A note on a model of local search

Andrea Roli

TR/IRIDIA/2004/23.01
December 2004
A note on a model of local search*

Andrea Roli
Dipartimento di Scienze
Università degli Studi “G.D’Annunzio”
Pescara – Italy
a.roli@unich.it

December 7, 2004

Abstract

Local search is a very efficient and effective strategy for attacking hard combinatorial problems. Nevertheless, its search behavior has not yet been formalized. This is not the case for complete algorithms which can be described as search-tree exploration strategies. In this work, we propose a formal model for local search, with the aim of providing a common and general framework for the analysis and comparison of local search algorithms. Besides, this model enables to bridge local search and complex networks.

1 Introduction

Local search methods have been proved effective on hard constraint satisfaction (CSP) and combinatorial optimization problems (COP). The literature on this subject is huge and the research in the field is very active. Nevertheless, a formal characterization of such algorithms is still lacking. This contrasts with the theoretical achievements on complete algorithms, such as branch and bound and constraint programming, in which the search process is formalized (usually) in terms of tree search. This formalization has a fundamental importance to understand general advantages and shortcomings of the search algorithms, as well as to compare them. We could say that such a model bridges the gap between theory and practice. So far, local search behavior has been described and studied mainly on the basis of metaphors, such as the (informal) notion of search landscape and peaks, valleys, plateaus, etc. The number of publications concerning formal modeling of local search is extremely small and furthermore it often addresses very specific topics (such as genetic algorithms [8] or SAT [7]).

We believe that a formal modeling of local search would enable us to understand the algorithm behavior on a problem instance, to compare algorithms on a common basis, to have a prediction about the behavior of algorithm variants and to design new effective strategies.

This paper aims at describing a formal model of local search. It also takes inspiration from and introduces some concepts already discussed in [8]. The model is composed of two parts:

- neighborhood graph (defined upon variables, domains, constraints, objective function and neighborhood function)
- search graph (defined upon the neighborhood graph and the search algorithm)

This contribution has the following structure. In Section 2 we define the neighborhood graph and in Section 3 the model is completed with the introduction of the search strategy, leading us to the definition of the search graph. Then, in Section 4, we discuss the main properties and characteristics that can be studied on this model. Finally, we conclude with a brief discussion (Section 5) and we describe some applications in Section 6.

*This note has been written during a visiting period at IRIDIA
Figure 1: Example of undirected graph representing a neighborhood graph (fitness landscape). Each node is associated with a solution $s_i$ and its corresponding objective value $f(s_i)$. Arcs represent transition between states by means of $\varphi$. Undirected arcs correspond to symmetric neighborhood structure.

2 Neighborhood graph

The local search process can be viewed as an exploration of a landscape aimed at finding an optimal solution, or a good solution, i.e., a solution with a quality above a given threshold.\footnote{For the rest of this paper, we will suppose, without loss of generality, that the goal of the search is to find an optimal solution. Indeed, the same conclusions we will draw can be extended to a set including also good solutions.}

A neighborhood graph (NG), also referred to as fitness landscape (FL) is defined by a triple: $\mathcal{L} = (S, \mathcal{N}, f)$, where:

- $S$ is the set of feasible states;\footnote{In the field of metaheuristics, feasible states are also called solutions.}
- $\mathcal{N}$ is the neighborhood function $\mathcal{N} : S \rightarrow 2^S$ that defines the neighborhood structure, by assigning to every $s \in S$ a set of states $\mathcal{N}(s) \subseteq S$.
- $f$ is the objective function $f : S \rightarrow \mathbb{R}^+$

The neighborhood graph can be interpreted as a graph in which nodes are states (labeled with their objective value) and arcs represent the neighborhood relation between states (see Figure 1).

