Variable Neighborhood Search: The Power of Change and Simplicity

Jack Brimberg[a], Said Salhi[b], Raca Todosijević[c,d], Dragan Urošević[e]

[a]*Department of Mathematics and Computer Science, Royal Military College of Canada, Kingston, Ontario, Canada*
[b]*Centre for Logistics & Heuristic Optimisation (CLHO), Kent Business School, University of Kent, Canterbury, UK*
[c]*Univ. Polytechnique Hauts-de-France, LAMIH, CNRS, UMR 8201, F-59313 Valenciennes, France*
[d]*INSA Hauts-de-France, F-59313 Valenciennes, France*
[e]*Mathematical Institute of the Serbian Academy of Sciences and Arts, Knez Mihajlova 36, Belgrade, Serbia*

Abstract

This review discusses and analyses three main contributions championed by Professor Mladenovic. These include variable neighbourhood search (VNS), variable formulation space (VFS), and finally, the less-is-more approach (LIMA). These three methodologies share two important ingredients, namely, simplicity and change. Given that Professor Mladenović is widely known for VNS, we will focus mainly on this methodology and its developments as well as a sampling of successful applications. This is followed by a shorter discussion covering VFS and LIMA. We introduce several remarks throughout the text to highlight certain aspects while also suggesting challenging research areas to examine.
Keywords: Metaheuristic; Variable neighborhood search; Less-Is-More; Variable Formulation Space

1. Introduction

Optimization problems arise in many real-life settings such as technology, the economy, science, industry, the environment, telecommunications, energy, transportation, et cetera. In addition, these problems arise in both private and public sectors, and may interest a few individuals or even huge enterprises. Although quite different in content from area to area, once written in mathematical terms they become very similar. More precisely, once written as mathematical models (or programs), they all have one or more objective functions to optimize, a solution space that identifies the decision variables that apply, and a set of constraints that limit the feasible choices allowed to the decision-maker to some subset of the solution space. Often, there is a huge number of possible solutions, and the question to answer is how to find an optimal one, or alternatively, just a "good" one. Usually, the given problem will have only one or just a few optimal solutions mysteriously buried somewhere within the feasible set, and surrounded by a large number of locally optimal solutions (also referred to as traps). The most tractable models have a special structure that guarantees any local optimum is also a global one. This is true, for example, with linear programs that often arise in practice. However, many practical problems do not have this special structure, and consequently, exact (or optimal) solutions cannot be found in reasonable time, even with the most powerful computers. More precisely, the mathematical models cannot be solved in polynomial time, i.e., they are NP hard.

*Corresponding author
Email addresses: Jack.Brimberg@rmc.ca (Jack Brimberg), S.Salhi@kent.ac.uk (Said Salhi), racatodosijevic@gmail.com (Raca Todosijević), draganu@mi.sanu.ac.rs (Dragan Urošević)

Thus, many classes of important problems can be solved exactly today only by performing a total enumeration, implicit or otherwise, of the feasible set. However, this becomes an impossible task in general, except for very small instances. As a result, and by necessity, another research direction has gained relevance, namely, the development of non-optimal techniques that provide approximate solutions, with or without estimates of how far they are from the optimal objective value. These so-called heuristic methods have become the most useful techniques of the day for tackling hard real-life optimization problems.

It is also worth stressing that though optimality cannot be always guaranteed, useful performance measures are usually adopted. These may include worst-case analysis, empirical testing, benchmarking and/or comparison against lower or upper bounds if these happen to be found. In brief, heuristics, which originated in ancient Greece, and which are based on the 'find and discover' principle, can be considered a combination of several attributes, namely, mathematical logic, computing, insight into the problem, and the design of selection rules with simplicity and ease of understanding in mind. For more information on heuristic search in general see Salhi (2017), while an overview on heuristics can be found in Salhi and Thompson (2022b).

In essence, these approximate methods execute a search strategy within the solution space aimed at finding a solution of good quality while following some rules that are usually problem independent. This feature allows the heuristics to be applicable to a large number of optimization problems arising in completely different areas. Another desirable feature of heuristics is their simplicity. The best heuristics are often the easiest to understand and to adapt to specific problems.

In this paper, we review some heuristic frameworks that follow the paradigms of "change" and "simplicity". The idea of "change" is not something new in the scientific literature, and even dates back to the age of Heraclitus of Ephesus, who proposed the idea that the only constant in life is the notion of change. The idea of change was also an inspiration to Professor Nenad Mladenović, who continuously sought ways to develop and improve the science and art of heuristic design. Among the many contributions he has made with collaborators over the years in the field of optimization, he will be best remembered for introducing and developing a general framework for heuristic design known as Variable Neighborhood Search (VNS). As the VNS methodology progressed and its list of accomplishments grew, the tendency among researchers in the field moved towards more complex and less user-friendly algorithms, for example, hybrid heuristics with multiple stages and numerous parameters to set. To urge the return to simplicity, Professor Nenad Mladenović with his collaborators proposed a *less-is-more methodology* (LIMA) to simplify as much as possible the design of heuristics that will be competitive with state-of-the-art methods, and ultimately, even improve on the quality of solutions.

As a tribute to Nenad, our dear colleague and friend, and in recognition of his outstanding contributions to the optimization literature, we attempt to review in this paper the progression of Variable Neighborhood Search from its initial introduction in 1995 to the present day. The aim is to provide a general overview or taste, and not an exhaustive list of papers published during this time on VNS-related topics. In addition, we believe that the current status of VNS, and related ideas pioneered by Nenad such as Variable Formulation Search and the *less-is-more* concept, are just "the tip of the iceberg", and many possibilities remain under the surface for future research. There is no doubt that Nenad would like to see energetic efforts in this direction. These yet undiscovered advances will confirm again that everything is subject to "change". In line with this, we will provide some critical discussion and suggestions for future research in the form of *"Remarks"* at the end of each section reserved for some methodology or selected sub-sections as appropriate.

The paper is organized as follows. The next section covers basic VNS including preliminaries and necessary ingredients, followed by section 3 on important modifications to the basic procedure. Section 4 presents more recent advances in VNS including its adaptation to nonlinear programming and hybridisation.

This is followed by Section 5 that discusses related VNS philosophies, VFS and LIMA. Section 6 provides some concluding remarks.

2. Basic variable neighborhood search (BVNS)

2.1. A little history:

Professor Nenad Mladenović is best known in the Operations Research (OR) field as the founder of the Variable Neighborhood Search (VNS) metaheuristic. He first introduced the idea at the *Optimization Days* 1995 conference held at the University of Montreal. The title of his presentation was *Variable Neighborhood Algorithm – A New Meta-Heuristic for Combinatorial Optimization*. This talk was prefaced in the conference proceedings by a short abstract that probably went unnoticed by most of the participants. The key sentence in the abstract goes as follows: *"Escaping from the local optima trap is done by changing the neighbourhood (metric) in a systematic way."* At that time, the conventional approach used when an improvement algorithm converged to a local optimum was to re-start the search from a new point randomly selected from the feasible set. In fact, random multi-start improvement algorithms (or local searches) are still popular today. Thus, the principal contribution of VNS may be viewed as the new approach it offers for evading the *local optimum trap*. That is, a series of random starting solutions is now replaced by a systematic selection of random points from pre-defined neighborhood structures centered at the current incumbent (or best found) solution. In this way, several attributes of the current anchor point can be retained, instead of starting anew from a completely random point. The fact that the VNS concept was slow to gain momentum perhaps was due to the simplicity of the concept. Even so, should not "simple to understand" be one of the main attractions of a good heuristic?

The VNS methodology was first formalized in a general framework in Mladenović and Hansen (1997). The first application had appeared a year earlier (Brimberg and Mladenovic (1996)). An improved shaking (or perturbation) procedure was tested on the same problem class in Brimberg et al. (2000). The computational results in this paper were outstanding! Several new best-known solutions were found on benchmark instances, and the improvements were quite significant. This type of performance became routine as the number of publications using VNS began to grow.

2.2. Some preliminaries

The general form of optimization problem that we are looking at may be given as follows:

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

where S, X, x and f denote, respectively, the solution space, the feasible set, a feasible solution, and a real-valued objective function. Depending on the set S, we distinguish the combinatorial optimization problem where set S is finite but extremely large, and the continuous optimization problem ($S = R^n$). Expressed as a mathematical program, X would be defined by a set of main constraints, and S by variable type constraints (e.g., non-negativity and integer constraints). Let $N(x)$ denote a neighborhood structure centered at a given solution $x \in X$. This neighborhood would be constructed using some distance function δ, resulting in the following definition:

$$N(x) = \{y \in X|\delta(x, y) \leq \Delta\}, \tag{2}$$

where Δ is a positive number (integer for combinatorial optimization problems). For example, when feasible solutions are arrays containing n attribute values, we could specify $\delta(x, y) = |x \setminus y|$ and $\Delta \in \{1, \ldots, n\}$. A solution $x^* \in X$ is a local minimum relative to neighborhood $N(x^*)$ for problem (1) if

$f(x^*) \leq f(x), \forall x \in N(x^*)$, while $x^* \in X$ is an optimal solution (global minimum) for problem (1) if $f(x^*) \leq f(x), \forall x \in X$.

Usually it is relatively easy to find a local minimum, but it becomes nearly to impossible to find a global one. Furthermore, there are inevitably a huge number of local minima induced by the objective function and the constraints, which act as traps, so that finding a "good" local minimum is also a daunting task. Clearly, the ability to escape local optima traps on the way to "better" solutions is an important ingredient of any selected heuristic. Based on a wealth of experimental results over the past two decades, we can say that the VNS framework is very good in this regard, and this would explain why VNS-based heuristics are often able to reach new best-known solutions that are significantly better than the state-of-the art.

Remark 1. *Empirical evidence available (Kirkpatrick and Toulouse (1985)) suggests that most local minima are relatively close to each other. In view of this, the systematic approach of exploring neighborhoods starting from the closest, and moving progressively further away, may give VNS a statistical edge over other metaheuristic-based methods. This would be an important but complex area for future empirical and theoretical studies.*

2.3. VNS ingredients

The following three steps are repeated in sequence in each iteration of a VNS-based heuristic until a stopping condition, such as a limit on execution time, is reached:

(1) Shaking procedure

(2) Improvement procedure

(3) Neighborhood change

An initial feasible solution is required to start the process, and this starting point is typically selected at random from the feasible set. An improvement algorithm is then used to find a local minimum associated with the local search being employed. At this point, the VNS iterations begin, each consisting of the three steps in sequence given above. We outline these steps next with respect to the basic VNS (BVNS) framework.

