Variable Neighborhood Search, a Chapter of "Handbook of Metaheuristics"

Pierre Hansen[1], Nenad Mladenović[1,2]

[1]GERAD and Ecole des Hautes Etudes Commerciales
3000 ch. de la Cote-Sainte-Catherine, Montréal H3T 2A7, Canada
pierreh@crt.umontreal.ca

[2]Mathematical Institute, Serbian Academy of Science
Kneza Mihajla 35, 11000 Belgrade, Yugoslavia
nenad@mi.sanu.ac.yu

Abstract: Variable neighborhood search (VNS) is a recent metaheuristic for solving combinatorial and global optimization problems whose basic idea is systematic change of neighborhood within a local search. In this survey paper we present basic rules of VNS and some of its extensions. Moreover, applications are briefly summarized. They comprise heuristic solution of a variety of optimization problems, ways to accelerate exact algorithms and to analyze heuristic solution processes, as well as computer-assisted discovery of conjectures in graph theory.

Résumé: La Recherche à voisinage variable (RVV) est une métaheuristique récente pour la résolution de problèmes d'optimisation combinatoire et globale, dont l'idée de base est le changement systématique de voisinage au sein d'une recherche locale. Dans ce chaptre, nous présentons les règles de base de RVV et de certaines de ses extensions. De plus, des applications sont brièvement résumées. Elles comprennent la résolution approchée d'un ensemble de problèmes d'optimisation, des manières d'accélerer les algorithmes exacts et d'analyser le processus de résolution des heuristiques ainsi qu'un système automatique de découverte de conjectures en théorie des graphes.

1 Introduction

An optimization problem may be formulated as follows:

$$\min\{f(x)|x \in X, X \subseteq S\}. \tag{1}$$

S, X, x and f are *solution space*, *feasible set*, *feasible solution* and real valued function, respectively. If S is a finite but large set a *combinatorial optimization* problem is defined. If $S = R^n$, we talk about *continuous optimization*. Most optimization problems are NP-hard and heuristic (suboptimal) solution methods are needed to solve them (at least for large instances or as an initial solution for some exact procedure).

Metaheuristics, or general frameworks for building heuristics to solve problem (1), are usually based upon a basic idea, or analogy. Then, they are developed, extended in various directions and possibly hybridised. The resulting heuristics often get complicated, and use many parameters. This may enhance their efficiency but obscures the reasons of their success.

Variable Neighborhood Search (VNS for short), a metaheuristic proposed just a few years ago [110], [112], is based upon a simple principle: systematic change of neighborhood within the search. Its development has been rapid, with several dozen papers already published or to appear. Many extensions have been made, mainly to allow solution of large problem instances. In most of them, an effort has been made to keep the simplicity of the basic scheme.

In this paper, we survey these developments. The basic rules of VNS methods are recalled in the next section. Extensions are considered in Section 3 and Hybrids in Section 4. Applications are reviewed in Section 5, devoted to heuristic solution of combinatorial and global optimization problems, and Sections 6 to 8, which cover innovative uses, i.e., tools for analysis of the solution process of standard heuristics, acceleration of column generation procedures, and computer-assisted discovery in graph theory. Desirable properties of metaheuristics are listed in Section 9 together with brief conclusions.

2 Basic Schemes

Let us denote with $\mathcal{N}_k$, $(k = 1, \ldots, k_{max})$, a finite set of pre-selected neighborhood structures, and with $\mathcal{N}_k(x)$ the set of solutions in the k^{th} neighborhood of x. (Most local search heuristics use only one neighborhood structure, i.e., $k_{max} = 1$.) Neighborhoods $\mathcal{N}_k$ may be induced from one or more metric (or quasi-metric) functions introduced into a solution space S. An *optimal solution* x_{opt} (or global minimum) is a feasible solution where a minimum of (1) is reached. We call $x' \in X$ a *local minimum* of (1) with respect to $\mathcal{N}_k$ (w.r.t. $\mathcal{N}_k$ for short), if there is no solution $x \in \mathcal{N}_k(x') \subseteq X$ such that $f(x) < f(x')$. Metaheuristics (based on local search procedures) try to continue the search by other means after finding the first local minimum. VNS is based on three simple facts:

Fact 1 *A local minimum w.r.t. one neighborhood structure is not necessary so with another;*

Fact 2 *A global minimum is a local minimum w.r.t. all possible neighborhood structures.*

Fact 3 *For many problems local minima w.r.t. one or several $\mathcal{N}_k$ are relatively close to each other.*

This last observation, which is empirical, implies that a local optimum often provides some information about the global one. This may for instance be several variables with the same value in both. However, it is usually not known which ones are such. An organized study of the neighborhood of this local optimum is therefore in order, until a better one is found.

In order to solve (1) by using several neighborhoods, facts 1 to 3 can be used in three different ways: (i) deterministic; (ii) stochastic; (iii) both deterministic and stochastic.

(i) The **Variable neighborhood descent** (VND) method is obtained if change of neighborhoods is performed in a deterministic way and its steps are presented on Figure 1.

Initialization. Select the set of neighborhood structures $\mathcal{N}'_k$, for $k = 1, \ldots, k'_{max}$, that will be used in the descent; find an initial solution x (or apply the rules to a given x);

Repeat the following sequence until no improvement is obtained:
(1) Set $k \leftarrow 1$;
(2) Repeat the following steps until $k = k'_{max}$:
(a) Exploration of neighborhood. Find the best neighbor x' of x ($x' \in \mathcal{N}'_k(x)$);
(b) Move or not. If the solution thus obtained x' is better than x, set $x \leftarrow x'$ and $k \leftarrow 1$; otherwise, set $k \leftarrow k + 1$;

Figure 1. Steps of the basic VND.

Most local search heuristics use in their descents a single or sometimes two neighborhoods ($k'_{max} \leq 2$). Note that the final solution should be a local minimum w.r.t. all k'_{max} neighborhoods, and thus chances to reach a global one are larger than by using a single structure. Beside this *sequential* order of neighborhood structures in VND above, one can develop a *nested* strategy. Assume e.g. that $k'_{max} = 3$; then a possible nested strategy is: perform VND from Figure 1 for the first two neighborhoods, in each point x' that belongs to the third ($x' \in \mathcal{N}_3(x)$). Such an approach is applied in [16], [81], and [9].

(ii) The **Reduced VNS** (RVNS) method is obtained if random points are selected from $\mathcal{N}_k(x)$, without being followed by descent, and its steps are presented on Figure 2.

Initialization. Select the set of neighborhood structures $\mathcal{N}_k$, for $k = 1, \ldots, k_{max}$, that will be used in the search; find an initial solution x; choose a stopping condition;
Repeat the following sequence until the stopping condition is met:
(1) Set $k \leftarrow 1$;
(2) Repeat the following steps until $k = k_{max}$:
(a) Shaking. Generate a point x' at random from the k^{th} neighborhood of x ($x' \in \mathcal{N}_k(x)$);
(b) Move or not. If this point is better than the incumbent, move there ($x \leftarrow x'$), and continue the search with $\mathcal{N}_1$ ($k \leftarrow 1$); otherwise, set $k \leftarrow k + 1$;

Figure 2. Steps of the Reduced VNS.

RVNS is useful for very large instances for which local search is costly. It is observed that the best value for the parameter k_{max} is often 2. In addition, the maximum number of iterations between two improvements is usually used as stopping condition. RVNS is akin to a Monte-Carlo method, but more systematic (see [114] where results obtained by RVNS were 30% better than those of the Monte-Carlo method in solving a continuous min-max problem). When applied to the p-Median problem, RVNS gave equally good solutions as the *Fast Interchange* heuristic of [136] in 20 to 40 times less time [84].

(iii) The **Basic VNS** (VNS) method [112] combines deterministic and stochastic changes of neighborhood. Its steps are given on Figure 3.

Initialization. Select the set of neighborhood structures $\mathcal{N}_k$, for $k = 1, \ldots, k_{max}$, that will be used in the search; find an initial solution x; choose a stopping condition;

Repeat the following sequence until the stopping condition is met:
(1) Set $k \leftarrow 1$;
(2) Repeat the following steps until $k = k_{max}$:
(a) Shaking. Generate a point x' at random from the k^{th} neighborhood of x ($x' \in \mathcal{N}_k(x)$);
(b) Local search. Apply some local search method with x' as initial solution; denote with x'' the so obtained local optimum;
(c) Move or not. If this local optimum is better than the incumbent, move there ($x \leftarrow x''$), and continue the search with $\mathcal{N}_1$ ($k \leftarrow 1$); otherwise, set $k \leftarrow k + 1$;

Figure 3. Steps of the basic VNS.

The stopping condition may be e.g. maximum CPU time allowed, maximum number of iterations, or maximum number of iterations between two improvements. Often successive neighborhoods $\mathcal{N}_k$ will be nested. Observe that point x' is generated at random in step 2a in order to avoid cycling, which might occur if any deterministic rule was used. Note also that the Local search step (2b) may be replaced by VND. Using this VNS/VND approach led to the most successful applications recently reported (see e.g. [3], [16], [24], [25] – [29], [81], [124], [125]).

3 Extensions

Several easy ways to extend the basic VNS are now discussed. The basic VNS is a descent, first improvement method with randomization. Without much additional effort it could be transformed into a descent-ascent method: in Step 2c set also $x \leftarrow x$" with some probability even if the solution is worse than the incumbent (or best solution found so far). It could also be changed into a best improvement method: make a move to the best neighborhood k^* among all k_{max} of them. Other variants of the basic VNS could be to find solution x' in Step 2a as the best among b (a parameter) randomly generated solutions from the k^{th} neighborhood, or to introduce k_{min} and k_{step}, two parameters that control the change of neighborhood process: in the previous algorithm instead of $k \leftarrow 1$ set $k \leftarrow k_{min}$ and instead of $k \leftarrow k + 1$ set $k \leftarrow k + k_{step}$.

While the basic VNS is clearly useful for approximate solution of many combinatorial and global optimization problems, it remains difficult or long to solve very large instances. As often, size of problems considered is limited in practice by the tools available to solve them more than by the needs of potential users of these tools. Hence, improvements appear to be highly desirable. Moreover, when heuristics are applied to really large instances their strengths and weaknesses become clearly apparent. Three improvements of the basic VNS for solving large instances are now considered:

(iv) The **Variable Neighborhood Decomposition Search** (VNDS) method [84] extends the basic VNS into a two-level VNS scheme based upon decomposition of the problem. Its steps are presented on Figure 4.

Initialization. Select the set of neighborhood structures $\mathcal{N}_k$, for $k = 1, \ldots, k_{max}$, that will be used in the search; find an initial solution x; choose a stopping condition;

Repeat the following sequence until the stopping condition is met:
(1) Set $k \leftarrow 1$;
(2) Repeat the following steps until $k = k_{max}$:
(a) Shaking. Generate a point x' at random from the k^{th} neighborhood of x ($x' \in \mathcal{N}_k(x)$); in other words, let y be a set of k solution attributes present in x' but not in x ($y = x' \setminus x$).
(b) Local search. Find the local optimum in the space of y either by inspection or by some heuristic; denote the best solution found with y' and with x'' the corresponding solution in the whole space S ($x'' = (x' \setminus y) \cup y'$);
(c) Move or not. If the solution thus obtained is better than the incumbent, move there ($x \leftarrow x''$), and continue the search with $\mathcal{N}_1$ ($k \leftarrow 1$); otherwise, set $k \leftarrow k + 1$;

Figure 4. Steps of the basic VNDS.

Note that the only difference between the basic VNS and VNDS is in step 2b: instead of applying some local search method in the whole solution space $\mathcal{S}$ (starting from $x' \in \mathcal{N}_k(x)$), in VNDS we solve at each iteration a subproblem in some subspace $V_k \subseteq \mathcal{N}_k(x)$ with $x' \in V_k$. When the local search used in this step is also VNS, the two-level VNS-scheme arises.

VNDS can be viewed as embedding the classical successive approximation scheme (which has been used in combinatorial optimization at least since the sixties, see e.g. [66]) in the VNS framework. Other simpler applications of this technique, where the size of the subproblems to be optimized at the lower level is fixed, are Large neighborhood search [128] and POPMUSIC [131].

(v) The **Skewed VNS** (SVNS) method [74], a second extension, addresses the problem of exploring valleys far from the incumbent solution. Indeed, once the best solution in a large region has been found it is necessary to go quite far to obtain an improved one. Solutions drawn at random in far-away neighborhoods may differ substantially from the incumbent and VNS can then degenerate, to some extent, into the Multistart heuristic (in which descents are made iteratively from solutions generated at random and which is known not to be very efficient). So some compensation for distance from the incumbent must be made and a scheme called Skewed VNS is proposed for that purpose. Its steps are presented in Figure 5.

Initialization. Select the set of neighborhood structures $\mathcal{N}_k$, for $k = 1, \ldots, k_{max}$, that will be used in the search; find an initial solution x and its value $f(x)$; set $x_{opt} \leftarrow x$, $f_{opt} \leftarrow f(x)$: choose a stopping condition and a parameter value α;

Repeat the following until the stopping condition is met:
(1) Set $k \leftarrow 1$;
(2) Repeat the following steps until $k = k_{max}$:
(a)*Shaking.* Generate a point x' at random from the k^{th} neighborhood of x;
(b)*Local search.* Apply some local search method with x' as initial solution; denote with x'' the so obtained local optimum;

(c) *Improve or not.* If $f(x'') < f_{opt}$ set $f_{opt} \leftarrow f(x)$ and $x_{opt} \leftarrow x''$;
(d) *Move or not.* If $f(x'') - \alpha\rho(x, x'') < f(x)$ set $x \leftarrow x''$ and $k \leftarrow 1$;
otherwise set $k \leftarrow k + 1$.

Figure 5. Steps of the Skewed VNS.

SVNS makes use of a function $\rho(x, x'')$ to measure distance between the incumbent solution x and the local optimum found x''. The distance used to define the $\mathcal{N}_k$, as in the above examples, could be used also for this purpose. The parameter α must be chosen in order to accept exploring valleys far from x when $f(x'')$ is larger than $f(x)$ but not too much (otherwise one will always leave x). A good value is to be found experimentally in each case. Moreover, in order to avoid frequent moves from x to a close solution one may take a large value for α when $\rho(x, x'')$ is small. More sophisticated choices for a function $\alpha\rho(x, x'')$ could be made through some learning process.

