On the Analysis of Backtrack Procedures for the Colouring of Random Graphs

Rémi Monasson

[1] CNRS-Laboratoire de Physique Théorique de l'ENS, 24 rue Lhomond, 75005 Paris, France
[2] CNRS-Laboratoire de Physique Théorique, 3 rue de l'Université, 67000 Strasbourg, France

Abstract. Backtrack search algorithms are procedures capable of deciding whether a decision problem has a solution or not through a sequence of trials and errors. Analysis of the performances of these procedures is a long-standing open problem in theoretical computer science. I present some statistical physics ideas and techniques to attack this problem. The approach is illustrated on the colouring of random graphs, and some current limitations and perspectives are presented.

1 Introduction

1.1 Why Studying Backtrack Algorithms?

Many computational tasks e.g. constrained satisfaction, scheduling, ... amount to a search for the optimum of a cost function which depends upon a set of variables taking values in a huge space of possible configurations. Finding the true optimum with certainty is not easy from an algorithmic point of view and can be done through 'clever' exhaustive search only. Indeed, as stated by D. Knuth in 1975,

"the majority of all combinatorial computing applications can apparently be handled only by what amounts to an exhaustive search through all possibilities. Such searches can readily be performed by using a well-known "depth-first" procedure (...) called backtracking" [1].

Three decades later, Knuth's statement still holds. It is a fundamental conjecture of theoretical computer science that exhaustive search is essentially the only way to solve many combinatorial problems, called NP-complete [2]. For such problems, backtracking is among the most efficient solving procedures, and often the only one. Unfortunately, the running time of backtrack-based algorithms is hardly predictable, as explained by the same author:

"Sometimes a backtrack program will run to completion in less than a second, while other applications of backtracking seem to go on forever. The author once waited all night for the output from such a program, only to discover that the answers would not be forthcoming for about 10^6 *centuries. A "slight increase" in one of the parameters of a backtrack routine might slow down the total running time by a factor of a thousand; conversely, a "minor improvement" to the algorithm might cause a hundredfold improvement in speed; and a sophisticated "major improvement" might actually make the program ten times slower"* [1].

R. Monasson, On the Analysis of Backtrack Procedures for the Colouring of Random Graphs, Lect. Notes Phys. **650**, 235–254 (2004)
http://www.springerlink.com/

The study of backtrack algorithms has a long and rich story in theoretical computer science [3]. Recently, the use of out-of-equilibrium ideas stemming from statistical physics has led to some improvement in our understanding of when and why a backtrack procedure is very fast, or slow to solve computational problems [4]. The purpose of this article is to illustrate this approach on the colouring problem, discuss its limitations, and propose some ways to circumvent those.

1.2 The Colouring Problems and Some Definitions

An example of combinatorial problem that can be solved by a backtrack procedure is the colouring of graphs (COL). An input of the K-COL decision problem consists in a graph G. The problem consists in finding a mapping from the set of vertices to the set of K colours such that no two neighbouring vertices (connected by an edge) have the same colour, or proving there exists no such mapping. K-COL is a NP-complete problem for any $K \geq 3$ [2], and we choose $K = 3$ in most of what follows. The operation of the backtrack procedure, called Davis-Putnam-Logemann-Loveland (DPLL) [5] on an input of the 3-COL problem is illustrated in Fig. 1.

In order to study in a quantitative way the performances of DPLL, we need first to define in a precise way the notion of running time, and then the features of the input graphs we want to colour.

Running Time and Search Tree

It is convenient to represent the history of the search process followed by DPLL, that is, the sequence of trials and errors by a search tree. Examples of search trees are given in Fig. 2. Nodes in the tree are attached to assignment of variables, while edges represent logical consequences (elimination of satisfied constraints, simplification of other constraints) resulting from these assignments. A good computer-independent measure of the complexity of resolution is the size of the search tree generated by DPLL. This search tree varies with the input of the problem under consideration *i.e.* the graph to be coloured, and the sequence of assignments carried out by the search procedure.

Random Graphs

In spite of being NP-complete, 3-COL is not always hard. Deciding whether a given graph is 3-colourable or not may sometimes be very easy. For instance, it is immediate to recognise that a square lattice is 3-colourable, irrespectively of its size (number of nodes), while a complete graph with 4 vertices (or more) is not. To obtain a reliable estimate of the performances of DPLL on 3-COL, we want to discard such instances.

A possibility is to estimate resolution complexity for some underlying probability distribution of instances. This 'average-case' behaviour depends, of course,

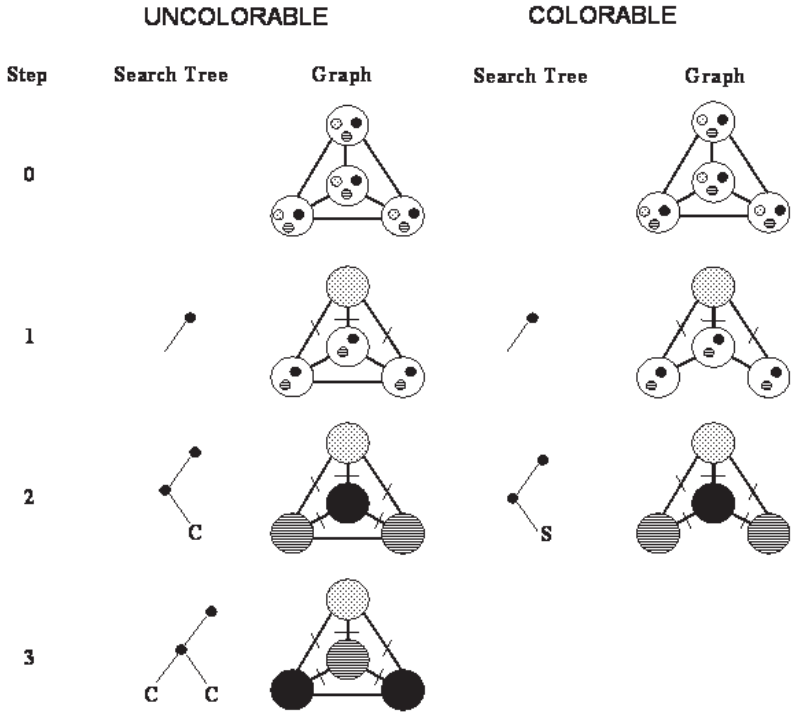

Fig. 1. Two examples which demonstrate how the DPLL algorithm acts onto a uncolourable (left side) and an colourable (right side) graph. The figure illustrates how the search tree grows with the operation of the algorithm. Available colours at each step are denoted by the patterns of the filled circles attached to vertices. When a vertex is coloured, it is removed from the graph, together with all its attached edges. In addition, the chosen colour is removed from the neighbours' sets of available colours. On the right side of the figure, a colourable graph is coloured by the algorithm. No contradiction is encountered, and the algorithm finds a solution without backtracking. On the left side, the algorithm tries to colour an uncolourable graph. When it first hits a contradiction (step 2) *i.e.* when two 1-colour vertices connected by an edge are left with the same available colour, the algorithm backtracks to the last-coloured vertex, and tries to colour it with the second available colour. When a contradiction is hit again, the algorithm terminates. Note, that in principle, it could backtrack to the first-coloured node, and try other colour options. However, due to colour gauge symmetry, this will not yield a solution.

