On the impact of small-world on local search

Andrea Roli

Dipartimento di Scienze
Università degli Studi "G. D'Annunzio"
Chieti–Pescara (Italia)
a.roli@unich.it

Abstract. The impact of problem structure on search is a relevant issue in artificial intelligence and related areas. Among the possible approaches to analyze problem structure, the one referring to constraint graph enables to relate graph parameters and characteristics with search algorithm behavior. In this work, we investigate the behavior of local search applied to SAT instances associated to graphs with small-world topology. Small-world graphs, such as friendship networks, have low characteristic path length and high clustering. In this work, we first present a procedure to generate SAT instances characterized by an interaction graph with a small-world topology. Then we show experimental results concerning the behavior of local search algorithms applied to this benchmark.

1 Introduction

The impact of problem structure on search is a relevant issue in artificial intelligence and related areas. In order to design and tune effective and efficient algorithms for constraint satisfaction problems (CSPs) and constrained optimization problems (COPs), the relations between structural problem instance features and algorithm performance have to be investigated. These relations have been studied from different perspectives. In particular, search algorithm behavior w.r.t. graph properties of some constraint satisfaction problems and constrained optimization problems has been discussed in [22, 21, 15, 18]. The definition of structure emerging from the literature on CSPs and COPs is usually based on the informal notion of a property enjoyed by non-random problems. Thus, *structured* is used to indicate that the instance is derived from a real-world problem or it is an instance generated with some similarity with a real-world problem. Commonly, we attribute the characteristic of structured to a problem that shows, at a given level of abstraction, regularities such as well defined subproblems, patterns or correlations among problem variables. In this work, we focus on one among the possible ways of characterizing the structure of a problem instance: We analyze the structure of links among its components, i.e., the network that connects the components. Some problems suggest a natural structural description, since they have a representation that can be directly used for structure analysis. A classical example are problems defined on graphs, such as the Graph Coloring Problem and the k-Cardinality Tree Problem. For CSPs, an interaction graph can be defined [11], in which nodes correspond to variables and edges connect two variables if there exists a constraint involving them[1]. Hence, the

[1] This graph is equivalent to the constraint graph in the case of binary constraints.

structure of any CSP can be characterized by a graph. Relevant features of a graph that can affect search behavior are, for example, the *average node degree* and its frequency, the *path length* and the *clustering*. The impact of node degree frequency on search has been studied in [22, 12, 17, 13]. In this work, we investigate the relations between the *small-world* property and search algorithm performance. Small-world graphs [23, 24] are characterized by the simultaneous presence of two properties: the average number of hops connecting any pair of nodes is low and the clustering is high. Social networks defined on the basis of friendship relationships are a typical example of graphs with a small-world topology. The impact of small-world topology on search problems (e.g., Graph Coloring Problem) has been discussed in [21], where it is shown that many CSPs and COPs have a small-world topology and the search cost can be characterized by a heavy-tail distribution [4].

In this work, we report experimental results concerning the behavior of local search applied to instances of the Satisfiability Problem (SAT) with an interaction graph characterized by a small-world topology. The aim of these experiments is to address the question whether small-world SAT instances are harder to solve than others and if this behavior is common across different local search algorithms.

The contribution of this work is twofold. First, we define a procedure to construct SAT instances with a lattice structure, along with a method to generate small-world SAT instances. Then, we test three different local search algorithms on the generated benchmark. Results show that the behavior strongly differentiates across the algorithms. In some cases, results show that many harder instances have a small-world structure. This empirical analysis shows that, even if local search can be affected by instance structure, an important role is played by the actual search space exploration strategy.

This paper is structured as follows. Sec.2 introduces the basic concepts of small-world graphs and graphs associated to SAT instances. In Sec.3, we describe the properties of the instances composing the testbed and the procedure used to generate them. Sec.4 presents experimental results obtained by applying three different local search algorithms, namely WalkSAT, GSAT and Iterated local search. We conclude by briefly discussing the results obtained and outlining future work.

2 Preliminaries

In this section, we succinctly introduce small-world graphs and the graph associated to SAT instances.