The neighborhood function $\mathcal{N}$ implicitly defines an operator $\varphi$ which takes a state $s_1$ and transforms it into another state $s_2 \in \mathcal{N}(s_1)$. Conversely, given an operator $\varphi$, it is possible to define a neighborhood of a variable $s_1 \in S$:

$$\mathcal{N}_\varphi(s_1) = \{ s_2 \in S \setminus \{s_1\} \mid s_2 \text{ can be obtained by one application of } \varphi \text{ on } s_1 \}$$

Usually, the operator is symmetric: if $s_1$ is a neighbor of $s_2$ then $s_2$ is a neighbor of $s_1$. In a graph representation (like the one depicted in Figure 1) undirected arcs represent symmetric neighborhood structures.

The notion of neighborhood graph enables to view local search algorithms as search processes exploring a graph. The search starts from an initial node and explores the graph moving from a node to one of its neighbors, until a termination condition is met.

There exists another definition of fitness landscape [8], that can also be applied to population-based metaheuristics. This definition can deal with states (nodes of the neighborhood graph)
representing populations of solutions, rather than single solutions. The key point is the introduction of multi-sets, which are sets with possible repetitions of elements. A multi-set substitutes a single solution and the operator transforms a multi-set into another one. Furthermore, the operator is defined as a function $\psi : M(S) \times M(S) \rightarrow [0, 1]$, which assigns a probability for each possible transition between states$^3$. This definition of fitness landscape enables to deal with population heuristics, such as genetic algorithms, in the same way as simple local search algorithms. An important difference between our model and the one proposed in [8] is that we distinguish between the neighborhood structure and the search strategy.

There are some important design issues in developing a search algorithm over a neighborhood graph: the solution representation, the neighborhood structure and the objective function. Furthermore, there are some ways to cope with constraints. For example, it is possible to map a CSP or a COP into a free optimization problem, where there are not constraints and infeasible solutions are penalized by modifying the objective function [6]. The neighborhood graph topology yields a formal representation of the characteristics of the feasible space. For example, side constraints (such as time windows in transportation problems) might disconnect parts of the neighborhood graph and thus affect the search strategy effectiveness.

It is worth underlining that, given an objective function, the choice of an operator determines the properties of the landscape. This is the “One Operator, One Landscape” concept, introduced in [8, 9]. The algorithm performance is strongly affected by the model chosen and, in general, no best choice exists which leads to the best performance with every algorithm/problem combination. This empirical conjecture is theoretically supported by the No Free Lunch Theorem [29].

Besides the topological features, other important characteristics that can be defined on the neighborhood graph are the global and local optima. Indeed, they are defined on the basis of the objective function and the neighborhood. Therefore, they have to be considered as invariants with respect to the search strategy.

3 Local search model

The exploration process of local search methods can be seen as the evolution in (discrete) time of a discrete dynamical system [3, 5]. The algorithm starts from an initial state and describes a trajectory in the state space, that is defined by the neighborhood graph. The system dynamics depends on the strategy used; simple algorithms generate a trajectory composed of two parts: a transient phase followed by an attractor (a fixed point, a cycle or a complex attractor). Algorithms with advanced strategies generate more complex trajectories which can not be subdivided in those two phases.

It is useful to define the search as a walk on the neighborhood graph. In general, the choice of the next state is a function of the search history (i.e., the sequence of the previous visited states) and the iteration step. Formally: $s(t + 1) = \phi(s(0), s(1), \ldots, s(t), t)$, where the function $\phi$ is defined on the basis of the search strategy. $\phi$ could also depend on some parameters and can be either deterministic or stochastic. We also distinguish between Markovian and non-Markovian search strategies. In the first case, the choice of the successor state only depends on the current state, i.e., $s(t + 1) = \phi(s(t), t)$.