on the input distribution and on the resolution algorithm considered [6,7]. Such distributions are usually unrealistic compared to structured instances from the real world, but are simple enough to allow for some analytical treatment. A popular input distribution for 3-COL are random graphs G *à la* Erdös-Renyì *i.e.* drawn with uniform probability among all the graphs having N vertices and E edges. The limit of interest is $N, E \to \infty$ at fixed ratio $c = 2E/N$ of edges per vertex [8–10]. Random 3-COL exhibits a phase transition phenomenon. For

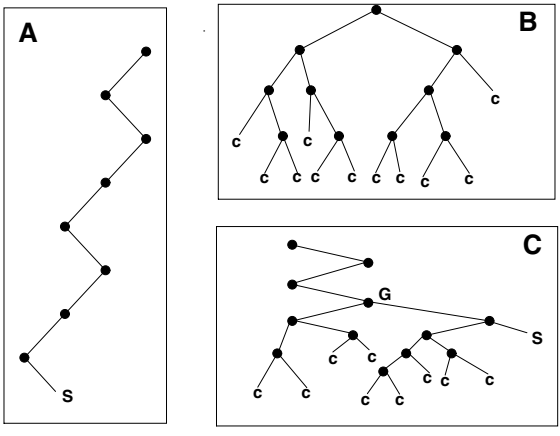

Fig. 2. Types of search trees generated by the DPLL solving procedure on 3-COL. **A.** *simple branch:* the algorithm finds easily a colouring without ever backtracking. **B.** *dense tree:* in the absence of proper colourings, DPLL builds a tree, including many branches ending with contradictory leaves, before stopping. **C.** *mixed case, branch + tree:* if many contradictions arise before reaching a solution, the resulting search tree can be decomposed into a single branch followed by a dense tree.

small values of the control parameter c, and for large input sizes, the answer to the decision problem (existence of a proper colouring) is almost surely yes. This holds as long as c remains smaller than a critical value $c_S \simeq 4.7$ called threshold, see [11] for a recent estimate of c_S with statistical mechanics techniques. Above the threshold, the answer is no with high probability.

1.3 Average Performances of DPLL. An Overview

A rigorous framework has been developed over the past years in theoretical computer science to understand the performances of DPLL in the yes region, at small enough ratios. There, decision is easily reached through a search tree as in Fig. 2A. The complexity of search is linear in the graph size, and essentially no backtracking takes place. This property allows for a rigorous analysis. It enormously simplifies the search by turning it into a Markovian process where a node in the graph is never seen twice, and the quenched character of the random graph to be coloured is not essential. The underlying structure (graph) is dynamically annealed. An informal account of the analysis of the operation of DPLL at small graph degree is presented in Sect. 2.

The situation is totally different above the threshold c_S. Proving the absence of solution requires the building up of a search tree like the one of Fig. 2B[3]. Though it is proven that the size of the search tree is exponentially large (in N) [12], not much more is known. In particular, no quantitative characterisation

[3] Massive backtracking is also present slightly below threshold where proper colourings are found at the price of intense computational effort, see search tree in Fig. 2C.

of this tree has been obtained by theoretical computer scientists so far. Due to massive backtracking, DPLL attempts to colour a node many different times at different stages of the search process. Furthermore, the Markovian character is lost: partial colourings of the graph, associated to nodes in the search tree, have to be stored to allow for the search to resume after a contradiction terminates a branch.

The understanding of DPLL operation in presence of massive backtracking is a formidable task from a probabilistic point of view. Recently, studies inspired from out-of-equilibrium statistical mechanics have permitted to tackle this problem to some extent. We present in Sect. 3 some of the ideas and results obtained along these lines. Finally, criticisms and remarks are given in Conclusion.

2 Colouring in the Absence of Backtracking

We first briefly review the 'rigorous' analysis of DPLL acting on an random input of 3-COL with small average vertex degree.

2.1 Main Features of the Search Heuristic

The action of the colouring procedure, illustrated in Fig. 1, is described as follows:

- *List of available colors:* while running, the algorithm maintains for each uncoloured vertices, a list of available colours, which consists of all the colours that can be assigned to this vertex. A node with $j(= 1, 2, 3)$ available colors is called j–colour node.
- *List-Updating:* to ensure that no adjacent vertices have the same colour, whenever a vertex is assigned a colour, this colour is removed from the lists (if present) attached to each of the uncoloured neighbours.
- *Colouring Order:* most constrained vertices *i.e.* with the least number of available colours are coloured first. At each step, a vertex is chosen among the most constrained vertices, and a colour is selected from the list of available colours. Both choices are done according to some heuristic rule, which can be unbiased (no preference is made between colours), or biased (following a hierarchy between colours), see next section.
- *Contradictions and Backtracking:* a contradiction occurs as soon as one of the lists becomes empty. Then, the algorithm backtracks to the most recently chosen vertex, which have more than one available colour (the closest node in the search tree - see definition below).
- *Termination Condition:* the algorithm stops when all vertices are coloured, or when all colouring possibilities have been tried.

Let us call Greedy heuristic the incomplete version of the above algorithm, obtained when the algorithm stops if a colouring is found (and outputs "Colourable"), or just after the first contradiction instead of backtracking (and

outputs "Don't know if colourable or not"). In contrast to the algorithm with backtracking, the Greedy heuristic is not able to prove the absence of solution, but is amenable to rigorous analysis [13,9]. In the simplest case, vertices and colours are chosen purely randomly without any bias between colours (Colouring Order step described above). This Greedy heuristic enjoys two key properties. The first one is a statistical invariance: throughout the execution of the algorithm, the uncoloured part of the graph is uniformly randomly distributed, with an average vertex degree equal to $c(1-t)$ where t is the fraction of coloured vertices. The second property is colour symmetry: the search heuristic is symmetric with respect to the different colours, and the initial conditions are symmetric as well. Hence, the evolution of the algorithm can be monitored by tracking of the three numbers $N_j(T)$ of j-colour nodes ($j = 1, 2, 3$) without distinction between the colours available to each of these nodes.