Given a graph $G = (V, E)$, where V is the set of nodes and E the set of edges, the *characteristic path length* $L(G)$ of G is formally defined as the median of the means of the shortest paths connecting each node $v \in V$ to all other nodes. The *clustering* coefficient is defined on the basis of the notion of neighborhood. The neighborhood Γ_v of a node $v \in V$ is the subgraph consisting of the nodes adjacent to v (not including v itself). The clustering of a neighborhood is defined as $\gamma_v = |E(\Gamma_v)| / \binom{k_v}{2}$, where $|E(\Gamma_v)|$ is the number of edges in Γ_v and k_v is the number of neighbors of v. Therefore, γ_v is the ratio between the number of edges of the neighborhood and the maximum number of edges it can have. The clustering coefficient γ of a graph G is defined as the average of the clustering values γ_v for all $v \in V$. For example, we compute L and γ for the

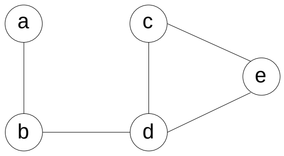

Fig. 1. Constraint graph associated to the SAT instance $(a \vee \neg b) \wedge (b \vee d) \wedge (c \vee \neg d \vee \neg e)$

graph depicted in Fig.1. The characteristic path length is the median of the average path lengths related to the nodes a, b, c, d and e, i.e., $L = \text{median}\{\frac{9}{4}, \frac{6}{4}, \frac{7}{4}, \frac{5}{4}, \frac{7}{4}\} = \frac{7}{4} = 1.75$. The clustering γ is the average of the neighborhood clustering values, i.e., $\gamma = \frac{1}{5}(0/\binom{1}{2} + 0/\binom{2}{2} + 1/\binom{2}{2} + 1/\binom{3}{2} + 1/\binom{2}{2}) \approx 0.467$.

Typically, random graphs are characterized by low characteristic path length and low clustering, whilst regular graphs (such as lattices) have high values for L and γ. Conversely, small-world graphs are characterized by low L and high γ.

In this paper, we apply the notion of small-world to a graph associated to SAT instances. SAT belongs to the class of NP-complete problems [9] and can be stated as follows: given a set of clauses, each of which is the logical disjunction of $k > 2$ *literals* (a literal is a variable or its negation), we ask whether an assignment to the variables exists that satisfies all the clauses. The graph we associate to a SAT instance is called the *interaction graph* [11] and it is defined as an undirected graph $\mathcal{G} = (\mathcal{V}, A)$, where each node $v_i \in \mathcal{V}$ corresponds to a variable and edge $(v_i, v_j) \in A$ $(i \neq j)$ if and only if variables v_i and v_j appear in a same clause (see Fig.1). Observe that the same graph corresponds to more than one formula, since nodes are connected by one arc even if the corresponding variables belong to more than one clause. Having a set of clauses associated to the same graph, makes this representation quite rough. Nevertheless, in the following, it will be shown that some properties of this graph can strongly affect the behavior of local search.

3 Small-world SAT instances

In order to explore the behavior of search algorithms on small-world SAT instances, we generated a benchmark by morphing between instances constructed on lattice graphs and random instances. The core idea of the morphing procedure is derived from [3], wherein a method that enables to generate instances gradually morphing from a source to a destination instance is presented. This procedure is also quite similar to the one used in [24] to generate small-world graphs by interpolating between lattice and random graphs (see Fig.2). Starting from a lattice graph, links are randomly removed and rewired. Small-world graphs can be obtained by randomly rewiring just a few links between nodes.

SAT instances with a small-world graph topology can be obtained by morphing between a SAT instance associated to a lattice graph and a random SAT instance. Therefore, we have first to define a procedure to construct SAT instances associated to a lattice graph. Instead of starting from a SAT formula, expressed as a conjunction of

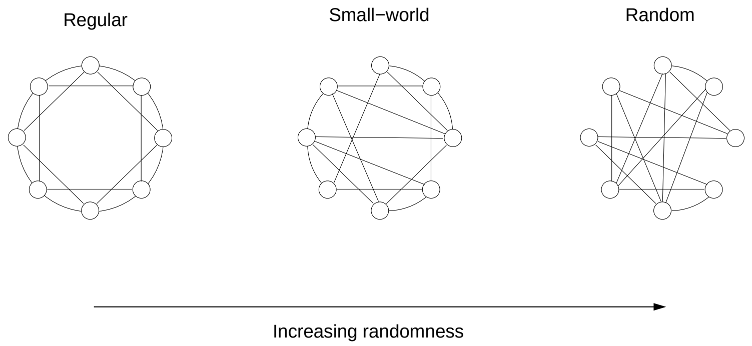

Fig. 2. Morphing between a lattice graph and a random graph. Small-world graphs can be obtained by randomly rewiring just a few links between nodes.

