Problem Structure and Search: Empirical Results and Open Questions

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

Abstract

The impact of problem structure on search is a relevant issue in AI research and related areas. Among the possible approaches to analyze problem structure, the one referring to constraint graph and similar graphs enables to relate graph parameters and characteristics with search algorithm behavior. In this work we present and discuss two examples of the impact of graph structure on the performance of local search and exact algorithms applied to Satisfiability problems (SAT). After the definition of the graph associated to SAT instances, we define its main parameters and characteristics. Then, by means of a *morphing* procedure we generate set of instances which smoothly interpolate between random and structure. In a first experiment we show that the node degree distribution affects the behavior of parallel local search. The second experiment evidences the impact of small-world properties on the solution cost of a complete solver.

1 Introduction

The definition of structure emerging from the literature on Constraint Satisfaction Problems and Combinatorial Optimization Problems 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 an instance generated with some similarity with a real-world problem. Commonly, we say that a problem is structured if it shows, under some abstraction, regularities such as well defined subproblems, patterns or correlations among problem variables. It is interesting to note that one of the strengths of Constraint Programming is indeed that it enables to capture this structure by means of global constraints.

The impact of problem structure on search performance has been studied from different perspectives. Studies on the impact of problem structure on heuristic search can be found, for example, in [2, 28, 25, 14]. Important results and observations on structure and problem hardness are reported in [9, 7, 8]. The effects of problem encoding are discussed in [10, 3]. Finally, the search algorithms behavior w.r.t. graph properties has been discussed in [24, 23].

In this work we present and discuss empirical results on the search behavior as a function of parameters of the constraint graph. The results concern the application of parallel local search and an exact algorithm to Satisfiability Problems (SAT). The benchmarks have been constructed by a *morphing* procedure [5] which enables us to smoothly interpolate between randomness and structure and to control single characteristics of the constraint graph.

The structure of the paper is as follows: in Sec. 2 we introduce the graph associated to SAT instances and the graph parameters which will be considered in the experiments. In Sec. 3 we present the first set of experiments concerning the behavior of parallel local search w.r.t. the node degree distribution of the constraint graph. Then, Sec. 4 reports experiments on the impact of small-world properties on a complete SAT solver, as well as local search algorithms. Finally, we conclude with the discussion of open points and future research directions.

2 Structural Properties of SAT Problems

In this work we focus on one among the possible ways of characterizing the structure of a problem: 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 are defined in terms of a data structure suitable for structure analysis. For instance, problems defined on graphs (e.g.,Graph Coloring Problem). In general, for CSPs a constraint graph can be defined, where nodes correspond to variables and edges connect two variables if there exists a constraint involving them[1].

For SAT problems a graph very similar to the CSP constraint graph can be defined. The graph associated with a SAT instance is an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where each node $v_i \in \mathcal{V}$ corresponds to a variable and edge $(v_i, v_j) \in \mathcal{A}$ $(i \neq j)$ if and only if variables v_i and v_j appear in a same clause. Observe that the same graph corresponds to more than one formula, since nodes are connected by only 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 sections, it will be shown that some properties of this SAT-associated graph strongly affect the behavior of local and complete search.

In the following we define the properties which will be considered relevant for the purposes of this work.

Given a non-oriented and simple graph, associated to a SAT instance, we consider the following parameters: node degree, characteristic path length and clustering.

In an instance with n variables, each node v_i, $i = 1, \ldots, n$, has a degree $q_i \in \{0, 1, \ldots, n-1\}$. For k-SAT problems, defined as conjunction of clauses with exactly k literal each, it holds $k-1 \leq q_i \leq n-1$. We define the *average connectivity*[2] of the instance as the average node degree of the corresponding graph, i.e., $q = \frac{1}{n}\sum_{i=1}^{n} q_i$. Moreover,

[1] In this work we do not focus on relationships between constraint graph properties and special cases of algorithm complexity, as discussed for instance in [4, 20].

[2] In models for generating random CSPs, the *average connectivity* corresponds to the *density*.

to make direct comparisons among instances with different number of variables, we also introduce the normalization of q: $\overline{q} = \frac{q}{n-1}$. In the following, we will use indifferently the expressions connectivity and node degree of an instance $\mathcal{I}$, being the second defined on the graph associated with $\mathcal{I}$. In order to compare the node degree distribution between instances, we consider the frequency of node degree $Freq(j)$ = 'frequency of a node connected to exactly j nodes' and the cumulative frequency $CumFreq(j)$ = 'frequency of a node connected to not more than j nodes'.

The connectivity gives a rough evaluation of the speed at which a modification occurring on a node affects the other nodes. The higher the connectivity, the stronger the "information spreading".

The characteristic path length $L(G)$ of a graph G can be informally defined as the average path length between any pair of nodes. We will assume that the graph is connected, therefore L is always finite. Indeed, if the graph is not connected, it can be decomposed in connected components representing independent subproblems. Formally, the characteristic path length L of a graph G is defined as the median of the means of the shortest paths connecting each vertex $v \in V(G)$ to all other vertices [26].

Finally, the clustering coefficient γ of a graph G quantifies the probability that, given node v_1 connected to v_2 and v_3, there is an edge between v_2 and v_3. For instance, friendship relations are characterized by a high value of γ. Formally, the clustering coefficient is defined on the basis of the notion of *neighborhood*. The *neighborhood* Γ_v of a node $v \in G$ is the subgraph consisting of the nodes adjacent to v (not including v itself). The clustering of a neighborhood is defined as:

$$\gamma_v = \frac{|E(\Gamma_v)|}{\binom{k_v}{2}},$$

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 $v \in G$.