Shaking procedure:

Let $\mathcal{N} = \{\mathcal{N}_1, \ldots, \mathcal{N}_{k_{max}}\}$ be a set of operators such that each operator $\mathcal{N}_k, 1 \leq k \leq k_{max}$, maps a given feasible solution x to a predefined neighborhood structure,

$$\mathcal{N}_k(x) = \{y \in X | \delta(x, y) \leq \Delta_k\}, k = 1, \ldots, k_{max}. \tag{3}$$

Note that the order of operators in set $\mathcal{N}$ also defines the order of examining the various neighborhood structures of a given solution x as specified in (3). Furthermore, the neighborhoods are purposely arranged in increasing distance from the incumbent solution x; that is,

$$\Delta_1 < \Delta_2 < \cdots < \Delta_{k_{max}}. \tag{4}$$

As an example, suppose we are considering the p-median problem on a network, and solution x contains the p (a parameter) nodes selected at which to locate new facilities. There are $C(n, p)$ possible solutions in feasible set X, where n is the number of nodes on the network. The objective is to find a solution that minimizes a weighted sum of distances from the n nodes to their closest facilities. This problem is NP-hard

(Kariv and Hakimi (1979)). The neighborhood $\mathcal{N}_1(x)$ contains all solutions y obtained by replacing one node in x with a node not in $x(\{1, 2, \ldots, n\} \setminus x)$. Similarly, $\mathcal{N}_k(x)$ contains all solutions y obtained by replacing k nodes in x with k nodes not in x. The parameter k_{max} is normally set to a maximum value that makes sense. In the p-median example, we would set $k_{max} = p$, thus eliminating one of the parameters. The neighborhood $N_{k_{max}}(x)$ would essentially be equivalent to selection of a completely random point in multi-start local search (MLS); i.e., setting $\mathcal{N} = \{\mathcal{N}_{k_{max}}\}$ would replace the VNS heuristic with a random MLS heuristic.

Also note that in the original implementations of VNS, b (a parameter ≥ 1) points would be selected from neighborhood $\mathcal{N}_k(x)$, but b is now normally set to 1, thereby simplifying the VNS heuristic further. The only remaining parameter to specify in a basic VNS would be the stopping criterion for the heuristic, which is normally set to a time limit, t_{max}. The simple shaking procedure consists in selecting a random solution from the kth neighborhood structure, $\mathcal{N}_k(x)$ (see Algorithm 1).

Algorithm 1: Shaking procedure

Function `Shake` $(x, k, \mathcal{N})$;
choose $x' \in \mathcal{N}_k(x)$ at random;
return x'.

Remark 2. *Further empirical studies on the finite time performance of VNS would be interesting. Here we would investigate the relative frequency of success (i.e., finding an improved solution that becomes the new incumbent) as a function of the shake neighborhood $\mathcal{N}_k(x)$. We know that a rapid improvement occurs initially, that is, there are several jumps to improved solutions, but the rate of improvement slows down and eventually stalls. These improvements likely occur from closer neighborhoods $\mathcal{N}_k(x)$. Some ideas come to mind to improve the finite time performance of VNS:*

- *Use of a sliding shake step where the range of parameter k shifts, e.g., Phase 1: $k = 1, \ldots, k_1$; Phase 2:$k = k_1 + 1, \ldots, k_2$, and so on. This could be referred to as sliding VNS.*
- *Setting the parameter $b = 1$ may improve the user-friendliness of the procedure, but we may also be missing a useful ingredient here. Another version of sliding VNS would introduce parameter values b_k for each neighborhood $\mathcal{N}_k(x)$. Parameter b_k could be set to larger values for neighborhoods that are statistically observed to be more successful, and shifting values of the b_k's might also be employed based on statistical evidence.*
- *Random selection of neighborhood $\mathcal{N}_k(x)$ based on empirically derived distributions has been suggested in the literature (Dražić et al. (2008)), but limited results are available.*

Remark 3. *There are several works proposing smart shaking instead of a completely random jump. For example, some authors consider adaptive shaking procedures (Abdelkhalek et al., 2015; Karakostas et al., 2020; Stenger et al., 2013); Pacheco et al. (2018) introduce memory in the shaking procedure; Jomaa et al. (2021) consider a learning effect process based on a probabilistic model to enhance the shaking; Thevenin and Zufferey (2019) introduce the notion of move attractiveness and use a trail system from where VNS learns which attributes are often present together in good solutions in order to generate a new solution. This type of learning is also successfully used in other techniques as a way to solve large problems by focussing on the promising variables only (also known as reduction techniques (Guastaroba and Speranza, 2012; Luis et al., 2011; Rosing and ReVelle, 1997; Wilbaut et al., 2009)). Recently, Cao et al. (2022) proposed an*

interesting memory based restart, which upon reaching the k_{max} limit, constructs a new solution based on the search history. Adaptive or 'smart' shaking should remain a research area of continuing interest.

Improvement step:

An improving search typically involves an examination of alternate solutions that are "near" to the current solution; i.e., the improving search uses local information obtained from a single neighborhood of surrounding solutions. This is precisely what the improvement step does in BVNS. Let $N_{loc}(x')$ denote the single neighborhood used in this step. Typically, in BVNS, we would conveniently set

$$N_{loc}(x') = \mathcal{N}_1(x'), \tag{5}$$

the first (or closest) neighborhood used in the shaking step, but this does not have to be the case. Alternatively, one could select

$$N_{loc}(x') = \mathcal{N}_1(x') \cup \mathcal{N}_2(x'), \tag{6}$$

for a more intensified search at the current solution (but requiring much more computing time), or a different neighborhood all together than the ones from the shaking step. Returning to the p-median example and using (5), the local search becomes the well-known and popular vertex swap algorithm of Teitz and Bart (1968). The local search typically uses either a best improvement strategy, which examines $N_{loc}(x')$ completely and replaces x' with the best solution found in $N_{loc}(x')$ provided it is better than x', or a first improvement strategy, which replaces x' with the first solution found in $N_{loc}(x')$ that is better than x'. The local search ends when no solution better than the updated current solution x' can be found in $N_{loc}(x')$.

Remark 4. *The first improvement strategy is often chosen in local search because it is faster than best improvement, and often leads to better local optima from random starting solutions. However, empirical results (Hansen and Mladenović (2006)) suggest that the reverse may be true when the starting solution used in the local search is of good quality. This suggests that a switch from first improvement to best could improve the performance of VNS when further improvement of the incumbent solution slows down drastically, implying that the quality of starting solutions from the shaking operation may also be relatively high.*

Neighborhood change step:

An iteration of VNS starts with the shake operation, which moves the incumbent solution x to a perturbed solution $x' \in \mathcal{N}_k(x)$. Then the improvement or local search step moves x' to a local minimum x''; that is,

$$f(x'') \leq \min\{f(y) : y \in N_{loc}(x'')\}. \tag{7}$$

At this point, the neighborhood change step takes over to decide how the search will continue in the next iteration of VNS. The routine commonly used is known as the *sequential neighborhood change step*. If $f(x'') < f(x)$ (the new solution is better than the incumbent), then $x \leftarrow x''$ (the new solution becomes the incumbent), and $k \leftarrow 1$ (the shake parameter is reset to 1); otherwise $(f(x'') \geq f(x))$, $k \leftarrow k + 1$ (the shake operation uses the next neighborhood in the sequence defined by $\mathcal{N} = \{\mathcal{N}_1, ..., \mathcal{N}_{k_{max}}\}$). Other forms of the neighborhood change step have been attempted in the literature. These include cyclic, pipe, random and skewed forms. Additional details may be found in the extensive survey paper by Hansen et al. (2017). However, the sequential procedure remains the most often used form by far.

Remark 5. *The different neighborhood change routines obviously lead to different descent paths. Empirical studies may provide interesting insights on this matter. For example, would a random selection of the routine to use in each iteration lead to better results?*

Joining the steps to create BVNS:

We can now join the steps outlined above to obtain the basic form of the VNS algorithm (see Algorithm 2).

Algorithm 2: Basic Variable Neighborhood Search.

Function VNS(x, k_{max}, $\mathcal{N}$, N)
repeat
  $k \leftarrow 1$;
  while $k \leq k_{max}$ **do**
    $x' \leftarrow$ Shake$(x, k, \mathcal{N})$;
    $x'' \leftarrow$ Local_Search(x', N);
    Neighborhood_change_sequential(x, x'', k);
until *stopping condition is fulfilled;*
return x.

Remark 6. *The superiority of VNS over multi-start local search (MLS) has been demonstrated several times in the literature. Brimberg et al. (2010) , explain the reason behind this phenomenon by assuming that local optima are relatively close to each other, or we could say, the majority of local optima are contained in a small subset of X. As a result, the probability that a small shake to a point relatively close to the incumbent solution will lead to a better local optimum is orders of magnitude higher than the probability associated with the large shake in MLS. This assumption is at the very core of VNS, and top priority should be given to prove its validity. Understanding why VNS always outperforms MLS is also fundamental to achieving a deeper understanding of the topology (or landscape) of the objective function over the feasible set X. In other words, VNS can serve as a useful exploratory tool for studying the structure of the problem under investigation. Keeping track in computational experiments of the distances between consecutive local optima, and the shake neighborhoods leading to better solutions, would be useful in this regard.*

Remark 7. *Here is another thought. The conventional approach used to verify if one algorithm is better than another is to test the two algorithms under similar conditions (e.g. same starting solutions and limit on execution time) on a series of benchmark instances. One of the algorithms would be declared the winner if the evidence (based on objective function values) was overwhelming in its favor. The proposal thrown out to the reader would be to replace this procedure by a standardized test where a new algorithm is compared to the same "base" algorithm (a standard MLS, for example) as all other competing algorithms. This would simplify test procedures and also eliminate possible ambiguities in the testing procedures.*

3. Modifications of Basic Variable Neighborhood Search

The variants below are obtained directly from BVNS by modifying (or omitting) one of the three steps of BVNS. Fixed neighborhood search essentially omits the neighborhood change step while Skewed variable neighborhood search modifies it by adding a new criterion for accepting changes to the incumbent solution. Reduced variable neighborhood search omits the improvement step (local search), while Variable neighborhood decomposition and General variable neighborhood searches modify it.

3.1. Fixed Neighborhood Search (FNS)

A simple modification of BVNS yields fixed neighborhood search (FNS), also known as iterated local search (Johnson and McGeoch (1997)). The switchover to FNS is accomplished by restricting the shake

operation in algorithm 2 to the single neighborhood,

$$\mathcal{N}(x) = \mathcal{N}_s(x), \tag{8}$$

where the same $s \in \{1, \ldots, k_{max}\}$ is used in each iteration. Thus, FNS is in between MLS, where a new starting point is selected at random from X (an extreme shake), and VNS where several (k_{max}) shake neighborhoods are in use. Moreover, $\mathcal{N}(x)$ should be moderately close to the incumbent solution so that the perturbed solution x' always retains many of the incumbent's favorable attributes.