clauses, we start from a graph with the desired topology and we use it as a skeleton for generating a SAT formula. The starting graph is a *lattice graph*. Lattice graphs have a very regular topology and every node is connected to a fixed (usually quite small) number of neighbors. Examples of lattice graphs are ring lattices (also called cycles) with adjunctive links connecting neighbors and hypercubes. Once obtained the graph with the given topology, we have to assign variables to nodes and to generate the clauses of the formula. The first step can be completed very easily by assigning variables in order: variable x_i is assigned to node i, for $i = 1,\ldots,n$. The generation of clauses, i.e., of a formula that can be mapped into the given lattice graph, is a bit more complex. First of all, we remind that the graph associated to a SAT instance, as previously defined, corresponds to a set of SAT instances. Therefore, it is important to define a given structure for the formula. Our choice is to follow the usual experimental settings for random generated SAT instances: 3-SAT formulas with controlled ratio m/n, where n is the number of variables and m the number of clauses.

In the following, we describe the algorithm to generate 3-SAT instances with given ratio m/n on a lattice graph. The generalization of the algorithm to k-SAT instances is straightforward. The high level algorithm is described in Alg.1. The algorithm is structured in two phases. In the first phase, a minimal set of clauses is generated to obtain a formula that can be represented by the given lattice graph. In the second phase, the additional required number of clauses is generated by adding clauses randomly chosen from the first set and by randomly changing the sign of literals.

In the first phase, clauses of three literals are constructed, by taking in turn each variable as a *pivot* and adding two subsequent variables (see Fig.3 and Fig.4). In order to avoid repetitions of clauses, for every variable x_i only subsequent variables $x_j, j > i$ (modulo n) are considered. Indeed, given the symmetry of the graph, the clauses involving the symmetric part of neighbors will be generated by using those neighbors as pivot (see Fig.5 and Fig.6). The instances composing the benchmark are generated by morphing between a lattice SAT instance and a random one. Each instance is obtained by taking from the lattice SAT instance all the clauses except for a prefixed number which

Algorithm 1 Generation of a 3-SAT instance on a lattice graph

INPUT: n, m, λ {λ is the number of neighbors, supposed even}
OUTPUT: 3-SAT formula $\Phi = \{C_1, \ldots, C_m\}$ with n variables and m clauses associated to a lattice graph with n nodes with λ neighbors each.

Build a lattice graph $G(n, \lambda)$ (on a circle) with n nodes with λ neighbors each;
Assign variables (clockwise) to nodes;
$\Phi \leftarrow \emptyset$
for $i = 1$ to $n-1$ **do**
  The neighbors of x_i are $\mathcal{N}^+ = \{x_{i+1}, \ldots, x_{i+\lambda/2}\}$ (mod n) and $\mathcal{N}^- = \{x_{i-1}, \ldots, x_{i-\lambda/2}\}$ (mod n);
  for each pair x_j, x_{j+1} in $\mathcal{N}^+$ **do**
    Construct the clause $C = x_i \vee x_j \vee x_{j+1}$
    Negate each variable in C with probability 0.5;
    $\Phi \leftarrow \Phi \cup C$
  end for
end for
{Now the number of clauses is $|\Phi| = n(\lambda/2 - 1)$ }
while $|\Phi| < m$ **do**
  repeat
    Pick randomly a clause C' in Φ;
    Negate each variable in C' with probability 0.5;
  until a new clause C' is generated
  $\Phi \leftarrow \Phi \cup C'$
end while

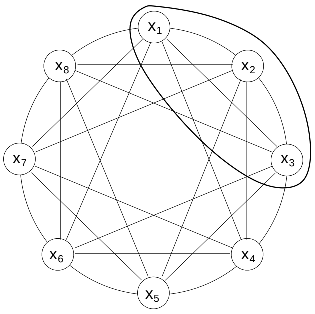

Fig. 3. Construction of the first clause involving variable x_1.

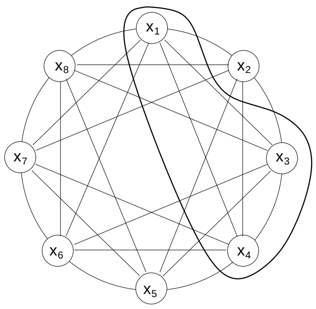

Fig. 4. Construction of the second clause involving variable x_1.

are randomly chosen from a random SAT instance with the same number of variables and clauses. This procedure is indeed very similar to the morphing procedure described in [3], but in this case we control the exact number of clauses taken from the destination instance. In this way it is possible to smoothly interpolate from lattice to random and observe the arising of small-world properties in SAT instances.

