle scheduling and the incorporation of a learning strategy. *J. of the Operational Research Society*, 49:700–708, 1998.
[19] T. Bäck, D.B. Fogel, and Z. Michalewicz. *Handbook of Evolutionary Computation*. Oxford University Press, New York, NY, 1997.
[20] J.F. Bard. An analysis of a rail car unloading area for a consumer products manufacturer. *J. of the Operational Research Society*, 48:873–883, 1997.
[21] J.F. Bard and T.A. Feo. Operations sequencing in discrete parts manufacturing. *Management Science*, 35:249–255, 1989.
[22] J.F. Bard and T.A. Feo. An algorithm for the manufacturing equipment selection problem. *IIE Transactions*, 23:83–92, 1991.
[23] J.F. Bard, L. Huang, P. Jaillet, and M. Dror. A decomposition approach to the inventory routing problem with satellite facilities. *Transportation Science*, 32:189–203, 1998.
[24] S. Binato, W.J. Hery, D. Loewenstern, and M.G.C. Resende. A greedy randomized adaptive search procedure for job shop scheduling. In C.C. Ribeiro and P. Hansen, editors, *Essays and surveys on metaheuristics*, pages 58–79. Kluwer Academic Publishers, 2002.
[25] S. Binato and G.C. Oliveira. A Reactive GRASP for transmission network expansion planning. In C.C. Ribeiro and P. Hansen, editors, *Essays and surveys on metaheuristics*, pages 81–100. Kluwer Academic Publishers, 2002.
[26] S. Binato, G.C. Oliveira, and J.L. Araújo. A greedy randomized adaptive search procedure for transmission expansion planning. *IEEE Transactions on Power Systems*, 16:247–253, 2001.
[27] M. Boudia, M.A.O. Louly, and C. Prins. A reactive GRASP and path relinking for a combined production-distribution problem. *Computers and Operations Research*, 34:3402–3419, 2007.
[28] J.L. Bresina. Heuristic-biased stochastic sampling. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, pages 271–278, 1996.
[29] 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.
[30] 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.
[31] C. Carreto and B. Baker. A GRASP interactive approach to the vehicle routing problem with backhauls. In C.C. Ribeiro and P. Hansen, editors, *Essays and surveys on metaheuristics*, pages 185–200. Kluwer Academic Publishers, 2002.
[32] I. Charon and O. Hudry. The noising method: A new method for combinatorial optimization. *Operations Research Letters*, 14:133–137, 1993.
[33] I. Charon and O. Hudry. The noising methods: A survey. In C.C. Ribeiro and P. Hansen, editors, *Essays and surveys on metaheuristics*, pages 245–261. Kluwer Academic Publishers, 2002.
[34] C. Commander, P. Festa, C.A.S. Oliveira, P.M. Pardalos, M.G.C. Resende, and M. Tsitselis. A greedy randomized algorithm for the cooperative communication problem on ad hoc networks. In *Eighth INFORMS Telecommunications Conference*, April 2006.
[35] I.A. Contreras and J.A. Díaz. Scatter search for the single source capacitated facility location problem. *Annals of Operations Research*, 157:73–89, 2008.