Typically, random graphs are characterized by low characteristic path length and low clustering, whilst regular graphs (such as lattice graphs) have high values for L and γ. Conversely, small-world graphs [27, 26] are characterized by low L and high γ.

In the following section we will discuss the impact of the node degree on the behavior of local search for SAT in connection with a phenomenon called *criticality and parallelism* in combinatorial optimization. Afterwards, we briefly present preliminary results on the effect of small-world phenomenon also in systematic (exact) search and approximate algorithms for SAT.

3 Node degree distribution and parallel moves

As first example of the impact of graph properties on search, we discuss some results related to the phenomenon called *criticality and parallelism* in combinatorial optimization.

3.1 Criticality and Parallelism in Combinatorial Optimization

The phenomenon called *criticality and parallelism* has been observed in the context of local search algorithms applied to combinatorial optimization problems [15, 13, 12], where local search is modified by applying some local moves in parallel. It has been shown that the effectiveness of these algorithms depends on the parallelism degree τ (number of simultaneous moves): if τ increases, the solution quality also increases up to a maximal point (corresponding to τ_{opt}) at which it starts to decrease. It has also been shown that τ_{opt} is negatively correlated with the *connectivity* among variables of the problem: the higher the connectivity, the lower τ_{opt}. The average connectivity of a problem estimates the direct influence among variables. In [15, 13, 12] it is also shown that the optimal parallelism value is associated to a phase transition.

In the following we use the expression "parallel local search" with the meaning of local search in which more than one local move is synchronously performed. These algorithms can be both sequentially and parallel implemented. The parallelization of local search can be achieved in different ways and the most important applied so far are:

- At each iteration, apply a local move on a set of variables (or a solution component) with probability p. This results in an average parallelism of pn, where n is the number of variables.
- Divide the problem in τ subsystems (which are, in general, not independent) and apply local search to optimize each of them independently.

An example of the first process is given in [15], where Simulated Annealing is applied on problems defined over binary variables. Instead of performing one random flip, every variable is flipped with probability p. The resulting average parallelism is pn, where n is the number of variables. Examples of the second approach can be found in [13, 12], where NK lattice models are optimized by subdividing them in τ *patches*, independently optimized.

A phenomenon with analogous characteristics has been discovered in parallel local search for SAT [18] and MAXSAT [19], where a parallel version of GSAT [22] (called PGSAT) has been applied to random instances. In PGSAT, variables are divided in τ subsets and, for each subset, we flip the variable that will decrease the greatest number of unsatisfied clauses. For random satisfiable SAT instances it has been experimentally shown that the best global performance (time, iterations, fraction of solved instances) is achieved with an optimal parallelism degree τ_{opt}. Furthermore, τ_{opt} is monotonically non increasing with the connectivity among variables.

3.2 Connectivity distribution

In order to compare the node degree distribution between instances, and especially non random instances, we consider the frequency of node degree $Freq(j)$ = 'frequency of a node connected to exactly j nodes' and the cumulative frequency $CumFreq(j)$ = 'frequency of a node connected to not more than j nodes'. Fig.1 shows the cumulative frequency vs. the normalized node degree for random 3-SAT instances retrieved from SATLIB [11]. Note that the curves are quite regular and, as the number of variables increases, they converge to a step function located at the average node degree. We can assume that the graph corresponding to a random 3-SAT instance is a random graph $G_{n,p}$ [17], where n is the number of nodes and p is the probability that any pair of nodes are connected. In fact, uniform random 3-SAT instances of SATLIB are generated by randomly selecting, for each clause, three literals among the complete set of $2n$ literals. Thus, every pair of variables has the same probability to belong to a same clause. For

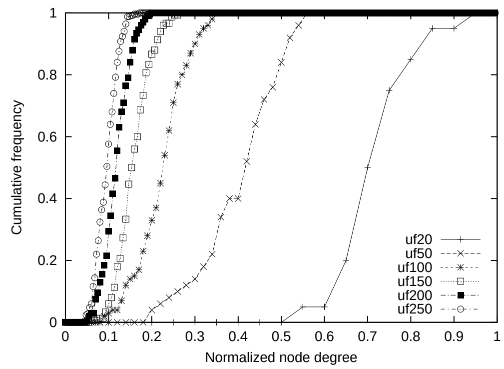

Figure 1: Cumulative frequency vs. normalized node degree for Uniform Random 3-SAT instances in the threshold region.

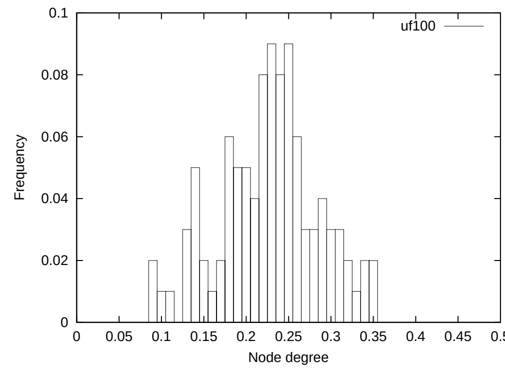

Figure 2: Frequency against normalized connectivity in random 3-SAT instance uf100-01. The instance has an average normalized connectivity of 0.229.

random graphs like $G_{n,p}$ the distribution probability of connectivity follows a Poisson distribution, i.e.,

$$\text{prob}\{\text{a node is connected exactly to other } j \text{ nodes}\} = e^{-\lambda}\lambda^j/j!$$

where the parameter λ is the expected node degree, therefore, in our case, $\lambda = (n-1)\overline{q} = q$. For instance, in Fig.2 the frequency of a 3-SAT instance with 100 variables is plotted.