2.2 Dynamics, Concentration, and Fluctuations of Node Populations

The evolution of these numbers in the course of the colouring was analysed by Achlioptas and Molloy [13]. evolution equations for the three populations of vertices read,

$$\begin{aligned} N_3(T+1) &= N_3(T) - w_2(T), \\ N_2(T+1) &= N_2(T) + w_2(T) - w_1(T) - \delta N_1(T), \\ N_1(T+1) &= N_1(T) + w_1(T) - (1 - \delta N_1(T)), \end{aligned} \tag{1}$$

where $\delta N_1(T) = 1$ if $N_1(T) = 0$ (a 2-colour vertex is coloured) and $\delta N_1(T) = 0$ if $N_1(T) \neq 0$ (a 1-colour vertex is coloured). Quantities $w_2(T)$ and $w_1(T)$ are the 'flows' of vertices from $N_3(T)$ to $N_2(T)$, and from $N_2(T)$ to $N_1(T)$ respectively (Fig. 3). These are stochastic numbers depending on the graph under consideration and on the random choices made by the Greedy heuristic.

As a result of the additivity of (1), some concentration phenomenon takes place in the large size limit. The numbers of j–colour nodes do not fluctuate too much,

$$N_j(T) = n_j(T/N)\ N + o(N). \tag{2}$$

where the n_j's are the population densities averaged over the graph (quenched disorder) and the choices of colours ("thermal" disorder). In other words, the densities of j–colour nodes are self-averaging quantities and we shall attempt at calculating their mean values only. Note that, in order to prevent the occurrence of contradictions, the number of 1-colour nodes must remain small and the density n_1 has to vanish.

Formula (1) also illustrates another essential feature of the dynamics of populations. Two time scales are at play. The short time scale, of the order of the unity, corresponds to the fast variations of the numbers of clauses $N_j(T)$ ($j = 1, 2, 3$). When time increases from T to $T + O(1)$ (with respect to the size N), all N_j's vary by $O(1)$ amounts. Consequently, the densities n_j of nodes, that

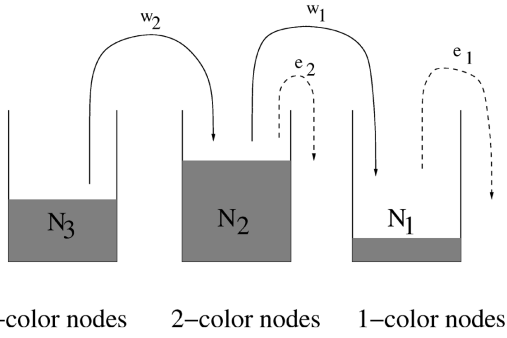

Fig. 3. Schematic view of the dynamics of the Greedy heuristic. Nodes are sorted into three recipients according to the number of available colours. Each time a node is coloured by DPLL, populations N_1, N_2, N_3 are modified, resulting in a dynamics of the recipients populations (lines with arrows). Bold lines represent the reduction of 3-nodes into 2-nodes, or 2-nodes into 1-nodes with flows denoted by w_2, w_1 respectively. A solution is found when all recipients are empty. The level of the rightmost recipient coincides with the number of 1-colour nodes. If this level is low (*i.e.* $O(1)$), the probability that two 1-colour nodes (with the same available colour) are adjacent on the graph is vanishingly small. When the level is high (*i.e.* $O(\sqrt{N})$), contradictions will occur with high probability. Flow e_1 (respectively e_2) is equal to unity if the node coloured by the Greedy heuristic is chosen from the 1-colour (resp. 2-colour) recipient.

is, their numbers divided by N, are changed by $O(1/N)$ only. The densities n_js evolve on a long time scale of the order of N and depend on the reduced time $t = T/N$ only.

Due to the concentration phenomenon underlined above, the densities $n_j(t)$ will evolve in a deterministic way with the reduced time t. On the short time scale, the relative populations $\Delta N_j(T) = N_j(T) - N\ n_j(T/N)$ fluctuate (with amplitude $\ll N$) and are stochastic variables. As said above the evolution process for these relative numbers of clauses is Markovian and the probability rates (master equation) are functions of slow variables only, *i.e.* of the reduced time t and of the densities n_2 and n_3. On intermediary time scales, much larger than unity and much smaller than N, the ΔN_js reach some stationary distribution that depend upon the slow variables.

To sum up, the dynamical evolution of the clause populations may be seen as a slow and deterministic evolution of the j–colour nodes densities to which are superimposed fast, small fluctuations. The equilibrium distribution of the latter adiabatically follows the slow trajectory. This scenario is sketched in Fig. 4.

2.3 Resolution Trajectories and Percolation

Due to the statistical invariance property, the average flows of vertices $w_2(T)$ and $w_1(T)$ can be easily calculated. Each time a node is coloured, its 3-colour neighbors are turned into 2-colour vertices. The average number of neighbours is $c(1-t)$, and the probability that a neighbour is a 3-colour nodes (prior to

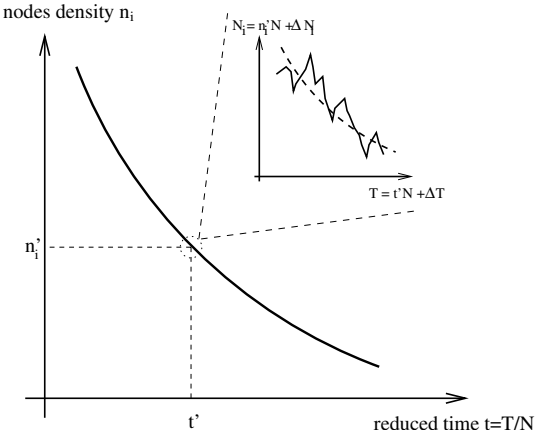

Fig. 4. Deterministic versus stochastic dynamics of the nodes population N_i as a function of the number of steps T of the algorithm. On the slow time scale (reduced time $t = T/N$), the density $n_i = N_i/N$ of (2- or 3-colour) nodes varies smoothly according to a deterministic law. Blowing up of the dynamics around some point t', n_i' shows the existence of small and fast fluctuations around this trajectory. Fluctuations are stochastic: their distribution depends upon the slow variables t', n_i'.

coloring) equals $N_3(T)/(N(1-t))$, leading to $w_2(T) = c\,N_3(T)/N$. A similar argument gives $w_1(T) = 2\,c\,N_2(T)/(3\,N)$. Thus, the evolution equations for the densities are

$$\frac{dn_3(t)}{dt} = -c\,n_3(t), \qquad \frac{dn_2(t)}{dt} = c\,n_3(t) - 1. \tag{3}$$

The solution of these differential equations, with initial conditions $n_3(0) = 1$, $n_2(0) = 0$, is $n_3(t) = e^{-c\,t}$, $\quad n_2(t) = 1 - t - e^{-c\,t}$. Eqs. (3) were obtained under the assumption that $n_2(t) > 0$ and hold until time $t = \tau$ defined through,

$$1 - \tau = e^{-c\,\tau}, \tag{4}$$

at which the density n_2 of 2-colour nodes vanishes. For $t > \tau$, 2-colour vertices do not accumulate anymore. They are coloured as soon as they are created. 1-colour vertices are almost never created, and the vertices coloured by the algorithm are either 2-, or 3-colour vertices. Thus, when $\tau < t < 1$, $n_2(t) = 0$, and $n_3(t) = 1 - t$ decreases to zero. A proper colouring is found at $t = 1$ *i.e.* when all nodes have been coloured, see trajectory in Fig. 5. Notice that, if $c < 1$, $\tau = 0$ and n_2 vanishes at all times.