For instance, let us consider a deterministic version of the iterative improvement local search. The trajectory starts from a point $s(0)$, exhaustively explores its neighborhood, picks the neighboring state $s'$ with minimal objective function value$^4$ and, if $s'$ is better than $s(0)$, it moves from $s(0)$ to $s'$. Then this process is repeated, until a minimum $\hat{s}$ (either local or global) is found. The trajectory does not move further and we say that the system has reached a fixed point. In this case, given an initial state, there is only one trajectory (i.e., a path along the graph) toward the fixpoint. The set of points from which $\hat{s}$ can be reached is called the basin of attraction of $\hat{s}$. On the other extreme is random walk, in which $\phi$ chooses randomly among the possible successors of

$^3M(S)$ denotes the (finite) set of multisets whose elements are drawn from $S$.

$^4$Ties are broken by enforcing a lexicographic order of states.
Figure 2: Representation of the two extreme cases of a search walk. Left: Representation of a step of deterministic iterative improvement. Note that, since the strategy is deterministic, the probabilities are all 0 but for one node. Right: Representation of a random walk step. The search is not biased, therefore every possible successor has the same probability to be chosen.

Figure 3: Representation of a generic search step: A distribution probability characterizes the choice of the successor node. \((p_1 + p_2 + p_3 + p_4 = 1)\) The distribution can be iteration dependent and function of the entire search history.

A state. The difference between the two extreme cases is graphically represented in Figure 2. In general, a search step can be represented as depicted in Figure 3.

Once we have introduced also the search strategy, the edges of the graph can be oriented and labeled with transition probabilities (whenever it is possible to evaluate them). This will lead to the definition of concepts such as basins of attraction, state reachability and graph navigation. In the following, the resulting graph will be referred to as search graph.

We would like to remark that, while the neighborhood graph topology is only dependent on the neighborhood structure and the problem model, the basins of attraction and other related concepts depend also on the particular algorithm used.

3.1 Example

To clarify the concepts introduced in the previous sections, we discuss the model of a very simple problem defined over boolean variables.

Let us consider a problem defined over three boolean variables \(x_1, x_2, x_3\). All the eight possible assignments are feasible and the objective function (to be minimized) is defined extensively in Table 1.

In this example, we use the neighborhood defined on unitary Hamming distance, that is the most used neighborhood for binary variables. Since three are the possible flips of any assignment, each node of the graph is connected with three other nodes. It is important to observe that this neighborhood structure generates a graph with a uniform degree. (The degree of a node is the number of edges connected to it.) Also note that the neighborhood structure defines only the graph topology and does not specify the criteria upon which a particular neighbor is chosen. These criteria are part of the search strategy. The neighborhood graph corresponding to this landscape is depicted in Figure 4. We observe that the arcs are not oriented, since the neighborhood is sym-
Table 1: Objective function values corresponding to the eight feasible states of the problem.

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

Figure 4: Neighborhood graph corresponding to a problem defined over three boolean variables. Nodes are labeled with the corresponding objective value. All the states are feasible.

Moreover, nodes are labeled with the corresponding objective function. The neighborhood graph has one global optimum (010) and two local optima (001 and 100).

The neighborhood graph just defined can be explored with different strategies. This exploration generates a second graph in which arcs can be oriented and be associated with a transition probability (i.e., the probability to move from a node to a neighboring one). For simplicity, we just delete arcs associated with a transition probability equal to zero. Self-arcs characterize fixpoint nodes.

Let us start with the simplest search strategy: Deterministic iterative improvement (DII). This algorithm exhaustively explores the neighborhood of a node and selects the neighbor with minimum objective function. If this value is less than the objective value of the incumbent node, then the selected neighbor becomes the new current node, otherwise the search stops. In this algorithm, we break ties by imposing a lexicographic order among variables. Namely, nodes are ordered by binary coding from 000 to 111 (we call this algorithm DII-lex). The resulting graph is drawn in Figure 5. Observe that, since the algorithm is deterministic, all arcs have probabilities of value 1. Also, note that the graph is not connected and it is composed of three disconnected subgraphs, each built around a local optimum. Indeed, once an initial state is chosen, there is only one path leading to a fixpoint that can be either a global or a local optimum. The global optimum can be reached from four states out of seven, one local optimum is isolated and the other one is reached from only one state.