Remark 8. *The selection of the shake neighborhood $\mathcal{N}(x)$ affects the level of diversity of FNS. As an example, consider the p-median problem. Setting the shake neighborhood $\mathcal{N}(x) = \mathcal{N}_1(x)$ (single vertex swap), and then using the same neighborhood for local search in the improvement step, would result in a probabilistic form of 2-swap local search. Setting $\mathcal{N}(x) = \mathcal{N}_2(x)$, and using 1-swap again for local search, gives a probabilistic 3-swap neighborhood search. Using $\mathcal{N}(x) = \mathcal{N}_{k_{max}}(x)$ converts the FNS to something near or equivalent to MLS and maximum diversity. It would certainly be interesting to compare the performance of FNS for different levels of shaking. This may also provide insights into the underlying problem structure. For example, if best results are obtained by applying a small shake, we might conclude that the objective function has a globally convex shape with 'noise' superimposed on it. A larger shake, meanwhile, might indicate that there are several 'nice' valleys that are relatively far apart. Skewed VNS, our next topic, is designed to handle this case. For further reading, also see Hansen and Mladenovic (2003).*

3.2. Skewed variable neighborhood search (SVNS)

SVNS was first introduced by Hansen et al. (2000). Up until then the neighborhood change step allowed a move to a new incumbent solution if, and only if, it was better than the current incumbent solution. Broadly speaking, in SVNS, a move to an inferior solution will also be permitted if this solution is only "slightly" inferior to the current incumbent solution and "sufficiently far" from it. The logic behind such moves is that they tend to diversify the search to promising regions that are far away from the "valley" where the search is currently locked. In order to perform SVNS, we only need to replace the current sequential neighborhood change step in Algorithm 2 (for BVNS) to a skewed neighborhood change step. In this new subroutine, a trial local minimum x'' is evaluated taking into account not only its objective value, but also the distance between it and the current incumbent solution x. The evaluation function used in the skewed neighborhood change step may be stated as:

$$g(x'', x) = f(x'') - f(x) - \alpha\delta(x'', x), \tag{9}$$

where α represents a (small) positive parameter, and $\delta(x'', x)$ gives the distance between solutions x'' and x, as defined previously. The skewed neighborhood change step allows x'' to become the new incumbent solution if $g(x'', x) < 0$. (Note that setting $\alpha = 0$ returns the neighborhood change step to its standard form.) See algorithm 3 for details.

Remark 9. *With SVNS, two parameters can be used to control the level of diversification. One is k_{max}, the number of neighborhoods available for the shake step. Selection of a large k_{max} increases the diversity by allowing randomly targeted searches in neighborhoods that are far from the incumbent solution. On the other hand, the skewed neighborhood change step with parameter α allows the search to move the "anchor point" to a new incumbent solution located in a different region of X that may provide a "richer" area to mine for local minima. A natural research question would relate to how these the two parameters (k_{max}, α)*

Algorithm 3: Skewed neighborhood change step

Procedure `Skewed_Neighborhood_change` (x, x'', k, α)
if $f(x) - f(x'') < \alpha\delta(x'', x)$ **then**
  $x \leftarrow x''$;
  $k \leftarrow 1$;
else
  $k \leftarrow k + 1$.

interact with each other to obtain best results. To illustrate, we could try smaller values (up to a point) of k_{max} coupled with larger values of α. This would allow more jumps to inferior solutions that could lead eventually to a very deep valley and high quality final solution. An issue of concern might be an increased likelihood of cycling back to previous incumbent solutions. Another issue to contemplate with SVNS is the timing of skewed moves. Premature movement could result in a missed opportunity to explore the region more thoroughly around a current incumbent solution.

3.3. Reduced variable neighborhood search (RVNS)

This is a speedy version of VNS where the improvement step is deleted. Thus, the neighborhood change step follows immediately after the shake step. Otherwise, RVNS is the same as BVNS. Note that the Monte Carlo method is a special case of RVNS, obtained by setting $k_{max} = 1$ and $\mathcal{N}_1(x) = X$. Because there is no improvement step, the RVNS may get stuck at a good incumbent solution x for some time. However, the iterations are so fast that many more points in varying regions of X are examined, and the net result is that RVNS tends to work surprisingly well (see e.g., (Cheimanoff et al., 2021; Hansen et al., 2001)).

Remark 10. *A post-RVNS procedure that performs local search on a large number of "good" solutions obtained during the RVNS, and that are dispersed through the feasible set X, might be worth considering.*

3.4. Variable neighborhood decomposition search (VNDS)

VNDS was first proposed by Hansen et al. (2001). Recall that in BVNS, the shaking operation obtains a random point in the kth neighborhood of the incumbent solution. The VNDS does the same thing, except that now the local search works on the subproblem containing only the k attributes that have been changed, while keeping fixed the remaining ones. The size of the subproblem changes systematically from smaller to larger in each iteration in accordance with the shake neighborhood parameter (k) in use. A local search is typically used to solve the subproblem in the improvement step. However, a second level of VNS may be applied instead, yielding a new two-level VNS approach. The steps of VNDS are given in Algorithm 4. Note that y corresponds to the initial point in the subproblem identified by the k attributes that are allowed to change. The attributes in the set $x' \setminus y$ identify the attributes that do not change. The improvement step is then applied in the reduced solution space. The solution y' returned by the improvement step is joined to $x' \setminus y$ to create solution $(x' \setminus y) \cup y'$ of the original problem.

Remark 11. *A current issue of importance deals not only with problem complexity but also the handling of large sets of data. Several optimization problems tend to be very complex because they are composed of many NP-hard sub-problems or because of the huge amount of available data. We believe that a viable approach for solving these problems would be to apply a decomposition approach such as VNDS. VNDS has been already successfully applied to several problems (see e.g., (Brimberg et al., 2006; Džamić et al., 2019; Lazić et al., 2010; Lazic et al., 2016; Lejeune, 2006; Plotnikov et al., 2019; Rupolo et al., 2021;*

Algorithm 4: Variable Neighborhood Decomposition Search

Function VNDS(x, k_{max}, $\mathcal{N}$)
repeat
  $k \leftarrow 1$;
  repeat
    $x' \leftarrow$ `Shake`$(x, k, \mathcal{N})$;
    $y \leftarrow x' \setminus x$;
    $y' \leftarrow$ `Improvement_procedure`(y);
    $x'' = (x' \setminus y) \cup y'$;
    `Neighborhood_change_sequential`(x, x'', k);
  until $k = k_{max}$;
until *stopping condition is fulfilled*;
return x.

Seeanner et al., 2013; Urošević et al., 2004)), but has been somehow forgotten while (Adaptive) Large Neighboorhood Search (LNS) seems to have gained much more attention. We believe the research track should also consider VNDS more deeply, as both approaches have many similar principles, some already present in VNDS since its introduction in 2001 (Hansen et al. (2001)).

Remark 12. *Note that the output solution x may not be a local optimum in the original solution space. Therefore, a final local search applied to x in the original solution space is advisable.*

3.5. General variable neighborhood search (GVNS)

Unlike BVNS, which applies a simple local search in a single neighborhood during the improvement step, GVNS intensifies this search by combining two or more neighborhoods. The motivation stems from the observation that a local minimum relative to one neighborhood structure is not necessarily a local minimum for another one, while a global minimum is a local minimum with respect to all neighborhood structures. Thus, a local search using several neighborhoods is more likely to find a global minimum than if it used only one of them. This intensified local search is referred to as Variable neighborhood descent (VND). The most common form of VND comprises a sequential search of the neighborhoods, and is referred to as Basic VND (BVND). Let $N = \{N_1, N_2, \ldots, N_{\ell_{max}}\}$ denote the set of selected neighborhoods arranged in the order they will be examined, and where l_{max} is a parameter similar to k_{max}. Typically, the neighborhoods are placed in order of increasing size. Each time a better solution is found, it becomes the new incumbent, and the search resumes from there with N_1. If no improvement is found in N_ℓ, the search moves to the next neighborhood $N_{\ell+1}$. If no improvement is found in each of the neighborhoods sequentially from N_1 to $N_{\ell_{max}}$, a local minimum for all ℓ_{max} neighborhoods is found, and this iteration of VND ends. This version of VND is fast as the search tends to occur more often in the smaller neighborhoods at the start of the sequence. The neighborhoods in N may also be nested, leading to a more intensive search from x, but also consuming much more CPU time. For further details on variants of VND, see (Hansen et al. (2017)).

3.6. Nested variable neighborhood search (NVNS)

The general idea behind NVNS is to apply a VNS procedure on all the points of a predefined neighborhood structure, instead of the normal application to one point - the incumbent solution (Brimberg et al., 2017; Todosijević et al., 2016a). To achieve efficient performance, it is crucial that the VNS applied at each point of the predefined neighborhood be very fast. Hence, it is desirable to assign a relatively small CPU

time limit to each such application. The VNS procedure, in effect, replaces a "local search" within the main loop. If the main loop also uses VNS, the procedure is referred to as a two-level VNS (Mladenović et al. (2014)).

Remark 13. *In the 'fast' VNS suggested above, the shake operation typically is assigned a small number of neighborhood structures that are relatively close to given point x. Instead of imposing a short execution time, it might be preferable to stop the procedure after a specified number of failures in order to ensure that stopping the procedure prematurely does not occur.*

Remark 14. *It would be very interesting to test and compare a random multi-start 'fast' VNS as a stand-alone procedure against standard local searches that are applied to a given problem class. Fast VNS could also be incorporated and tested within VND procedures (see e.g., Brimberg et al. (2017)).*

3.7. Combinations of variants

Some combinations of the VNS variants above have appeared in the literature. For example, in Brimberg et al. (2015), a Skewed general VNS (SGVNS) is proposed to solve the Maximally Diverse Grouping Problem. This combination of increased intensification in the local search coupled with increased diversification from skewed moves to inferior solutions would seem to make good sense. Intensifying the search in a new and promising region of the feasible set should be a good thing. It turns out to work very well in this case where several new best-found solutions are obtained by the SGVNS (also see e.g., (Derbel et al., 2019; Mladenović et al., 2022; Smiti et al., 2020)).

Remark 15. *Other combinations of the above variants may be interesting to try. For example, we could consider combining the two changes in the improvement step from VNDS and GVNS to get General variable neighborhood decomposition search (GVNDS), or going further, add skewed moves in the neighborhood change step to obtain Skewed general variable neighborhood decomposition search (SGVNDS). The intensified search in the decomposed problem combined with skewed moves to new regions of the solution space could well result in significant improvements over the state-of-the-art. Of course, this comes at a cost of increased complexity of the heuristic, but adding pieces that have synergy together would be worth it.*

4. Recent Advances in variable neighborhood search

This section discusses recent advances within the VNS framework, and combined with the next section, brings us more-or-less to the present day.