The interpretation of the change taking place at time τ is simple. Let us first assume that $c < 1$, then $\tau = 0$. The random graph to be coloured is essentially made of small trees[4]. Colouring of such a tree is straightforwardly done by the Greedy procedure. Since the graph is essentially a tree[5], colouring starts from one

[4] There may also exist small odd cycles e.g. triangles easily colourable with 3 colours.
[5] The presence of unicycle does not affect the property of being 3-colorable.

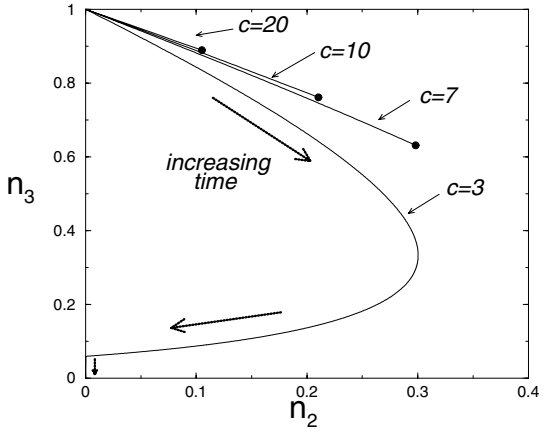

Fig. 5. Trajectories of dominant search branches generated by DPLL in the uncolourable phase ($c > c_3 \simeq 4.7$) compared to a search trajectory in the easy colourable phase ($c < c_L \simeq 3.85$). Horizontal and vertical axis represent the densities n_2 and n_3 of 2- and 3-colour nodes respectively. Trajectories are depicted by solid curves, and the arrows indicate the direction of motion (increasing depth of the search tree); they originate from the left top corner, with coordinates ($n_2 = 0, n_3 = 1$), since all nodes in the initial graph are 3-colour nodes. Dots at the end of the uncolourable trajectories ($c = 7, 10, 20$) symbolise the halt point at which condition $n_2 < 3 \ln 2/c$ ceases to be fulfilled, and the search tree stops growing. Note that as the initial connectivity increases, the trajectories halt at earlier stage, implying the early appearance of contradictions as the problem becomes over-constrained (large connectivity values). The colourable trajectory (shown here for $c = 3$) represents the under-constrained region of the problem, where the very first search branch is able to find a proper colouring (bottom left corner with coordinates ($n_2 = 0, n_3 = 0$)).

node and then proceeds, branch after branch. The lists of available colours of not-yet-coloured nodes contain at least two colours. During this process, n_3 decreases linearly with time T, while $N_2(T)$ is bounded from above by the number of nodes in the tree. Hence the density of 2-colour nodes vanishes. Assume now that $c > 1$. A randomly drawn graph is now made of a giant percolating component including a fraction of the vertices equal to τ defined in eqn (4), while the fraction of remaining nodes, $1 - \tau$, almost surely belong to small trees. During the initial stage of the colouring process e.g. $1 \ll T \ll N$, the probability that a node belonging to the giant component is chosen by the Greedy heuristic tends to one. On the average, the number of its neighbours equals c, and this coincides with the number N_2 of 2-colour nodes created. Next, only one of these nodes is coloured, resulting in an average net creation rate of $c - 1$ 2-colour nodes. Hence N_2 initially grows, and the density n_2 becomes finite. This goes on until all the nodes in the percolating component have been assigned some colour *i.e.* up to time τ. Later on, the Greedy algorithm is left with the colouring of the remaining small trees (resolution trajectory at zero 2-colour density in Fig. 5).

Motion equations (3) are valid as long as no contradiction occurs. The number of 1-colour vertices must remain small throughout the execution of the algorithm. Clearly, 1-colour nodes are created slowly enough to be coloured ans eliminated, and do not accumulate provided that $w_1(t) < 1$. For $c < c_L \approx 3.847$, this condition is never violated, and the probability that the algorithm succeeds in finding an appropriate colouring without backtracking is positive [6]. For $c_L < c < c_S$, the condition is violated at $t = t_d(c)$ which depends on c, and 1-colour vertices start to accumulate. As a result, the probability for contradictions becomes large, and backtracking enters into play with a search tree sketched in Fig. 2C.

3 Colouring in the Presence of Massive Backtracking

The analysis of the DPLL algorithm in presence of backtracking was initiated on the random SAT problem. The case of 3-COL is qualitatively similar. Hereafter, we study the average complexity of showing that a random graph G with average degree c is not 3-colourable (as happens with high probability if $c > c_S$).

3.1 From Depth-First to Breadth-First Search: The Markovian Evolution Matrix

The probabilistic analysis of DPLL in the uncolourable regime appears to be a formidable task since the search tree of Fig. 2B is the output of a complex, sequential process: nodes and edges are added by DPLL through successive descents and backtrackings (depth-first search). We have imagined a different, breadth-first building up of the refutation tree, which results in the same complete tree but can be mathematically analysed. In our imaginary process, the tree grows in parallel, layer after layer (Fig. 6). At time $T = 0$, the tree reduces to a root node, to which is attached the initial colouring E consisting in assigning to one randomly chosen vertex, say vertex number 1, a colour, say, Red, and an attached outgoing edge. Nothing else is known at the beginning of the search process. We suppose that the graph G is connected, and not 3-colourable. At time T, that is, after having coloured T vertices of the graphs attached to each branch, the tree is made of $B(T)$ ($\leq 2^T$) branches, each one carrying a partial colouring. At next time step $T \to T + 1$, a new layer is added by colouring, according to DPLL heuristic, one more node along every branch. As a result, a branch may keep growing through 1-colour node colouring, get hit by a contradiction and die out, or split if the colouring proceeds through 2-colour node colouring.

This parallel growth process is Markovian, and can be encoded in an instance–dependent evolution operator **H**. A detailed definition and construction of **H** for the SAT problem is presented in [15]. We hereafter expose the main steps:

[6] See ref. [14] for a recent study of the occurrence of exponentially hard resolutions with massive backtracking in this range of connectivities.