In SAT problems generated by an encoding procedure from other problems there are two sources of structure: the inherent structural properties of the original problem and the relations among variables introduced by the encoding procedure. As noted in [3], the inherent structure of the problem might be partially lost in the encoded formulation. However, independently of the origins of structure, the SAT instances we consider clearly show node degree distributions very different with respect to random instances. Fig.3 shows the curve of cumulative frequency for structured SAT instances taken from SATLIB, produced by encoding a Blocks World Planning Problem (*huge*), a Logistics Planning Problem (*logistics-a*) and Inductive Inference (*ii16a1*). The plotted curves show apparent differences with those of random SAT problems. They are not as regular as random ones and they have gaps and plateaus, especially in the uppermost part of the curve. Structured instances thus have a more spread and non-uniform connectivity distribution.

The instance *ii16a1* is the most peculiar and differs the most from random instances. It has 1650 variables and a normalized average connectivity $\overline{q}_{ii16a1} = 0.0239$. Its cumulative frequency is shown in Fig.4, along with the cumulative frequency of a random 3-SAT instance of the same size and normalized connectivity (instance *3sat1650*). Fig.5 and Fig.6 plot the respective frequency of node degree. We can note that the node degree frequency of the structured instance is highly asymmetric and has a peak close to 0.018, corresponding to the large gap in the cumulative frequency. Therefore, *ii16a1* has a very large number of nodes with lower connectivity than the average. Conversely, the node degree frequency of the random instance is regular (it approximately fits the Poisson distribution with high mean) and the highest peak in frequency is very close to

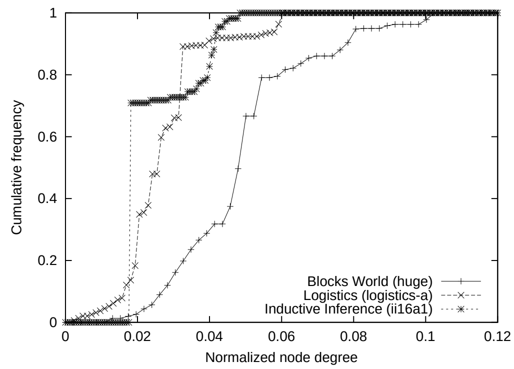

Figure 3: Cumulative frequency vs. normalized node degree in structured instances.

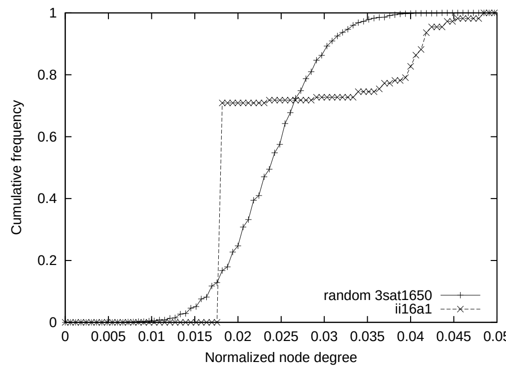

Figure 4: Cumulative frequency vs. normalized node degree in a structured instance (*ii16a1*) and in a random 3-SAT instance *3sat1650* of same size ($n = 1650$) and connectivity ($\overline{q} \approx 0.0239$).

the mean.

3.3 Morphing from Random to Structure

To investigate the impact of node degree distribution on parallel local search, we generated instances with a controlled amount of *structure*, by means of a technique called *morphing* [5]. This methods enables to generate instances gradually morphing from a source to a destination instance by varying a parameter $p \in [0, 1]$. The lower p used to generate an instance, the more similar to the source. To be applied on SAT problems, the method needs instances with the same number of variables (n) and clauses (m). A new SAT instance is generated by selecting each of the m clauses either from the source or the destination. The clause is chosen from the destination instance with probability p. We generated a satisfiable random 3-SAT instance with 1650 variables and 19368 clauses (*3sat1650_large*), the same number as *ii16a1*. With $p = 1$ we obtain *ii16a1* and with $p = 0$ *3sat1650_large*. Since p controls the number of clauses belonging to the structured instance *ii16a1*, it also measures the amount of structure in the generated instance. Fig.7 shows the node degree cumulative frequency of source (random), destination (structure) and instances generated by the morphing method. The node degree frequency of the considered instances is plotted in Fig.8. The results of 500 trials of PGSAT for different values of parallelism are shown in Fig.9. Since the instances generated by the morphing procedure are no longer guaranteed to be satisfiable, the plots report the average solution error (number of unsatisfied clauses) returned by the algorithm. The results are still valid, as shown in [19]. Observe that the optimal parallelism increases with p, therefore we can conjecture that the high peak of the node frequency of the structured instance strongly affects the optimal parallelism. In fact, the optimal parallelism for *ii16a1* is higher than that of the related random instance, which has an average degree greater than the one corresponding to the peak in *ii16a1*.