4.1. Adaptive neighborhood change step

The sequential neighborhood change step, which is the most popular procedure in practice, tends to favor the closer neighborhoods than the ones further away from the incumbent solution. This is because every time an improved solution is obtained (or the solution meets a specified skewed acceptance criterion), the neighborhood change step moves the incumbent solution to this new point, and the next iteration returns to the first (and closest) shake neighborhood ($\mathcal{N}_1$) of the new incumbent solution. If r_i denotes the total number of shake operations in $\mathcal{N}_i$ until the stopping criterion is reached, then clearly, $r_1 \geq r_2 \geq \cdots \geq r_{k_{max}}$. However, this makes little sense if, for example, $\mathcal{N}_1$ rarely, if ever, leads to an improved solution. The cyclic neighborhood change step (Mjirda et al., 2017; Todosijević et al., 2017b) takes care of this problem to some extent by always moving to the next shake neighborhood in the list, irrespective of a move occurring, or not, to a new incumbent solution. Thus, all the shake neighborhoods are equally represented ($r_1 = r_2 = \cdots = r_{k_{max}}$). Meanwhile, the pipe neighborhood change step (Mjirda et al., 2017; Todosijević

et al., 2016b) returns to the same neighborhood each time a shake operation to that neighborhood results in a move to an improved solution. In this way, the most effective neighborhoods tend to be used more frequently in the shake operation. We should specify that the neighborhood change steps do not occur only in shaking procedure, but also in VND.

The *adaptive neighborhood change* step uses a formal approach to decide which shake neighborhood will be selected in the next iteration by assigning a merit number to each neighborhood. Each time $\mathcal{N}_k$ is applied, its merit number increases if this iteration leads to an improved solution (or skewed move), and decreases, if it does not. The neighborhoods are arranged in order of merit, so that the next iteration will select the neighborhood with the highest merit. Thus, the adaptive change step operates the same as the pipe change step as long as the current neighborhood is successful and its merit number is increasing. However, as soon as it produces a failure, the adaptive change step selects the next neighborhood as the one with the highest merit number (which can still be the current one). Meanwhile, the pipe change step moves to the next neighborhood $\mathcal{N}_{k+1}$ as in the cyclic step.

Remark 16. *From our discussions so far, it is clear that the VNS framework is very flexible, in that it allows many different options to be implemented, and hence, opens several questions: Which neighborhood structures should be used? In which order should they be examined (i.e., which neighborhood change step to use)? What should be the values of the k_{max} and ℓ_{max} parameters? These are still open research questions to which answers are usually found by some preliminary testing or using a dedicated software such as IRACE (López-Ibáñez et al. (2016)). Either way, the best settings for a particular application are typically determined beforehand in a pre-processing phase (or set to a default value), and fixed for the rest of the execution. The adaptive neighborhood change step removes some of the rigidity in this process by allowing the sequencing of neighborhoods in the shaking operation to change or evolve during the execution of the heuristic. This view can be extended to incorporate more sophisticated aspects as will be seen next.*

Remark 17. *Another research line that is relatively recent, and that tries to provide answers dynamically to the preceding questions from the information gathered during the execution of a heuristic, involves coupling machine learning (ML) techniques and heuristics (Talbi (2021)). For example, to determine the VNS parameters, a machine learning (ML) technique called trained random forest regressor is used in Cheimanoff et al. (2021), while Q-learning is used in Öztop et al. (2020). The shaking parameter k_{max} is adjusted in an adaptive manner in Polacek et al. (2008), while Qiu et al. (2018) consider an adaptive adjustment of the skewing parameter α. This research area, known as hyper-parameter optimization, has received a lot of attention in the computer science arena.*
It is also interesting that adaptive change of the order of neighborhood structures is successfully implemented in (Aziz et al., 2017; Bezerra et al., 2023; Chen et al., 2020; Gu et al., 2019; Han et al., 2022; Li and Tian, 2016; Sze et al., 2016; Todosijević et al., 2016b; Wu and Che, 2020; Yazdani et al., 2017). Obviously, the adaptive change procedure above may also be replaced by machine learning techniques. The most common ML technique used to learn the order of neighborhoods is Q-learning (Alicastro et al., 2021; dos Santos et al., 2014; Meng et al., 2022). Kalatzantonakis et al. (2023) propose recently a new reinforcement learning GVNS where five Upper Confidence Bound algorithms are used to create a reliable strategy for adaptive neighborhood selection.

Remark 18. *Although the works on smart neighborhood selection in the intensification or diversification phase are relatively numerous, there are only two that simultaneously consider smart neighborhood selection in both these phases (Karakostas and Sifaleras, 2022; Pugliese et al., 2022). So, we believe there is much work to be done in this direction. Also, very little, if any, work has been attempted that investigates simultaneously smart parameter selections for k_{max}, ℓ_{max}, and skewed parameter α. This should be worth*

revisiting. In addition, this may all be further enhanced by coupling VNS with smart techniques for neighborhood investigation in the intensification phase that includes smart moves to avoid non-promising ones, generating new neighborhood structures based on the previous search process, etc. (Lei, 2015; Lotfi and Behnamian, 2022; Lucas et al., 2020).

4.2. Primal-dual variable neighborhood search

This variant of VNS is aimed at solving pure and mixed integer linear programs where the dual formulation of the problem can be used to provide a tight lower bound of the optimal objective function value, and hence, provide a good estimation of the quality of an obtained primal solution. This method was first introduced by Hansen et al. (2007) to solve the simple plant location problem (SPLP), which was selected because it is known to be integer friendly (ReVelle (1993)). Large-scale instances of SPLP that were out of the range of specialized algorithms were solved to optimality by their novel approach. In the first phase of primal-dual VNS, a VNS heuristic is used to obtain a (high-quality) primal feasible solution. This solution is then used to deduce a dual infeasible solution. A second VNS heuristic, which is designed to work on the dual problem, then reduces the dual infeasibility. This is followed by an exact method (a novel sliding simplex approach) that solves the relaxed dual problem and thus generates an improved lower bound. Finally, a standard branch-and-bound algorithm is launched to find an optimal solution of the original problem using the tight upper and lower bounds obtained from the heuristic's primal solution and the exact dual solution, respectively.

Remark 19. *Primal-dual VNS was also successfully applied to the (integer friendly) classical p-median problem (Hansen et al. (2009)). However, it appears that interest to extend the methodology to other classes of problems has not materialized. This situation suggests an obvious direction for further work. Problem classes that are not as integer friendly as the SPLP and p-median, for example, the p-centre and p-dispersion problems, and many useful variants of classical models, should also be investigated, as the bounds generated by solving the dual may provide some useful information. Solving exactly the dual relaxation of a large-scale problem may be as challenging as solving the primal (LP) relaxation. In this case, it could be advantageous to develop a good heuristic method to approximately solve the dual.*

4.3. Parallelization of variable neighborhood search

Parallelization of VNS can be a useful technique for tackling large instances of an optimization problem in reasonable time or obtaining better solutions within the same time limit. It is obvious that if tasks are performed in parallel instead of sequentially, there can be a saving in total execution time to obtain the same or better solution quality. In the literature, various parallelization strategies have been implemented within VNS heuristics. These include:

- parallelizing the local search, which could be particularly useful for large neighborhoods,
- augmenting the number of solutions drawn from the current neighbourhood of the shaking operation, and implementing local search in parallel from each one,
- updating the information about the best solution found, and
- parallelizing the search in diverse regions of the solution space.

Other more sophisticated approaches have been proposed (see e.g., (Davidović and Crainic, 2012; Kalatzantonakis et al., 2020; Schryen, 2020)). Applications of parallel VNS include the p-median problem, the car sequencing problem, vehicle routing related problems, job shop scheduling problems, dynamic

memory allocation problems, portfolio optimization, planning of large-scale power distribution systems, order batching, and more (Akbay et al., 2020; Crainic et al., 2004; García-López et al., 2002; Herrán et al., 2020; Pirkwieser and Raidl, 2009; Polacek et al., 2008; Rupolo et al., 2021; Sánchez-Oro et al., 2015; Sevkli and Aydin, 2007; Todosijević et al., 2017a; Wagner and Mönch, 2023; Yazdani et al., 2010; Zang et al., 2022), as well as multi-objective optimization problems (see e.g., (Eskandarpour et al., 2013; Sánchez-Oro et al., 2022)).

Remark 20. *Skewed VNS could be an excellent candidate for applying parallelization in a new way. When a skewed move is made, the search from the current incumbent stops in favor of the new anchor point. With parallelization, this would not be necessary. The search could continue in both regions by adding a computer. Furthermore, each time a new search is added, the skewed move criterion would have to be satisfied with respect to all anchor points to ensure that the new anchor point is sufficiently far from all others. A new stopping criterion might be attempted where the VNS heuristic ends if no new best solution has been found and no skewed move has been made during a specified time.*

Coupling parallelization techniques with existing and new VNS variants can undoubtedly remain an exciting and challenging research avenue.

4.4. Adaptation of VNS to Nonlinear Optimisation

For this topic, we assume that $f : R^n \to R$ is a continuous function in (1) with no other assumptions attached. In other words, f does not need to be convex or smooth.

To deal with this problem, both the shaking step as well as the local search step need to be modified accordingly to cater for the continuous space. Here, besides the setting of t_{max} and k_{max}, the choice of the local search solver is also required. For instance, first order methods such as steepest descent, Fletcher-Reeves, and variable metric or direct search methods, such as Nelder-Mead, Hooke-Jeeves, and Rosenbrock, can be used. Also in the shaking step, there are two issues that need serious consideration: (i) how the neighborhood structures $\mathcal{N}_k$ that are induced from some metric such as the ℓ_p norm are defined, and (ii) which probbability distribution $\mathcal{P}_k$ to adopt for the generation of the random solutions from $\mathcal{N}_k(x)$.

Neighborhood structures $\mathcal{N}_k$ are usually induced from the ℓ_∞ norm. Radii around the incumbent are found, taking into account the box constraints. One way is to divide each coordinate in k_{max} intervals from both sides of the incumbent, forming k_{max} concentric hyper-cubes. Interesting implementations of VNS for solving general unconstrained NLPs, known as `Glob-VNS`, are explored in (Audet et al., 2008; Mladenović et al., 2003), whereas the case of constrained NLPs can be found in those VNS variants that do not use metric distances to define neighborhoods but consider instead exterior penalty functions (Mladenović et al., 2008; Toksarı and Güner, 2007) or trust region methods (Audet et al., 2008; Bierlaire et al., 2010). These are usually incorporated into `GLOB-VNS` (Carrizosa et al. (2012)) to produce the so called Gaussian VNS (`Gauss-VNS`).