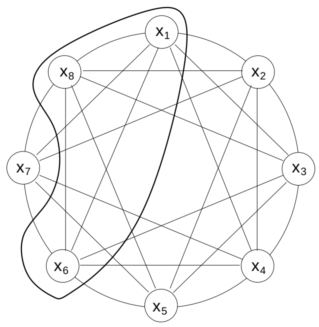

Fig. 5. Construction of the third clause involving variable x_1. The pivot is variable x_6.

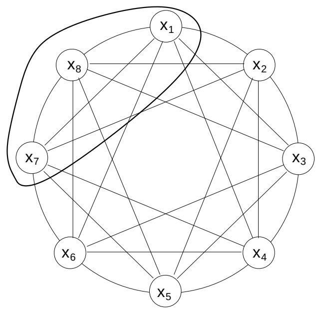

Fig. 6. Construction of the fourth clause involving variable x_1. The pivot is variable x_7.

In order to have a quantitative measure of the small-world characteristic, we introduce the *proximity ratio* μ [21], defined as the ratio between clustering and characteristic path length, normalized with the same ratio corresponding to a random graph, i.e., $\mu = (\gamma/L)/(\gamma_{rand}/L_{rand})$. In Fig.7, the clustering and the characteristic path length of SAT instances gradually interpolating from lattice to random are plotted (in semi-log scale). We observe that L drops very rapidly with the introduction of clauses from the random instance. Conversely, γ maintains a relatively high value for a larger amount of perturbation. The instances with low length and high clustering are characterized by the small-world property. This is also indicated by the proximity ratio curve, which approximately assumes its maximum in that region.

We generated four sets of instances (respectively with 100, 200, 500 and 800 variables), each obtained by morphing between a lattice 3-SAT and a random 3-SAT with same number of variables and clauses. All the generated instances are satisfiable (unsatisfiable instances have been filtered by means of a complete solver). The ratio between the number of clauses and the number of variables is 3, lower than the so-called critical ratio (which is close to 4.3 for 3-SAT instances [1, 8, 5]). This is due to the structure of lattice SAT instances which turned out to be almost all unsatisfiable at the critical ratio. In Fig.8, the proximity ratio of the instances composing the benchmark is plotted. The value μ ranges approximately from 0.5 to 10. We can observe the typical behavior of instances interpolating between regular and random instances. In the next section we present experimental results on the behavior of local search algorithms on the benchmark defined.

4 Experimental Results

Some constraint satisfaction problems and constrained optimization problems with small-world topology have been found to require a higher computational search cost with respect to "non small-world" ones [21, 22]. Since those results only concern complete algorithms, we question whether this behavior could also be observed in the case of approximate algorithms, namely local search.

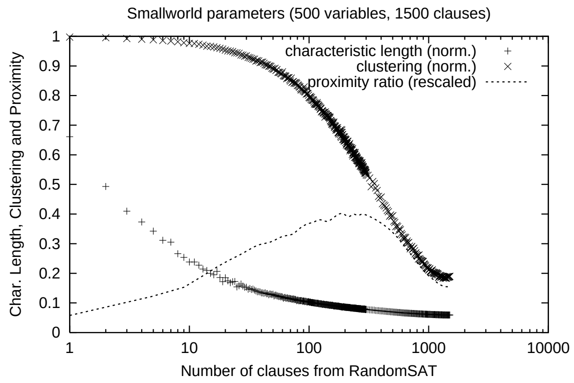

Fig. 7. Characteristic path length L, clustering γ and proximity ratio μ for instances generated by morphing from a lattice SAT instance to a random SAT instance of 500 variables and 1500 clauses.

We performed a series of experiments aimed at checking whether small-world SAT instances are harder to solve than both regular (lattice) and random ones. In the following, we will use the notion of *hardness* referred to the algorithm at hand. We estimate the hardness by means of the search cost, namely the number of iterations required for the algorithm to find a satisfying assignment. Since the algorithms we deal with are stochastic, we run each of them 1000 times on the same instance and we took the median value. We emphasize that we use the concept of hardness referring to a given algorithm $\mathcal{A}$ and we say that an instance I_1 is harder than I_2 if the search cost (as defined above) for solving I_1 via $\mathcal{A}$ is higher than that of I_2. Even if this definition of hardness is grounded to the algorithm used, in general it is possible to observe that a class of instances is harder than another class for a set of algorithms. This case reveals that there is a characteristic of the class that makes the instances difficult for all the considered algorithms.