We can conclude by asserting two points. First, regardless of the instance type, the

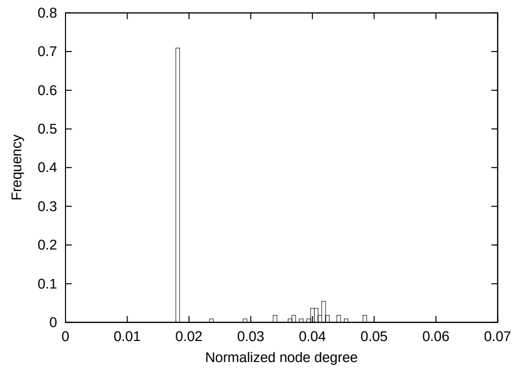

Figure 5: Frequency vs. normalized node degree in the instance *ii16a1*.

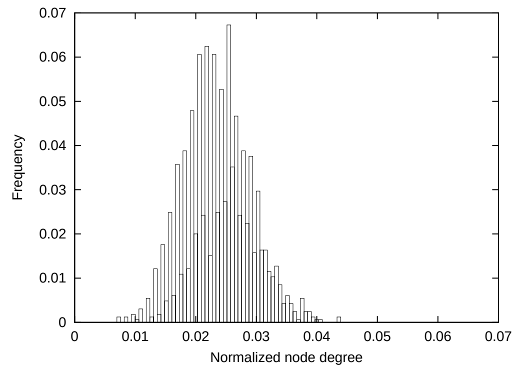

Figure 6: Frequency vs. normalized node degree in the instance *3sat1650*.

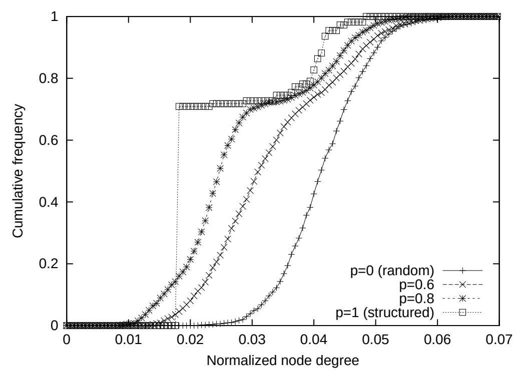

Figure 7: Cumulative frequency vs. normalized node degree for instances generated by morphing. $p \in [0, 1]$ controls the randomness/structure ratio: the higher p, the higher the amount of structure.

average connectivity is a rough, yet indicative, parameter for the optimal parallelism. Second, we have found experimental evidence that in structured instances the highest peaks in the node degree distribution have a strong impact on the optimal parallelism value. Of course, we can not claim any statistical proof, that needs an exhaustive and deeper experimental analysis.

In the next section, we will focus on a different feature of the graph, the small-world property, investigating the impact of this characteristic on the search performance.

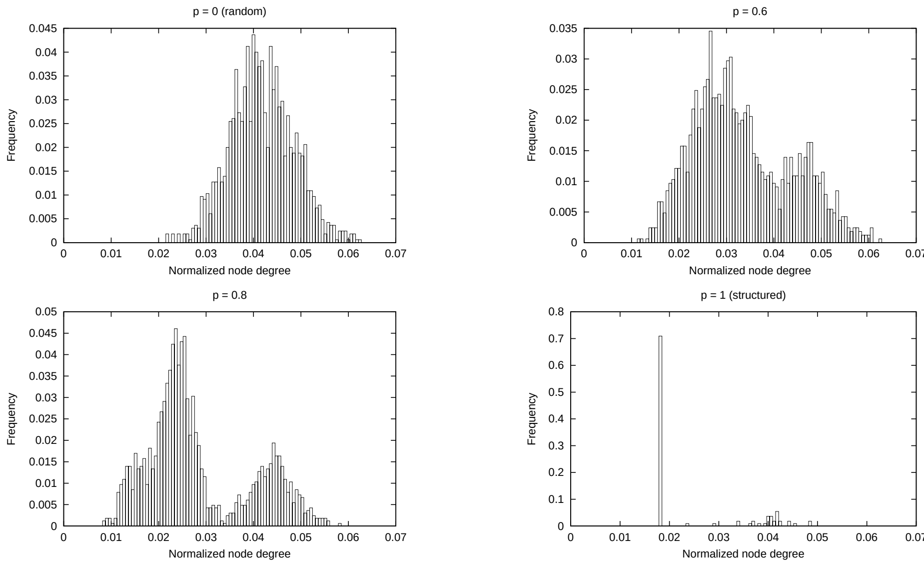

Figure 8: Frequency against normalized node degree for instances generated by morphing from a random instance (*3sat1650_large*) to a structured one (*ii16a1*).