Remark 21. *The idea here is to replace the class* $\{\mathcal{N}_k(x)\}_{1\leq k\leq k_{\max}}$ *of neighborhoods of point* x *by a class of* probability distributions $\{\mathcal{P}_k(x)\}_{1\leq k\leq k_{\max}}$. *The next random point in the shaking step is generated using the probability distribution* $\mathcal{P}_k(x)$ *instead.*

Other enhancements of the above two variants are proposed in Dražić et al. (2016) while a successful implementation to mixed-integer nonlinear programs is explored in Liberti et al. (2010).

4.5. VNS-based Hybridisation

The hybridisation of VNS with other heuristics as well as with exact methods is pursued in several papers. In this subsection, we briefly discuss these two hybridisation-based classes.

4.5.1. Hybridisation of VNS with heuristics

The main idea here is to have at least one of the three VNS steps (i.e., shaking to a neighbouring solution, the local search step, or the neighborhood change step) complemented with another heuristic such as tabu search, path relinking, simulated annealing, greedy randomized adaptive search procedures (GRASP), genetic algorithm, particle swarm optimization, among others. For example, Li et al. (2014) combine VNS with chemical-reaction optimization and estimation of distribution (EDA) to solve the hybrid flow shop scheduling problem. In Oliveira et al. (2015), GVNS is hybridized with GRASP to solve the targeted offers problem in direct marketing campaigns. Belhaiza et al. (2014) propose a hybrid variable neighborhood-tabu search heuristic for the vehicle routing problem with multiple time windows, while Xiao et al. (2014) use a variable neighbourhood-simulated annealing algorithm to solve capacitated vehicle routing problems. In Kocatürk et al. (2021), the uathors propose hybridization of VNS with the Greedy Randomized Adaptive Memory Programming Search for the multi-depot heterogeneous VRP with backhauls. A hybrid VNS tabu search is proposed in Schermer et al. (2019) for the vehicle routing problem with drones. An interesting hybrid VNS-Lagrangean heuristic framework is proposed in Nogueira et al. (2022) for a scheduling problem. In Sun et al. (2023), a hybrid genetic algorithm with variable neighborhood search for the flexible job shop scheduling problem is proposed. Lu et al. (2021) propose a hybrid discrete black hole algorithm and variable neighborhood search approach to solve an integrated problem of scheduling and maintenance. For other VNS hybridization schemes, we refer the reader to (Hansen et al., 2010; Salhi and Thompson, 2022a).

4.5.2. Hybridisation of VNS with exact methods

VNS has also been integrated with exact methods leading to a heuristic class known as matheuristics. Here, either at least one of the components of the heuristic, the VNS in this case, needs to incorporate an exact type approach such as B&B, or at least one of the steps of the exact method such as B&B needs to include components from a heuristic. An overview on heuristic hybridisation is provided in (Salhi and Thompson (2022a)).

For completeness, we focus on the 0–1 mixed integer program (MIP) where useful implementations are carried out and interesting results are discovered. A useful metric that is commonly used throughout these matheuristics is the Hamming distance between two solutions x and x' such that $x_j, x'_j \in \{0,1\},\ j \in \mathcal{I}$. This is defined as

$$\delta(x,x') = \sum_{j\in\mathcal{I}} |x_j - x'_j| = \sum_{j\in\mathcal{I}} (x_j(1-x'_j) + x'_j(1-x_j)), \tag{10}$$

with the subset $\mathcal{I}$ containing the indices of binary variables from the entire index set of variables.

Some of these approaches are briefly provided here, but more details can be found in their respective references.

(a) Variable Neighborhood Branching. In 2006, Hansen et al. (2006) proposed a variable neighborhood search heuristic combined with Local Branching (LB) (Fischetti and Lodi (2003)), called *Variable Neighborhood Branching*. This can be seen as a generalization of LB with the added advantage of performing a more systematic neighborhood exploration.

(b) Hybrid Variable Neighborhood decomposition search heuristics. In 2010, Lazić et al. (2010) combined variable neighborhood decomposition search (VNDS) with the CPLEX MIP solver. At each iteration of the VNDS procedure, the distances $\delta_j = |x_j - \bar{x}_j|$ between the current incumbent solution values and the corresponding LP-relaxation solution values are computed. These distance values are used to fix some variables in subsequent iterations. Here, at each iteration, k variables whose indices correspond to the indices of k smallest δ_j values, are fixed to values of the current incumbent solution x and then the resulting (reduced) problem is solved using the CPLEX MIP solver. If an improvement of the current solution is

achieved, a Variable Neighborhood Descent branching is launched as the local search in the whole solution space and the process is repeated. If this is not the case, more flexibility is provided by reducing the number of fixed variables in the current subproblem. Interesting reduction schemes that restrict the search space by adding pseudo cuts, in order to avoid multiple explorations of the same areas, is successfully exploited in Hanafi et al. (2010b).

(c) Variable Neighborhood Pump. In 2010, Hanafi et al. (2010a) combined Variable neighborhood branching (VNB) (Hansen et al. (2006)) and Feasibility pump heuristics (Fischetti et al. (2005)) in their *Variable Neighborhood Pump* (VNP) to find an initial feasible solution. The idea is to round the optimal solution of the LP-relaxation of the initial 0-1 MIP problem and then apply one iteration of the VNP to obtain a near-feasible vector. A variable neighbourhood branching, adapted for 0-1 MIP feasibility (Hanafi et al. (2010a)), is then used on this new solution to locate a feasible solution of the original problem. If this is not found, a pseudo-cut is added to the current subproblem and the process is repeated.

(d) Diving heuristics. In 2016, Lazic et al. (2016) proposed diving heuristics to obtain a first MIP feasible solution. Diving heuristics are based on the systematic hard variable fixing (diving) process.

Similarly to earlier work, the idea is to first round the optimal solution of the LP-relaxation, and at each iteration of the diving procedure, compute the distances $\delta_j =\mid \tilde{x}_j - \overline{x}_j \mid$ values which are then indexed so that $\delta_1 \leq \delta_2 \leq \ldots \leq \delta_{|\mathcal{I}|}$. The diving algorithm then solves the subproblems $P(\tilde{x}, \{1, \ldots, k\})$ obtained from the original problem P, where the first k variables are fixed to the current incumbent solution values. If a feasible solution is found, the search terminates; otherwise, a pseudo-cut is added and the next subproblem is examined. More details as well as potential enhancements on this algorithm can be found in Lazic et al. (2016).

Remark 22. *The investigation on variable fixation mechanisms to reduce the resulting MIP subproblems while providing guided flexibility in such schemes can be challenging but worth exploring.*

5. Two VNS-related methodologies and "where from here?"

In this section, we shall present two methodologies that are related to VNS and which are also very close to Professor Mladenovic 's heart. These are "variable formulation space" and "Less-is-More".

5.1. Variable Formulation Space

Many optimization problems, e.g., min-max problems, demonstrate a flat landscape. It means that, given a formulation of the problem, many neighbors of a solution have the same objective function value. When this happens, it is difficult to determine which neighborhood solution is more promising to continue the search. To address this drawback, the use of alternative formulations of the problem within VNS is proposed and named Variable Formulation Search (VFS). This fits well, as many optimization problems have more than one formulation. In such a case, we can also stress that, similar to the philosophy adopted by VNS, a local optimum with respect to one formulation may not necessarily be a local optimum with respect to another formulation. The idea then is to extend the concept of neighbourhood to cater for formulation as well. In other words, this is achieved by combining a change of neighborhood within the VNS framework, with the use of alternative formulations.

Two formulations are defined as equivalent if an optimal solution of one is an optimal solution of the other, and vice versa. This notion is exploited by Mladenović et al. (2005) for the circle packing problem where two formulations are used, namely, Cartesian and polar coordinate formulations. The idea is to switch between the two while using the obtained solution of one formulation as an initial solution in the other. A similar attempt was also made earlier for solving the Cutwidth Minimization Problem in Pardo

et al. (2013). More information on the various variants, implementations and applications of VFS can be found in (Mladenović et al., 2005; Mladenović et al., 2010; Pardo et al., 2013).

Remark 23. *Investigations on how to accept solutions and on how to move between the formulations are challenges that should be further explored. This area of investigation is in the early stages, and we believe much more work can be accomplished here.*

5.2. Less-Is-More Approach

The main idea of the Less-is-More Approach, LIMA for short, is based on using as few ingredients as possible to provide the best possible outcome. For instance, this approach has been successfully applied in many areas such as Genetic Networks (Bornholdt (2005)), Medicine Treatment (Kaiser (2017)), Cancer Immunotherapy (Shi (2020)), Neuro-sciences (Chong (2011)), Signaling (Bannard and Cyster (2012)), Education (McCartney (2011)), Functional Materials (Bassiri-Gharb (2020)), to cite a few. Also, the title "Less-is-more" appears in over two dozens articles in the prestigious *Science* journal. However, it is worth stressing that this philosophy is seldomly used for solving optimization problems (OP).

It is also worth mentioning that in heuristic search design in general, LIMA fits well, as a heuristic usually is defined by several characteristics, namely, its speed, accuracy, simplicity, flexibility (i.e., ease of adaptation), effectiveness and robustness, ease to produce, etc. For example, general papers or chapters on heuristics highlight this issue, e.g. see Salhi (2017), while desirable meta-heuristic properties can be found in Hansen et al. (2019). However, there is no formal approach to consider these attributes. Hence, Professor Mladenović, with his students and colleagues, has attempted to formally provide a mathematical framework of one of the attributes, namely, simplicity, to be added alongside the two commonly used ones, speed (CPU time) and accuracy (solution quality). The resulting "Less-is-More Approach" (LIMA) provides a road map connecting Optimization to Artificial Intelligence and Machine Learning.

The usual way to compare different methods to solve an OP is to assess the relative solution quality (or algorithm effectiveness) expressed in terms of a percentage deviation (known also as accuracy) and algorithm efficiency measured by CPU time (also referred to as speed). The LIMA approach adds a comparison of the simplicity of each method to the mix.

Remark 24. *LIMA reinforces the principle that including many ideas in the search does not necessarily lead to better computational results. This claim is often observed in heuristics as there is a lack of correlation between complexity and effectiveness. On the contrary, less can often even yield more.*

An interesting dominance relation that measures an algorithm's simplicity is contained in the following two definitions described in Mladenović et al. (2022):

1- The cardinality of the set of ingredients U used to solve optimization problem P (given in (1)) on data set D by method A is called the *LIMA number* of A.