(vi) **Parallel VNS** (PVNS) methods are a third extension. Several ways for parallelizing VNS have recently been proposed [106], [32] in solving the p-Median problem. In [106] three of them are tested : (i) parallelize local search; (ii) augment the number of solutions drawn from the current neighborhood and do local search in parallel from each of them and (iii) do the same as (ii) but updating the information about the best solution found. The second version gave the best results. It is shown in [32] that assigning different neighborhoods to each processor and interrupting their work as soon as an improved solution is found gives very good results: best known solutions have been found on several large instances taken from TSP-LIB [122]. Three Parallel VNS strategies are also suggested for solving the Traveling purchaser problem in [116].

4 Hybrids

As change of neighborhood in the search for good solutions to (1) is a simple and a very powerful tool, several authors have added such a feature to other metaheuristics than VNS.

In this section we review the resulting Hybrids at a general level; more details concerning specific applications are given in the next section.

(i) **VNS and Tabu search.** Tabu search ([62], [63], [72]) is a metaheuristic that has a huge number of applications (see e.g. [64]). It explores different types of memories in the search, i.e., *recency* based (short-term), *frequency* based, *long-term* memories etc. Usually it uses one neighborhood structure and, with respect to that structure, performs descent and ascent moves building a trajectory. In principle, there are two ways of making hybrids of VNS and TS: use TS within VNS or use VNS within TS. Recently four reports on hybrids of the first kind and two on hybrids of the second kind have been proposed. For solving the Route-median problem, TS is used instead of Local search within VNS (step 2b of Figure 3) in [126] (the resulting method is called VNTS), while in [20], in solving the Nurse rostering problem, in each among several neighborhoods simple descent or TS are used alternatively. A Multi-level TS is proposed in [97] for solving the Continuous min-max problem, where each level represents a ball of different size in the vicinity of the current solution, i.e., different neighborhoods, and a tabu list is constructed for each level. Reactive

variable neighborhood descent (ReVND) is proposed in [14] for solving the Vehicle routing problem with time windows. In repeating a sequence of four proposed local searches, the information on unsuccessful pairs of edges is memorized so that in the next repetition those pairs are not considered. In that way the size of each neighborhood is reduced.

Note that nested VND can easily be used in a TS framework since cycling can be avoided by controlling only one neighborhood. In solving the Multisource Weber problem, several heuristics that use nested VND within TS are proposed in [16]. Moves are considered as 'tabu' regarding the *relocation* neighborhood only, while for each solution of that neighborhood *reallocation* and *alternate* moves are used in a sequence. It is also observed in [37] that the same Tabu list can be used for several different neighborhoods used sequentially (i.e., Or-opts and interchanges) for solving problems where the solution is stored as a permutation of its attributes or variables. This makes possible a hybrid of TS and VND, i.e., VND can be used within TS.

(ii) **VNS and GRASP.** GRASP is a two phase metaheuristic [48]. In the first phase solutions are constructed using a greedy randomized procedure and in the second, solutions are improved by some local search or enumerative method. A natural way of hybridizing GRASP and VNS is to use VNS in the second phase of GRASP. Such an approach has been performed in solving the Steiner tree problem in graphs [108], [125], the Phylogeny problem [3], the Prize-collecting Steiner tree problem [24], the Traveling purchaser problem [116] and the Max-Cut problem [49]. The results reported show improvements of the GRASP/VNS hybrid over the pure GRASP.

(iii) **VNS and Constraint programming.** In the last few years, Constraint programming (CP) has attracted high attention among experts from many areas because of its potential for solving hard real-life problems. A *constraint* is a logical relation among several variables, each taking a value in a given domain. The important feature of constraints is their declarative manner, i.e., they specify what relationship must hold without specifying a computational procedure to enforce it. One aim of CP is to solve problems by stating constraints and finding solutions satisfying all these constraints. It has been noted that local search techniques can successfully be used in CP (see for example [118], [119] and [128]). In [57], two operators have been suggested and used in a local descent (VND), within Large neighborhood search (LNS) and CP framework for solving the Vehicle routing problem with time windows (VRPTW). They are called LNS-GENI and SMART (SMAll RouTing). The idea of LNS [128], which can be seen as a general decomposition heuristic, is first to 'destroy' the current solution by removing a given number of solution attributes and then rebuilt it in the best possible way by solving smaller problems. Both phases are problem dependent, but some general rules are recommended in [128]. In [12], in order to minimize the total travel costs of VRPTW, destruction of the solution is performed in a more systematic way: it starts from $k = 2$ attributes, then 3, etc.; once the improvement in rebuilding phase is obtained, $k = 2$ is set again. This version is very close to VNDS. A simpler version of LNS (called LNS/CP/GR) is suggested in [105] for solving a *Valued Constraint satisfaction problem* (VCSP). In VCSP the penalty (a valuation) must be paid for each unsatisfied variable. Then the objective is to find values of variables such that the sum of penalties is minimum. In rebuilding the solution, authors use a greedy proce-

dure. However, in future work, they could include VNS, which has been done recently by other authors. In [107] so-called VNS/LDS+CP heuristic is proposed and tested on the Frequency assignment problem.

5 Finding good solutions

In this section and the three following ones, we review applications of VNS, the number of which has rapidly increased since 1997, when the first paper on VNS was published. We begin with traditional ones, i.e., finding good solutions to combinatorial and global optimization problems, that is near optimal ones or possibly optimal ones but without a proof of optimality. These applications are grouped by field and within each field papers are considered chronologically.

5.1 Traveling salesman and Vehicle routing problems.

We shall consider here Traveling salesman problem (TSP), Vehicle routing problem (VRP), Arc routing problem (ARP) and some of their extensions that have been solved by VNS.

Traveling salesman problem. Given n cities with intercity distances, the traveling salesman problem (TSP) is to find a minimum cost tour x (i.e., a permutation of the cities which minimizes the sum of the n distances between adjacent cities in the tour). It is a classical NP-hard problem.

A heuristic for the Euclidean TSP called GENIUS was developed in [59]. It is a sophisticated insertion followed by local deletion/insertion and correction procedure. The size of the neighborhood in GENIUS depends on a parameter p (the number of cities already in the tour closest to the city that is considered for possible deletion or insertion). We immediately get a set of neighborhood structures for VNS by denoting with $\mathcal{N}_p(x)$ all tours obtained by deletion/insertion with parameter value p. Details can be found in [112] where results on the same type of test problems as reported in [59] are given. VNS gives a 0.75% average improvement over GENIUS within a similar CPU time. Moreover, improvements are obtained for all problem sizes.

Probably the most popular heuristic for solving TSP is 2-opt, where in turn two links between cities in the current tour x are removed and these cities reconnected by adding links in the only other way which gives a tour. Since 2-opt is a local search descent heuristic, it stops in a local minimum. In [78] the basic VNS rules using 2-opt (or a quicker variant, in which only the shortest edges are used) as local search routine, are applied. Average results for random Euclidean problems over 100 trials for $n = 100, \ldots, 500$ and 10 trials for $n = 600, \ldots, 1000$ are reported. Average improvements in value of 2.73% and 4.43% over the classical 2-opt heuristic within a similar computing time are obtained by these two versions respectively.

In [19] a new local search heuristic, the so-called *k-hyperopt* is proposed. Then a VND heuristic uses 2-opt, 2-hyperopt and 3-hyperopt neighborhoods in descent, while a VNS heuristic uses k-hyperopt for shaking and 2-hyperopt for a local search. The new

methods are compared with Iterated local search and Multistart 2-opt. Results are reported on standard test instances. It appears that tours obtained are comparable with iterated Lin-Kernighan [103] in terms of tour quality, but CPU time is larger.

In a work in progress [58], a similar VND heuristic is applied within two different decomposition schemes, i.e., VNDS: on the one hand subproblems corresponding to k successive points on the current tour are selected and re-optimized; on the other hand the same is done but with the k closest points from a randomly chosen one. It appears that the second decomposition gives the best results.

Traveling salesman problem with back-hauls. The GENIUS heuristic [59] was applied to the *TSP with back-hauls* in [60]. In this problem customers (or cities) are divided into three disjoint sets: depot, line-haul and back-haul customers. Starting from the depot, a tour must be designed such that all line-haul customers are visited before all back-haul customers. This time VNS gives a 0.40% average improvement over GENIUS with a 30% increase in computing time [112]. Improvements are obtained for all problem sizes.

Route-median problem. Given a set of cities with inter-cities distances, the Route-Median problem consists in choosing a subset of cities that are included in a cycle and allocating the remaining ones each to the closest chosen city in order to minimize the length of the route with an upper bound on the sum of the distances from the cities not in the route to the cities to which they are assigned. An exact branch and cut solution method has been proposed in [101]. A metaheuristic approach that combines VNS and TS and uses several neighborhood structures is proposed in [126].

Vehicle routing problem with time windows. The Vehicle routing problem (VRP) consists in designing least cost routes from one depot to a given set of customers. The routes must be designed in such a way that each customer is visited only once by exactly one vehicle. There are usually capacity constraints on the load that can be carried by a vehicle, and each customer has a known demand. The time window variant to the problem (VRPTW) imposes the additional constraint that each customer must be visited within a given time interval.

Four methods for solving VRPTW have recently been proposed that include VNS ideas as well. In [57] two operators which make use of Constraint programming and local search to explore their neighborhood are proposed. These operators are combined in a VND framework. Computational results show that this method is comparable with other heuristics, often producing better solutions in terms of distance traveled. Another heuristic [14], in its third phase, uses four neighborhoods in descent. In addition, not-improving neighborhood solutions are memorized, so that in the next pass they are not visited. The resulting method is called Reactive VND (ReVND). The obvious fact that a local minimum w.r.t. one objective is not necessary so for another, can also be a powerful tool for escaping from the local minima trap. This has been explored in [14]. In the second phase the objective is minimum number of vehicles, in the third (where ReVNS is proposed) the objective is to minimize the total travel distance, and in the last phase a new objective function is defined that considers both criteria with given weights. Results are reported on standard data sets. It appears that the proposed procedure outperforms other recent

heuristics for VRPTW, and that four new best known solutions are found. In [30] the same idea is used (i.e., escape from the local minima trap by changing both the neighborhood structure and the objective function), but with different local searches and in a different way: the classical k-opt exchanges improve the solution (for $k \leq 3$) in terms of number of vehicles and then another objective is considered. Beside the two objectives mentioned above, two other functions are used as well. No parameter tuning is required and no random choice is made. The algorithm has been tested on benchmark problems with favorable results, when compared with those obtained by most recent heuristics. In the most recent paper for solving VRTPTW [12], the idea of changing both the objective function and the neighborhoods within the search is explored further, and probably the best results to date on Solomon's benchmark instances are reported. The heuristic has two phases. In the first one, five neighborhood structures (2-opt, Or-opt, relocation, interchange and cross-over) and three objective functions (number of routes, maximum of the sum-of-squares route sizes, minimum of minimal delay) are considered. All five neighborhoods are not used as in VND (not in a sequential nor nested way), i.e., a single one among them is chosen at random and explored for all three objectives. The resulting procedure uses the Simulated annealing framework. In the second phase the authors implement a modified Large neighborhood search (LNS) [128] heuristic (minimizing the total distance traveled) which, according to them, make it very close to VNS.

Vehicle routing problem with backhauls (VRPB). In this problem the vehicles are not only required to deliver goods to (linehaul) customer, but also to pick up goods at (backhaul) customers locations. In [33] Reactive tabu search (RTS) and VND are developed and compared on some VRP test instances from the literature with different percentage of linehaul customers (50%, 66% and 80%). VND uses insertion and interchange moves in descent. It appears that RTS and VND are 1.8% and 5.3% above known optimal value.

Arc routing problem. In Arc routing problems the aim is to determine a least cost traversal of all edges or arcs of a graph, subject to some side constraints. In the Capacitated arc routing problem (CARP) edges have a non-negative weight and each edge with a positive weight must be traversed by exactly one of several vehicles starting and ending their trip at a depot, subject to the constraint that total weight of all edges serviced by a vehicle cannot exceed its capacity. In [61] the VNS heuristic developed in [109] for the undirected CARP was compared with a Tabu search heuristic [87]. The conclusions are as follows: (i) on small test instances ($7 \leq n \leq 27$), TS and VNS perform similarly (the deviation from optimality is identical and computing times are about 20% smaller for VNS); (ii) on larger test instances VNS is better than TS in terms of solution quality, but also faster (average deviation from optimality and computing times on the 270 instances were 0.71% and 349.81 seconds for TS and 0.54% and 42.50 seconds for VNS).

Linear ordering problem. Given a squared $n \times n$ matrix D, the Linear ordering problem (LOP) consists in finding permutations of rows and columns of D such that the sum of all elements above the main diagonal is maximum. LOP has a large number of applications such as triangulation of input-output matrices, archaeological seriation, scheduling etc. In [65], a basic VNS heuristic for solving LOP is proposed. Neighborhoods are derived by k row interchanges, and local searches performed using the same neighborhood as in the TS

approach from [102]. The outer loop of the basic VNS has not been used, i.e., the procedure stops the first time all k_{max} neighborhoods were explored. It is shown, on the same 49 test instances, that both TS and VNS (k_{max} is set to 10) have similar performance: TS is slightly better in terms of the solution quality (number of optimal solutions found by TS was 47 and 44 for VNS); VNS is faster (average time was 0.93 seconds for TS and 0.87 for VNS).

Traveling purchaser problem (TPP) can be seen as an extension of TSP. Given a set of m items to be purchased at n markets, the cost of item k at market j ($p_{kj}, k = 1, .., m,\ j = 1, .., n$) and inter-market travel costs t_{ij}, the problem is to purchase all m products by visiting a subset of the markets in a tour (starting from the source j=0), such that the total travel and purchase costs are minimized. This problem includes many well-known NP-hard problems such as uncapacitated facility location, set covering and group Steiner tree problems as its special cases [121].