[36] G.L. Cravo, G.M. Ribeiro, and L.A. Nogueira Lorena. A greedy randomized adaptive search procedure for the point-feature cartographic label placement. *Computers and Geosciences*, 34(4):373–386, 2008.
[37] H. Delmaire, J.A. Díaz, E. Fernández, and M. Ortega. Reactive GRASP and tabu search based heuristics for the single source capacitated plant location problem. *INFOR*, 37:194–225, 1999.
[38] A.S. Deshpande and E. Triantaphyllou. A greedy randomized adaptive search procedure (GRASP) for inferring logical clauses from examples in polynomial time and some extensions. *Mathematical and Computer Modelling*, 27:75–99, 1998.
[39] M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, 2004.
[40] H. Faria, S. Binato, M.G.C. Resende, and D.J. Falcão. Power transmission network design by a greedy randomized adaptive path relinking approach. *IEEE Transactions on Power Systems*, 20(1):43–49, 2005.
[41] T.A. Feo and J.F. Bard. Flight scheduling and maintenance base planning. *Management Science*, 35:1415–1432, 1989.
[42] T.A. Feo and J.L. González-Velarde. The intermodal trailer assignment problem: Models, algorithms, and heuristics. *Transportation Science*, 29:330–341, 1995.
[43] T.A. Feo and M.G.C. Resende. A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8:67–71, 1989.
[44] T.A. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *J. of Global Optimization*, 6:109–133, 1995.
[45] 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.
[46] T.A. Feo, K. Sarathy, and J. McGahan. A GRASP for single machine scheduling with sequence dependent setup costs and linear delay penalties. *Computers & Operations Research*, 23:881–895, 1996.
[47] T.A. Feo, K. Venkatraman, and J.F. Bard. A GRASP for a difficult single machine scheduling problem. *Computers & Operations Research*, 18:635–643, 1991.
[48] D. Ferone, P. Festa, and M.G.C. Resende. Hybrid metaheuristics for the far from most string problem. In *Proceedings of 8th International Workshop on Hybrid Metaheuristics*, volume 7919 of Lecture Notes in Computer Science, pages 174–188, 2013.
[49] D. Ferone, P. Festa, and M.G.C. Resende. Hybridizations of GRASDP with path-relinking for the far from most string problem. *International Transactions in Operational Research*, page to appear, 2015.
[50] P. Festa. On some optimization problems in molecular biology. *Mathematical Bioscience*, 207(2):219–234, 2007.
[51] P. Festa, J.F. Gonçalves, M.G.C. Resende, and R.M.A. Silva. Automatic tuning of GRASP with path-relinking heuristics with a biased random-key genetic algorithm. In P. Festa, editor, *Proceedings of 9th International Symposium on Experimental Algorithms*, volume 6049 of Lecture Notes in Computer Science, pages 338–349, 2010.
[52] P. Festa, P.M. Pardalos, L.S. Pitsoulis, and M.G.C. Resende. GRASP with path-relinking for the weighted MAXSAT problem. *ACM J. on Experimental Algorithmics*, 11:1–16, 2006.
[53] P. Festa, P.M. Pardalos, and M.G.C. Resende. Algorithm 815: FORTRAN subroutines for computing approximate solution to feedback set problems using GRASP. *ACM Transactions on Mathematical Software*, 27:456–464, 2001.
[54] P. Festa, P.M. Pardalos, M.G.C. Resende, and C.C. Ribeiro. Randomized heuristics for the MAX-CUT problem. *Optimization Methods and Software*, 7:1033–1058, 2002.
[55] P. Festa and M. G. C. Resende. An annotated bibliography of grasp – part i: algorithms. *International Transactions in Operational Research*, 16(1):1–24, 2009.
[56] P. Festa and M. G. C. Resende. An annotated bibliography of grasp – part ii: applications. *International Transactions in Operational Research*, 16(2):131–172, 2009.
[57] P. Festa and M.G.C. Resende. GRASP: An annotated bibliography. In C.C. Ribeiro and P. Hansen, editors, *Essays and Surveys on Metaheuristics*, pages 325–367. Kluwer Academic Publishers, 2002.
[58] P. Festa and M.G.C. Resende. Hybridizations of GRASP with Path-Relinking. In E.-G. Talbi, editor, *Hybrid Metaheuristics – Studies in Computational Intelligence*, volume 434, pages 135–155. Springer, 2013.