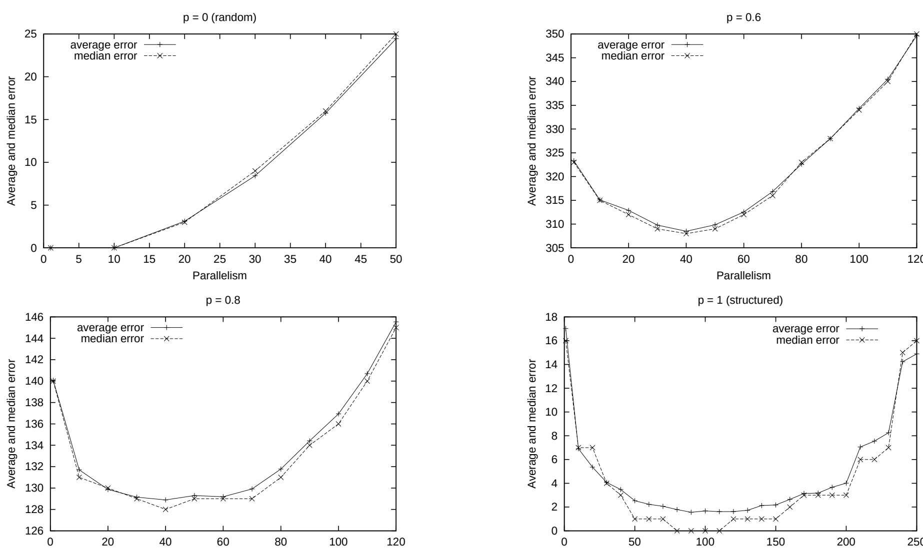

Figure 9: Average and median error of PGSAT run with different values of τ on instances generated by morphing from a random instance (*3sat1650_large*) to a structured one (*ii16a1*).

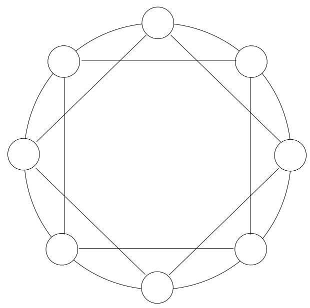

Figure 10: Example of lattice graph. Each node has 4 neighboring nodes.

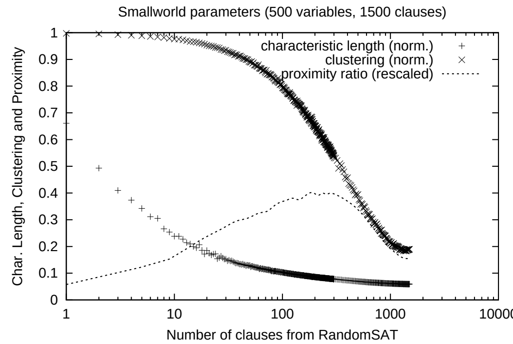

Figure 11: Characteristic path length L, clustering C and proximity ratio μ for instances generate by morphing from a lattice SAT instance and a random SAT instance of 500 variables and 1500 clauses.

4 Small-world

A second set of experiments concerns the investigation of the behavior of complete and approximate search applied to SAT instances generated on the basis of small-world graphs [27, 26]. Small-world graphs are characterized by low characteristic path length and high clustering. Therefore, they exhibit a mixture of properties from random and highly structured graphs.

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. This procedure is indeed very similar to the one used in [27] to generate graphs by interpolating between lattice and random graphs. Lattice 3-SAT instances have been generated on the basis of a lattice graph (see Fig. 10). A boolean variable is associated to each node and clauses are generated in such a way that, for every pair of neighboring nodes, a clause exists that involves both the variables. The instances composing the benchmark are obtained by introducing into a lattice SAT instance a prefixed number of clauses randomly chosen from a random SAT instance with the same number of variables and clauses. In this way it is possible to morph from lattice to random topology with the finest tuning and observe the arising of small-world properties in SAT instances.

In order to have a quantitative measure of the small-world characteristic we introduce the *proximity ratio* μ [23], defined as the ratio between clustering and characteristic path length, normalized with the same ratio corresponding to a random graph, i.e., $\mu = (C/L)/(C_{rand}/L_{rand})$. In Fig. 11 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, C maintains a relatively high value for a higher amount of *perturbation*. The instances with low length and high clustering are characterized by the small-world property. This is also indicated by the maximum in the proximity ratio