Several constructive heuristics for the TSP are adapted in the initial solution building (within GRASP) and several neighborhoods are used for local search (within VND) for solving TPP in [129] and [116]. Among many possible variants that have been investigated in the sequential implementation are two from each class of heuristic (GRASP, VNS, Hybrid GRASP/VNS and TS [135]) on 16 random test instances with 50 to 150 markets and items. Each heuristic was restarted 5 times and average results reported. Among 8 sequential codes the GRASP/VNS hybrid had the best performance, while among 5 parallel methods, VNS was the best in average.

5.2 Location and Clustering problems

p-Median problem. Given a set L of m potential locations for p facilities and a set U of given locations for n users, the p-Median problem (PM) is to locate simultaneously the p facilities in order to minimize the total transportation distance (or cost) from users to facilities, each user being served by the closest facility. Solutions of PM are thus characterized by 0–1 vectors x with p components among $|L|$ equal to 1 indicating where facilities are located.

There are several papers that use VNS for solving the PM problem. In the first one [77], the basic VNS is applied and extensive statistical analysis of various strategies performed. Neighborhood structures are defined by moving $1, 2, \ldots, k_{\max}$ facilities and correspond to sets of 0–1 vectors at Hamming distance $2, 4, \ldots, k_{\max}$ from x. The descent heuristic used is 1–interchange, with the efficient *Fast Interchange* (FI) computational scheme [136]. Results of a comparison of heuristics for OR-Lib and some TSP-Lib problems are reported. It appears that VNS outperforms other heuristics. In order to solve larger PM problem instances, in [84] both RVNS and VNDS are applied. Subproblems with increasing number of users (that are solved by VNS) are obtained by merging the sets of users (or market areas) associated with k ($k = 1, \ldots, p$) medians. Results on 1400, 3038 and 5934 users instances from the TSP library show VNDS improves notably upon VNS in less computing time, and gives much better results than FI, in the same time that FI takes for a single descent. Moreover, Reduced VNS (RVNS), which does not use a descent

phase, gives results similar to those of FI in much less computing time. Two versions of Parallel VNS for PM are proposed in [106], [32] (see Section 3 above).

Multisource Weber problem. The multisource Weber (MW) problem (also known as continuous location-allocation problem) is the continuous counterpart of PM: instead of locating the p facilities at some locations of L, they can be located anywhere in the plane. An early application of VNS to MW is given in [17]. Several other ones are discussed at length in [16]. It appears that the choice of neighborhoods is crucial. Reassignment of customers to facilities a few at a time is a poor choice, as it entails only marginal changes in the solutions considered. Much better results are obtained when the facilities themselves are moved. As they may be located anywhere in the plane target locations are needed. An easy and efficient choice is locations of customers where there is no facility as yet. Using this neighborhood structure, several basic TS and VNS heuristics were developed and an extensive empirical study carried out to evaluate various heuristics - old, recent, and new - in a unified setting. The different methods (i.e., Genetic search, three Tabu search variants, four VNS variants etc.) were compared on the basis of equivalent CPU times. Results of this study indicate that VNS can be effectively used to obtain superior solutions. For instance on a series of 20 problems with 1060 users the average error (by comparison with best known solution) is of 0.02% only for the best VNS, while it can rise to more than 20% for some well-known heuristics of the literature. Average error for a Genetic algorithm was 1.27% and for the best TS 0.13%.

Minimum-sum-of-squares clustering problem. Given a set of n points in Euclidean q-dimensional space, the minimum sum-of-squares clustering problem (MSSC) is to partition this set into classes, or clusters, such that the sum of squared distances between entities and the centroids of their clusters is minimum. Among many heuristics for MSSC, the k-Means local search heuristic [93] is the most popular. It is an interchange heuristic, where points are reassigned to another cluster than their own, one at a time, until a local optimum is reached. Another popular heuristic, called H-Means [2], selects an initial partition, computes the centroids of its clusters, then reassigns entities to the closest centroid and iterates until stability. A new local search heuristic, called J-Means is proposed in [81]: centroids are relocated at some of the given points, which do not yet coincide with one of them. Results of a comparison of k-Means, H-Means, $H+K$-Means (where H-Means and k-Means are applied in sequence) and two versions of VNS are given in [81]. VNS–1 is an extension of k-Means and gives slightly better results than $H+K$-Means; VNS–2 which extends J-Means proves to be the best heuristic.

Fuzzy clustering problem. The Fuzzy clustering problem (FCP) is an important one in pattern recognition. It consists in assigning (allocating) a set of patterns (or entities) to a given number of clusters such that each of them belongs to one or more clusters with different degrees of membership. The objective is to minimize the sum of squared distances to the centroids, weighted by the degrees of membership. The fuzzy clustering problem was initially formulated in [42] as a mathematical programming problem and later generalized in [13]. The most popular heuristic for solving FCP is the so-called Fuzzy C-means (F-CM) method [23]. It alternatively finds membership matrices and centroids until there is no more improvement in the objective function value. A new descent local search heuristic called

Fuzzy J-means (F-JM) is proposed in [9] where the neighborhood is defined by all possible centroid-to-entity relocations (see also [81], [16]). This 'integer' solution is then moved to a continuous one by an alternate step, i.e., by finding centroids with given memberships. Fuzzy VNS rules are applied as well (F-JM is used as local search subroutine).

p-Center problem. The p-Center problem consists in locating p facilities and assigning clients to them in order to minimize the maximum distance between a client and the facility to which he (or she) is allocated (i.e., the closest facility). This model is used for example in locating fire stations or ambulances, where the distance from the facilities to their farthest allocated client should be minimum. In [111] basic VNS and Tabu search (e.g., the so called Chain substitution Tabu Search, [113]) heuristics are presented. Both methods use the 1-interchange (or vertex substitution) neighborhood structure. It is shown how this neighborhood can be used even more efficiently than for solving the p-Median problem. Based on the same computing time, comparison between the Multistart 1-interchange, the Chain interchange TS (with one and two Tabu lists) and VNS are reported on standard test problems from the literature. It appears that both TS and VNS outperform the Multistart approach and give similar results with a slight edge in favor of VNS for the larger instances.

Quadratic assignment problem. The Quadratic Assignment Problem can be described as follows: Given two $n \times n$ matrices A and B, find a permutation π^* minimizing the sum of the $a_{ij} \cdot b_{\pi_i \pi_j}$. A parameter free basic VNS (k_{max} is set to n) is suggested in [130], where two new methods based on Fast Ant systems (FANT) and Genetic-Descent hybrid (GDH) are also proposed. All three methods use the same simple descent local search procedure. On a series of *structured* instances from the literature, results of good quality for all three methods are reported, i.e., the average errors with respect to the best known solutions are 0.185%, 0.167% and 0.184% for FANT, GDH and VNS respectively with time necessary for 1,000 calls to the improving procedure of FANT. Best results, i.e., 0.104% error were obtained with the previous Hybrid Ant system (HAS-QAP) of [56].

Balanced MBA student teams problem. In some schools and universities, students from the same grade must sometimes be divided into several teams within a classroom in such a way that each team provides a good representation of the classroom population. A problem is to form these teams and to measure the quality of their balance. In [39] mathematical models are proposed that take into account the attributes assigned to the students. Two different ways of measuring the balance among teams are proposed: *min-sum* and *min-max* models. Two formulations are considered and both exact and heuristic solution methods are developed. The exact method consists in solving a *Set Partitioning* problem based on the enumeration of all possible teams. In order to solve large problem instances, both VNS and VNDS heuristics are also developed. In the basic VNS, interchange of ℓ ($\ell = 1, \ldots, \ell_{\max}$) student pairs is used for the perturbation of an incumbent solution. The local search is performed within $\mathcal{N}_1$, and an initial solution is obtained by using a random partition. Since the gap was larger than 1% on a 65 student instance with 19 attributes and 13 teams, VNDS was tested as well. The use of this extended VNS scheme led to obtain results of better quality (0.48% larger than optimum) in moderate time (29 seconds for VNDS versus 872 seconds for the exact method).

Simple plant location problem. The well-known simple plant location problem (SPLP) is to locate facilities among a given set of cities in order to minimize the sum of set-up costs and distribution cost from the facilities to a given set of users, whose demand must be satisfied. The case where all fixed-costs are equal and the number of facilities is fixed is the p-Median problem. In work in progress, VNS heuristics for the p-median are extended to the SPLP. In addition to interchange moves, opening and closing of facilities are considered and a VNDS similar to that one suggested in [84] developed. Moreover, complementary slackness conditions are exploited to find a bound on the error through solution of a reduced dual problem. In this way solution within 0.04% of optimality could be obtained for problems with up to 15,000 users (see Table 1).

	Fix		*Objective values*			*Time (sec.)*			*Time total*		
n	*costs*	p	CPLEX	RVNS	VNDS	CPLEX	RVNS	VNDS	*Best*	*All*	*Gap*
500	223	347	99794.0	107984	99794	0.1	0.2	0.3	0.6	1.5	0.0000
1000	316	391	223206.0	247790	223206	0.3	0.7	1.3	2.3	6.6	0.0000
2500	500	498	542166.5	614880	542267	2.8	3.2	34.8	40.8	66.4	0.0185
5000	707	600	1028221.0	1217007	1028255	19.5	6.9	239.6	266.0	382.1	0.0033
10000	1000	752	1914908.0	2163915	1915562	167.5	36.3	1530.4	1734.2	2212.4	0.0342
15000	1224	844	2733060.5	3134626	2733979	1071.9	61.3	5103.4	6236.6	7390.1	0.0336

Table 1: SPLP on Euclidean random instances with equal fixed costs ($\sqrt{n}/1000$); Dual is reduced by both upper and lower bounds; Sun 533 Mhz; stopping rules: number of iterations without improvement = 20 for both VNDS and VNS; k_{max}=20 for VNS; $k_{max} = \min\{p, 25\}$ for VNDS.

One-dimensional bin-packing problem (BPP). Packing items into boxes or bins is a task that occurs frequently in distribution and production. BPP is the simplest bin-packing problem: given a number of items $i = 1, \ldots, n$ of integer sizes t_i, what is the minimum number of bins, each having the same integer capacity c, necessary to pack all items.

In developing the basic VNS for BPP, usual neighborhood structures are used in [52]: add, drop and interchange (swap) of items between bins. By using sophisticated data structure, neighborhoods considered are significantly reduced. In an intensified shaking step an item is selected at random and its best position (w.r.t. add/drop/swap restricted neighborhood) found; to get x' that belongs to $\mathcal{N}_k(x)$, this step is repeated k times. Local search uses the same neighborhoods. In addition, since a characteristic of the BPP is existence of large plateaus (many different configurations, in terms of assignment of items to bins, correspond to the same number of bins), an auxiliary objective function is introduced, i.e., maximization of the sum of squared slacks of the bins. Initial solutions for VNS are obtained by modification of an existing one (called MBS'), which already produces good solutions. When tested on 1370 benchmark instances from two sources, VNS proved capable of achieving the optimal solution for 1329 of them, and could find for 4 instances solutions better than the best known ones. According to the authors, this is a remarkable performance when set against other methods. For example, when compared with the Tabu search heuristic from [127] on 1210 hardest test instances, it appears that in 1010, 1125 and 1170 instances the best solutions are found by MBS', TS and MBS'+VNS respectively.

5.3 Graphs and Networks.

Let $G = (V, E)$ be a connected (un)directed graph with vertex set V and edge set E. Solution of many optimization problems on G correspond to some subset of vertices $V' \subseteq V$ or subset of edges $E' \subseteq E$ that satisfies certain conditions. A usual way to supply a solution space with some metric, and thus make possible development of VNS heuristics, is to define a distance function as the cardinality of the (symmetric) difference between any two solutions V_1 and V_2 or E_1 and E_2, i.e., node-based or edge-based neighborhoods:

$$\rho_1(V_1, V_2) = |V_1 \setminus V_2| \quad (or \quad \rho_1 = |V_1 \Delta V_2|) \tag{2}$$

$$\rho_2(E_1, E_2) = |E_1 \setminus E_2| \quad (or \quad \rho_2 = |E_1 \Delta E_2|) \tag{3}$$

where Δ denotes the symmetric difference operator. In the applications of VNS that follow, ρ_1 or ρ_2 or both metrics are used for inducing different neighborhood structures $\mathcal{N}_k$:

$$V_2 \in \mathcal{N}_k(V_1) \Leftrightarrow \rho_1(V_1, V_2) = k \quad and/or \quad E_2 \in \mathcal{N}_k(E_1) \Leftrightarrow \rho_2(E_1, E_2) = k. \tag{4}$$

Oil pipeline design problem. Brimberg *et al* [15] consider a given set of offshore platforms and on-shore wells, producing known (or estimated) amounts of oil, to be connected to a port. Connections may take place directly between platforms, well sites and the port or may go though connection points at given locations. The configuration of the network and sizes of pipes used must be chosen in order to minimize construction costs. This problem is expressed as a mixed integer program, and solved both heuristically by Tabu search and VNS methods and exactly by a branch-and-bound method. Tests are made with data from the South Gabon oil field and randomly-generated problems. VNS gives best solutions and TS is close.

Phylogeny problem. The phylogeny (or evolutionary tree) problem, in its simplest form, can be formulated as follows: given A, an $n \times m$ 0-1 matrix, construct a tree with minimum number of 'evolutionary' steps, that satisfies the following conditions: (i) each node of the tree corresponds to an n-dimensional 0-1 vector, where vectors given by rows of A should be associated to leaves; (ii) degrees of all nodes that are not leaves are equal to three; (iii) two nodes are connected by an edge if their corresponding vectors differ in only one variable. The trees that satisfies (i)-(iii) and the criterion used are called phylogeny and parsimony respectively. The leaves of an evolutionary tree represent groups of species, populations of distinct species, etc (denoted by taxons as well), while interior nodes represent hypothetical (unknown) ancestors. In [3] a VND method that uses three neighborhoods is developed, and compared with the best among three local searches used alone. It appears that VND is much faster and gives solutions with better quality. In the same paper, the authors developed and compared three metaheuristic approaches to the phylogeny problem: GRASP (with the best local search among three for the second phase, hybrid GRASP with VND and VNS. The method developed as VNS use the same three neighborhoods for Shaking as for VND ($k_{max} = k'_{max} = 3$) and does therefore not use the full potential of VNS. Nevertheless it gives the best results. Comparative results on eight test problems are summarized in Table 2 (taken from [3]).