We now consider a simple randomized version of iterative improvement, in which ties are
broken randomly (RII). The corresponding graph is depicted in Figure 6. The main difference
between the previous case is that the introduction of randomness generates nodes with more than
one outgoing arc, each associated to a transition probability.\footnote{For each node, the sum of the transition probability values of the outgoing arcs equals 1.} For instance, from node 111 it is
possible to reach both node 110 and node 101 with probability 0.5. From node 110 the optimal
solution can be reached, but if the move toward node 101 is chosen, then the search will stop on
a local optimum (100 or 001).

Since the neighborhood graph is completely expanded, it is also possible to calculate the
probability of reaching node \( j \) from node \( i \). For instance, node 111 can reach the global optimum
010 with probability \( \text{prob}(111 \rightarrow 010)=\frac{1}{2} \times 1 = \frac{1}{2} \). Moreover, we have \( \text{prob}(111 \rightarrow 001)=\frac{1}{2} \times \frac{1}{2} = \frac{1}{4} \)
and \( \text{prob}(111 \rightarrow 100)=\frac{1}{2} \times \frac{1}{2} = \frac{1}{4} \). Finally, we observe that there is only one path connecting
any pair of nodes. Such a property could not be valid when different algorithms are adopted to
explore the neighborhood graph, as happens when applying iterative improvement with random
choice of any better neighbor (RII-any), depicted in Figure 7. In this case, we have: \( \text{prob}(111 \rightarrow 010)=\frac{1}{4} \), \( \text{prob}(111 \rightarrow 001)=\frac{1}{4} \) and \( \text{prob}(111 \rightarrow 100)=\frac{1}{4} \).

We consider now the application of a more complex search strategy: Simulated annealing
(SA). Simulated annealing [10] is a stochastic local search that allows moves resulting in states
of worse quality than the incumbent one, with the aim of escaping from local minima. The
probability of doing such a move is decreased during the search. In essence, SA randomly samples
the neighborhood of the current state \( s \) and moves to the picked neighbor \( s' \) if \( f(s') \leq f(s) \)
or, if \( f(s') > f(s) \), with probability equal to \( \exp(-\frac{f(s')-f(s)}{T}) \). \( T \) is the so-called temperature
parameter that is decreased during the search. This process is analogous to the annealing process
of metals and glass, which assume a low energy configuration when cooled with an appropriate
cooling schedule. Regarding the search process, this means that the algorithm is the result of two
combined strategies: random walk and iterative improvement. In the first phase of the search,
the bias toward improvements is low and it enables the exploration of the search space; this
erratic component is slowly decreased thus leading the search to converge to a (local) minimum.
The application of SA to our example generates a quite complex search graph with loops. In
Figure 8, we drew the search graph corresponding to \( T \) equal to 10, which can be considered a
high temperature for the instance at hand. By evaluating the transition probabilities for decreasing
values of \( T \), we observe that the probabilities converge to the ones of RII-any. Indeed, they are
the same for \( T = 0.001 \).

Since the temperature is quite high, many moves toward worse neighbors are allowed and are
associated to not negligible probabilities. We can observe that for states surrounded by better
Figure 6: Graph resulting from the application of randomized iterative improvement (RII).

Figure 7: Graph resulting from the application of randomized iterative improvement with random choice among better neighbors (RII-any).
neighbors (such as state 000) the transition probabilities have all the same value, equal to $\frac{1}{3}$. For the global optimum 010, the probability of self-loop is higher than the probability of moving to any neighbor. The case of SA is more complex than the previous ones, not just because of bi-directional transitions between states, but also because, in general, the transition probabilities are function of the iteration step.

This simple example describes the main characteristics of the proposed model of local search. In the next sections we will define and discuss some graph properties that are relevant for analyzing the search behavior.