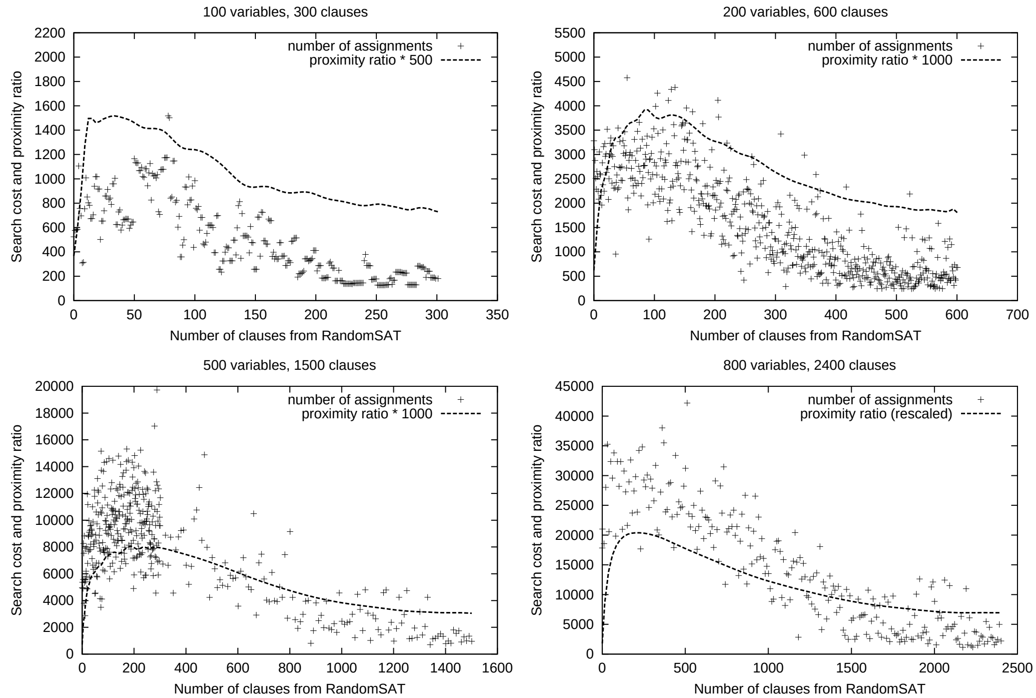

Figure 12: Search cost (number of assignments needed to solve the instance) for instance gradually interpolating between lattice SAT and random SAT. The proximity ratio (rescaled) is also plotted.

curve, which approximately assumes its maximum in correspondence of that region.

We generated four sets of instances, each obtained by morphing between a lattice SAT and a random SAT with same number of variables and clauses. All the generated instances are satisfiable[3]. The instances have been solved by BerkMin solver [6], one of the most efficient complete SAT solvers available nowadays. The search cost has been evaluated as the number of variable assignments performed by the algorithm before solving the instance. In Fig. 12 the search cost for every set of instances is plotted. We clearly observe that, after few perturbations the small-world property appears, as proved by the proximity ratio curve. Exactly in that region, the search cost is maximal. We can conjecture, as previously done in [23], that small-world instances are harder to solve. This conjecture is confirmed by the correlation between search cost and proximity ratio, shown in Fig.13: the higher the proximity ratio, the higher the search cost.

A possible explanation for this behavior can be formulated with respect to the heuristic used for branching on variable values, which makes use of local information. Indeed, since the small-world topology is characterized by highly clustered parts of the graph connected by few links, a local decision is usually biased on the cluster property and it

[3]Unsatisfiable instances have been filtered by means of a complete solver. The ratio between clauses and variables is 3, lower than the so-called critical ratio [16, 1] (which is close to 4.3 for 3-SAT instances). This is due to the structure of lattice SAT instances which seem almost all unsatisfiable at the critical ratio.

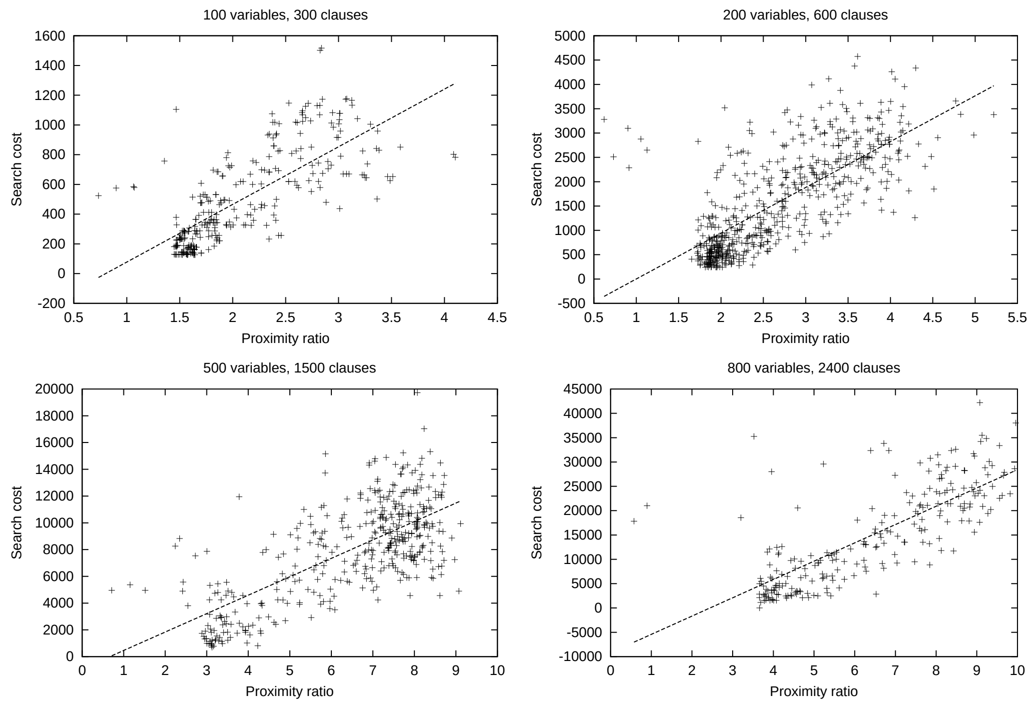

Figure 13: Correlation between search cost and proximity ratio.

might be wrong for the whole collection of connected clusters.

It is interesting to note also that a behavior with similar characteristics has been observed when local search is applied, though results are not as clear as in the previous case. We solved the same benchmarks with two local search algorithms, namely WalkSAT [21] and GSAT [22]. Results are shown in Fig. 14 and Fig.15 respectively[4]. The algorithms have been stopped after a maximum number of moves without improvement. In each plot we reported the number of successes (out of 1000 runs) and the proximity ratio. Considering the results obtained with WalkSAT, we note that in the proximity of the small-world region are located some among the hardest instances (i.e., the success rate is the lowest). This behavior is particularly apparent on the 200-600 instances, where the lowest success rate instances are located approximately around the maximum proximity. GSAT performance is not as good as the WalkSAT one and it always reach a lower success rate. Nevertheless, we can observe that the most difficult instances are located in the small-world region[5].