Heuristic	% Deviation	# of Best values	Time (minutes)
GRASP	0.6	5	481.8
GRASP/VND	0.6	2	321.9
VNS	0.5	6	714.4

Table 2: Phylogeny problem: final average results on 8 test problems.

Maximum clique problem. Let $G = (V, E)$ denote a graph with vertex set $V = \{v_1, v_2, \ldots, v_n\}$ and edge set $E = \{e_1, e_2, \ldots, e_m\}$. A set $C \subseteq V$ of vertices is a *clique* if any two of them are adjacent. A clique C is *maximum* if it has the largest possible number of vertices. Finding the maximum clique is a classical problem in applied graph theory. A basic VNS heuristic that combines greedy with the simplicial vertex test in its descent step is proposed and tested on standard test problems from the literature in [85]. Despite its simplicity, the proposed heuristic outperforms most of the well-known approximate solution methods. One exception is the recent Reactive Local Search (RLS) heuristic [7] which obtains results of equal quality in a quite different way.

Steiner tree problem on graphs. Given a graph $G = (V, E)$, nonnegative weights w_{ij} associated with the edges (i, j) of E and a subset $X \subseteq V$ of terminal nodes, the Steiner problem (SPG) is to find a minimum weighted subtree of G which spans all terminal nodes. The solution X and corresponding tree are called Steiner nodes and Steiner minimum tree respectively. Application can be found in many areas, such as telecommunication network design, computational biology, VLSI design, etc. (see e.g. [90] for a survey).

In [108] and later in [125], two types of neighborhoods, i.e., *node-based* (N) (2) and *path-based* (P) [134], are successfully used for solving SPG as in VND. For example, the average % of improvement over a constructive initial solution on 454 instances was 10.24% for the N-P order, while only P local search led to a 2.59% improvement. Computing time for both heuristics was the same. This N-P procedure is used within parallel GRASP in [108] and improved by Path relinking in [125], but no attempt was made to develop a full VNS, which could be an interesting task for future work.

Degree-constrained minimum spanning tree problem (DCMST) consists in finding a minimum spanning tree such that the degree of each vertex of the tree is less than or equal to a given integer b_i, for all i. Note that for $b_i = 2$ the problem becomes the Hamiltonian path problem. In [124] three neighborhood structures based on the k-edge exchange operator, $k = 1, 2, 3$ (or k-elementary tree transformations) are used in developing local search procedures within a VND heuristic. This procedure is then embedded into a VNS as a local search, where again k-edge exchange moves are used for perturbation (shaking) with parameter $k_{max} = 0.4(|V| - 1)$. VNS stops the first time $\mathcal{N}_{k_{max}}$ has been explored without improvement. VNS and VND are then extensively compared with known methods from the literature, i.e., a genetic search, problem space search heuristics and simulated annealing, on four different sets of randomly generated problems: CRD, SYM, STR and SHRD. VND and VNS were better and much faster on almost all test instances.

Problem		GA-F		VNS	
b_i	*dim*	% deviation	time	% deviation	time
3	3	0.17	257.55	0.00	7.58
3	4	1.10	271.85	-0.01	7.39
3	5	0.07	314.29	0.00	7.09
3	6	0.58	203.85	-0.01	11.52
3	7	0.58	233.01	0.00	8.88
4	3	0.00	245.80	0.00	5.55
4	4	0.38	240.35	0.00	6.79
4	5	0.02	254.46	0.00	7.55
4	6	0.51	233.18	0.00	8.62
4	7	0.51	232.48	0.00	9.15
5	3	0.00	220.06	0.00	4.14
5	4	0.05	213.79	0.00	6.45
5	5	0.00	219.51	0.00	7.08
5	6	0.46	206.02	0.00	7.73
5	7	0.48	207.28	0.00	8.83

Table 3: DCMST problem: test problems STR, for $|V| = 100$; parameter *dim* designates the dimension of the space for randomly generated data; sign '-' shows that the best known solution is improved.

Some observations from their empirical study are as follows: (i) VND alone succeeded in finding optimal solutions for all problems already solved exactly (in the STR class) in less than one second of processing time, while the best among three other heuristics, i.e., the problem space search heuristic, found suboptimal solutions within up to 300 seconds; (ii) on SHRD class of instances, VND reduced the average relative error of Local search (with a single neighborhood $k = 1$) from 4.79% to 0.27% with a very small increase in computing time; (iii) for the same series of instances, VNS found 23 out of 24 best solutions; (iv) for instances CRD and SYM, VNS got optimal solution on 69 out of 70 instances; (v) VNS found optimal or best known solutions for all instances of class STR; it improved the average best solution values found by GA-F [99] for 35 out of the 45 pairs of instances considered. Some of the results that compare GA-F and VNS on STR instances are given in Table 3 (a part of Table 5.7 from [124]).

Max-cut problem. Given an undirected graph $G = (V, E)$ and weights w_{ij} on the edges $(i, j) \in E, i, j \in V$ the Max-cut problem is to find a subset of vertices S that maximizes the sum of the edge weights in the cut $(S, \bar{S})$. This problem has wide applications including VLSI design. Three heuristics for solving Max-cut are developed and compared in [49]: GRASP, basic VNS and a hybrid of GRASP and VNS (G-VNS). Solutions are represented as binary $|V|$-vectors and neighborhood structures for VNS are induced from the Hamming distance introduced into the solution space. Local search is performed in the neighborhood with distance 1. G-VNS uses VNS in the local search phase of GRASP. Results on 4 instances from the literature are reported. Preliminary computational results indicate VNS is a robust heuristic for Max-Cut. Indeed VNS is not very sensitive to the choice of initial solution, and thus, G-VNS did not perform better than VNS for the same number

of iterations.

Cable layout problem. Given the plan of buildings, the location of the equipment and a structure of cable supports of fixed capacities, cables of all kind have to be routed (voltage, communication, etc.). The Cable layout problem (CLP) consists in finding the design of cables that minimize the total cost, such that numerous technical requirements are met. If the cost of a cable is a linear function of its length, then the CLP can be considered as a capacitated multi-path problem. This property is used in [31] to decompose the problem, and then a VNDS heuristic is proposed. Each sub-problem consists in solving several min-cost multi-paths problems. Graphs with up to 1,000 nodes are randomly generated with up to 4% edge density (which is close to real problems) in order to compare VNDS with a previously suggested TS based heuristic [86]. Solutions obtained by VNDS were better than TS in 15 out of 16 instances, in much less computing times. Moreover, for larger instances, it is shown (from the dual bounds obtained from Lagrangian relaxation), that the duality gap is always less than 1%.

k-Cardinality tree problem. Given an undirected weighted graph $G = (V, E)$ with vertex set V, edge set E and weights $w_i \in R$ associated to V or to E, the *Minimum weighted k-Cardinality tree problem* (k-CARD for short) consists of finding a subtree of G with exactly k edges whose sum of weights is minimum. There are two versions of this problem: *vertex-weighted* and *edge-weighted*, if weights are associated to V or to E respectively. The k-CARD problem is strongly NP-hard [51]. However, if G is a tree then it is polynomially solvable [51]. In [115] two VNS methods for the edge-weighted k-CARD problem, are proposed: the basic VNS (denoted by VNS-1) and VNS that uses VND as a local search (VNS-2). In VNS-1 the solution space (i.e. the set of all trees with k edges) is supplied with a distance function based on edge difference between any two trees and all neighborhood structures are induced from it. Since the minimal weighted k-cardinality tree is a spanning tree on any subgraph of G with $k+1$ vertices, the solution space may be reduced to the set of all spanning trees and the set of spanning trees $\mathcal{T}_k$ may be supplied with another metric, i.e., as the cardinality of the difference of their vertex sets. In developing their VND method for k-CARD, the authors use three neighborhood structures. The first two neighborhoods are induced by *edge distance*, and the third one is induced by a *vertex distance*. In Table 4 comparative results of VNS heuristics, two Tabu search methods (TS-1 and TS-2 for short) and the Dynamic-Dijkstra-Path (DDP for short) constructive heuristic are given. (In [45] it is shown that DDP has the best performance when compared with several others constructive heuristics). TS-1 is an implementation of Chain interchange TS rules suggested in [113] and the second TS (denoted by TS-2) is taken from [104]. Test instances include random graphs with 500, 1000, 1500 and 2000 vertices, where degree of each node is set to 20 (see [104] for details). In Table 4 average results on 10 random instances for each n and k are reported. Large problem instances could not be solved by DDP and TS-2 in reasonable time and for that reason we did not report on them in Table 4. It appears that VNS-2 performs best in average for all but one sizes of problems.

Prize-collecting Steiner tree problem on graphs. Given a graph $G = (V, E)$, non-negative weights w_{ij} associated with the edges (i, j) of E and a nonnegative prizes π_i

n	k	*Objective value*					t_{max} (sec)
		VNS-1	VNS-2	TS-1	TS-2	DDP	
500	50	1577.10	1561.50	1600.40	1713.70	**1535.90**	78.
	100	3424.80	**3395.00**	3604.80	3736.40	3397.50	93.
	150	5553.30	**5525.30**	5826.10	5915.80	5541.50	117.
	200	7974.60	**7959.20**	8231.50	8626.80	7980.70	148.
	250	10814.80	**10795.90**	11203.30	11349.10	10829.40	170.
1000	100	5863.30	**5844.30**	5912.50	-	5847.20	585.
	200	11947.80	**11915.40**	12079.60	-	11922.00	655.
	300	18230.20	**18220.10**	18393.60	-	18222.60	775.
	400	24758.90	**24748.30**	25002.30	-	24763.90	930.
	500	31572.90	**31566.80**	31886.10	-	31591.90	1138.
1500	150	8209.40	**8197.20**	8260.90	-	-	3000.
	300	16569.60	**16557.60**	16686.40	-	-	3500.
	450	25107.80	**25039.80**	25293.04	-	-	4000.
	600	33842.80	**33804.10**	34099.20	-	-	5000.
	750	42820.10	**42785.40**	43118.30	-	-	6000.
2000	200	23722.60	**23586.30**	23924.90	-	-	3000.
	400	48188.40	**48077.80**	48867.90	-	-	3500.
	600	73486.50	**73344.10**	74461.30	-	-	4000.
	800	99670.40	**99535.80**	100950.40	-	-	5000.
	1000	126938.20	**126804.70**	128311.60	-	-	6000.

Table 4: k-Cardinality tree problem: average results on 10 random instances. Best values are boldfaced.

associated with the vertices i of V, the *Prize-collecting Steiner tree problem* (PCSTP), is to find a subtree of G which minimizes the sum of the weights of its edges, plus the prizes of vertices not spanned. If the subset of vertices X to be spanned is known, we have the *Steiner tree problem*. PCSTP has an important application in design telecommunication of local access networks. In [24], among other heuristics, VNS has been proposed for PCSTP. In the solution space, represented by all spanning trees with cardinality $k = 1, 2, \ldots, |V|$, the neighborhood of $T(X)$ is defined as a set of spanning trees $T(X')$ having one different vertex ($|X \setminus X'| = 1$ or $|X' \setminus X| = 1$. The k^{th} ordered neighborhood is also defined by vertex deletions or additions. Each tree considered is always transformed into another tree by a so-called peeling algorithm, which iteratively eliminates leaves whose weights are larger than corresponding prizes. The basic VNS developed is used as a post-optimization procedure, i.e., it starts with a solution obtained by a known constructive (2-approximation primal-dual) algorithm (denoted with GW) plus iterative improvement with two types of perturbations (It. Impr.) plus local search with path-relinking (LS+PR). Summary results from [24] on three known classes of hard instances are given in Table 5. It appears that in each class, VNS improved the best known solution for three instances.

5.4 Scheduling

Single machine scheduling problems. Given a set of n jobs that has to be processed without any interruption on a single machine, which can only process one job at a time,

		It. Impr.		LS+PR			VNS		
Series	Instances	# Opt	% Impr.	# Opt	% Impr.	Time	# Opt	% Impr.	Time
JMP	26	6	1.2	18	2.3	157.8	21	2.5	94.1
C	34	2	7.8	27	8.6	956.8	30	10.5	796.6
D	35	6	4.7	17	7.0	7668.1	20	7.5	2749.2

Table 5: Prize-collecting Steiner tree problem: Improvements over GW.

several NP-hard versions of Single machine scheduling problem that differ in objective function are solved by VND and VNS in [10]. Since the solution can be represented as a permutation of jobs, three neighborhoods of different cardinalities are used within VND, i.e., 1-opt (or transpose), insert (or Or-opt) and interchange neighborhoods have cardinalities $n-1$, $(n-1)^2$ or $n(n-1)/2$, respectively. They are used in sequence, without return to the smallest one after improvement (as in the basic VND). It appears, from extensive computations, that VND significantly improves solutions compared with those obtained by single neighborhood local searches in small additional computing time. Another interesting observation is that ordering 1-opt, Or-opt, interchange is better than 1-opt, interchange, Or-opt, despite the fact that the cardinality of the Or-opt neighborhood is larger than that of the interchange one.

Nurse rostering problems (NRP) consist of assigning varying shift types to hospital personnel with different skills and work regulations, over a certain planning period. Typical shift types are Morning, Day and Night shift. There are many constraints in the NRP which are divided (in [20]) into hard (which can never be violated and expressed in the set of constraints) and soft (which are preferably not violated). The objective is to minimize the number of violated soft constraints. The commercial nurse rostering package *Plane*, which is implemented in many Belgian hospitals, makes use of Tabu search [18]. For the same problem VNS is applied in [20], where several neighborhoods are constructed in order to especially satisfy some of the constraints. It appears that VND enables the system to find schedules which are hidden for single neighborhood heuristics. Moreover, the authors conclude that "it is often more beneficial to apply simple descent heuristics with [a] variety of neighborhoods than to use sophisticated heuristics which are blind for large parts of the search space".