2- We say that algorithm B is *LIMA-dominant* over algorithm A in solving problem P on data set D, if and only if,

(a) accuracy or quality: the objective function $f_B \leq f_A$;

(b) speed: the running time $t_B \leq t_A$;

(c) simplicity: the cardinalities of the ingredient sets (LIMA numbers) satisfy $|U_B| \leq |U_A|$;

(d) At least one of conditions (a), (b), (c) is satisfied strictly.

Note that the LIMA-dominance relation could be too strong if applied to each run and each test instance. One way forward is to introduce the concept of weak or *average LIMA-dominance*, i.e., after many runs on the same problem, we verify that average values satisfy $f_B \leq f_A$ and $t_B \leq t_A$.

Remark 25. *The problem can be considered as a challenging three-objective optimization problem defined by (a), (b) and (c).*

There are several ways of tackling a multi-objective problem including weighted multi objective, goal programming and lexicographic, among others. If one wants to stress attribute (c), namely the simplicity, the aim would be to solve the following problem:

$$\min_{U' \subseteq U} \{|U'| \; | f_B^{(k)} \leq f_*^{(k)}, t_B^{(k)} \leq t_*^{(k)}; k = 1, 2, \ldots M\}, \tag{11}$$

where index k identifies instance $T^{(k)} \in T$; $T = \{T^{(1)}, T^{(2)}, \ldots, T^{(M)}\}$ is a bank of benchmark instances; $f_* = (f_*^{(1)}, f_*^{(2)}, \ldots, f_*^{(M)})$ is a vector of objective function values obtained by state-of-the-art algorithm A; $f_B = (f_B^{(1)}, f_B^{(2)}, \ldots, f_B^{(M)})$ is a vector of objective function values obtained by algorithm B using only the ingredients in U'; and $t_* = (t_*^{(1)}, t_*^{(2)}, \ldots, t_*^{(M)})$, $t_B = (t_B^{(1)}, t_B^{(2)}, \ldots, t_B^{(M)})$ are, respectively, the running times of algorithm A and B,

The objective in (11) is to find subset U' with the minimum number of ingredients (i.e., the smallest LIMA number) that satisfies the stated conditions. If $|U'| < |U_*|$ (the number of ingredients used in A), then algorithm B dominates algorithm A in the LIMA sense. As noted above the 'total' dominance implied in (11) may be replaced by a less stringed requirement such as 'average' dominance or dominance by statistical hypothesis testing. For further details, including some successful applications of LIMA, see Mladenović et al. (2022).

5.3. Where from here?

Since the inception of VNS in 1995, we have seen a tremendous rise in the number of different heuristics that have been developed within the VNS framework and the number of successful applications of these heuristics. As pointed out in Hansen and Mladenovic (2003), there are eight desirable properties of a metaheuristic (simplicity, precision, coherence, efficiency, effectiveness, robustness, user-friendliness and innovation), and VNS possesses to a large degree all of them. Therefore, we are very optimistic that this growth will continue for many years to come. One area of further developments will certainly be in the design of new VNS-based heuristics and hybrid heuristics, which combine VNS with other ingredients. This will be driven in some part by new classes of problems where VNS can play a useful role. For example, a promising research direction not yet mentioned would involve the application of VNS for solving optimization problems under uncertainty. Problem data may be subject to uncertainty or the model under investigation may contain important stochastic elements, but this aspect is in many cases relatively less explored. Hence, we believe that one way forward to account for uncertainty is to couple VNS with other techniques from robust/stochastic optimization. Some works in this direction do exist, but there are very few compared to their counterpart, the deterministic case. Some successful applications in this area may be found in e.g., (Antoniadis et al., 2022; Gruler et al., 2018, 2020, 2017; Kizys et al., 2022; Panadero et al., 2020; Tayebi Araghi et al., 2021).

Heuristics are designed to solve optimization problems, but they may have other uses too. For example, VNS has been applied successfully to enhance graph theory by following a so-called Artificial Intelligence approach. An aim would be to automate such an approach (Hansen and Mladenovic (2003)). Such non-conventional use refers to the last desirable property of a metaheuristic mentioned above, namely, that one of innovation. In this respect, we predict that VNS will play an important role in new applications that do not necessarily involve searching for optimal solutions of mathematical models. As suggested in other parts of this paper, one such possible direction would be to apply VNS as an exploratory tool to study the structure of mathematical models. A better theoretical or empirical understanding of the structure of these

models will in turn lead to improved designs of algorithms to solve them. We believe this will be a fruitful area for future research.

6. Conclusions

In this review paper, honoring the life achievement of Professor Mladenović, we mainly concentrate on important developments in variable neighbourhood search (VNS), its origin, various adaptations and novel advances. The various successes achieved by VNS are also noted through references to a wide range of real applications where superior results were obtained. Though the main focus is on VNS, we also briefly describe two related areas pioneered by Professor Mladenović that were also close to his heart, namely, variable formulation space (VFS) and less-is-more approach (LIMA).

Throughout the text, we also highlighted several remarks needing further reflections. We hope that these remarks will convince readers that, for VNS and related methodologies, new and exciting applications and avenues of research most definitely lie ahead.

Acknowledgment

This paper was written at the request of the Editor-in-Chief, Professor Francisco Saldanha da Gama. We thank him for this opportunity to pay tribute to Professor Nenad Mladenović's pioneering work in Variable Neighborhood Search. In addition, we are very thankful to Professor Francisco Saldanha da Gama for his valuable suggestions and comments that helped us to improve both the content and the presentation of the paper.

References

Abdelkhalek, O., Masri, H., and Krichen, S. (2015). An adaptive variable neighborhood search for solving the multi-objective node placement problem. *Electronic Notes in Discrete Mathematics*, 47:189–196.

Akbay, M. A., Kalayci, C. B., and Polat, O. (2020). A parallel variable neighborhood search algorithm with quadratic programming for cardinality constrained portfolio optimization. *Knowledge-Based Systems*, 198:105944.

Alicastro, M., Ferone, D., Festa, P., Fugaro, S., and Pastore, T. (2021). A reinforcement learning iterated local search for makespan minimization in additive manufacturing machine scheduling problems. *Computers & Operations Research*, 131:105272.

Antoniadis, N., Cordy, M., Sifaleras, A., and Le Traon, Y. (2022). A variable neighborhood search simheuristic algorithm for reliability optimization of smart grids under uncertainty. *International Transactions in Operational Research*, 29(4):2172–2200.

Audet, C., Béchard, V., and Le Digabel, S. (2008). Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search. *Journal of Global Optimization*, 41(2):299–318.

Aziz, R. A., Ayob, M., Othman, Z., Ahmad, Z., and Sabar, N. R. (2017). An adaptive guided variable neighborhood search based on honey-bee mating optimization algorithm for the course timetabling problem. *Soft Computing*, 21(22):6755–6765.

Bannard, O. M. and Cyster, J. G. (2012). When less signaling is more. *Science*, 336:1120–1121.

Bassiri-Gharb, N. (2020). Less can be more in functional materials. *Science*, 369:252–253.

Belhaiza, S., Hansen, P., and Laporte, G. (2014). A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows. *Computers & Operations Research*, 52:269–281.

Bezerra, S. N., Souza, M. J. F., and de Souza, S. R. (2023). A variable neighborhood search-based algorithm with adaptive local search for the vehicle routing problem with time windows and multi-depots aiming for vehicle fleet reduction. *Computers & Operations Research*, 149:106016.

Bierlaire, M., Thémans, M., and Zufferey, N. (2010). A heuristic for nonlinear global optimization. *INFORMS Journal on Computing*, 22(1):59–70.

Bornholdt, S. (2005). Less is more in modeling large genetic networks. *Science*, 310:449–451.

Brimberg, J., Hansen, P., and Mladenović, N. (2006). Decomposition strategies for large-scale continuous location–allocation problems. *IMA Journal of Management Mathematics*, 17(4):307–316.

Brimberg, J., Hansen, P., and Mladenovic, N. (2010). Attraction probabilities in variable neighborhood search. *4OR*, 8(2):181–194.

Brimberg, J., Hansen, P., Mladenović, N., and Taillard, E. D. (2000). Improvements and comparison of heuristics for solving the uncapacitated multisource weber problem. *Operations Research*, 48(3):444–460.

Brimberg, J. and Mladenovic, N. (1996). A variable neighborhood algorithm for solving the continuous location-allocation problem. *Studies in Locational Analysis*, 10:1–12.

Brimberg, J., Mladenović, N., Todosijević, R., and Urošević, D. (2017). A general framework for nested variable neighborhood search. *Electronic Notes in Discrete Mathematics*, 58:159–166.

Brimberg, J., Mladenović, N., and Urošević, D. (2015). Solving the maximally diverse grouping problem by skewed general variable neighborhood search. *Information Sciences*, 295:650–675.

Cao, L., Ye, C.-m., Cheng, R., and Wang, Z.-k. (2022). Memory-based variable neighborhood search for green vehicle routing problem with passing-by drivers: A comprehensive perspective. *Complex & Intelligent Systems*, pages 1–19.

Carrizosa, E., Dražić, M., Dražić, Z., and Mladenović, N. (2012). Gaussian variable neighborhood search for continuous optimization. *Computers & Operations Research*, 39(9):2206–2213.

Cheimanoff, N., Fontane, F., Kitri, M. N., and Tchernev, N. (2021). A reduced VNS based approach for the dynamic continuous berth allocation problem in bulk terminals with tidal constraints. *Expert Systems with Applications*, 168:114215.

Chen, B., Qu, R., Bai, R., and Laesanklang, W. (2020). A variable neighborhood search algorithm with reinforcement learning for a real-life periodic vehicle routing problem with time windows and open routes. *RAIRO-Operations Research*, 54(5):1467–1494.

Chong, L. (2011). Exosomes Deliver. *Science*, 332:515.

Crainic, T. G., Gendreau, M., Hansen, P., and Mladenović, N. (2004). Cooperative parallel variable neighborhood search for the p-median. *Journal of Heuristics*, 10(3):293–314.

Davidović, T. and Crainic, T. G. (2012). MPI parallelization of variable neighborhood search. *Electronic Notes in Discrete Mathematics*, 39:241–248.

Derbel, H., Jarboui, B., and Bhiri, R. (2019). A skewed general variable neighborhood search algorithm with fixed threshold for the heterogeneous fleet vehicle routing problem. *Annals of Operations Research*, 272(1-2):243–272.

dos Santos, J. P. Q., de Melo, J. D., Neto, A. D. D., and Aloise, D. (2014). Reactive search strategies using reinforcement learning, local search algorithms and variable neighborhood search. *Expert Systems with Applications*, 41(10):4939–4949.