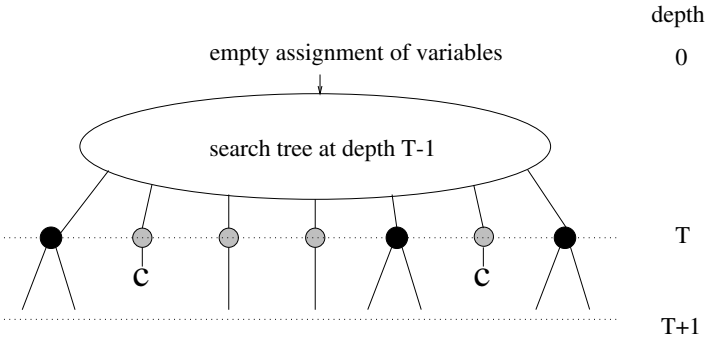

Fig. 6. Imaginary, parallel growth process of a search tree proving uncolourability used in the theoretical analysis. Colouring proceeds along the DPLL rule, but branches evolve in parallel. T denotes the depth in the tree, that is the number of nodes coloured by DPLL along each branch. At depth T, one node is chosen on each branch among 1-colour nodes (grey circles not represented in Fig. 2), or 2-, 3-colour nodes (splitting, black circles as in Fig. 2). If a contradiction occurs, the branch gets marked with C and dies out. The growth of the tree proceeds until all branches carry C leaves. The resulting tree is identical to the one built through the usual, sequential operation of DPLL.

- A 4^N dimensional–vector space $\mathbf{V}$ is introduced. Each vector $|S\rangle$ in the spanning basis is in one–to–one correspondence with a partial colouring $S = (s_1, s_2, \ldots, s_N)$ of the N vertices, where $s_i = R$, G, B if vertex i is coloured with colour R, G, B respectively, or $s_i = U$ (Unknown) if vertex i has not been assigned any colour yet.
- Let S be a partial colouring for the graph under consideration with $s_j = U$ for some vertex j. Then $S^{(j,x)}$ is the partial colouring obtained from S by assigning colour $x(= R, G, B)$ to vertex j. Call $h_n(j|S)$ and $h_v(x|S, j)$ the probabilities that the Greedy heuristic respectively chooses vertex j when presented the graph with partial colouring S, and then colour it with colour x.
- For a partial colouring S, we denote by $N_i(S)$ the number of uncoloured nodes having i available colours *i.e.* having coloured neighbours of $3 - i$ different colours.
- The evolution operator $\mathbf{H}$ encodes the action of DPLL. Its matrix elements in the spanning basis are, see Fig. 7,
  1. if S is an improper colouring of G, that is if two adjacent vertices have the same colour in S, $\langle S'|\mathbf{H}|S\rangle = 1$ if $S' = S$, 0 otherwise.
  2. if S is a partial but not improper colouring of G, $\langle S'|\mathbf{H}|S\rangle = h_n(j|S) \times h_v(x|S, j)$ if $N_1(S) \geq 1$ and $S' = S^{(j,x_1)}$, $h_n(j|S)$ if $N_1(S) = 0$ and ($S' = S^{(j,x_2)}$ or $S' = S^{(j,x_2')}$), 0 otherwise. Here S, S' are the partial colourings attached to $|S\rangle, |S'\rangle$, x_1 (respectively x_2, x_2') denote the colour(s) available to the 1-colour (resp. 2-colour) node[7].

[7] Notice that, under our assumption that the graph is connected, there will always be at least one node with two or one available colours.

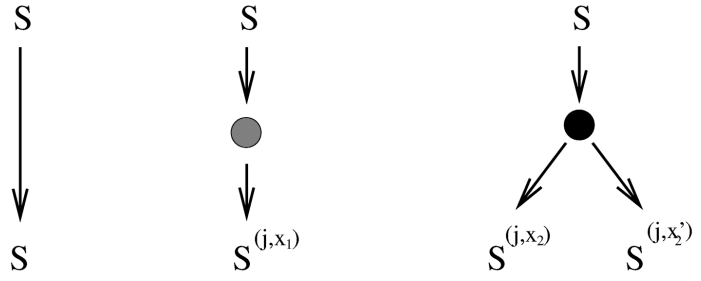

Fig. 7. Transitions allowed by the heuristic-induced evolution operator. Grey and black nodes correspond to the case of colouring of 1-colour and 2-colour nodes respectively, as in Fig. 6. **A**. If partial colouring S is improper for G, it is left unchanged. **B**. If the partial colouring has not led to any contradiction yet and there is at least 1-colour node, a 1-colour node is chosen, say j, and coloured as required, say with colour $s_j = x$ (grey node). The output partial colouring is $S^{j,x}$. **C**. If the partial colouring is not improper and there is no 1-colour node, a 2-colour node is coloured through splitting (black node). Two partial colourings are generated, S^{j,x_2} and $S^{j,x_2'}$ where $s_j = (x_2, x_2')$.

3.2 Gluing Partial Proofs of Uncolourability and Time Reversal

In this section, we show that the expectation value over the random colourings of variables of the size (number of leaves) of the search tree produced by DPLL to prove the uncolourability of G is equal to

$$B = \sum_S \langle S|\mathbf{H}^N|E\rangle, \tag{5}$$

where $\mathbf{H}^N$ denotes the N^{th} matrix power of $\mathbf{H}$, the sum runs over all 4^N partial colourings S, and the rightmost vector $|E\rangle = |R, U, U, \ldots, U\rangle$ corresponds to the initial colouring of G [15].

Let S be a partial colouring. We call refutation tree built from S a complete search tree that proves the uncolourability of G conditioned to the fact that DPLL is allowed to colour only nodes which are uncoloured in S. The height of the search tree is the maximal number of colourings leading from the root node (attached to partial colouring S) to a contradictory leaf. Let $b_T(S)$ be the average size (number of leaves) of refutation trees of height $\leq T$ that can be built from partial colouring S. Let us call W the set of improper colourings of G *i.e.* of partial colourings S assigning the same colour to two adjacent vertices. Clearly, $b_T(S) = 1$ if $S \in W$ and $b_T(S) \geq 2$ otherwise.

Assume now T is an integer larger or equal to 1, S a partial colouring with $N_1(S)$ 1-colour nodes. Our parallel representation of DPLL allows us to write simple recursion relations:

1. if $S \in W$, $b_T(S) = 1 = b_{T-1}(S)$.
2. if $S \notin W$ and $N_1(S) \geq 1$,

$$b_T(S) = \sum_{j=1}^{N} \sum_{x_1} h_n(j|S)\, h_v(x_1|S,j)\; b_{T-1}\big(S^{(j,x_1)}\big). \tag{6}$$

3. if $S \notin W$ and $N_1(S) = 0$,

$$b_T(S) = \sum_{j=1}^{N} h_n(j|S)\; \Big[b_{T-1}\big(S^{(j,x_2)}\big) + b_{T-1}\big(S^{(j,x_2')}\big) \Big]. \tag{7}$$

In the above equations, x_1, x_2, x_2' denote colours as defined in Sect. 3.1. These three different cases are symbolised on Fig. 7A, B and C respectively. From the definition of $\mathbf{H}$, these recursion relations are equivalent to

$$b_T(S) = \sum_{S'} \langle S'|\mathbf{H}|S\rangle\; b_{T-1}(S'), \tag{8}$$

for any partial colouring S. Let $|b_T\rangle$ be the vector of $\mathbf{V}$ whose coefficients on the spanning basis $\{|S\rangle\}$ are the $b_T(S)$'s. In particular, $|b_0\rangle$ is the sum of all improper colourings in W. Then identity (8) can be written as $|b_T\rangle = \mathbf{H}^\dagger\; |b_{T-1}\rangle$ where $\mathbf{H}^\dagger$ is the transposed of the evolution operator. The apparition of the time reversal operator $\mathbf{H}^\dagger$ is very natural since we glue partial refutation trees to build bigger and bigger ones to finally refute G from the initial (almost) empty colouring E.