Multiprocessor scheduling problem with communication delays. This problem consists in finding a static schedule of an arbitrary task graph onto a homogeneous multiprocessor system, such that the makespan (i.e., the time when all tasks are finished) is minimum. The task graph contains a precedence relation as well as communication delays (or data transferring time) between tasks if they are executed on different processors. The multiprocessor architecture is assumed to contain identical processors with a given distance matrix (i.e., the number of processors along the shortest path) between each two processors [35]. For solving this NP-hard problem [133], a basic VNS heuristic is developed in [36], where a k-swap neighborhood structure is used for shaking and a reduced 1-swap for local search. It is compared with Multistart local search (MLS), Tabu search (TS) [36] and Genetic al-

Set	# Instances	Best-known	GA	Init.	Init.+	VNS	VNS+
J60	480	10.76	11.89	12.98	12.18	11.13	10.94
J120	600	32.17	36.74	40.02	37.38	33.88	33.10

Table 6: RCPSP: % deviation from critical-path lower bound.

gorithms [1] (PSGA). Initial solutions for all methods are obtained by modification of the well-known constructive heuristic *critical path* (CP). Starting from it a *swap* local search is run (LS). Two types of task graphs are generated: (i) with known optimal solution on the 2-dimensional hypercube (i.e. with 4 processors) and given density as proposed in [100]; (ii) with given density 0.2, 0.4, 0.5, 0.6 and 0.8. It appears that for both types of random test instances VNS performs best.

Resource-constrained scheduling problem (RCPSP) is concerned with n non-preemptive activities and m renewable resources. Given the availability of each resource k (R_k), processing time for each activity j in time units (d_j), amount of resource k needed for activity j, during d_j (r_{jk}) and for each activity j a sets of immediate predecessors (P_j) or a set of immediate successors (S_j), find for each activity j its start time s_j ($s_1 = 0$) such that the makespan of the project $T = s_n$ is minimum.

Since the RCPSP is an important and difficult NP-hard problem, a considerable amount of research on it has appeared in the literature (for a recent survey see e.g. [21]). In [53], a VNS heuristic is developed for the RCPSP. The solution is represented as a feasible sequence of activities (that respect precedence conditions), and in order to find makespan, an additional procedure is used. The disadvantage of such a representation is that the same schedule could be obtained from several permutations. Neighborhood for a local search is defined as a sequence of feasible 1-opt (or transpose) moves, i.e., an activity is inserted between its two closest ones (to the left and to the right) in the current permutation until all left activities are from P_j or right activities from S_j. In addition, this neighborhood is reduced by a parameter α which defines the maximum number of such 1-opt moves. These moves allow an efficient updating of the objective function. In order to derive a solution from $\mathcal{N}_k$, the same sequence of moves is repeated k times. In the Shaking step, k (instead of one, as in the basic VNS) solutions are generated and the best among them is used as an initial solution for the Local search. Very good results by VNS are reported on 4 classes of problems from the literature: J30 (480 instances), J60 (480), J90 (480), J120 (600). Some average comparative results, given in Table 6, include GA as well, which was identified in [96] as the most effective.

Capacitated lot-sizing problem with setup times (CLSP-ST). The trend towards just-in-time manufacturing has led to a drastic reduction of setup times in many manufacturing processes. Given are a fixed costs r_{it} for producing item i in period t ($i = 1, .., n$; $t = 1, .., T$), variable costs (per unit production cost and per unit inventory holding cost in period t), the demand for each item in each period and the amount of each resource available. In the CLSP-ST it is required to determine the lot sizes of each item in each

Method	Better than TTM	Same as TTM	Worse than TTM	Not solved	Gap % avg.	Gap % Max	Time (s.) avg.	Time (s.) Max
TTM	-	-	-	0	4.16	31.87	0.60	3.61
DBKZ	35	2	453	31	9.11	47.86	0.13	1.90
M-DBKZ	100	39	352	1	5.22	29.28	2.00	16.03
SM	437	42	13	0	2.77	31.40	0.81	11.66
SM+VNS	447	35	10	0	2.51	22.90	5.84	117.36

Table 7: CLSP-ST: Comparative aggregate results from [88]

time period, such that the total cost is minimized and the capacity constraints satisfied.

Very few papers address the CLSP-ST. Usually Lagrangian relaxation with subgradient optimization is employed to calculate a lower bound and different heuristics are developed in attempt to find good feasible solutions. In [88], beside Lagrangian relaxation, a new smoothing heuristic (SM) followed by a VNS is suggested. In the Shaking step, the k setups corresponding to k smallest additional costs are switched off (and the corresponding transshipment problem is solved), while SM is used again as a local search within VNS. Only one loop of the basic VNS is applied with $k_{max} = 7$, i.e., the first time k reaches 7, the procedure stops. All tests were carried out on a subset of the 751 benchmark test instances. Five methods were tested, two known and three new: TTM [132], DBKZ [40], M-DBKZ (modification of DBKZ), SM and SM+VNS. It appears that, among 492 hardest problem instances, SM and SM+VNS improve 437 and 447 solutions, being 13 and 10 times worse, respectively. In Table 7 comparative aggregate results w.r.t. TTM on 492 hardest problem instances are reported.

5.5 Artificial intelligence

Weighted Max-SAT problem. The satisfiability problem, in clausal form, consists in determining if a given set of m clauses (all in disjunctive or all in conjunctive form) built upon n logical variables has a solution or not [55]. The maximum satisfiability problem consists in finding a solution satisfying the largest possible number of clauses. In the *weighted maximum satisfiability* problem (WMAXSAT) [123] positive weights are assigned to the clauses and a solution maximizing the sum of weights of satisfied clauses is sought. Results of comparative experiments with VNS and TS heuristics on instances having 500 variables, 4500 clauses and 3 variables per clause, in direct or complemented form, are given in Table 8 [74]. It appears that using a restricted neighborhood consisting of a few directions of steepest descent or mildest ascent in the Shaking step does not improve results, but using this idea in conjunction with SVNS improves notably upon results of basic VNS and also upon those of a TS heuristic.

Learning Bayesian networks. Let us consider a set of random variables $V = \{x_1, x_2, \ldots, x_n\}$ and a set of parameters which together specify a joint probability distribution over the random variables. Bayesian networks (BNs), also known as Belief or Causal networks,

	VNS	VNS-low	SVNS-low	TS
Number of instances where best solution is found	6	4	23	5
% Average error in 10 trials	0.2390	0.2702	0.0404	0.0630
% Best error in 10 trials	0.0969	0.1077	0.0001	0.0457
Total number of instances	25	25	25	25

Table 8: Results for *GERAD* test problems for WMAXSAT ($n = 500$).

are knowledge representation tools able to manage the dependence and independence relationships among the random variables [117]. BN is represented by a directed acyclic graph (dag). Once the BN is specified, it constitutes an efficient device to perform inference tasks. The problem is to develop automatic methods for building BNs capable of learning directly from given data ($D = \{v^1, v^2, \ldots, v^m\}$, containing m instances of V), as an alternative or a complement to the method of eliciting opinions from experts. This NP-hard problem is receiving increasing attention in Artificial intelligence. In [22] the current dag (solution) is represented by a permutation θ of n variables. The quality of an ordering is measured by the so-called *scoring metric*, f, i.e., a function, $f(\theta, D)$, defined for dags and a search in the space of dags compatible with θ performed. Then the basic VNS scheme (called VNS based on ordering, VNSO for short) is applied. The set of neighborhood structures is defined by $k - interchange$ moves in θ and local search performed for $k = 1$. VNSO has been successfully compared with two other methods from the literature.

5.6 Continuous optimization

Bilinear programming problem. Structured global optimization problems, while having several and often many local optima, possess some particular structure which may be exploited in heuristics or exact algorithms. One such problem, of considerable generality, is the bilinear programming problem (BLP) with bilinear constraints. This problem has three sets of variables, x, y and z, with cardinalities n_1, n_2 and n_3 respectively. When all variables of y are fixed it becomes a linear program in x and z. When all variables of z are fixed it becomes a linear program in x and y. This property suggests the well-known *Alternate* heuristic:

1. *Initialization:* Choose values of variables of z (or y);
2. *LP-1*: solve the linear program in (x, y) (or in (x, z));
3. *LP-2*: For y (or z) found in the previous step, solve the linear program in (x, z) (or in (x, y));
4. If stability is not reached (within a given tolerance) return to 2.

Obviously this algorithm may be used in a Multistart framework. To apply VNS one may observe that neighborhoods $\mathcal{N}_k(x, y, z)$ of a solution (x, y, z) are easy to define. They correspond to k pivots of the linear program in (x, y) or in (x, z), for $k = 1, 2, \ldots, k_{max}$. One can then apply the basic VNS of Figure 3 in a straightforward way. The local search routine is the alternate heuristic described above. Results on randomly generated test problems are presented in Table 9 from [6]. It appears that VNS improves in almost

Parameters					CPU time		% Error	
n	m	n_1	n_2	n_3	MALT	VNS	MALT	VNS
50	36	30	10	10	0.7	1.0	1.09	0.00
60	36	30	20	10	1.6	2.4	10.05	0.00
70	36	30	30	10	0.9	5.8	20.38	0.00
80	36	30	40	10	1.8	2.7	51.22	0.00
90	36	30	50	10	1.8	3.7	43.77	0.00
100	36	40	50	10	3.6	11.9	18.45	0.00
110	36	50	50	10	3.7	7.3	15.90	0.00
120	36	60	50	10	10.0	22.4	39.12	0.00
130	36	70	50	10	15.0	30.0	34.56	0.00
140	36	80	50	10	8.8	13.9	21.01	0.00
150	36	90	50	10	6.8	10.0	42.87	0.00
Average					4.35	9.19	23.23	0.00

Table 9: BLP: results for 10 repetitions of ALT (MALT) and VNS: each line reports average results on 4 random test problems with same parameters.

all cases and sometimes very substantially upon the solution provided by the Multistart Alternate heuristic.

Pooling problem. The pooling problem, which is fundamental to the the petroleum industry, describes a situation where products possessing different attribute qualities are mixed together in a series of pools in such a way that the attribute qualities of the blended products of the end pools must satisfy given requirements. It is well known that the pooling problem can be modeled through bilinear programming. A simple alternating procedure and a VNS heuristic are developed to solve large instances, and compared with the well-known method of successive linear programming [6]. This is an application of BBLP above.

Continuous Min-Max problem. The so-called multi-level Tabu search (MLTS) heuristic has been proposed in [97] for solving a continuous min-max optimization problem (in R^n with m periodic functions) of spread spectrum radar polyphase code design problem [43]. Improvements obtained with the basic VNS in both solution quality and computing time on the same set of test problems are reported in [114] and [82]. Different neighborhoods are derived from the Euclidean distance. A random point is selected from such neighborhoods in the *Shaking* step; then the gradient (feasible direction) local search with a given step size is performed on the functions that are active (functions that have a maximum value in the current point); this step is repeated until the number of active functions in the current point is equal to n (with some tolerance). It appears that VNS outperforms MLTS on all test instances.

6 Understanding heuristics and metaheuristics

The advent of metaheuristics has led, for a large variety of problems, to the design of heuristics with a much improved empirical performance, without however that theoretical

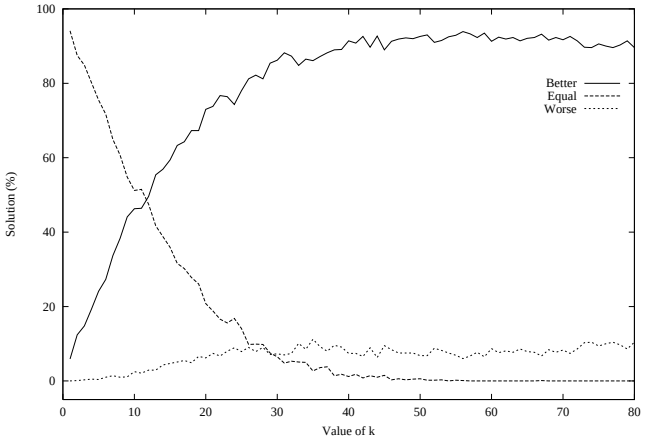

Figure 6. A mountain profile

reasons for such improvements be clearly understood. First steps towards a better understanding are to study the topography of local optima, and valleys or mountains, for given classes of problems, as well as the trajectories followed by the heuristics and the resulting phases of improvement or deterioration. We next describe two new tools, i.e., *mountain* (or *valley*) profiles and *distance-to-target visualization* which have been found helpful in deriving better heuristics or variants of basic schemes for VNS. We then turn to the study of an unexpected phenomenon, i.e., that when selecting moves in a heuristic *first improvement can be better than best improvement.* Finding the reason why this is so does not lead directly to a competitive heuristic for the problem under study, i.e., the traveling salesman problem, but does pinpoint a potential defect of many heuristics.

6.1 Mountain or valley profiles

When applying VNS, the descent from the randomly selected solution x' can lead back to the local optimum x around which the current neighborhoods are centered, or to another local optimum x" the value of which can be better or not than that of x. It is thus natural to study the probabilities of these three outcomes as a function of distance from x to x'. It is also worthwhile to observe whether x" is closer to x than x' (which may be interpreted as x'' belonging to the same large valley, with local rugosities in relief) or not (which may be viewed as indicating a new large valley has been found). Mountain profiles give such information on the basis of 1000 ascents from points in each successive neighborhood. Examples of such profiles for the *weighted maximum satisfiability problem* are given in Figure 6, from [74].

Profiles appear to vary considerably with the quality of the local optimum x: when it is bad it suffices to go a little away from x to obtain, with high probability, a better local optimum. When it is good, or very good, one must go quite far to find a new large valley and, moreover, the probability of finding a solution better than the incumbent is then low. This illustrates a weakness of the basic VNS scheme, already mentioned above: it tends

to degenerate into *Multistart* when the distance from x to x' becomes large. The solution, also presented above, is to resort to the SVNS scheme.