We can not claim any generality from these experiments, nevertheless these results evidence that for at least one general purpose SAT solver and two different local search algorithms, small-world SAT instances are more difficult to solve. Moreover, these results bring to attention that also macro-parameters of the constraint graph may affect the search performance.

[4]Due to lack of space we omit the plots with correlation between proximity and search cost.
[5]The 800-2400 instances are indeed not solved in the range corresponding to small-world.

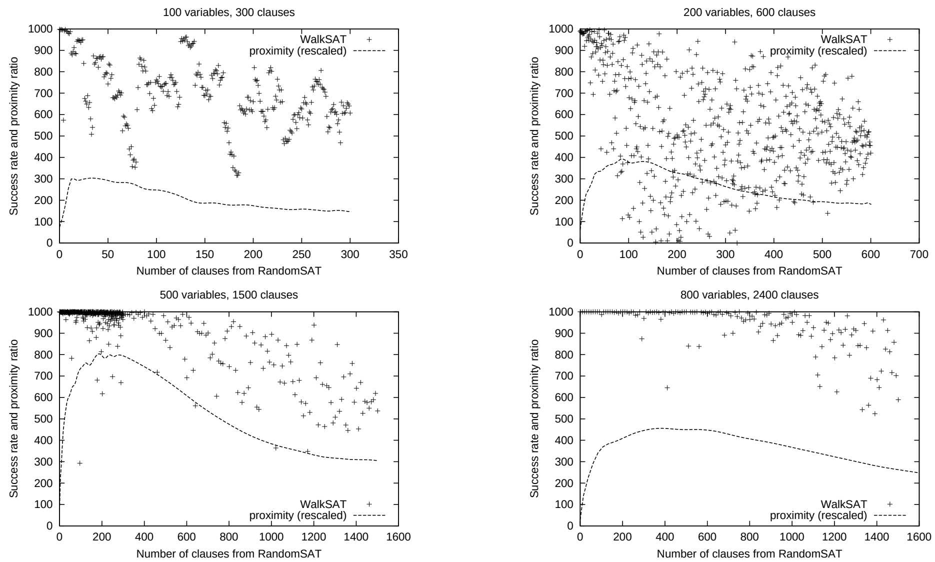

Figure 14: Success rate (out of 1000 runs) of WalkSAT on instances gradually interpolating between lattice SAT and random SAT. The proximity ratio (rescaled) is also plotted.

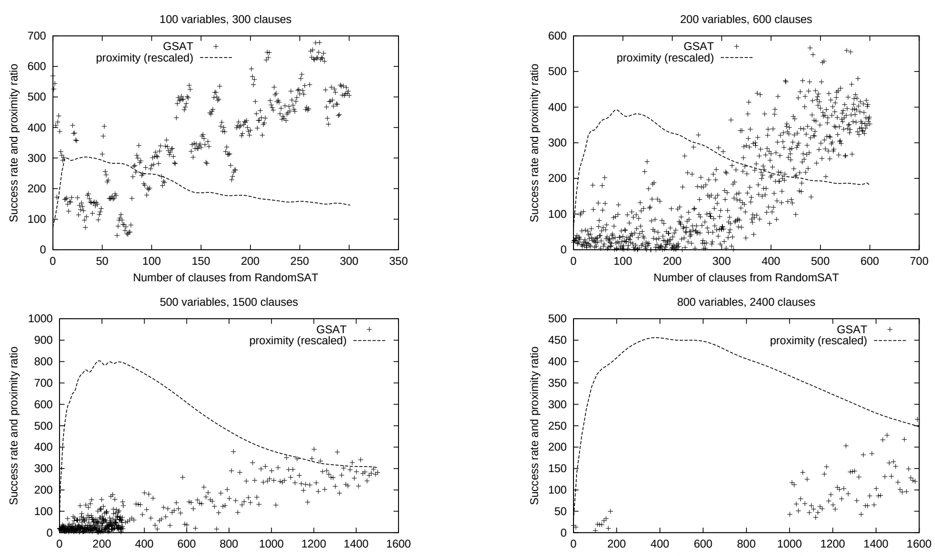

Figure 15: Success rate (out of 1000 runs) of GSAT on instances gradually interpolating between lattice SAT and random SAT. The proximity ratio (rescaled) is also plotted.

5 Discussion and open questions

This work discusses just two examples of the use of a general methodology which tries to exploit constraint graph parameters to extract useful information on the problem structure. We have shown that two important properties, namely the node degree distribution and the small-world property, affect the search performance of local search (with parallel moves) and systematic search.

We believe that the study of constraint graph properties, also for CSPs which are are not explicitly formulated on graphs, can be effectively used with two main objectives:

- *a posteriori*: what are the characteristics of problem benchmarks? Are they suitable to represent real-world problems?
- *a priori*: how to exploit structure to guide heuristic search ?

Several open questions arise from these empirical results, concerning both systematic and approximate algorithms and also their integration. We briefly outline some among the most relevant:

- Which are the connections between constraint graph properties and search space characteristics?
- Is it possible to explore the strengths and weaknesses of the heuristics w.r.t. constraint graph properties?
- A deep understanding of the effects of problems encoding is needed in order to design effective hybrid solvers. To what extent the graph properties can help?
- The constraint graph might be a not suitable general abstraction, indeed various alternative formulations to study the structure of a problem can be used (e.g., weighted graphs).