Dražić, M., Dražić, Z., Mladenović, N., Urošević, D., and Zhao, Q. H. (2016). Continuous variable neighbourhood search with modified Nelder–Mead for non-differentiable optimization. *IMA Journal of Management Mathematics*, 27(1):75–88.

Dražić, M., Lavor, C., Maculan, N., and Mladenović, N. (2008). A continuous variable neighborhood search heuristic for finding the three-dimensional structure of a molecule. *European Journal of Operational Research*, 185(3):1265–1273.

Džamić, D., Aloise, D., and Mladenović, N. (2019). Ascent–descent variable neighborhood decomposition search for community detection by modularity maximization. *Annals of Operations Research*, 272(1):273–287.

Eskandarpour, M., Zegordi, S. H., and Nikbakhsh, E. (2013). A parallel variable neighborhood search for the multi-objective sustainable post-sales network design problem. *International Journal of Production Economics*, 145(1):117–131.

Fischetti, M., Glover, F., and Lodi, A. (2005). The feasibility pump. *Mathematical Programming*, 104(1):91–104.

Fischetti, M. and Lodi, A. (2003). Local branching. *Mathematical programming*, 98:23–47.

García-López, F., Melián-Batista, B., Moreno-Pérez, J. A., and Moreno-Vega, J. M. (2002). The parallel variable neighborhood search for the p-median problem. *Journal of Heuristics*, 8(3):375–388.

Gruler, A., Panadero, J., de Armas, J., Moreno Pérez, J., and Juan, A. (2018). Combining variable neighborhood search with simulation for the inventory routing problem with stochastic demands and stock-outs. *Computers and Industrial Engineering*, 123:278–288.

Gruler, A., Panadero, J., de Armas, J., Pérez, J., and Juan, A. (2020). A variable neighborhood search simheuristic for the multi-period inventory routing problem with stochastic demands. *International Transactions in Operational Research*, 27(1):314–335.

Gruler, A., Quintero-Araújo, C., Calvet, L., and Juan, A. (2017). Waste collection under uncertainty: A simheuristic based on variable neighbourhood search. *European Journal of Industrial Engineering*, 11(2):228–255.

Gu, X., Huang, M., and Liang, X. (2019). An improved genetic algorithm with adaptive variable neighborhood search for FJSP. *Algorithms*, 12(11):243.

Guastaroba, G. and Speranza, M. (2012). Kernel search for the capacitated facility problem. *Journal of Heuristics*, 18:877–917.

Han, X., Han, Y., Zhang, B., Qin, H., Li, J., Liu, Y., and Gong, D. (2022). An effective iterative greedy algorithm for distributed blocking flowshop scheduling problem with balanced energy costs criterion. *Applied Soft Computing*, 129:109502.

Hanafi, S., Lazić, J., and Mladenović, N. (2010a). Variable neighbourhood pump heuristic for 0–1 mixed integer programming feasibility. *Electronic Notes in Discrete Mathematics*, 36:759–766.

Hanafi, S., Lazić, J., Mladenović, N., Wilbaut, C., and Crévits, I. (2010b). Hybrid variable neighbourhood decomposition search for 0–1 mixed integer programming problem. *Electronic Notes in Discrete Mathematics*, 36:883–890.

Hansen, P., Brimberg, J., Urošević, D., and Mladenović, N. (2007). Primal-dual variable neighborhood search for the simple plant-location problem. *INFORMS Journal on Computing*, 19(4):552–564.

Hansen, P., Brimberg, J., Urošević, D., and Mladenović, N. (2009). Solving large p-median clustering problems by primal–dual

variable neighborhood search. *Data Mining and Knowledge Discovery*, 19:351–375.

Hansen, P., Jaumard, B., Mladenovic, N., and Parreira, A. (2000). Variable neighborhood search for weighted maximum satisfiability problem. *Les Cahiers du GERAD*, 62.

Hansen, P. and Mladenovic, N. (2003). A tutorial on variable neighborhood search. *Les Cahiers du GERAD*, 711:2440.

Hansen, P. and Mladenović, N. (2006). First vs. best improvement: An empirical study. *Discrete Applied Mathematics*, 154(5):802–817.

Hansen, P., Mladenović, N., Brimberg, J., and Pérez, J. A. M. (2019). *Variable Neighborhood Search*, pages 57–97. Springer International Publishing, Cham.

Hansen, P., Mladenović, N., and Pérez, J. A. M. (2010). Variable neighbourhood search: methods and applications. *Annals of Operations Research*, 175(1):367–407.

Hansen, P., Mladenović, N., and Perez-Britos, D. (2001). Variable neighborhood decomposition search. *Journal of Heuristics*, 7(4):335–350.

Hansen, P., Mladenović, N., Todosijević, R., and Hanafi, S. (2017). Variable neighborhood search: basics and variants. *EURO Journal on Computational Optimization*, 5(3):423–454.

Hansen, P., Mladenović, N., and Urosević, D. (2006). Variable neighborhood search and local branching. *Computers & Operations Research*, 33(10):3034–3045.

Herrán, A., Colmenar, J. M., Martí, R., and Duarte, A. (2020). A parallel variable neighborhood search approach for the obnoxious p-median problem. *International Transactions in Operational Research*, 27(1):336–360.

Johnson, D. S. and McGeoch, L. A. (1997). The traveling salesman problem: A case study in local optimization. *Local search in combinatorial optimization*, 1(1):215–310.

Jomaa, W., Eddaly, M., and Jarboui, B. (2021). Variable neighborhood search algorithms for the permutation flowshop scheduling problem with the preventive maintenance. *Operational Research*, 21(4):2525–2542.

Kaiser, J. (2017). When less is more. *Science*, 355:1144–1146.

Kalatzantonakis, P., Sifaleras, A., and Samaras, N. (2020). Cooperative versus non-cooperative parallel variable neighborhood search strategies: a case study on the capacitated vehicle routing problem. *Journal of Global Optimization*, 78(2):327–348.

Kalatzantonakis, P., Sifaleras, A., and Samaras, N. (2023). A reinforcement learning-variable neighborhood search method for the capacitated vehicle routing problem. *Expert Systems with Applications*, 213:118812.

Karakostas, P. and Sifaleras, A. (2022). A double-adaptive general variable neighborhood search algorithm for the solution of the traveling salesman problem. *Applied Soft Computing*, 121:108746.

Karakostas, P., Sifaleras, A., and Georgiadis, M. C. (2020). Adaptive variable neighborhood search solution methods for the fleet size and mix pollution location-inventory-routing problem. *Expert Systems with Applications*, 153:113444.

Kariv, O. and Hakimi, S. L. (1979). An algorithmic approach to network location problems. I: the p-centers. *SIAM journal on applied mathematics*, 37(3):513–538.

Kirkpatrick, S. and Toulouse, G. (1985). Configuration space analysis of travelling salesman problems. *Journal de Physique*, 46(8):1277–1292.

Kizys, R., Doering, J., Juan, A., Polat, O., Calvet, L., and Panadero, J. (2022). A simheuristic algorithm for the portfolio optimization problem with random returns and noisy covariances. *Computers & Operations Research*, 139:105631.

Kocatürk, F., Tütüncü, G. Y., and Salhi, S. (2021). The multi-depot heterogeneous VRP with backhauls: formulation and a hybrid vns with gramps meta-heuristic approach. *Annals of Operations Research*, 307(1-2):277–302.

Lazić, J., Hanafi, S., Mladenović, N., and Urošević, D. (2010). Variable neighbourhood decomposition search for 0–1 mixed integer programs. *Computers & Operations Research*, 37(6):1055–1067.

Lazic, J., Todosijević, R., Hanafi, S., and Mladenović, N. (2016). Variable and single neighbourhood diving for MIP feasibility. *Yugoslav Journal of Operations Research*, 25(2):131–157.

Lei, D. (2015). Variable neighborhood search for two-agent flow shop scheduling problem. *Computers & Industrial Engineering*, 80:125–131.

Lejeune, M. A. (2006). A variable neighborhood decomposition search method for supply chain management planning problems. *European Journal of Operational Research*, 175(2):959–976.

Li, J.-q., Pan, Q.-k., and Wang, F.-t. (2014). A hybrid variable neighborhood search for solving the hybrid flow shop scheduling problem. *Applied Soft Computing*, 24:63–77.

Li, K. and Tian, H. (2016). A two-level self-adaptive variable neighborhood search algorithm for the prize-collecting vehicle routing problem. *Applied Soft Computing*, 43:469–479.

Liberti, L., Nannicini, G., and Mladenović, N. (2010). A good recipe for solving MINLPs. In Maniezzo, V., Stuetze, T., and Voss, S., editors, *MATHEURISTICS: Hybridizing metaheuristics and mathematical programming, Operations Research/Computer Science Interface Series*, pages 231–244. Springer, Berlin.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., and Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.

Lotfi, M. and Behnamian, J. (2022). Collaborative scheduling of operating room in hospital network: Multi-objective learning variable neighborhood search. *Applied Soft Computing*, 116:108233.

Lu, S., Pei, J., Liu, X., and Pardalos, P. M. (2021). A hybrid DBH-VNS for high-end equipment production scheduling with machine failures and preventive maintenance activities. *Journal of Computational and Applied Mathematics*, 384:113195.

Lucas, F., Billot, R., Sevaux, M., and Sörensen, K. (2020). Reducing space search in combinatorial optimization using machine learning tools. In *International conference on learning and intelligent optimization*, pages 143–150. Springer.

Luis, M., Salhi, S., and Gabor, N. (2011). A guided reactive GRASP for the capacitated multi source weber problem. *Computers & Operations Research*, 38:1014–1024.

McCartney, M. (2011). Calendar effects. *Science*, 334:1324–1324.

Meng, K., Tang, Q., and Zhang, Z. (2022). Robust assembly line balancing problem considering preventive maintenance scenarios with interval processing time. *Engineering Applications of Artificial Intelligence*, 116:105417.

Mjirda, A., Todosijević, R., Hanafi, S., Hansen, P., and Mladenović, N. (2017). Sequential variable neighborhood descent variants: an empirical study on the traveling salesman problem. *International Transactions in Operational Research*, 24(3):615–633.

Mladenović, N., Dražić, M., Kovačevic-Vujčić, V., and Čangalović, M. (2008). General variable neighborhood search for the continuous optimization. *European Journal of Operational Research*, 191(3):753–770.

Mladenović, N., Drezner, Z., Brimberg, J., and Urošević, D. (2022). Less is more approach in heuristic optimization. In *The Palgrave Handbook of Operations Research, S. Salhi and J. Boylan (Eds)*, pages 469–499. Palgrave Macmillan, Cham.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100.