The average size of refutation trees of height T obtained without any *a priori* knowledge on G is simply $b_T(E) = \langle E|b_T\rangle$. Since refutation trees cannot have height larger than the number of nodes N, it is easy to show that the average size of the proof of uncolourability of G generated by DPLL is

$$B \equiv b_N(E) = \langle E|(\mathbf{H}^\dagger)^N|b_0\rangle = \sum_S \langle E|(\mathbf{H}^\dagger)^N|S\rangle = \sum_S \langle S|\mathbf{H}^N|E\rangle \tag{9}$$

as claimed in equation (5).

3.3 Dynamical Annealing and the Search Growth Process

Calculation of the expectation value of the N^{th} power of $\mathbf{H}$, and of its average over the instance distribution is a hard task. We therefore turned to a simplifying approximation, called dynamical annealing. Call population vector $\boldsymbol{N}(S)$ of a partial colouring S the three dimensional vector $\boldsymbol{N} = (N_1, N_2, N_3)$ where N_j is the number of nodes with j available colours. The quantity we focus on is $\bar{B}(\boldsymbol{N}; T+1)$, the expectation number of branches at depth T in the search tree (Fig. 6) carrying partial colourings with population vector $\boldsymbol{N} = (N_1, N_2, N_3)$. Within the dynamical annealing approximation, the evolution of the $\bar{B}$'s is Markovian,

$$\bar{B}(\boldsymbol{N};T+1)=\sum_{\boldsymbol{N}'}\bar{\mathbf{H}}\left[\boldsymbol{N},\boldsymbol{N}';T\right]\bar{B}(\boldsymbol{N}';T). \tag{10}$$

The entries of the evolution matrix $\bar{\mathbf{H}}[\boldsymbol{N},\boldsymbol{N}';T]$ can be calculated from the definition of the evolution matrix $\mathbf{N}$ [15]. They can be interpreted as the average number of branches with population vector $\boldsymbol{N}$ that DPLL will generate through the colouring of one node from a partial colouring with population vector $\boldsymbol{N}'$. We find

$$\bar{\mathbf{H}}(\boldsymbol{N},\boldsymbol{N}';T)=\sum_{w_2=0}^{N_3'}\binom{N_3'}{w_2}\left(\frac{c}{N}\right)^{w_2}\left(1-\frac{c}{N}\right)^{N_3}\delta_{N_3'-N_3-w_2}\times \tag{11}$$

$$\left\{\{(1-\delta_{N_1'})\sum_{w_1=0}^{N_2'}\binom{N_2'}{w_1}\left(\frac{2c}{3N}\right)^{w_1}\left(1-\frac{2c}{3N}\right)^{N_2'-w_1}\delta_{N_2-N_2'-(w_2-w_1)}\delta_{N_1-N_1'-w_1+1}+\right.$$

$$\left.2\,\delta_{N_1'}\sum_{w_1=0}^{N_2'-1}\binom{N_2'-1}{w_1}\left(\frac{2c}{3N}\right)^{w_1}\left(1-\frac{2c}{3N}\right)^{N_2'-w_1-1}\delta_{N_2-N_2'-(w_2-w_1-1)}\delta_{N_1-N_1'-w_1}\right\}$$

where δ_N is the Kronecker delta function. Note that (11) is written under the condition that no 3-colour nodes are chosen by the algorithm throughout the growth process. This condition is consistent with the assumption that the graph G is connected, and with the fact that initially one node is coloured.

Let us examine how a step of the algorithm affects the size of the three populations N_1, N_2, N_3. Since the average connectivity is $O(1)$ *i.e.* each vertex is connected on average only to $O(1)$ vertices, when a vertex is coloured, the number of vertices whose status (the number of available colours) is subsequently changed is bounded from above by the number of neighbours of the coloured vertex. Hence a reasonable assumption is that the densities $n_i = N_i/N$ change by $O(1)$ after $T = t \times N$ vertices are coloured. In addition, we expect that, as soon as $N_1(T)$ becomes very large, contradictions are very likely to occur, and the growth process stops. Throughout the growth process, $N_1 = O(1)$ almost surely. Thus $n_1 = 0$ with high probability. The corresponding Ansatz for the number of branches is,

$$\tilde{B}(\boldsymbol{N};T)=e^{N\,\omega(n_2,n_3;t)+o(N)} \tag{12}$$

where non-exponential terms in N depend on the populations of i-colour nodes ($i = 1, 2, 3$). At the initial stage of the tree building up, there is a single outgoing branch from the root node, carrying a fully uncoloured graph. Thus, $\bar{B}(\boldsymbol{N};T = 0) = 1$, and

$$\omega(n_2,n_3;t=0)=\begin{cases}0 & \text{if } (n_2,n_3)=(0,1)\ ,\\ -\infty & \text{if } (n_2,n_3)\neq(0,1)\ .\end{cases}$$

Insertion of Ansatz (12) into evolution equation (10) leads to the following partial differential equation for the logarithm $\omega(n_2, n_3; t)$ of the average number of branches with densities n_2, n_3 of 2- ,3-colours nodes [16],