Some future research directions concern the experimental evaluation of the application of parallel moves also in constructive algorithms (e.g., more than one assignment per step). Furthermore, we would like to investigate whether phenomena analogous to criticality and parallelism appear also in systematic search, e.g., when randomness is introduced. Finally, we believe that a very promising issue is the introduction of learning techniques to design effective heuristics and dynamically exploit structure properties.

References

[1] D. Achlioptas and C. Moore. The asymptotic order of the random k -SAT threshold. In *Proceedings of FOCS 2002*, pages 779–788, 2002.

[2] J.C. Beck and M.S. Fox. Dynamic problem structure analysis as a basis for constrained-directed scheduling heuristics. *Artificial Intelligence*, 117:31–81, 2000.

[3] R. Béjar, A. Cabiscol, C. Fernàndez, F. Manyà, and C.P. Gomes. Capturing structure with satisfiability. In T. Walsh, editor, *Principles and Practice of Constraint Programming – CP 2001, 7th International Conference*, Lecture Notes in Computer Science, pages 137–152. Springer, 2001.

[4] E.G. Freuder. A sufficient condition for backtrack-free search. *Journal of the ACM*, 29(1):24–32, 1982.

[5] I. P. Gent, H. H. Hoos, P. Prosser, and T. Walsh. Morphing: Combining structure and randomness. In *Proceedings of AAAI99*, pages 654–660, 1999.

[6] E. Goldberg and Y. Novikov. BerkMin: A Fast and Robust SAT Solver. In *Design Automation and Test in Europe (DATE)*, pages 142–149, 2002.

[7] C.P. Gomes. Structure, duality, and randomization – common themes in AI and OR. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, 2000.

[8] C.P. Gomes and B. Selman. Problem structure in the presence of perturbations. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, 1997.

[9] C.P. Gomes and D. Shmoys. Completing quasigroups or latin squares: A structured graph coloring problem. In *Proceedings of the Computational Symposium on Graph Coloring and Extensions*, 2002.

[10] H.H. Hoos. SAT-encodings, search space structure, and local search performance. In *Proceedings of IJCAI-99*, pages 988–993, 1999.

[11] H.H. Hoos and T. Stützle. SATLIB: An online resource for research on SAT. In I. Gent, H. van Maaren, and T. Walsh, editors, *SAT2000: Highlights of Satisfiability Research in the Year 2000*, pages 283– 292. IOS Press, 2000.

[12] S.A. Kauffman. *At home in the universe*. Oxford Press, 1995.

[13] S.A. Kauffman and W. Macready. Technological evolution and adaptive organizations. *Complexity*, 26(2):26–43, March 1995.

[14] K. Leyton-Brown, E. Nudelman, and Y. Shoham. Learning the empirical hardness of optimization problems. In P. Van Henteryck, editor, *Proceedings of CP02 - Eighth International Conference on Principles and Practice of Constraint Programming*, volume 2470 of *Lecture Notes in Computer Science*. Springer, 2002.

[15] W.G. Macready and S.A. Kauffman A.G. Siapas. Criticality and parallelism in combinatorial optimization. *Science*, 271:56–59, 1996.

[16] D.G. Mitchell, B. Selman, and H.J. Levesque. Hard and easy distributions of sat problems. In *Proceedings, Tenth National Conference on Artificial Intelligence*, pages 459–465. AAAI Press/MIT Press, July 1992.

[17] M.E.J. Newman, S.H. Strogatz, and D.J. Watts. Random graphs with arbitrary degree distributions and their applications. *Phys. Rev. E*, 64, 2001.

[18] A. Roli. Criticality and parallelism in GSAT. *Electronic Notes in Discrete Mathematics*, 9, 2001.

[19] A. Roli and C. Blum. Critical Parallelization of Local Search for MAX–SAT. In *AI*IA2001: Advances in Artificial Intelligence*, volume 2175 of *Lecture Notes in Artificial Intelligence*, pages 147–158. Springer, 2001.

[20] T. Schiex. A note on CSP graph parameters. Technical report, INRA, Toulouse (France), 1999.

[21] B. Selman, H.A. Kautz, and B. Cohen. Noise strategies for local search. In *Proc. 12th National Conference on Artificial Intelligence, AAAI'94, Seattle/WA, USA*, pages 337–343, 1994.

[22] B. Selman, H.J. Levesque, and D. Mitchell. A new method for solving hard satisfiability problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 440–446, Menlo Park, California, 1992. American Association for Artificial Intelligence, AAAI Press.

[23] T. Walsh. Search in a small world. In *proceedings of IJCAI99*, pages 1172–1177, 1999.

[24] T. Walsh. Search on high degree graphs. In *Proceedings of IJCAI-2001*, 2001.

[25] J.P. Watson, L. Barbulescu, A.E. Howe, and L.D. Whitley. Algorithm Performance and Problem Structure for Flow-Shop Scheduling. In *Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, 1999.

[26] D.J. Watts. *Small Worlds: The Dynamics of Networks between Order and Randomness*. Princeton University Press, 1999.

[27] D.J. Watts and S.H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

[28] C. Williams and T. Hogg. Exploiting the deep structure of constraint problems. *Artificial Intelligence*, 70:72–117, 1994.