4 Properties of interest

Once the neighborhood graph and the search graph are defined, some properties are particularly relevant for the analysis of local search behavior. Among such properties we consider topological characteristics of the graph and the basins of attraction of local optima. The topological properties can refer either to the neighborhood graph defined upon the neighborhood structure or the graph derived by the application of a particular search strategy to the neighborhood graph (the search graph). Both the cases are extremely relevant for the analysis of local search behavior. The first describes properties that emerge from the problem model and the neighborhood function chosen (i.e., the representation chosen) and are therefore invariant to the search strategy. The second case refers, instead, to properties that also depend on the search strategy.

4.1 Topology

The topology can be studied by means of techniques stemming from graph theory and complex networks [26, 13, 1, 25, 24, 4, 2]. The neighborhood graph topology gives an indication of the ‘navigability’ of the graph. For instance, a regular topology such as the hypercube indicates that the graph is connected, therefore there is a path between any node and any global optimum. (Whether this property is exploited by the search strategy is another issue) More important, given this structure of the graph, there are many alternative paths connecting any pair of nodes. Such a
regular and redundant structure can be particularly suitable for local search. A disconnected graph could be particularly difficult to explore, since a mechanism to move the search trajectory between disconnected regions should be provided. Irregular graph topologies, such as scale-free and small-world graphs [25, 4], can strongly affect the graph exploration process. For example, nodes with high degree (so-called hubs) may play a key role for the reachability of optimal solutions. Hence, it is clear that the properties of the neighborhood structure can be characterized by studying the neighborhood graph topology. Results and tools developed in the field of complex networks can be usefully applied for the analysis of topological properties of the search and search graphs.

4.1.1 Example

Let us consider the example of Section 3.1 and add the following constraint to the problem: \( x_1 \lor (x_2 \oplus x_3) \). This constraint can be regarded as a side constraint that forbids a specific configuration of variable values. Indeed, it makes the assignments 000 and 011 infeasible. The resulting neighborhood graph is represented in Figure 9. The optimal solution can now be reached only through state 110, since the other paths have been cut.

The issue of the effect of constraints on the connectivity, and in general on the topology, of the neighborhood graph is quite complex. Some of the main questions arising are: Are there families of constraints (specific classes of side constraints, implied constraints, etc.) that negatively/positively affect the graph topology? Can we define and study the graph robustness against constraints?

4.2 Basins of attraction

The concept of basin of attraction (BOA) has been introduced in the context of dynamical systems, in which it is defined referring to an attractor. Concerning our model of local search, we will use the concept of basin of attraction of any node of the search graph. Moreover, for this definition to be valid for any state of the search graph, we have to relax the requirement that the goal state is an attractor. Therefore, the basin of attraction will also depend on the particular termination condition of the algorithm. In the following examples, we will suppose to apply a termination condition such that the algorithm is stopped as soon as a stagnation condition is detected, that is when no improvements to the solutions are found after a maximum number of steps. This termination condition is the closest to the concept of steady state in dynamical systems. We will

---

\(^6^{\text{Alternatively, a solution construction mechanism is required such that the search starts in a subgraph containing an optimal solution.}}\)
Initially consider the case of deterministic systems, then we extend the definition to stochastic systems.

**Definition** Given a deterministic algorithm \( \mathcal{A} \), the basin of attraction \( \mathcal{B}(\mathcal{A}|s) \) of a point \( s \), is defined as the set of states that, taken as initial states, give origin to trajectories that include point \( s \). The cardinality of a basin of attraction represents its size (in this context, we always deal with finite spaces).

Given the set \( S^* \) of the global optima, the union of the BOA of global optima \( I^* = \bigcup_{i \in S^*} \mathcal{B}(\mathcal{A}|i) \) represents the set of desirable initial states of the search. Indeed, a search starting from \( s \in I^* \) will eventually find an optimal solution. Since it is usually not possible to construct an initial solution that is guaranteed to be in \( I^* \), the ratio \( |I^*|/|S| \) can be taken as an indicator of the probability to find an optimal solution. On the extreme case, if we start from a random solution, the probability to find a global optimum is exactly \( |I^*|/|S| \). Therefore, the higher this ratio, the higher the probability of success of the algorithm.