We applied three different local search procedures, that are based on different heuristic strategies. The algorithms we considered are WalkSAT [19], GSAT [20] and Iterated local search (ILS, [7, 14]). GSAT was the first effective local search algorithm proposed for SAT. It applies a greedy strategy, by flipping the variable that, if flipped, leads to the greatest gradient in the number of satisfied clauses. GSAT suffers from being frequently trapped in confined areas of the search space, therefore its performance is often not satisfactory for large instances. WalkSAT is based on the principle of repair: It randomly chooses one unsatisfied clause and flips one variable within it. There are some different heuristics for the choice of the variable to flip [6]. In our implementation, we applied a GSAT-like heuristic, i.e., the variable that produces the largest increment in the number of satisfied clauses is flipped (no random walk is performed). WalkSAT has usually a far better performance than GSAT. Nevertheless, both algorithms lack a global strategy that could guide them during the exploration of the search space. Algorithms equipped with such a strategy are commonly called metaheuristics [2]. In order

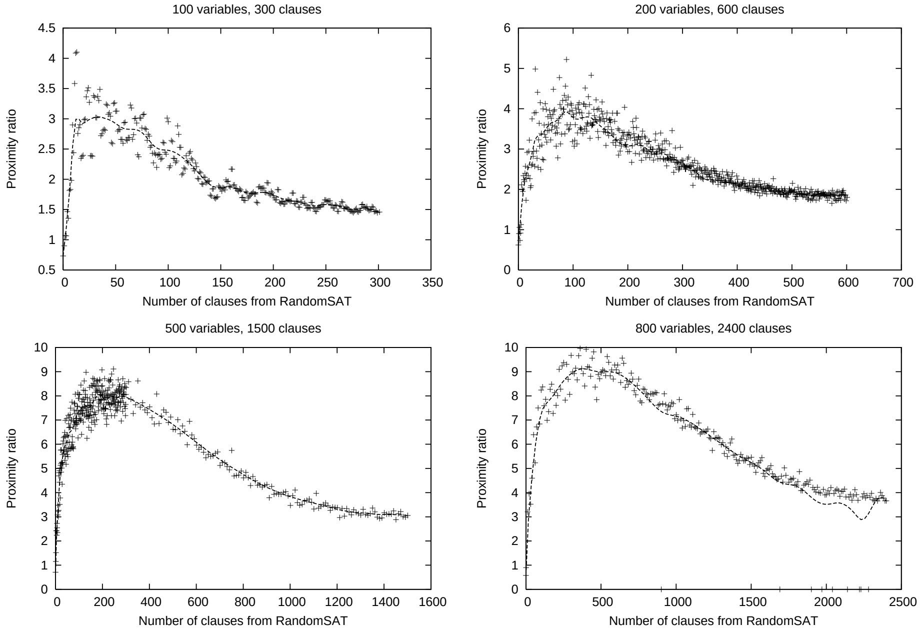

Fig. 8. Proximity ratio of the instances composing the benchmark.

to extend the diversity of the techniques compared, we applied also an ILS designed to attack SAT and MAXSAT problems [14, 15]. In essence, this metaheuristic is a tabu search-based WalkSAT guided by a strategy that tunes both the tabu tenure and the intensification/diversification balance by using the search history.

Before describing the results concerning local search, it is important to point out that in [15, 16] a complete algorithm has been applied to this benchmark and results strikingly show that small-world SAT instances are the hardest (i.e., the search cost, evaluated as number of variable assignments performed by the algorithm before solving the instance, is the highest).

Results are shown in Figs. 9, 10 and 11. In the plots, we reported for each algorithm the median iterations (over 1000 runs) on every instance. The algorithms run until a feasible solution was found[2]. Results are very interesting and show the complexity of empirical analysis of local search behavior. First of all, we note that the behavior across the three algorithms is very different. In the first two plots of Fig.9, we observe that some of the hardest instances for WalkSAT are located in the small-world area. Nevertheless, for the instances of size 500 and 800, the search cost regularly increases –linearly in semi-log scale– while morphing from lattice to random. This peculiar behavior requires a deeper investigation and from these preliminary results we can only conjecture that

[2] In the case of GSAT, due to the extremely high execution time, we stopped the algorithm at a maximum number of non-improving moves and we reported the success ratio, i.e., the number of successful runs out of 1000. Thus, in this case, the lower the value, the harder the instance.