[59] C. Fleurent and F. Glover. Improved constructive multistart strategies for the quadratic assignment problem using adaptive memory. *INFORMS J. on Computing*, 11:198–204, 1999.
[60] M.R. Garey and D.S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness.* W.H. Freeman and Company, New York, 1979.
[61] J.B. Ghosh. Computational aspects of the maximum diversity problem. *Operations Research Letters*, 19:175–181, 1996.
[62] F. Glover. Tabu search – Part I. *ORSA J. on Computing*, 1:190–206, 1989.
[63] F. Glover. Tabu search – Part II. *ORSA J. on Computing*, 2:4–32, 1990.
[64] F. Glover. Tabu search and adaptive memory programing – Advances, applications and challenges. In R.S. Barr, R.V. Helgason, and J.L. Kennington, editors, *Interfaces in Computer Science and Operations Research*, pages 1–75. Kluwer, 1996.
[65] F. Glover. Multi-start and strategic oscillation methods – Principles to exploit adaptive memory. In M. Laguna and J.L. Gonzáles-Velarde, editors, *Computing Tools for Modeling, Optimization and Simulation: Interfaces in Computer Science and Operations Research*, pages 1–24. Kluwer, 2000.
[66] F. Glover and M. Laguna. *Tabu Search.* Kluwer Academic Publishers, 1997.
[67] F. Glover and M. Laguna. *Tabu search.* Kluwer Academic Publishers, 1997.
[68] F. Glover, M. Laguna, and R. Martí. Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39:653–684, 2000.
[69] A. Goëffon, J.-M. Richer, and J.-K. Hao. Progressive tree neighborhood applied to the maximum parsimony problem. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5(1):136–145, 2008.
[70] M.X. Goemans and D.P. Williamson. The primal dual method for approximation algorithms and its application to network design problems. In D. Hochbaum, editor, *Approximation algorithms for NP-hard problems*, pages 144–191. PWS Publishing Co., 1996.
[71] D.E Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley, 1989.
[72] J.F. Gonçalves and M.G.C. Resende. Biased random-key genetic algorithms for combinatorial optimization. *J. of Heuristics*, 17(5):487–525, 2011.
[73] P.L. Hammer and D.J. Rader, Jr. Maximally disjoint solutions of the set covering problem. *J. of Heuristics*, 7:131–144, 2001.
[74] P. Hansen and N. Mladenović. An introduction to variable neighborhood search. In S. Voss, S. Martello, I. H. Osman, and C. Roucairol, editors, *Meta-heuristics, Advances and trends in local search paradigms for optimization*, pages 433–458. Kluwer Academic Publishers, 1998.
[75] P. Hansen and N. Mladenović. Developments of variable neighborhood search. In C.C. Ribeiro and P. Hansen, editors, *Essays and Surveys in Metaheuristics*, pages 415–439. Kluwer Academic Publishers, 2002.
[76] J.P. Hart and A.W. Shogan. Semi-greedy heuristics: An empirical study. *Operations Research Letters*, 6:107–114, 1987.
[77] M.J. Hirsch, C.N. Meneses, P.M. Pardalos, M.A. Ragle, and M.G.C. Resende. A continuous GRASP to determine the relationship between drugs and adverse reactions. In O. Seref, O.E. Kundakcioglu, and P.M. Pardalos, editors, *Data mining, systems analysis, and optimization in biomedicine*, volume 953 of *AIP Conference Proceedings*, pages 106–121. Springer, 2007.
[78] F. Hutter, H.H. Hoos, K. Leyton-Brown, and T. Stützle. ParamILS: An Automatic Algorithm Configuration Framework. *Journal of Artificial Intelligence Research*, 36:267–306, 2009.
[79] B.W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning problems. *Bell System Technical Journal*, 49(2):291–307, 1970.
[80] S. Kirkpatrick. Optimization by simulated annealing: Quantitative studies. *J. of Statistical Physics*, 34:975–986, 1984.
[81] J.G. Klincewicz. Avoiding local optima in the p-hub location problem using tabu search and GRASP. *Annals of Operations Research*, 40:283–302, 1992.
[82] J.G. Klincewicz and A. Rajan. Using GRASP to solve the component grouping problem. *Naval Research Logistics*, 41:893–912, 1994.
[83] G. Kontoravdis and J.F. Bard. A GRASP for the vehicle routing problem with time windows. *ORSA J. on Computing*, 7:10–23, 1995.

[84] M. Laguna, T.A. Feo, and H.C. Elrod. A greedy randomized adaptive search procedure for the two-partition problem. *Operations Research*, 42:677–687, 1994.

[85] M. Laguna and J.L. González-Velarde. A search heuristic for just-in-time scheduling in parallel machines. *J. of Intelligent Manufacturing*, 2:253–260, 1991.

[86] M. Laguna and R. Martí. GRASP and path relinking for 2-layer straight line crossing minimization. *INFORMS J. on Computing*, 11:44–52, 1999.

[87] M. Laguna and R. Martí. A GRASP for coloring sparse graphs. *Computaional Optimization and Applications*, 19:165–178, 2001.

[88] 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.

[89] X. Liu, P.M. Pardalos, S. Rajasekaran, and M.G.C. Resende. A GRASP for frequency assignment in mobile radio networks. In S. Rajasekaran, P.M. Pardalos, and F. Hsu, editors, *Mobile Networks and Computing*, volume 52 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, pages 195–201. American Mathematical Society, 2000.