In the case of stochastic local search, we may define a probabilistic basin of attraction, as a generalization of the previous case.

**Definition** Given a (stochastic) algorithm \( \mathcal{A} \), the basin of attraction \( \mathcal{B}(\mathcal{A}|s; p^*) \) of a point \( s \), is defined as the set of states that, taken as initial states, give origin to trajectories that include point \( s \) with probability \( p \geq p^* \). Also in this case, we define the union of the BOA of global optima: \( I^*(p) = \bigcup_{i \in S^*} \mathcal{B}(\mathcal{A}|i; p) \). For simplicity, in the following we will write \( \mathcal{B}(s; p^*) \) instead of \( \mathcal{B}(\mathcal{A}|s; p^*) \) when the algorithm involved is clear from the context.

This definition includes the previous one as a special case. Indeed, if \( p^* = 1 \) we are interested in finding the states generating trajectories that will eventually reach \( s \). It is also important to note that if \( p_1 > p_2 \), then \( \mathcal{B}(s; p_1) \subseteq \mathcal{B}(s; p_2) \).

Given a local search algorithm \( \mathcal{A} \), the topology and structure of the search landscape determine the effectiveness of \( \mathcal{A} \). In particular, the reachability of optimal solutions is the key issue. Therefore, the characteristics of the BOA of optimal solutions are of dramatic importance. Our definition of basins of attraction enables both a complete and analytical study —when probabilities can be computed on the basis of the search strategy description— and statistical analysis (e.g., by sampling).

### 4.2.1 Example

We consider again the example discussed in Section 3.1 and we study the characteristics of the global optimum basins of attraction, when the three iterative improvement versions are applied. In the case of DII-lex, only the deterministic BOA can be considered: \( \mathcal{B}(\text{DII-lex}|010) = \{000, 011, 110, 111\} \). The size of this BOA is 4, therefore this case can be considered quite favorable for DII-lex starting from a random solution, since the probability of reaching the optimal solution is \( \frac{4}{16} = 0.25 \). (We do not consider the number of states that have to be visited to reach the optimal solution)

In the case of RII, a different picture emerges. Indeed, as the algorithm is stochastic, probabilistic BOA have to be considered. The smallest BOA is the one corresponding to probability equal to 1: \( \mathcal{B}(\text{RII}|010; 1) = \{000, 011, 110\} \). Then, we consider the BOA of probability equal to 0.5 \( \mathcal{B}(\text{RII}|010; 0.5) = \{000, 011, 110, 111\} \). It is important to observe that \( \mathcal{B}(\text{RII}|010; p) = \mathcal{B}(\text{RII}|010; 0.5) \) for \( p \leq 0.5 \). Therefore, even if we reduce the probability of reaching the global optimum, no other initial state can lead to it except for the ones belonging to \( \mathcal{B}(\text{RII}|010; 0.5) \). We conclude this analysis by considering the ‘any-better’ version of RII. The basins of attraction for RII-any are the following: \( \mathcal{B}(\text{RII-any}|010; 1) = \emptyset, \mathcal{B}(\text{RII-any}|010; 0.5) = \{110\}, \mathcal{B}(\text{RII-any}|010; 0.25) = \{000, 011, 100, 110, 111\} \). The last BOA is the one with the maximal size, since no values for \( p^* \)

---

\(^7\)We always suppose that the goal state is part of its own basin of attraction.
exist such that the BOA obtained is larger than $B(\text{RII-}y|010; 0.25)$.

It is interesting to study the effect of the side constraint added to the original problem on the search graphs considered in Section 3.1. The constraint makes the states 000 and 011 infeasible, which are then removed from the search and search graph. The corresponding graphs are depicted in Figures 10, 11 and 12. In general, constraints could play positive or negative role on search, depending on their impact on topology and basins of attraction (and also other search graph properties). The issue is, trivially, to control this effect with the aim of enabling the search to reach optimal solutions with high probability and efficiently.