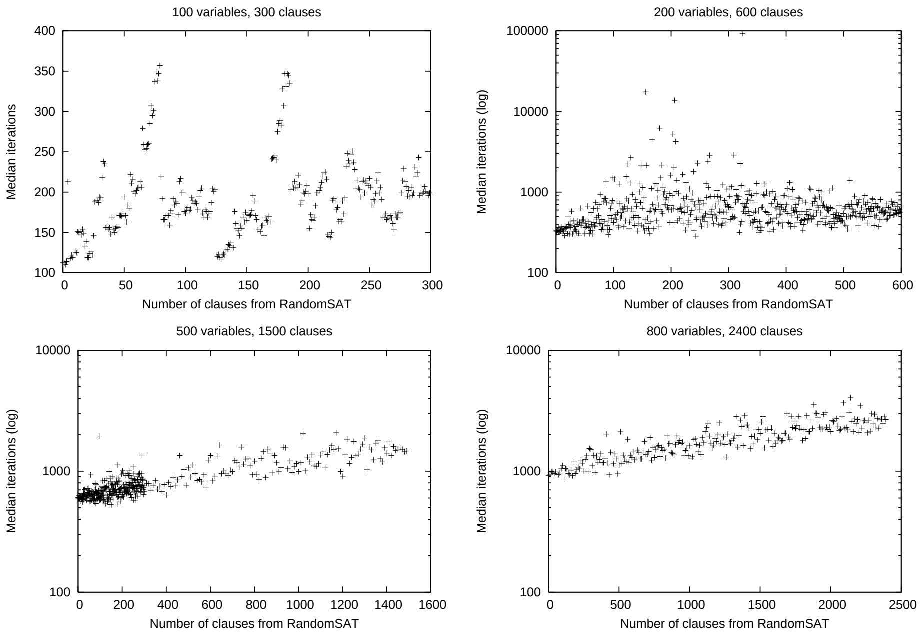

Fig. 9. Search cost of WalkSAT across the instances, from lattice to random structure. Points represent median iterations over 1000 runs. Log-scale on the y-axis has been used when necessary.

size scaling amplifies a characteristic of the instances such that the closer the instance to a lattice, the easier for WalkSAT.[3] GSAT and ILS show a mild tendency of requiring higher search cost in the vicinity of the small-world area, as shown in Fig.10 and Fig.11, respectively. The hardest instances for GSAT and ILS are the ones located in the first part of the plots, i.e., the instances with strong lattice/small-world topologies[4]. In some plots, we also observe that the instances corresponding to the maximal proximity ratio are the hardest on average[5]. Nevertheless, this behavior is not regular nor clear and the statistical correlation between search cost and proximity ratio is quite low.

The peculiar behavior observed is a clear signal that different factors other than small-world topology affect algorithm behavior. Among the main factors, we consider the search landscape characteristics induced by the SAT instance and the actual search process performed by the algorithm on the landscape. In fact, the landscape characteristics and the strategy used to explore it are the main elements that affect local search behavior. The relations between instance structure and search landscape are an extremely important research issue, that is subject of ongoing work (see, for instance, [10]).

[3] For example, the regular *chaining* pattern of clauses in lattice instances could make the repair process easier.

[4] The 800-2400 instances are indeed not solved by GSAT in the range corresponding to small-world.

[5] This observation is also confirmed by evaluating a moving window average.