[90] H. Ramalhinho Lourenço, J.P. Paixão, and R. Portugal. Multiobjective metaheuristics for the bus-driver scheduling problem. *Transportation Science*, 35:331–343, 2001.

[91] R. Martí and M. Laguna. Heuristics and meta-heuristics for 2-layer straight line crossing minimization. *Discrete Applied Mathematics*, 127(3):665–678, 2003.

[92] S.L. Martins, P.M. Pardalos, M.G.C. Resende, and C.C. Ribeiro. Greedy randomized adaptive search procedures for the Steiner problem in graphs. In P.M. Pardalos, S. Rajasekaran, and J. Rolim, editors, *Randomization methods in algorithmic design*, volume 43 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, pages 133–145. American Mathematical Society, 1999.

[93] S.L. Martins, M.G.C. Resende, C.C. Ribeiro, and P. Pardalos. A parallel GRASP for the Steiner tree problem in graphs using a hybrid local search strategy. *Journal of Global Optimization*, 17:267–283, 2000.

[94] S.L. Martins, C.C. Ribeiro, and M.C. Souza. A parallel GRASP for the Steiner problem in graphs. In A. Ferreira and J. Rolim, editors, *Proceedings of IRREGULAR'98 – 5th International Symposium on Solving Irregularly Structured Problems in Parallel*, volume 1457 of *Lecture Notes in Computer Science*, pages 285–297. Springer-Verlag, 1998.

[95] T. Mavridou, P.M. Pardalos, L.S. Pitsoulis, and M.G.C. Resende. A GRASP for the biquadratic assignment problem. *European J. of Operational Research*, 105:613–621, 1998.

[96] N. Mladenović and P. Hansen. Variable neighborhood search. *Computers and Operations Research*, 24:1097–1100, 1997.

[97] J. Mockus, E. Eddy, A. Mockus, L. Mockus, and G.V. Reklaitis. *Bayesian discrete and global optimization*. Kluwer Academic Publishers, 1997.

[98] S.K. Monkman, D.J. Morrice, and J.F. Bard. A production scheduling heuristic for an electronics manufacturer with sequence-dependent setup costs. *European J. of Operational Research*, 187(3):1100–1114, 2008.

[99] L.F. Morán-Mirabal, J.L. González-Velarde, and M.G.C. Resende. Automatic tuning of GRASP with evolutionary path-relinking. In *Proceedings of 8th International Workshop on Hybrid Metaheuristics*, volume 7919 of Lecture Notes in Computer Science, pages 62–77, 2013.

[100] M. López-Ibá nez, J. Dubois-Lacoste, T. Stützle, and M. Birattari. The irace package, iterated race for automatic algorithm configuration. Technical report, Technical Report TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, 2011.

[101] I.H. Osman, B. Al-Ayoubi, and M. Barake. A greedy random adaptive search procedure for the weighted maximal planar graph problem. *Computers and Industrial Engineering*, 45(4):635–651, 2003.

[102] P.M. Pardalos, L.S. Pitsoulis, and M.G.C. Resende. A parallel GRASP for MAX-SAT problems. *Lecture Notes in Computer Science*, 1184:575–585, 1996.

[103] P.M. Pardalos, L.S. Pitsoulis, and M.G.C. Resende. Algorithm 769: Fortran subroutines for approximate solution of sparse quadratic assignment problems using GRASP. *ACM Transactions on Mathematical Software*, 23:196–208, 1997.

[104] P.M. Pardalos, K.G. Ramakrishnan, M.G.C. Resende, and Y. Li. Implementation of a variance reduction based lower bound in a branch and bound algorithm for the quadratic assignment problem. *SIAM J. on Optimization*, 7:280–294, 1997.

[105] E. Pinãna, I. Plana, V. Campos, and R. Martì. GRASP and path relinking for the matrix bandwidth minimization. *European J. of Operational Research*, 153(1):200–210, 2004.

[106] M. Prais and C.C. Ribeiro. Parameter variation in GRASP implementations. In *Extended Abstracts of the Third Metaheuristics International Conference*, pages 375–380, 1999.

[107] M. Prais and C.C. Ribeiro. Parameter variation in GRASP procedures. *"Investigación Operativa"*, 9:1–20, 2000.