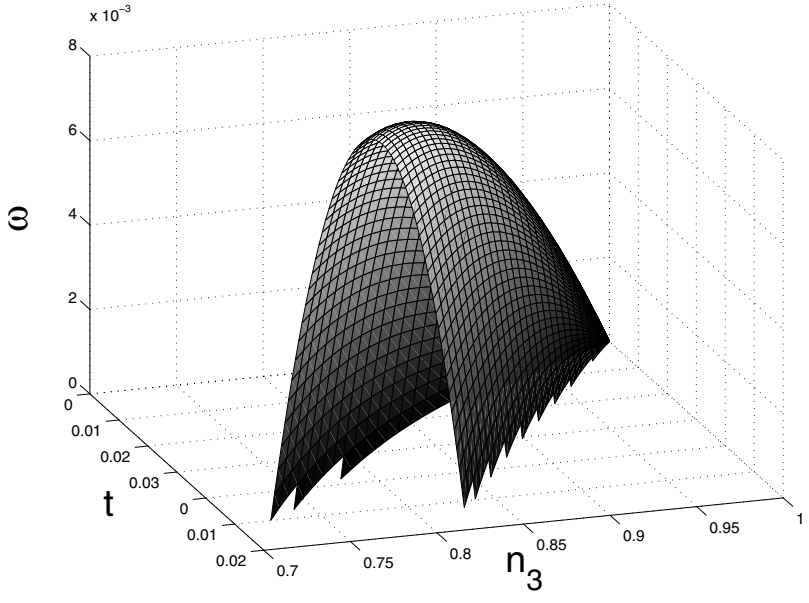

Fig. 8. Function ω (log. of number of branches with densities $n_2 = 1 - t - n_3$, n_3 of 2- and 3-colour nodes at depth t in the search tree) as a function of n_3 and t for $c = 10$. The top of the curve at given time t, $\omega^*(t)$, is reached for the dominant branch 3-colour density $n_3^*(t)$. The evolution of ω is shown till $t = t_h$ at which dominant branches in the search tree stop growing (die from the onset of contradictions). The maximal ω at t_h, $\omega^*(t_h)$, is the theoretical prediction for the complexity.

$$\frac{\partial \omega}{\partial t} = \frac{\partial \omega}{\partial n_2} + \ln 2 - \frac{c}{3}\, n_2 + c\, n_3 \left[\exp\left(\frac{\partial \omega}{\partial n_3} - \frac{\partial \omega}{\partial n_2} \right) - 1 \right] . \tag{13}$$

The surface ω, growing with "time" t above the plane n_2, n_3 describes the whole distribution of branches. Here, this distribution simplifies due to nodes conservation. The sum $n_2 + n_3$ of 2- and 3-colour nodes densities necessarily equals the fraction $1 - t$ of not-yet coloured nodes. Therefore, ω is a function of n_3 and t only, whose expression is obtained through the exact resolution of (13) with the above initial condition (see Sect. 4 for comments on the resolution),

$$\omega(n_3; t) = \frac{c}{6}\, t\, (1 - 2\,t - 4\,n_3) - n_3\, \ln\, n_3 - (1 - n_3)\, \ln (1 - n_3) - (1 - t - n_3) \ln 2 + (1 - n_3)\, \ln \left[3 \left(1 - e^{-\,2\,t\,c/3} \right) \right] . \tag{14}$$

Figure 8 exhibits $\omega(n_3, t)$ for $c = 10$.

The maximum $\omega^*(t)$ over n_2, n_3 of $\omega(n_2, n_3; t)$ at depth t in the tree gives the logarithm of the average number of branches at depth t divided by N. The search tree keeps growing as long as no contradictions are encountered *i.e.* as long as 1-colour vertices do not accumulate. This amounts to say that dominant branches are not suppressed by contradictions and become more and more nu-

Table 1. Analytical results and simulation results of the complexity ω for different connectivities c in the uncolourable phase. The analytical values of ω_{THE} are derived from theory; ω_{EXP} is obtained through experimental measures of the search tree size and averages over input graphs [16].

c	ω_{THE}	ω_{NOD}
20	$2.886 * 10^{-3}$	$3 * 10^{-3} \pm 3 * 10^{-4}$
15	$5.255 * 10^{-3}$	$5.8 * 10^{-3} \pm 5 * 10^{-4}$
10	$1.311 * 10^{-2}$	$1.5 * 10^{-2} \pm 1 * 10^{-3}$
7	$2.135 * 10^{-2}$	$3. * 10^{-2} \pm 3.6 * 10^{-3}$

merous through 2-colour nodes colouring, $d\omega^*/dt > 0$. Call t_h the halt time at which this condition ceases to be fulfilled. The logarithm $\omega^*(t_h)$ of the number of dominant branches at $t = t_h$, when divided by $\ln 2$, yields our analytical estimate for the complexity of resolution. Agreement between theory and numerics is very good at large c (see Table 1) but deteriorates at small c. However, the high computational complexity of the algorithm for small c values, does not allow us to obtain numerical results for large sizes N, and affects the quality of the large N extrapolation of ω. As c increases, contradictions emerge in an earlier stage of the algorithm, the probability that the same vertex appears in different branches reduces, and the analytical prediction becomes exact. As a consequence of the early appearance of contradictions, the complexity ω decreases with c. At very large c, we find

$$\omega(c) \asymp \frac{3 \ln 2}{2} \frac{1}{c^2} \simeq \frac{1.040}{c^2} \quad , \tag{15}$$

and therefore that the (logarithm of the) complexity exhibits a power law decay with exponent 2 as a function of connectivity c.

3.4 More Than Three Colours

The scaling exponent appearing in eqn (15) strongly depends on the number of colours, here three. The whole procedure described above can be extend to the study of K-COL, where K is the number of colours. Let us concentrate on the case of large degrees c, where the output of the dynamical annealing procedure is conjectured to be exact. We find that the logarithm of the average complexity scales as

$$\omega(c) \asymp \Omega(K)\ c^{-(K-1)/(K-2)} \tag{16}$$

where the constant Ω depends on the details of the Greedy heuristic. For the heuristic analysed in Sect. 1.2 with preferential colouring of nodes with the smallest number of available colours,

$$\Omega(K) = \frac{K(K-2)}{K-1} \left[\frac{2 \ln 2}{K-1}\right]^{1/(K-2)} \quad . \tag{17}$$

A less sophisticated Greedy heuristic corresponds to colouring nodes with one available color if any, any node otherwise. The multiplicative factor entering eqn (16) is then,

$$\Omega(K) = \frac{K(K-2)}{K-1}\ \ln K\ \left[\frac{K\ \ln K}{(K-1)^2}\right]^{1/(K-2)} \quad . \tag{18}$$

Recent works suggest that the correctness of the decay exponent in eqn (16) could be rigorously established [17].

4 Conclusions: What Is Missing?

In this article, we have presented a quantitative study of the search tree growth process accompanying the backtrack resolution of the random graph colouring problem, especially in presence of massive backtracking. Here are some remarks regarding the approach:

1. From a mathematical point of view, it is worth noticing that monitoring the growth of the search tree requires a partial differential equation, while ordinary differential equations are sufficient to account for the evolution of a single branch [18]. Yet, the partial differential equation is of the first-order[8], and can be solved using the characteristics method based on the use of an appropriate set of ordinary differential equations only. Dominant branches at time $t' > 0$ are indeed completely described by the set of coupled ordinary differential equations, see eqn (13),

$$\frac{dn_3(t)}{dt} = -c\ n_3(t)\ \exp\big(\psi(t)\big)\ , \tag{19}$$

$$\frac{d\psi(t)}{dt} = c\ \exp\big(\psi(t)\big) - \frac{2}{3}\ c, \tag{20}$$

for all intermediate times $0 < t < t'$ with the boundary conditions $n_3(0) = 1$ and $\psi(t') = 0$. The new field ψ is related to the derivative of the surface ω at density n_3. Notice the equivalence of equation (19) when $\psi = 0$ and the dynamical equation (3) describing a single branch. In other words, the equations for the evolution of a search tree look like to the ones of a single branch in the absence of backtracking up to the presence of a statistical bias imposed by the field ψ.
Remarkably, the idea of an effective branch was at the base of the phenomenological approach proposed by Knuth three decades ago [1]. The statistical physics approach is, to some extent, an analytical realization of this idea.