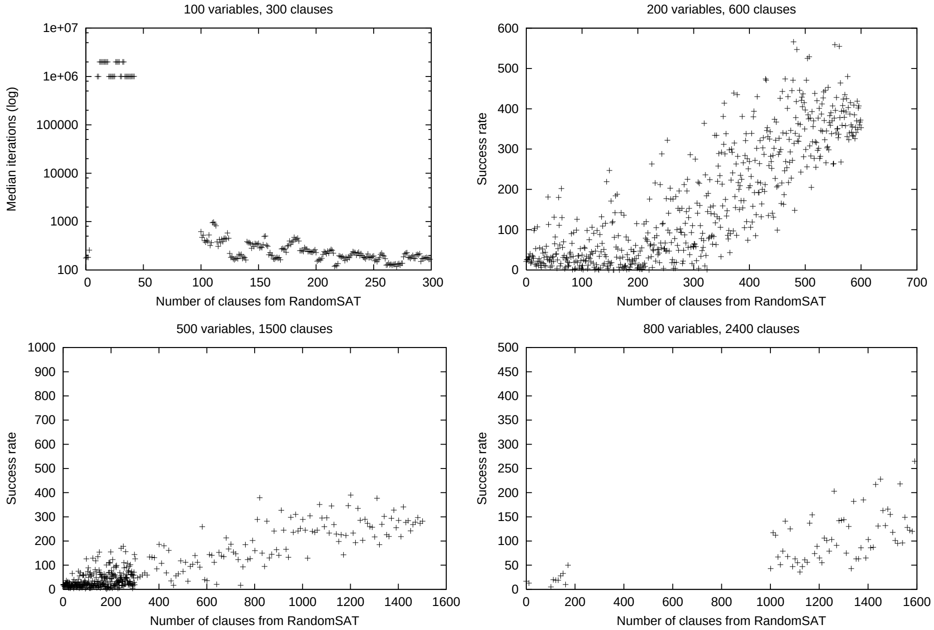

Fig. 10. Search cost of GSAT across the instances, from lattice to random structure. In the uppermost left plot, points represent median iterations over 1000 runs. The remaining plots report an estimation of the search cost in terms of success ratio, i.e., the number of instances solved – given a termination condition defined as the maximum number of non-improving moves.

5 Conclusion and Future work

In this work, we have presented a procedure to generate SAT instances associated to an interaction graph with small-world topology. Small-world SAT instances are constructed by introducing clauses from random instances into lattice based ones.

We tackled the benchmark instances with three different local search algorithms and observed their behavior across the whole spectrum, from regular lattice to random topologies. Our aim was to check whether there is a positive correlation between search cost and small-world topology of SAT instances, as observed in the case of complete solvers. Results showed primarily that the behavior of local search algorithms is fairly different. In some cases, we observed that most of the hardest instances are concentrated in the lattice/small-world area. Nevertheless, this result is not as clear as in the case of complete solvers and further investigations are required. If the conjecture on the positive correlation between instance hardness and proximity ratio is true, then the phenomenon may be explained considering the locality of decisions taken by the heuristics, as supposed in [21]: a locally good decision taken w.r.t. the clustering properties might be wrong with respect to the whole graph.

The study of relations between structure and local search behavior is still a partially unexplored area. First of all, in this work we have just considered one of the possi-

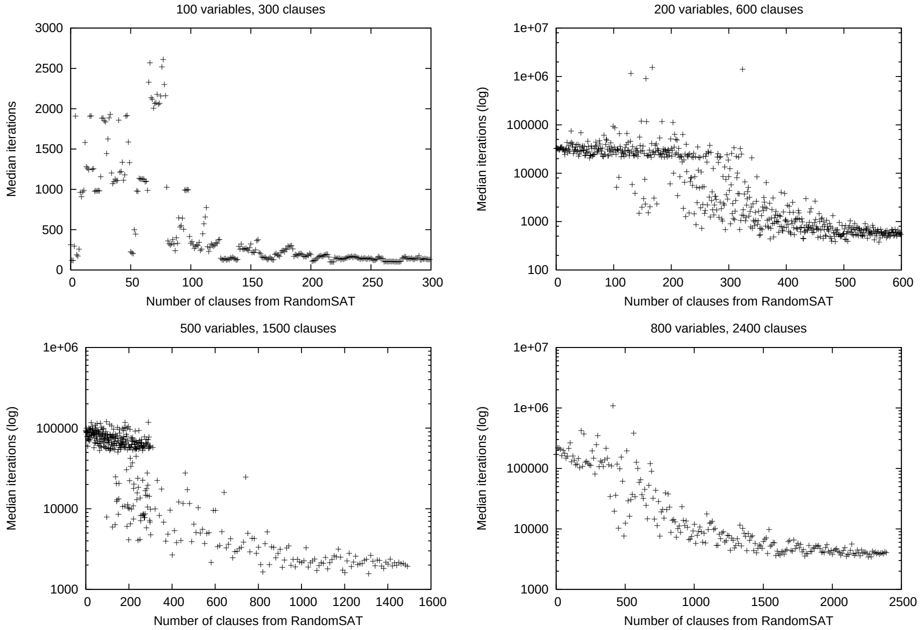

Fig. 11. Search cost of ILS across the instances, from lattice to random structure. Points represent median iterations over 1000 runs. Log-scale on the y-axis has been used when necessary.

ble ways of characterizing structure. Other definitions for graphs are possible (such as weighted graphs), to capture different problem features and to extend our results to problems other than SAT. Moreover, concerning local search algorithms, we believe that the core issue to explain the algorithm behavior is the investigation of the relations between problem structure and search landscape, and, in turn, search landscape and the actual strategy used to explore it.