6.2 Distance-to-target visualization

When designing heuristics for well studied classes of problems, small to medium size instances for which the optimal solution is known are often available. Such instances are used to find how often the optimal solution is reached, or what is the average percentage of error of the heuristic. But much more information can be obtained if the optimal solutions themselves are considered, and not only their values. In ongoing work on a VNS approach to the Traveling salesman problem [58] a *distance-to-target visualization* tool has been developed. This tool presents on screen the optimal solution for the training instance under study, the current solution and the symmetric difference between these two solutions. This indicates how much improvement is to be made and where. Moreover, a routine allows also the representation of the *difference* of solutions at an iteration of the heuristic and at the next. Finally, as representations of solutions for large instances may be hard to read, and many problems, in particular Euclidean ones, allow for a natural decomposition, a *focusing* routine allows representations of the above-mentioned information for chosen subproblems, e.g. in some region of the plane.

Visualizing successive steps in VND and VNS show their strengths and weaknesses, e.g. that after 2-opt is applied within VNS for the TSP much work remains to be done. Suggestions for further neighborhoods to use may then be obtained from a careful study of remaining differences between solution and target.

6.3 First vs best improvement

In addition to using the visualization tools described above, one can proceed to a step-by-step evaluation of the work done by a VNS or other heuristic. Moreover, variants at each of these steps can be studied. Detailed information, which would not be apparent from the global performance of the heuristic, can then be gathered. It can lead to the discovery of unexpected phenomena and the insight provided by their explanation can in turn lead to principles for building better heuristics. A typical case is the observation that when applying the 2-opt heuristic to the TSP [79], selecting at each iteration the first improvement, i.e., the first exchange of two edges which reduces the objective function value gives better results than selecting the best improvement, i.e., the exchange of two edges which reduces most the objective function (beginning from a randomly chosen solution). An explanation, corroborated by further detailed analysis, is easily obtained: if the best exchange is chosen, two edges of small but not necessarily very small length are introduced in the tour and are difficult to remove at later iterations; if a first improvement is chosen, due to ranking of edges by order of increasing lengths, a very small edge and an average length edge are introduced in the tour and the latter is easy to remove at a later iteration. Systematic study of such phenomena in a variety of heuristics could lead to many improvements.

7 Improving exact algorithms

It is well known that many combinatorial optimization problems such as the p-median or the Multisource Weber problem [41], the clustering or partitioning problem with various objectives [73], [44], the air-crew scheduling problem [38], etc. can be solved by combining column generation with integer programming. As the optimal solution of the linear relaxation is often highly degenerate, convergence may be slow and even when it is found many iterations may be needed to prove that it is indeed optimal. The dual of the column generation procedure is the outer approximation method of [95]. Slow convergence comes from the generation of many hyper-planes far from the cone vertexed at the optimal dual solution. One would therefore want to estimate this last solution and focus the generation procedure by penalizing hyperplanes far from it. If extracting the optimal dual solution is difficult, then the current dual values can be used to stabilize the generation procedure instead of focusing it.

Consider a primal problem P and its dual D, solved by column generation:

$$
\begin{array}{llrcl}
 & & \min c^T x & & \\
(P) & \text{s.t.} & Ax & = & b \\
 & & x & \geq & 0
\end{array}
\qquad\qquad
\begin{array}{llrcl}
 & & \min b^T \pi & & \\
(D) & \text{s.t.} & A^T \pi & \leq & c \\
 & & (\pi) & &
\end{array}
$$

In [41] a stabilization scheme is proposed which remains entirely within the linear programming column generation framework. It merges a perturbation and an exact penalty method.

A primal problem $\tilde{P}$ and its dual $\tilde{D}$ are defined as follows:

$$
\begin{array}{llrcl}
 & & \min c^T \tilde{x} - \delta_- y_- + \delta_+ y_+ & & \\
(\tilde{P}) & \text{s.t.} & A\tilde{x} - y_- + y_+ & = & b \\
 & & y_- & \leq & \varepsilon_- \\
 & & y_+ & \leq & \varepsilon_+ \\
 & & \tilde{x}, y_-, y_+ & \geq & 0
\end{array}
$$

$$
\begin{array}{llrcl}
 & & \max b^T \tilde{\pi} - \varepsilon_- w_- - \varepsilon_+ w_+ & & \\
(\tilde{D}) & \text{s.t.} & A^T \tilde{\pi} & \leq & c \\
 & & -\tilde{\pi} - w_- & \leq & -\delta_- \\
 & & \tilde{\pi} - w_+ & \leq & \delta_+ \\
 & & (\tilde{\pi}), w_-, w_+ & \geq & 0
\end{array}
$$

In the primal y_- and y_+ are vectors of slack and surplus variables with upper bounds ε_- and ε_+ respectively. These variables are penalized in the objective function by vectors $[-\delta_-, \delta_+]$. When applying column generation, after finding no more column with negative reduced cost, the parameters δ and ε are updated following problem-specific rules.

For this *stabilized column generation* scheme to work well, good estimates of the optimal dual variables should be obtained at the outset. Ways to find heuristically ranges

for these dual variables by sensitivity analysis are sketched in [41] for a few problems. For the p-Median problem this led to exact solution of instances with over 3000 users versus 900 in the literature; for its counterpart in the plane, i.e., the Multisource Weber problem ([98], [75]) it led to exact solution of instances with over 1000 users versus 30 in the literature. For the minimum sum-of-squares clustering problem combining stabilized column generation, with the ACCPM interior point method, hyperbolic 0-1 programming quadratic 0-1 programming and VNS in several steps [44] led to exact solution of problems with up to 150 entities, including the famous 150 iris example of [50].

8 Computer-aided discovery in graph theory

The AutoGraphiX system. Recall that a graph invariant is a variable defined on the family of all graphs (or on an appropriate sub-family) and the value of which does not depend on the numbering of vertices or edges. Numerous problems in graph theory can be viewed, or involve, optimization of invariants on a possibly constrained family of graphs. Therefore, VNS can be used for approximate solution of such problems. This led to the development of the AutoGraphiX (AGX) system and to a series of papers, next reviewed, on its principles and applications.

As explained in [29] , AGX uses the basic schemes of both VNS and VND. The descent is done by first drawing a graph G at random (with a given number of vertices and edges), computing its value for the invariant under study and then examining the effect on this value of bringing some elementary change to G: removal or addition of an edge, displacement of an edge, detour, i.e., replacement of an edge between two vertices by a path of length 2 joining them through some non-adjacent vertex, and so on. Neighborhoods so defined are ranked from the smallest to the largest. The best move is determined in each of them in turn and if it improves the value of the invariant studied the corresponding change is made in G. After a local optimum has been found, neighborhoods are defined using the Hamming distance, i.e., by considering removal or addition of edges to G, and VNS is applied. The only parameter is the maximum number of edges to be removed or added.

AGX can be applied to the following problems: (a) find a graph satisfying given constraints; (b) find optimal or near optimal values for an invariant subject to constraints; (c) refute a conjecture; (d) suggest a conjecture (or sharpen one); (e) suggest a proof. For instance, three conjectures of the system Graffiti [46] [47] are refuted in [29] , several strengthened and one of these proved, for the particular case of trees, exploiting knowledge of moves needed to find local optima.

Automated conjecture finding. Study of a set of extremal or near-extremal graphs G obtained with AGX taking the numbers n of vertices and m of edges as parameters often suggests conjectures. Moreover, there can be corroborated or refuted by performing various changes interactively. However, it would be preferable to have an entirely automated system. Three ways to automate conjecture finding are outlined in [28]: (a) a numerical method, which exploits the mathematics of Principal Component Analysis in order to find a basis of affine relations between graph invariants; (b) a geometric method which

consists in finding the convex hull of the points representing extremal or near-extremal graphs in invariants space, with a 'gift-wrapping' algorithm. Hyperplanes of this convex hull correspond to conjectures; (c) a geometric approach, which consists in recognizing the families of graphs to which belong the extremal ones and using known relations between graph invariants in those families to find new relations.

Chemical graph theory. 1. Energy. The π-electronic energy E is defined in chemical graph theory as the sum of absolute values of the eigenvalues of the adjacency matrix [68]. It has been extensively studied by chemists and mathematicians. As explained in [25] AGX led to the discovery of several simple, but as yet unknown relations, including the following:

$$E \geq 2\sqrt{m} \quad and \quad E \geq \frac{4m}{n}$$

which were easily proved.

Chemical graph theory. 2. Randić index. The Randić index [120] and its generalization are probably the most studied invariants of Chemical graph theory. Assign to each edge of a graph G a weight equal to the inverse of the geometric mean of the degrees of its vertices. Then, the Randić index of G is the sum of all such weights. AGX was used to find extremal trees for this index [27]. Maximum values are obtained by paths and minimal ones by two kinds of caterpillars and one of modified caterpillars with an appended 4-star. This last result was proved by an original way of using linear programming. Indeed, variables are associated with the numbers of edges with given degrees of end vertices (in chemical graphs these degrees are bounded by 4). This proof technique has already been used in three other papers of Chemical graph theory.

In [5] are presented several bounds on the Randić index of chemical trees in terms of this index for general trees and of the ramification index, which is the sum for all vertices of the excess of their degree over 2. Use of AGX [76] leads to correct some of these results and obtain much strengthened and best possible versions of the others.

Chemical graph theory. 3. Polyenes with maximum HOMO--LUMO gap. Using AGX, [54] a study was made of fully conjugated acyclic π systems with maximum HOMO-LUMO gap. In the simplest Hückel model, this property of the eigenvalue spectrum of the adjacency matrix of the corresponding chemical tree corresponds to reactivity. AGX gives for all even n a 'comb', i.e., a path on $n/2$ vertices to each of which is appended a single pendant edge. From this, the conjecture that the maximum gap tends to $2\sqrt{2} - 2$ when n goes to infinity can be deduced.

Trees with maximum index. Trees are bipartite and hence bicolorable, say in black and white. Color-constrained trees have given numbers of black and white vertices. In [34] AGX is used to study color-constrained trees with extremal index, or largest eigenvalue of the adjacency matrix. Six conjectures are obtained, five of which are proved.

Applying the numerical method for automated conjecture finding to the extremal trees found gave the conjecture

$$\alpha = \frac{1}{2}(m + n_1 + D - 2r)$$

where α is the stability number, n_1 the number of pendant vertices, D the diameter and r the radius of the tree. It is unlikely that a conjecture with so many invariants would be obtained without computer aid.

Trees with palindromic Hosaya polynomial. The Hosaya (or distance) polynomial of a graph is defined by

$$H(G) = a_0 + a_1 x + \ldots + a_D x^D$$

where $a_o = n$, $a_1 = m$ and $a_k (k = 3, ..., D)$ is the number of pairs of vertices at distance k. It was conjectured in [67] and [69] that there is no tree with a palindromic Hosaya polynomial , i.e., such that $a_0 = a_D, a_1 = a_{D-1}$ and so on. AGX refuted this conjecture [26]. All counter-examples have D even. This case is easier to satisfy than that where D is odd. Indeed, in the former case, there is a free coefficient in the center, and in the latter not. Then, define the distance to the palindrome condition as

$$z_p = \sum_{k=0}^{\lfloor \frac{D}{2} \rfloor} |a_k - a_{D-k}|$$

AGX gives for all $n = 10$ to $n = 50$ trees with $z_p = \lceil \frac{n}{2} \rceil$ (and higher values, due to border effects for $n \leq 10$). The conjecture

$$z_p \geq \lceil \frac{n}{2} \rceil$$

appears to be hard to prove.

Graffiti 105. The transmission of distance of a vertex of a graph G is the sum of distances from that vertex to all others. Conjecture 105 of Graffiti [47] is that in all trees the range of degrees does not exceed the range of transmission of distances. In [4] AGX is used to get hints on how to prove this conjecture and study variants of it. Minimizing range of transmission minus range of degrees always gives stars. It is then easy to prove that the conjecture holds with equality for stars only. Moreover, one gets the conjectures: (a) if T is not a star then range of transmission minus range of degree is at least $\lceil \frac{n}{2} \rceil - 2$, and (b) if T has maximum degree $D \leq \lceil \frac{n}{2} \rceil$ then range of transmission minus range of degree is not less than $n - 3D - 1$.

These results are easily proved, and, with a little more work, the extremal graphs characterized.

These examples illustrate the help the VNS-based system AGX can bring to discovery in graph theory. Many further ones are given in the cited papers.

9 Conclusions

Heuristics are an essential tool in applied optimization, and for many large and often messy problems encountered in practice the only applicable one. Their main aim is to provide, in reasonable time, near-optimal solutions to constrained or unconstrained optimization problems. Moreover, they may also be very useful within complex exact algorithms, to accelerate many steps. Finally, they are an important building block in optimization-based

knowledge discovery systems. Optimization problems are ubiquitous in Operations Research and its applications to numerous fields. Artificial intelligence tends to focus more on constraint satisfaction problems. Heuristics then seek a feasible solution. Both types of problems are less different than might appear at first view as Constraint satisfaction problems can be expressed as optimization problems in a variety of ways, e.g., as minimization of a sum of artificial variables representing constraint violations.

While traditional heuristics, such as, e.g., simple descent methods, are blocked in the first local optimum found, this is not the case for heuristics built within the metaheuristics paradigms. All of them provide methods, deterministic or stochastic, for getting out of poor local optima. As such local optima often differ considerably in value from the global optimum, particularly if there are many, the practical impact of metaheuristics has been immense. In contrast to this success, the theory of metaheuristics is lagging. While good heuristics are often obtained, with some ingenuity and a lot of parameter setting, the reason(s) why they work as well as they do are largely unknown. In this respect, the situation is even worse for hybrids. So, some reflection on desirable properties of metaheuristics, which would guarantee both their practical and theoretical interest, may be in order. A tentative list of such properties is the following:

(i) *Simplicity:* the metaheuristic should be based on a simple and clear principle, which should be largely applicable;

(ii) *Precision:* steps of the metaheuristic should be formulated in precise mathematical terms, independent from the possible physical or biological analogy which was an initial source of inspiration. Meaningless statements (e.g. "kicking the function") or overly vague ones (e.g. "choosing a point based on the history of the search") should be avoided;

(iii) *Coherence:* all steps of heuristics for particular problems should follow naturally from the metaheuristic's principle;

(iv) *Efficiency:* heuristics for particular problems should provide optimal or near-optimal solutions for all or at least most realistic instances. Preferably, they should find optimal solutions for most problems of benchmarks for which such solutions are known, when available;

(v) *Effectiveness:* heuristics for particular problems should take moderate computing time to provide optimal or near-optimal solutions;

(vi) *Robustness:* performance of heuristics should be consistent over a variety of instances, i.e., not just fine-tuned to some training set and less good elsewhere;

(vii) *User-friendliness:* heuristics should be clearly expressed, easy to understand and, most important, easy to use. This implies they should have as few parameters as possible and ideally none;

(viii) *Innovation:* preferably, the metaheuristic's principle and / or the efficiency and effectiveness of the heuristics derived from it should lead to new types of applications.