[108] M. Prais and C.C. Ribeiro. Reactive GRASP: An application to a matrix decomposition problem in TDMA traffic assignment. *INFORMS J. on Computing*, 12:164–176, 2000.

[109] G.G. Pu, Z. Chong, Z.Y. Qiu, Z.Q. Lin, and J.F. He. A hybrid heuristic algorithm for HW-SW partitioning within timed automata. In *Proceedings of Knowledge-based Intelligent Information and Engineering Systems*, volume 4251 of *Lecture Notes in Artificial Intelligence*, pages 459–466. Springer-Verlag, 2006.

[110] M.G.C. Resende and T.A. Feo. A GRASP for satisfiability. In D.S. Johnson and M.A. Trick, editors, *Cliques, Coloring, and Satisfiability: The Second DIMACS Implementation Challenge*, volume 26 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, pages 499–520. American Mathematical Society, 1996.

[111] 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.

[112] M.G.C. Resende, L.S. Pitsoulis, and P.M. Pardalos. Approximate solution of weighted MAX-SAT problems using GRASP. In J. Gu and P.M. Pardalos, editors, *Satisfiability problems*, volume 35 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, pages 393–405. American Mathematical Society, 1997.

[113] 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.

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

[115] M.G.C. Resende and C.C. Ribeiro. Greedy randomized adaptive search procedures. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 219–249. Kluwer Academic Publishers, 2003.

[116] 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.

[117] C.C. Ribeiro and M.G.C. Resende. Fortran subroutines for approximate solution of graph planarization problems using GRASP. *ACM Transactions on Mathematical Software*, 25:341–352, 1999.

[118] C.C. Ribeiro and M.C. Souza. Variable neighborhood search for the degree constrained minimum spanning tree problem. *Discrete Applied Mathematics*, 118:43–54, 2002.

[119] C.C. Ribeiro, E. Uchoa, and R.F. Werneck. A hybrid GRASP with perturbations for the Steiner problem in graphs. *INFORMS Journal on Computing*, 14:228–246, 2002.

[120] C.C. Ribeiro, E. Uchoa, and R.F. Werneck. A hybrid GRASP with perturbations for the Steiner problem in graphs. *INFORMS J. on Computing*, 14:228–246, 2002.

[121] C.C. Ribeiro and S. Urrutia. Heuristics for the mirrored traveling tournament problem. *European J. of Operational Research*, 179:775–787, 2007.

[122] R.Z. Ríos-Mercado and J.F. Bard. Heuristics for the flow line problem with setup costs. *European J. of Operational Research*, pages 76–98, 1998.

[123] R.Z. Ríos-Mercado and J.F. Bard. An enhanced TSP-based heuristic for makespan minimization in a flow shop with setup costs. *J. of Heuristics*, 5:57–74, 1999.

[124] L.I.D. Rivera. Evaluation of parallel implementations of heuristics for the course scheduling problem. Master's thesis, Instituto Tecnologico y de Estudios Superiores de Monterrey, Monterrey, Mexico, 1998.

[125] A.J. Robertson. A set of greedy randomized adaptive local search procedure (GRASP) implementations for the multidimensional assignment problem. *Computational Optimization and Applications*, 19:145–164, 2001.

[126] D. Sosnowska. Optimization of a simplified fleet assignment problem with metaheuristics: Simulated annealing and GRASP. In P.M. Pardalos, editor, *Approximation and complexity in numerical optimization*. Kluwer Academic Publishers, 2000.

[127] A. Srinivasan, K.G. Ramakrishnan, K. Kumaram, M. Aravamudam, and S. Naqvi. Optimal design of signaling networks for Internet telephony. In *IEEE INFOCOM 2000*, volume 2, pages 707–716, March 2000.

[128] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Mathematica Japonica*, 24:573–577, 1980.

[129] T.L. Urban. Solution procedures for the dynamic facility layout problem. *Annals of Operations Research*, pages 323–342, 1998.

(Paola Festa) Department of Mathematics and Applications, University of Napoli FEDERICO II, Italy.

E-mail address: paola.festa@unina.it

(Mauricio G.C. Resende) Amazon.com and Industrial & Systems Engineering, University of Washington, Seattle, WA USA.

E-mail address: mgcr@uw.edu