Mladenović, N., Petrović, J., Kovačević-Vujčić, V., and Čangalović, M. (2003). Solving spread spectrum radar polyphase code design problem by tabu search and variable neighbourhood search. *European Journal of Operational Research*, 151(2):389–399.

Mladenović, N., Plastria, F., and Urošević, D. (2005). Reformulation descent applied to circle packing problems. *Computers & Operations Research*, 32(9):2419–2434.

Mladenović, N., Todosijević, R., Urošević, D., and Ratli, M. (2022). Solving the capacitated dispersion problem with variable neighborhood search approaches: From basic to skewed VNS. *Computers & Operations Research*, 139:105622.

Mladenović, N., Todosijević, R., and Urošević, D. (2014). Two level general variable neighborhood search for attractive traveling salesman problem. *Computers & Operations Research*, 52:341–348.

Mladenović, N., Urošević, D., Pérez-Brito, D., and García-González, C. (2010). Variable neighbourhood search for bandwidth reduction. *European Journal on Operational Research*, 200:14–27.

Nogueira, T. H., Ramalhinho, H. L., de Carvalho, C. R., and Gomez Ravetti, M. (2022). A hybrid VNS-Lagrangean heuristic framework applied on single machine scheduling problem with sequence-dependent setup times, release dates and due dates. *Optimization Letters*, 16:59–78.

Oliveira, T., Coelho, V., Souza, M. J. F., Boava, D. L. T., Boava, F., Coelho, I. M., and Coelho, B. N. (2015). A hybrid variable neighborhood search algorithm for targeted offers in direct marketing. *Electronic Notes in Discrete Mathematics*, 47:205–212.

Öztop, H., Tasgetiren, M. F., Kandiller, L., and Pan, Q.-K. (2020). A novel general variable neighborhood search through q-learning for no-idle flowshop scheduling. In *2020 ieee congress on evolutionary computation (CEC)*, pages 1–8. IEEE.

Pacheco, J., Porras, S., Casado, S., and Baruque, B. (2018). Variable neighborhood search with memory for a single-machine scheduling problem with periodic maintenance and sequence-dependent set-up times. *Knowledge-Based Systems*, 145:236–249.

Panadero, J., Juan, A., Bayliss, C., and Currie, C. (2020). Maximising reward from a team of surveillance drones: A simheuristic approach to the stochastic team orienteering problem. *European Journal of Industrial Engineering*, 14(4):485–516.

Pardo, E. G., Mladenović, N., Pantrigo, J. J., and Duarte, A. (2013). Variable formulation search for the cutwidth minimization problem. *Applied Soft Computing*, 13(5):2242–2252.

Pirkwieser, S. and Raidl, G. R. (2009). Multiple variable neighborhood search enriched with ILP techniques for the periodic vehicle routing problem with time windows. In *Hybrid Metaheuristics*, pages 45–59. Springer.

Plotnikov, R., Erzin, A., and Mladenovic, N. (2019). VNDS for the min-power symmetric connectivity problem. *Optimization Letters*, 13(8):1897–1911.

Polacek, M., Benkner, S., Doerner, K. F., and Hartl, R. F. (2008). A cooperative and adaptive variable neighborhood search for the multi depot vehicle routing problem with time windows. *Business Research*, 1(2):207–218.

Pugliese, L. D. P., Ferone, D., Festa, P., Guerriero, F., and Macrina, G. (2022). Combining variable neighborhood search and machine learning to solve the vehicle routing problem with crowd-shipping. *Optimization Letters*.

Qiu, Y., Wang, L., Xu, X., Fang, X., and Pardalos, P. M. (2018). A variable neighborhood search heuristic algorithm for production routing problems. *Applied Soft Computing*, 66:311–318.

ReVelle, C. (1993). Facility siting and integer-friendly programming. *European Journal of Operational Research*, 65(2):147–158.

Rosing, K. and ReVelle, C. (1997). Heuristic concentration: Two stage solution construction. *European Journal of Operational Research*, 97:75–86.

Rupolo, D., Junior, B. R. P., Contreras, J., and Mantovani, J. R. S. (2021). A new parallel and decomposition approach to solve

the medium-and low-voltage planning of large-scale power distribution systems. *International Journal of Electrical Power & Energy Systems*, 132:107191.

Salhi, S. (2017). Heuristic search: The emerging science of problem solving. *Palgrave Macmillan, Cham*.

Salhi, S. and Thompson, J. (2022a). The new era of hybridisation and learning in heuristic search design. In *The Palgrave Handbook of Operations Research, S. Salhi and J. Boylan (Eds)*, pages 501–538. Palgrave Macmillan, Cham.

Salhi, S. and Thompson, J. (2022b). An overview of heuristics and metaheuristics. In *The Palgrave Handbook of Operations Research, S. Salhi and J. Boylan (Eds)*, pages 353–403. Palgrave Macmillan, Cham.

Sánchez-Oro, J., López-Sánchez, A. D., and Colmenar, J. M. (2022). A multi-objective parallel variable neighborhood search for the bi-objective obnoxious p-median problem. *Optimization Letters*, pages 1–31.

Sánchez-Oro, J., Sevaux, M., Rossi, A., Martí, R., and Duarte, A. (2015). Solving dynamic memory allocation problems in embedded systems with parallel variable neighborhood search strategies. *Electronic Notes in Discrete Mathematics*, 47:85–92.

Schermer, D., Moeini, M., and Wendt, O. (2019). A hybrid VNS/Tabu search algorithm for solving the vehicle routing problem with drones and en route operations. *Computers & Operations Research*, 109:134–158.

Schryen, G. (2020). Parallel computational optimization in operations research: A new integrative framework, literature review and research directions. *European Journal of Operational Research*, 287(1):1–18.

Seeanner, F., Almada-Lobo, B., and Meyr, H. (2013). Combining the principles of variable neighborhood decomposition search and the fix&optimize heuristic to solve multi-level lot-sizing and scheduling problems. *Computers & Operations Research*, 40(1):303–317.

Sevkli, M. and Aydin, M. E. (2007). Parallel variable neighbourhood search algorithms for job shop scheduling problems. *IMA Journal of Management Mathematics*, 18(2):117–133.

Shi, L. Z. (2020). Less is more for adoptive immunotherapy? *Science Translational Medicine*, 12.

Smiti, N., Dhiaf, M. M., Jarboui, B., and Hanafi, S. (2020). Skewed general variable neighborhood search for the cumulative capacitated vehicle routing problem. *International Transactions in Operational Research*, 27(1):651–664.

Stenger, A., Vigo, D., Enz, S., and Schwind, M. (2013). An adaptive variable neighborhood search algorithm for a vehicle routing problem arising in small package shipping. *Transportation Science*, 47(1):64–80.

Sun, K., Zheng, D., Song, H., Cheng, Z., Lang, X., Yuan, W., and Wang, J. (2023). Hybrid genetic algorithm with variable neighborhood search for flexible job shop scheduling problem in a machining system. *Expert Systems with Applications*, 215:119359.

Sze, J. F., Salhi, S., and Wassan, N. (2016). A hybridisation of adaptive variable neighbourhood search and large neighbourhood search: Application to the vehicle routing problem. *Expert Systems with Applications*, 65:383–397.

Talbi, E.-G. (2021). Machine learning into metaheuristics: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(6):1–32.

Tayebi Araghi, M., Tavakkoli-Moghaddam, R., Jolai, F., and Hadji Molana, S. (2021). A green multi-facilities open location-routing problem with planar facility locations and uncertain customer. *Journal of Cleaner Production*, 282:124343.

Teitz, M. B. and Bart, P. (1968). Heuristic methods for estimating the generalized vertex median of a weighted graph. *Operations Research*, 16(5):955–961.

Thevenin, S. and Zufferey, N. (2019). Learning variable neighborhood search for a scheduling problem with time windows and rejections. *Discrete Applied Mathematics*, 261:344–353.

Todosijević, R., Benmansour, R., Hanafi, S., Mladenović, N., and Artiba, A. (2016a). Nested general variable neighborhood search for the periodic maintenance problem. *European Journal of Operational Research*, 252(2):385–396.

Todosijević, R., Hanafi, S., Urošević, D., Jarboui, B., and Gendron, B. (2017a). A general variable neighborhood search for the swap-body vehicle routing problem. *Computers & Operations Research*, 78:468–479.

Todosijević, R., Mjirda, A., Mladenović, M., Hanafi, S., and Gendron, B. (2017b). A general variable neighborhood search variants for the travelling salesman problem with draft limits. *Optimization Letters*, 11:1047–1056.

Todosijević, R., Mladenović, M., Hanafi, S., Mladenović, N., and Crévits, I. (2016b). Adaptive general variable neighborhood search heuristics for solving the unit commitment problem. *International Journal of Electrical Power & Energy Systems*, 78:873–883.

Toksarı, M. D. and Güner, E. (2007). Solving the unconstrained optimization problem by a variable neighborhood search. *Journal of Mathematical Analysis and Applications*, 328(2):1178–1187.

Urošević, D., Brimberg, J., and Mladenović, N. (2004). Variable neighborhood decomposition search for the edge weighted k-cardinality tree problem. *Computers & Operations Research*, 31(8):1205–1213.

Wagner, S. and Mönch, L. (2023). A variable neighborhood search approach to solve the order batching problem with heterogeneous pick devices. *European Journal of Operational Research*, 304(2):461–475.

Wilbaut, C., Salhi, S., and Hanafi, S. (2009). An iterative variable-based fixation heuristic for the 0-1 multidimensional knapsack problem. *European Journal of Operational Research*, 199:339–348.

Wu, X. and Che, A. (2020). Energy-efficient no-wait permutation flow shop scheduling by adaptive multi-objective variable neighborhood search. *Omega*, 94:102117.

Xiao, Y., Zhao, Q., Kaku, I., and Mladenovic, N. (2014). Variable neighbourhood simulated annealing algorithm for capacitated vehicle routing problems. *Engineering Optimization*, 46(4):562–579.

Yazdani, M., Amiri, M., and Zandieh, M. (2010). Flexible job-shop scheduling with parallel variable neighborhood search algorithm. *Expert Systems with Applications*, 37(1):678–687.

Yazdani, M., Khalili, S. M., Babagolzadeh, M., and Jolai, F. (2017). A single-machine scheduling problem with multiple unavailability constraints: A mathematical model and an enhanced variable neighborhood search approach. *Journal of Computational Design and Engineering*, 4(1):46–59.

Zang, X., Jiang, L., Ratli, M., and Ding, B. (2022). A parallel variable neighborhood search for solving covering salesman problem. *Optimization Letters*, 16:175–190.