Variable Neighborhood Search (VNS) is a recent metaheuristic which strives to obtain the qualities listed above. It is based on a simple and relatively unexplored principle: systematic change of neighborhood during the search. (Note that precise rules for such change are crucial; several authors have proposed on occasion to combine different types of moves, or neighborhoods, within the same heuristic, without however doing so systematically, nor that being the main idea of their heuristics.)

Reviewing the eight desirable properties of metaheuristics, it appears that VNS possesses them to a large degree. Indeed, its principle is simple and all steps of the basic and extended schemes rely upon it. Moreover, they are state in precise mathematical terms. VNS has proved efficient in solving the problems of several benchmarks with optimal or very close to optimal results, and within moderate (or at least reasonable) computing times. Moreover, its performance appears to be robust, its basic principles are easy to apply, and very easy to use, parameters being kept to a minimum and sometimes absent. Simplicity is probably its main feature. Indeed, VNS gets as good or better results than most other metaheuristics on many problems and in a much simpler way. This explains its potential for innovation already illustrated by several new type of applications: analysis of moves and their selection for the TSP, stabilized or focussed column generation and the computer-aided scientific discovery program AGX.

Metaheuristics are a fairly young research field, with many new ideas and frequent recourse to intuition rather than deduction. Attempts at organizing this field are numerous, but as the main concepts are rarely precisely defined and there are as yet very few significant theorems, no framework has gained general acceptance. Rather, each metaheuristic has its own viewpoint and ability to explain many heuristics in its own vocabulary as well as to absorb ideas from the whole field (notably under the form of hybrids). Moreover, priority claims tend to proliferate. They are often based on such vague evidence that they are hard to evaluate.

Focusing for instance, on descent methods or memory structures corresponds to different viewpoints and metaheuristics. The closest one to VNS appears to be Iterated local search (ILS) ([91], [92], [11]). At the price of completely forgetting VNS's main idea - systematic change of neighborhood - one can squeeze it into the ILS framework. Conversely, one could view ILS heuristics as either badly defined (as the crucial step of perturbation of local optima is often not specified or described by some vague metaphor) or particular cases of VNS, with usually a single neighborhood. This would be equally arbitrary. It appears that the babelian character of research in metaheuristics is a, hopefully temporary, mild evil. While it lasts, clear-cut successes on particular problems will be probably more important to evaluate metaheuristics than lengthy controversies. Finally, when considering the eight desirable qualities listed above, we believe that comparative efficiency should not have the dominant, sometimes exclusive role, it gets in many papers. The aim of research should be insight, not competition. In our view other qualities of heuristics and metaheuristics than efficiency can be more important in the long run, particularly, simplicity, precision, coherence and above all, innovation.

References

[1] I. Ahmad and M. Dhodhi. Multiprocessor scheduling in a genetic paradigm. *Parallel Computing* 22: 395-406, 1996.

[2] M.R. Anderberg. *Cluster analysis for application*, Academic Press, New York, 1973.

[3] A. Andreatta and C. Ribeiro. Heuristics for the phylogeny problem, to appear in *Journal of Heuristics*.

[4] M.Aouchiche, G.Caporossi and P.Hansen. Variable neighborhood search for extremal graphs. 8. Variations on Graffiti 105, to appear in *Congressus Numeratium*, 2001.

[5] O.Araujo and J.A. de la Penã. Some bounds on the connectivity index of a chemical graph. *J. of Chemical Information and Computer Sci.* 38: 827-831, 1998.

[6] C. Audet, J. Brimberg, P. Hansen and N. Mladenović. Pooling problem: alternate formulation and solution methods. *Les Cahiers du GERAD G-2000-23*, 2000.

[7] Battiti R. and Protasi M. Reactive local search for the maximum clique problem, *Algorithmica* 29 (4), 610–637, 2001.

[8] Beasley, J.E. A note on solving large p-median problems, *European Journal of Operational Research* 21: 270-273, 1985.

[9] N. Belacel, N. Mladenović and P. Hansen. Fuzzy J-Means: A new heuristic for Fuzzy clustering, (to appear in *Pattern Recognition*).

[10] M. den Basten and T. Stutzle. Neighborhoods revisited: Investigation into the effectiveness of Variable neighborhood descent for scheduling, MIC'2001, Porto, pp. 545-549, 2001.

[11] M. den Basten, T. Stutzle and M. Dorigo. Design of iterated local search, EvoSTIM2001, 2^{nd} EW on Scheduling and Timetabling (to appear in *Lecture Notes CS*, 2001.

[12] R. Bent and P. Van Hentenryck. A two stage hybrid local search for the vehicle routing problem with time windows, Technical report, CS-01-06, Dept. of Computer Science, Brown University, 2001.

[13] J.C. Bezdek. *Pattern recognition with Fuzzy objective function algorithm.* Plenum, New York, 1981.

[14] O. Braysy. *Local search and variable neighborhood search algorithms for the vehicle routing with time windows*, Acta Wasaensia 87, Universitas Wasaenis, Vaasa, 2001.

[15] J. Brimberg, P. Hansen,K.-W. Lih, N. Mladenović, M. Breton. An oil pipeline design problem , *Les Cahiers du GERAD* G-2000-73, Montréal, Canada, 2000.

[16] J. Brimberg, P. Hansen, N. Mladenović and É. Taillard. Improvements and comparison of heuristics for solving the Multisource Weber problem. *Oper. Res.*, 48 (3): 444-460, 2000.

[17] J. Brimberg and N. Mladenović. A Variable neighborhood algorithm for solving the continuous location-allocation problem. *Stud. in Locat. Analysis*, 10: 1–12, 1996.

[18] E.K. Burke, De Causmaecker and G. V. Berghe. A hybrid tabu search algorithm for the nurse rostering problem. *Lecture notes in AI*, 1585, 187-194, 1999.

[19] E.K. Burke and P. Cowling and R. Keuthen. Effective local and guided Variable neighborhood search methods for the asymmetric traveling salesman problem, in the Proceedings of the Evo Workshops, Springer, Lecture Notes in Computer Science, pp. 203-212, 1999.

[20] E. Burke, P. De Causmaecker, S. Petrović and G.V.Berghe. Variable neighborhood search for nurse rostering problem, MIC'2001, Porto, pp. 755-760, 2001.

[21] P. Brucker, A. Drexl, R. Mohring, K. Neumann and E. Pesch. Resource-constrained project scheduling: notation, classification, models and methods, *European J. Oprnl. Res.*, 112: 3-41, 1999.

[22] L. M. de Campos and J.M.Puerta, Stochastic local search algorithms for linear belief networks: searching in the space of orderings, S. Benferhat and P. Besnard (Eds.): ESCQARU 2001, *Lecture Notess in AI* 2143, pp. 228-239, Springer-Verlag Berlin Heidelberg 2001.

[23] R.L. Canon, J.C. Bezdek and J. V. Dave. Efficient implementation of th fuzzy C-means clustering algorithm, *IEEE Trans. Pattern Recognition Mach. Intell.* 8 (2): 248-255, 1986.

[24] S. Canuto, M. Resende and C. Ribeiro, Local search with perturbations for the prize-collecting Steiner tree problem in graphs, to appear in *Networks*

[25] G. Caporossi, D. Cvetković, I. Gutman and P. Hansen. Variable neighborhood search for extremal graphs. 2. Finding graphs with extremal energy. *J. Chem. Inf. Comput. Sci.*, 39: 984-996, 1999.

[26] G. Caporossi, A.A. Dobrynin, I. Gutman and P. Hansen. Trees with palindromic Hosoya polynomials. *Graph Theory Notes of New-York*, 37: 10-16, 1999.

[27] G. Caporossi, I. Gutman and P. Hansen. Variable neighborhood search for extremal graphs. 4. Chemical trees with extremal connectivity index. *Computers and Chemistry*, 23: 469-477, 1999.

[28] G. Caporossi and P. Hansen. Finding relations in polynomial time. In *Proceedings of the XVI International Joint Conference on Artificial Intelligence*, 780-785, 1999.

[29] G. Caporossi and P. Hansen. Variable neighborhood search for extremal graphs. 1. The AutoGraphiX system. *Discrete Mathematics*, 212: 29-44, 2000.

[30] R. Cordone, R. W. Calvo. A Heuristic for the Vehicle routing problem with time windows, *Journal of Heuristics* 7(2): 107-129, 2001.

[31] M-C. Costa, F-R. Monclar and M. Zrikem. Variable neighborhood search for the optimization of cable layout problem, MIC'2001, Porto, pp. 749-753, 2001.

[32] T. Crainic, M. Gendreau, P. Hansen, N. Hoeb, N. Mladenović. Parallel Variable neighborhood search for the p-Median, *MIC'2001*, Porto, July 16-21, pp. 595-599, 2001.

[33] J. Crispim and J. Brandao. Reactive Tabu search and variable neighborhood descent applied to the vehicle routing problem with backhauls, MIC'2001, Porto, pp. 631-636, 2001.

[34] D. Cvetković, S. Simić, G. Caporossi and P. Hansen. Variable neighborhood search for extremal graphs. 3. On the largest eigenvalue of color-constrained trees. *Linear and Multi-linear Algebra*, 2001) (in press).

[35] T. Davidović. Scheduling heuristic for dense task graphs. *Yugoslav J. of Oper. Res.*, 10: 113–136, 2000.

[36] T. Davidović, P. Hansen and N. Mladenović. Variable neighborhood search for multiprocessor scheduling with communication delays. MIC'2001, Porto, pp. 737-741, 2001.

[37] T. Davidović, P. Hansen and N. Mladenović. Heuristic methods for multiprocessor scheduling with communication delays. (in preparation), 2001.

[38] G. Desaulniers, J. Desrosiers, Y. Dumas, S. Marc, B. Rioux and F. Soumis, F. Crew pairing at air France. *European Journal of Operational Research* 97, 245–259, 1997.

[39] J. Desrosiers, N. Mladenović and D. Villeneuve. Design of balanced MBA student teams. *MIC'2001*, Porto, July 16-21, pp. 281-285, 2001.

[40] M. Diaby, H.C. Bahl, M.H. Karwan and S. Zionts. Capacitated lot-sizing and scheduling by Lagrangian relaxation, *European J. Oprnl. Res.*, 59: 444-458, 1992.

[41] O. du Merle, D. Villeneuve, J. Desrosiers and P. Hansen. Stabilized column generation. *Disc. Math.*, 194: 229-237, 1999.

[42] J.C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, *J. Cybernet.* 3 (3): 32-57, 1974.

[43] M.L. Dukić and Z. S. Dobrosavljević. A method of a spread-spectrum radar polyphase code design. *IEEE J. on Selected areas in Comm.* 8 (5) 743-749, 1990.

[44] O. du Merle, P. Hansen, B. Jaumard and N. Mladenović. An interior point algorithm for Minimum sum-of-squares clustering. *SIAM J. Scient. Comp.* 21: 1485-1505, 2000.

[45] M. Ehrgott, J. Freitag, H. Hamacher, and F. Maffioli, Heuristics for the k-cardinality tree and subgraph problems. *Asia-Pacific J. of Op. Res.*, 14: 87-114, 1997.

[46] S. Fajtlowicz. On conjectures of Graffiti. *Discrete Mathematics*, 72: 113-118, 1988.

[47] S. Fajtlowicz. Written on the wall. Version 09-2000 (regularly updated file accessible via e-mail from clarson@math.uh.edu), 2000.

[48] T. Feo and M. Resende. Greedy randomized adaptive search. *J. Global Optim.* 6: 109–133, 1995.

[49] P. Festa, P. Pardalos, M. Resende and C. Ribeiro, GRASP and VNS for Max-cut, Proceedings of MIC'2001, pp. 371-376, 2001.

[50] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Ann. Eugenics* VII part II: 179–188, 1936.

[51] M. Fischetti, H. Hamacher, K. Jornsten and F. Maffioli. Weighted k-cardinality trees: complexity and polyhedral structure. *Networks*, 24:11-21, 1994.

[52] K. Fleszar and K.S. Hindi. New heuristics for one-dimensional bin-packing, to appear in *Comp. & Oprln. Res.*, 2001.

[53] K. Fleszar and K.S. Hindi. Solving the resource-constrained project scheduling problem by a variable neighborhood search, *Europ. J. Oprnl. Res.* (accepted s.t. revision), 2001.

[54] P.Fowler, P.Hansen, G.Caporossi and A.Sondini. Polyenes with maximum HOMO-LUMO gap. (Variable Neighborhood Search for extremal graphs 7.). *Les Cahiers du GERAD*, Montréal, Canada, 2001, to appear in *Chemical Physics Letters*, 2001.

[55] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, New-York, 1978.

[56] L.M. Gambardela, E. Taillard and M. Dorigo. Ant colonies for the quadratic assignment problem, *J. Oprnl. Res. Soc.*, 50, 167-176, 1999.

[57] M. Gendreau, G. Pesant and L.-M. Rousseau. Using constraint based operators with variable neighborhood search to solve the vehicle routing problem with time windows, to appear in *J. of Heuristics.*

[58] M.Gendreau, P.Hansen, M.Labbé and N.Mladenović. Variable neighborhood search for the traveling salesman problem, (in preparation), 2001.