[8] This statement is correct in the large size limit only. Finite size corrections would introduce second derivative terms with $1/N$ multiplicative coefficients. See [19] for a similar situation.

2. It is likely from the above discussion and numerical experiments that our theory is exact at large connectivity c but requires some corrections for small connectivities. These corrections come from the fact that we have neglected statistical correlations between branches in the search tree resulting from the average over the random graph to be colored. Though these correlations are expected to vanish at large c, they should give some contribution for finite c to the equations describing the dynamical evolution of the search tree. What is the structure of this contribution? A precise answer to this question will come from a detailed study of operator **H** along the lines recently initiated for another out-of-equilibrium system called the Contact Process [20]. It may be guessed that the output of such a calculation will be the emergence a systematic $1/c$ expansion for n_3 with non Markovian retarded terms on the right hand side of equation (19) involving the value of the density of 3-color nodes at all times $< t$. This structure is expected from the similarity of the present problem and the one treated in [20], and the fact that neglecting correlations between branches in the search tree precisely amounts to neglecting all non-Markovian terms in the evolution of the search tree[9].
3. From a qualitative point of view, the average complexity of DPLL for the 3-COL problem is linear for $c < c_L$ with finite probability, and exponential for $c > c_L$. Right at the location of the cross-over, that is, for connectivity $c = c_L$, we expect an average complexity growing as a stretched exponential of the size of the graph. A precise determination of the exponent is under way.
4. The present study is part of a general effort to understand the extremal statistics of correlated variables, and is related to studies on random binary trees to which statistical mechanics ideas have recently been applied [21]. From this point of view, it would be interesting to see to what extent the results presented in this article are changed when the input graph distribution is modified e.g. for random graphs with preferential attachment and power-law degree distributions.
5. Last of all, the study of the operator **H** is interesting regardless of its computer science interpretation. **H** is a non Hermitean evolution operator, and the halt of dominant branches is deeply related to a localization vs. delocalization transition of its dominant eigenvector [4].

In view of the above remarks, one may be quite confident that a major understanding of the average-case performances of backtracking algorithms will be obtained in the next future. To what extent these results will find rigorous support is however not clear.

[9] To be more precise, as stated in Sect. 3.1, the parallel coloring process is Markovian for a fixed graph but is not any longer once the average over the underlying graph has been carried out.

Acknowledgements

This work is the fruit of a long standing collaboration with S. Cocco on the Satisfiability problem. Results on graph Colouring have been obtained in collaboration with L. Ein-Dor. The discussion on the $1/c$ expansion is based on a recent work with C. Deroulers. I am grateful to O. Dubois and C. Moore for stimulating and very useful discussions. Partial support from the ACI Jeunes Chercheurs "Algorithmes d'optimisation et systèmes désordonnés quantiques" is acknowledged.

References

1. Knuth, D.E. Estimating the efficiency of backtrack programs, *Math. Comp.* **29**, 12-136 (1975).
2. Garey, M. R. and Johnson, D. S, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, San Fransico (1979).
3. Knuth, D.E. Selected Papers on Analysis of Algorithms, Center for the Study of Language and Information Lecture Notes 102, Stanford CA (2000).
4. Cocco, S. and Monasson, R. Trajectories in phase diagrams, growth processes and computational complexity: how search algorithms solve the 3-Satisfiability problem, *Phys. Rev. Lett.* **86**, 1654 (2001); Analysis of the computational complexity of solving random satisfiability problems using branch and bound search algorithms, *Eur. Phys. J. B* **22**, 505 (2001).
5. Davis, M., Logemann, G., Loveland, D. A machine program for theorem proving. *Communications of the ACM* **5**, 394-397 (1962).
6. Mitchell, D., Selman, B. and Levesque, H. Hard and Easy Distributions of SAT Problems, *Proc. of the Tenth Natl. Conf. on Artificial Intelligence (AAAI-92)*, 440-446, The AAAI Press / MIT Press, Cambridge, MA (1992).
7. Turner, J. S. Almost All k-Colorable Graphs Are Easy to Color, *Journal of Algorithms* **9**, 63–82 (1988).
8. Achlioptas, D. and Friedgut, E. A sharp threshold for k-colorability, *Random Structures and Algorithms* **14(1)**, 63–70 (1999).
9. Achlioptas, D. and Moore, C. Almost all graphs with average degree 4 are 3-colorable *Proc. on 34th Annual ACM Symposium on Theory of Computing, May 19-21* , Montreal, Quebec, Canada, ACM, Montreal, 199–208 (2002)
10. Culbersome, J. C. and Gent, I. P. Frozen development in graph coloring, *Theor. Comp. Sci.* **265(1-2)**, 227–264 (2001).
11. Mulet, R., Pagnani, A., Weigt, M. and Zecchina R. Coloring random graphs. *Phys. Rev. Lett.* **89**, 268701 (2002).
12. Chvàtal, V. and Szmeredi, E. Many hard examples for resolution, *Journal of the ACM* **35**, 759–768 (1988).
13. Achlioptas, D. and Molloy, M. Analysis of a List-colouring Algorithm on a Random Graph, *Proc. of FOCS 97* 204 (1997).
14. Jia, H., and Moore, C. How much backtracking does it take to color sparse random graphs? Rigorous results on heavy tails. *preprint* (2003).
15. Cocco, S. and Monasson R. Heuristic average-case analysis of backtrack resolution of random 3-Satisfiability instances, to appear in *Theoretical Computer Science* (2004).

16. Ein-Dor, L. and Monasson, R. The dynamics of proving uncolorability of large random graphs. I. symmetric colouring heuristic, *J. Phys. A* **36**, 11055 (2003).
17. Beame, P., Culberson, J., Mitchell, D. and Moore, C. The resolution complexity of random graph k-colorability, *preprint* (2004).
18. Achlioptas, D. Lower bounds for random 3-SAT via differential equations, *Theor. Comp. Sci.* **265**, 159–185 (2001).
19. Griffiths, R.B., Weng, C-H. and Langer, J.S. Relaxation times for metastable states in the mean-field model of a ferromagnet, *Phys. Rev.* **149**, 301 (1966).
20. Deroulers, C. and Monasson, R. Field theoretic approach to metastability in the contact process. *Phys. Rev. E* **69** 016126 (2004).
21. Majumdar, S.N. and Krapivsky, P.I. Extreme value statistics and traveling fronts: an application to computer science. *Phys. Rev. E* **65**, 036127 (2002).