References

1. D. Achlioptas and C. Moore. The asymptotic order of the random k-SAT threshold. In *Proc. of FOCS02*, pages 779–788, 2002.
2. C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.
3. I. P. Gent, H. H. Hoos, P. Prosser, and T. Walsh. Morphing: Combining structure and randomness. In *Proc. of AAAI99*, pages 654–660, 1999.
4. C.P. Gomes, B. Selman, N. Crato, and H. Kautz. Heavy-Tayled phenomena in Satisfiability and Constraint Satisfaction Prpblems. *Journal of Automated Reasoning*, 24:67–100, 2000.
5. T. Hogg, B. A. Huberman, and C. P. Williams. Phase transitions and the search problems. *Artificial Intelligence*, 81(1–2), 1996.
6. H. H. Hoos and T. Stützle. Towards a characterisation of the behaviour of stochastic local search algorithms for SAT. *Artificial Intelligence*, 112:213–232, 1999.

7. H. R. Lourenço, O. Martin, and T. Stützle. Iterated local search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, volume 57, pages 321–353. Kluwer Academic Publishers, Norwell, MA, 2002.
8. D. G. Mitchell, B. Selman, and H. J. Levesque. Hard and easy distributions of SAT problems. In *Proc. of AAAI92*, pages 459–465. AAAI Press/MIT Press, July 1992.
9. M.R.Garey and D.S.Johnson. *Computers and intractability; a guide to the theory of NP-completeness*. W.H. Freeman, 1979.
10. S. Prestwich and A. Roli. Symmetry breaking and local search spaces. In *Proceedings of CPAIOR 2005*, volume 3524 of *Lecture Notes in Computer Science*. Springer–Verlag, 2005.
11. I. Rish and R. Dechter. Resolution versus search: Two strategies for SAT. *J. Automated Reasoning*, 24:225–275, 2000.
12. A. Roli. Criticality and parallelism in GSAT. *Electronic Notes in Discrete Mathematics*, 9, 2001.
13. A. Roli. Criticality and parallelism in structured SAT instances. In P. Van Henteryck, editor, *Proc. of CP02*, volume 2470 of *Lecture Notes in Computer Science*, pages 714–719. Springer-Verlag, 2002.
14. A. Roli. Design of a new metaheuristic for MAXSAT problems (extended abstract). In P. Van Henteryck, editor, *Proceedings of CP02*, volume 2470 of *Lecture Notes in Computer Science*, page 767. Springer-Verlag, 2002.
15. A. Roli. Metaheuristics and structure in satisfiability problems. Technical Report DEIS-LIA-03-005, University of Bologna (Italy), May 2003. PhD Thesis - LIA Series no. 66.
16. A. Roli. Problem structure and search: Empirical results and open questions. In *Proceedings of CPAIOR03*, Montreal (Canada), 2003.
17. A. Roli and C. Blum. Critical Parallelization of Local Search for MAX–SAT. In F. Esposito, editor, *AI*IA2001: Advances in Artificial Intelligence*, volume 2175 of *Lecture Notes in Artificial Intelligence*, pages 147–158. Springer-Verlag, 2001.
18. A. Roli. Links between complex networks and combinatorial optimization. In *Proc. of Workshop on Experimental Analysis of Algorithms for Artificial Intelligence* – AI*IA working group on Knowledge Representation and Reasoning, Università di Ferrara, Italy, June 10 2005.
19. B. Selman, H.A. Kautz, and B. Cohen. Noise strategies for local search. In *Proc. of AAAI-94*, pages 337–343, 1994.
20. B. Selman, H. J. Levesque, and D. Mitchell. A new method for solving hard satisfiability problems. In *Proc. of AAAI92*, pages 440–446, Menlo Park, California, 1992. AAAI Press.
21. T. Walsh. Search in a small world. In *Proc. of IJCAI99*, pages 1172–1177, 1999.
22. T. Walsh. Search on high degree graphs. In *Proc. of IJCAI01*, 2001.
23. D.J. Watts. *Small Worlds: The Dynamics of Networks between Order and Randomness*. Princeton University Press, 1999.
24. D.J. Watts and S.H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.