[59] M. Gendreau, A. Hertz and G. Laporte. New Insertion and postoptimization procedures for the Traveling salesman problem. *Oprns. Res.* 40: 1086-1094, 1992.

[60] M. Gendreau, A. Hertz and G. Laporte, The traveling salesman problem with back-hauls, *Computers Oper. Res.* 23: 501–508, 1996.

[61] G. Ghiani, A. Hertz and G. Laporte. Recent algorithmic advances for arc routing problems. *Les Cahiers du GERAD G-2000-40*, Montréal, Canada, 2000.

[62] F. Glover. Tabu search - Part I, *ORSA J. Comput.* 1, 190–206, 1989.

[63] F. Glover. Tabu search - Part II, *ORSA J. Comput.*, 2, 4–32, 1990.

[64] F. Glover and M. Laguna. *Tabu search.* Kluwer, Boston, 1997.

[65] C.G. Gonzales and D.P. Brito, A Variable neighborhood search for solving the linear ordering problem, MIC'2001, Porto, pp. 181-185, 2001.

[66] R.E. Griffith & R.A. Stewart. A nonlinear programming technique for the optimization of continuous processing systems, *Management Science*, 7, 379–392, 1961.

[67] I. Gutman. A Contribution to the study of palindromic graphs. *Graph Theory Notes of New York*, XXIV:6, New York Academy of Science, 51-56, 1993.

[68] I. Gutman and O.E.Polansky. *Mathematical concepts in organic chemistry*, Berlin: Springer, 1986.

[69] I. Gutman, E. Estrada and O. Ivanciuc. Some properties of the Wiener polynomial of trees. *Graph Theory Notes of New York*, XXXVI:1, New York Academy of Sciences, 7-13, 1999.

[70] I. Gutman and O. Miljković. Molecules with smallest connectivity index. *Match*, 41: 57-70, 2000.

[71] I. Gutman, O. Miljković, G. Caporossi and P. Hansen. Alkanes with small and large connectivity index. *Chemical Physics Letters*, 306: 366-372, 1999.

[72] P. Hansen and B. Jaumard. Algorithms for the maximum satisfiability problem. *Computing* 44: 279–303, 1990.

[73] P. Hansen and B. Jaumard. Cluster analysis and mathematical programming, *Mathematical Programming*, 79: 191-215, 1997.

[74] P. Hansen, B. Jaumard, N. Mladenović and A. Parreira. Variable neighborhood search for Weighted maximum satisfiability problem, *Les Cahiers du GERAD* G-2000-62, Montréal, Canada, 2000.

[75] P. Hansen, S. Krau and O. du Merle. Stabilized column generation algotithm for the multisource Weber problem (in preparation).

[76] P. Hansen and H. Melot. Variable neighborhood search for extremal graphs. 6. Analyzing bounds on the connectivity index, *Les Cahiers du GERAD* (forthcoming).

[77] P. Hansen and N. Mladenović. Variable neighborhood search for the p-Median, *Location Sci.*, 5: 207-226, 1997.

[78] P. Hansen and N. Mladenović. An introduction to variable neighborhood search. in S. Voss *et al.* (eds.), *Metaheuristics, Advances and Trends in Local Search Paradigms for Optimization*, pp. 433-458, Kluwer, Dordrecht, 1999.

[79] P. Hansen and N. Mladenović. First improvement may be better than best improvement: An empirical study, *Les Cahiers du GERAD* G-99-40, Montréal, Canada, 1999.

[80] P. Hansen and N. Mladenović. Variable neighborhood search: Principles and applications. *European J. of Oper. Res.*, 130: 449-467, 2001.

[81] P. Hansen and N. Mladenović. J-Means: A new local search heuristic for minimum sum-of-squares clustering, *Pattern Recognition*, 34: 405-413, 2001.

[82] P. Hansen and N. Mladenović. Industrial applications of the variable neighborhood search metaheuristics, G. Zaccour (eds.), *Decisions & control in management science*, pp. 261-274, Kluwer Academic Publishers, Boston/Dordrecht/London, 2001.

[83] P. Hansen and N. Mladenović. Developments of variable neighborhood search, C. Ribeiro, P. Hansen (eds.), *Essays and surveys in metaheuristics*, pp. 415-440, Kluwer Academic Publishers, Boston/Dordrecht/London, 2001.

[84] P. Hansen, N. Mladenović and D. Perez-Brito. Variable neighborhood decomposition search. *J. of Heuristics*, 7 (4): 335-350, 2001.

[85] P. Hansen, N. Mladenović and D. Urošević. Variable neighborhood search for the Maximum clique. *Les Cahiers du GERAD* G-2001-25, Montréal, Canada, 2001.

[86] A. Hertz, M-C. Costa and M. Mitaz. Bounds and heuristics for the shortest capacitated path problem, MIC'99, Rio de Janeiro, 1999.

[87] A. Hertz G. Laporte and M. Mittaz. A Tabu search heuristic for the Capacitated arc routing problem. *Operations research*, 48: 129-135, 2000.

[88] K.S. Hindi, K. Fleszar and C. Charalambous. An effective heuristic for the CLSP with setup times, *Europ. J. Oprnl. Res.* (accepted s.t. revision), 2001.

[89] C-T. Hsieh, C-C Chin and K-M Shen. Generalized fuzzy Kohonen clustering networks, *IEICE Trans. Fundamentals*, V. E81-A (10), 1998.

[90] F.K. Hwang, D.S. Richards and P. Winter. *The Steiner tree problem*, North-Holland, Amsterdam, 1992.

[91] T. Ibaraki, M. Kubo, T. Masuda, T. Uno and M. Yagiura. Effective local search algorithms for the vehicle routing problem with general time windows, MIC'2001, Porto, 293-297, 2001.

[92] S. Imahori, M. Yagiura and T. Ibaraki. Local search heuristics for the rectangle packing problem with general spatial costs, MIC'2001, Porto, 471-476, 2001.

[93] R.C. Jancey. Multidimensional group analysis. *Australian J. Botany*, 14, 127–130, 1966.

[94] Johnson D. and Trick M. (eds.), Cliques, coloring and satisfiability: Second DIMACS implementation challenge, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 26, 1996.

[95] , J. E. Kelley. The cutting-plane method for solving convex programs. *Journal of the SIAM*, 8, 703–712, 1960.

[96] R. Kolish and S. Hartman. Heuristic algorithms for solving the resource-constrained project scheduling problem: classification and computational analysis, *Project scheduling: recent models, algorithms and applications*, Kluwer, 1999.

[97] V. Kovačević, M. Čangalović, M. Ašić, D. Dražić and L. Ivanović. Tabu search methodology in global optimization. *Computers & Math. with Appl.* 37: 125-133, 1999.

[98] S. Krau, *Extensions du problems de Weber*, Ph D. Thèse, École Polytechnique de Montréal, 1997.

[99] M. Krishnamoorthy, A.T. Ernst and Y.M. Sharaiha. Comparison of algorithms for the degree constrained spanning trees, Proceedings of International Conference on Optimisation Techniques and Applications (Perth, Western Australia, 1998), L. Caccetta, *et al.* (Eds.), Curtin University of Technology, pp. 859-866, 1998.

[100] Y.-K. Kwok and I. Ahmad. Efficient scheduling of arbitrary task graphs to multiprocessors using a parallel genetic algorithm. *J. Parallel and Distributed Computing*, 47: 58-77, 1997.

[101] M. Labbé, G. Laporte, I. Rodriques and J. Salazar. Median cycle problems. SMG Report, Université Libre de Bruxelles, Belgium, 1999. (http://smg.ulb.ac.be)

[102] M. Laguna, R. Martin and V. Campos. Intensification and diversification with elite tabu search solutions for the linear ordering problem, *Comp. & Oprln. Res.* 26, 1217-1230, 1999.

[103] S.Lin and B.W.Kernighan. An effective heuristic for the traveling salesman problem. *Oper. Res.* 21: 498-516, 1973.

[104] K. Jörnsten and A. Lokketangen. Tabu search for weighted k-cardinality trees. *Asia-Pacific J. of Op. Res.*, 14 (2): 9-26, 1997.

[105] L. Lobjois, M. Lemaitre and G. Verfaille. Large neighborhood search using constraint propagation and greedy reconstruction for valued CSP resolution. Research report ONERA, Toulouse, France, 2001.

[106] F.G.Lopez, B.M.Batista, J.A.Moreno Pérez and J.M.Moreno Vega. The parallel variable neighborhood search for the p-median problem. Research Report, University of La Laguna, Spain, 2000, (to appear in *J. of Heuristics*).

[107] S. Loudin and P. Boizumault. VNS/LDS + CP: A hybrid method for constraint optimization in anytime contexts, MIC'2001, Porto, pp. 761-765, 2001.

[108] 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, *J. of Global Optimization*, 17: 267-283, 2000.

[109] M. Mittaz. Problèmes de cheminements optimaux dans des réseaux avec contraintes associées aux arcs. Ph.D. Thesis, Department of Mathematics, École Polytechnique Fédérale de Lausanne, Switzerland.

[110] N. Mladenović. A Variable neighborhood algorithm - a new metaheuristic for combinatorial optimization. Abstracts of papers presented at Optimization Days, Montréal, p. 112, 1995.

[111] N. Mladenović, M. Labbé and P. Hansen. Solving the p-center problem by Tabu search and Variable neighborhood search, (to appear in *Networks*), 2001.

[112] N. Mladenović and P. Hansen. Variable neighborhood search. *Computers Oper. Res.* 24: 1097–1100, 1997.

[113] N. Mladenović, J.P. Moreno, and J. Moreno-Vega. A Chain-interchange heuristic method, *Yugoslav J. Oper. Res.* 6: 41–54, 1996.

[114] N. Mladenović, J. Petrović, V. Kovačević-Vujčić and M. Čangalović. Solving Spread spectrum radar polyphase code design problem by Tabu search and Variable neighborhood search. SMG Report, R-00-09, Universite libre Bruxelles, Belgium, 2000, (to appear in *European J. of Oper. Res.*).

[115] N. Mladenović and D. Urošević. Variable neighborhood search for the k-cardinality tree, MIC'2001, Porto, pp. 749-753, 2001.

[116] L.S. Ochi, M.B. Silva and L. Drummond. Metaheuristics based on GRASP and VNS for solving Traveling purchaser problem, MIC'2001, 489-494, Porto, 2001.

[117] J. Pearl, *Probabilistic reasoning in intelligent systems*, Morgan and Kaufman, 1998.

[118] G. Pesant and M. Gendreau. A View of local search in Constraint programming, principles and practice of Constraint Programming, *Lecture Notes in Computer Science*, 1118: 353-366, Springer-Verlag, 1996.

[119] G. Pesant and M. Gendreau. A Constraint programming framework for Local search methods, *J. of Heuristics* 5: 255-279, 1999.

[120] M.Randić. On characterization of molecular branching. *J. of the American Chemical Society* 97: 6609-6615, 1975.

[121] R. Ravi and F. S. Salman. Approximation algorithms for the traveling purchaser problem and its variants in network design, *Lecture Notes in Computer Science*, 1643: 29–40, Springer, 1999.

[122] G. Reinelt. TSLIB - A Traveling salesman library. *ORSA J. Comput.* 3: 376–384, 1991.

[123] M.G.C. Resende, L.S. Pitsoulis, and P.M. Pardalos. Approximate solution of Weighted max-sat problems using GRASP, In *Satisfiability Problem: Theory and Applications*, Dingzhu Du *et al.* (Eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science 35, American Mathematical Society, Providence, Rhode Island, 1997.

[124] C. Ribeiro and C.Souza. Variable neighborhood descent for the degree-constrained minimum spanning tree problem, to appear in *Discrete Applied Mathematics*, 2001.

[125] C. Ribeiro, E. Uchoa and R. Werneck. A hybrid GRASP with perturbations for the Steiner problem in graphs. Technical report, Computer Science Department, Catholic University of Rio de Janeiro, 2001.

[126] I. Rodriguez, M. Moreno-Vega and J. Moreno-Perez. Heuristics for Routing-median problems. SMG Report, Université Libre de Bruxelles, Belgium, 1999.

[127] A. Scholl, R. Klein and C. Jurgens. BISON: A fast hybrid procedure for exactly solving the one-dimensional bin packing problem, *Comp. & Oprln. Res.*, 24: 627-645, 1997.

[128] P.Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *Principles and practice of constraint programming* (CP'98), 417-431, 1998.

[129] M.B. Silva, L. Drummond and L.S. Ochi. Variable neighborhood search for the Traveling purchacer problem, 27^{th} Int. Conf. on Comp. Indust. Engineering, 2000.

[130] E. Taillard and L. Gambardella. Adaptive memories for the quadratic assignment problem. *Technical report* I-87-97, IDSIA, Lugano, Switzerland, 1999.

[131] E. Taillard and S. Voss. POPMUSIC - Partial optimization metaheuristic under special intensification conditions, C. Ribeiro, P. Hansen (eds.), *Essays and surveys in metaheuristics*, pp. 613-630, Kluwer Academic Publishers, Boston/Dordrecht/London, 2001.

[132] W. Trigeiro, J. Thomas and J. McClain. Capacitated lot-sizing with setup times, *Mangm. Sci.*, 35: 353-366, 1989.

[133] J. D. Ullman. NP-complete scheduling problems. *J. Comput. Syst. Sci.* 10 (3), 384–393, 1975.

[134] M.G.A. Verhoeven, M.E.M. Severens and E.H.L. Aarts. Local search for Steiner trees in graphs. V.J. Rayward-Smith *et al.* (Eds.), *Modern heuristic search methods*, pp. 117-129, John Wiley and sons, 1996.

[135] S. Voss. Dynamic Tabu search strategies for the traveling purchaser problem, *Ann. Oper. Res.* 63: 253-275, 1996.

[136] R. Whittaker. A fast algorithm for the greedy interchange for large-scale clustering and median location problems. *INFOR* 21: 95-108, 1983.

[137] N. Zufferey, C. Avanthay and A. Hertz. Variable neighborhood search for graph colouring (submitted).

Index