5 Discussion

The model we propose is aimed at providing a formal and general framework in which local search algorithms can be compared and their behavior studied, in analogy with complete methods that are modeled as search-tree exploration processes. In [8], a very similar model — the so called fitness landscape — has been proposed, along with an algorithm for reverse hillclimbing that returns the
Figure 12: Graph resulting from the application of randomized iterative improvement with random choice among better neighbors (RII-any).

basins of attraction of local optima when hillclimbing algorithms are applied. Also the discussion in [8] is of particular interest for the topic of this note. However, the fitness landscape model does not explicitly separate the neighborhood graph from the search graph.

We believe that the conceptual separation between the neighborhood graph and the search graph is particularly important for the analysis of algorithm behavior. In fact, from the neighborhood graph characteristics it is possible to extract pieces of information which are algorithm invariant, such as the density of global and local optima and the topological properties of the graph. The latter have impact on the reachability of optimal solutions. The actual search space explored by the algorithm is, instead, the search graph, in which basins of attraction of global optima and other topological characteristics can be analyzed.

We imagine two scenarios for the application of the model. One concerns the complete enumeration of the neighborhood graph and thus can be applicable only to small-size instances. The second scenario, which is complementary to the first, deals with the sampling of the neighborhood/search graph. In both the cases, there is a lot of room for new studies and application of knowledge stemming from different fields, such as complex networks and sampling. Techniques from statistics, such as Markov chains and estimation of rare events, can also be fruitfully applied.

6 Applications

In this section we outline some applications of the proposed model. Some of them are still conjectured. The list has not to be considered exhaustive.

6.1 Symmetry-breaking and local search

The model of neighborhood and search graph has been applied to study the effect of symmetry-breaking constraints on local search in [22]. It has been shown that a particular class of symmetry-breaking constraints (namely, permutations over variables) seems not to perturb the topology of the neighborhood graph. Therefore, if the effect of such constraints penalizes local search, the reason has to be found in the search strategy (i.e., in the properties of search graph). Indeed, by applying a simple iterative improvement search, it has been observed that the total basin of attraction of global optima is reduced in the search graph induced by the model with symmetry-breaking constraints. The reduction factor is bigger than that of search space reduction, therefore iterative improvement has a lower probability of finding an optimal solution, despite the search
space reduction. This work is still preliminary and it is part of an ongoing research and further investigations are needed for the assessment of the results.

6.2 Instance vs. neighborhood/search graph structure

The formalization of search space in terms of neighborhood and search graph can be effectively applied to relate instance structure and the search space explored by local search. Indeed, it has been found that some structural properties of the instance have direct correlation with some local search behavioral characteristics. Nevertheless, other structural properties seem to have less impact on local search than on systematic solvers. Some studies in this direction can be found in [21, 20, 17, 18, 23, 16, 11, 19, 14]. Some other important topics, such as backbones [12], backdoors [28] and solution sampling [27] could be tackled and investigated by using this model of local search.

6.3 A bridge between complex networks and local search

Our model may enable an interesting connection between complex networks and local search. The literature on complex networks [2, 24] is a rich source of results that could be fruitfully applied also in the field of local search algorithms. Besides studies on network connectivity, some results and algorithms on evolution and navigation of complex networks could be transferred to local search. Moreover, some technical tools commonly used in graph analysis, such as the analysis of spectral properties of adjacency matrices, could be useful also for the analysis of neighborhood/search graphs.

6.4 Neighborhoods and constraints

A very interesting topic is the classification of neighborhoods on the basis of the topological properties they induce on the neighborhood graph. Moreover, also the effect of constraints can be analyzed with the same approach. This is particularly interesting since it enables us to view neighborhoods and constraints from a unitary point of view. Some interesting results concerning the use of propagation for the generation of neighborhoods [15] can be seen as an example of this approach.

Acknowledgments

I thank Marco Dorigo for useful comments and suggestions on this work.

References


