Satisfiability-unsatisfiability transition in the adversarial satisfiability problem

Marco Bardoscia[*]
Abdus Salam International Centre for Theoretical Physics, Strada Costiera 11, 34151 Trieste, Italy

Daniel Nagaj[†]
Dept. of Physics, University of Vienna, Boltzmanngasse 5, 1090 Vienna, Austria and Institute of Physics, Slovak Academy of Sciences, Dúbravská cesta 9, 845 11 Bratislava, Slovakia

Antonello Scardicchio[‡]
Abdus Salam International Centre for Theoretical Physics, Strada Costiera 11, 34151 Trieste, Italy and INFN, Section of Trieste, Strada Costiera 11, 34151 Trieste, Italy

(Received 4 October 2013; published 24 March 2014)

Adversarial satisfiability (AdSAT) is a generalization of the satisfiability (SAT) problem in which two players try to make a Boolean formula true (resp. false) by controlling their respective sets of variables. AdSAT belongs to a higher complexity class in the polynomial hierarchy than SAT, and therefore the nature of the critical region and the transition are not easily parallel to those of SAT and worthy of independent study. AdSAT also provides an upper bound for the transition threshold of the quantum satisfiability problem (QSAT). We present a complete algorithm for AdSAT, show that 2-AdSAT is in **P**, and then study two stochastic algorithms (simulated annealing and its improved variant) and compare their performances in detail for 3-AdSAT. Varying the density of clauses α we claim that there is a sharp SAT-UNSAT transition at a critical value whose upper bound is $\alpha_c \lesssim 1.5$, suggesting a much stricter upper bound for the QSAT transition than those previously found.

DOI: 10.1103/PhysRevE.89.032128 PACS number(s): 02.50.−r, 75.10.Nr, 89.70.Eg, 03.67.−a

I. INTRODUCTION

The study of random ensembles of decision problems has grown into a fertile field of investigation where the methods of statistical physics have found applications to the theory (and practice) of hard combinatorial problems. This resulted in a wealth of intuition on the nature of the typical complexity of hard decision problems and in a new, efficient family of algorithms [1,2]. One such problem is a random ensemble of satisfiability (SAT), where Boolean formulas are generated in a random way and tested for a solution. If the formula is restricted to be of the form of a conjunction of an arbitrary number of clauses, and each clause is the logical disjunction of K variables, the problem is denoted by K-SAT. The ensemble is determined once the number of clauses per variable is fixed. As this ratio is increased the formulas go from being typically satisfiable to being typically unsatisfiable [2–4]. This is the SAT-UNSAT phase transition.

Recent progress in the study of *quantum* decision problems [5] has lead to the definition of the quantum generalization of K-SAT (we call it K-QSAT) [6]. This problem is proven to be $\mathbf{QMA}_1$ complete for $K \geqslant 3$ [7], with $\mathbf{QMA}_1$ the quantum analog of **NP** [8]. A random ensemble of K-QSAT was introduced and studied in Refs. [9–11], where it has been shown to have a SAT-UNSAT phase transition. Moreover, it has a phase with product-state solutions. The quantum version of the Lovász local lemma in Ref. [12] provides a lower bound for the SAT-UNSAT transition which implies that sufficiently close to the phase transition the solutions of the problem must be entangled (this is proven for sufficiently large K and believed true for all $K \geqslant 3$).

Following a theorem in Ref. [9], an upper bound on the quantum SAT-UNSAT threshold can be found in terms of *the most frustrated classical formula* on a given hypergraph (as will be explained in the following, the graph is determined by the membership relations between variables and clauses). This problem is interesting for at least two other reasons. First, it is a random problem of satisfiability formulas with more than one existential quantifier (it actually has two quantifiers), and, second, it is a problem of extreme-value statistics on the familiar random ensemble of K-SAT formulas.

The problem was named adversarial SAT (AdSAT) in Ref. [13], where it has been studied by means of belief and survey propagation. There it is claimed that the problem has a SAT-UNSAT transition for $K = 3$ at a clauses/variables ratio $\alpha = 3.39 \pm 0.01$ (compare with the familiar SAT-UNSAT transition of K-SAT at $\alpha = 4.27$). If this result is correct, then AdSAT improves only marginally on the upper bound for the transition in QSAT. In fact, an upper bound is found in Ref. [14] at $\alpha = 3.59$. However, from the numerics in Ref. [9] the threshold for 3-QSAT is closer to $\alpha = 1.0$ than to any one of these numbers, signaling that the physics of QSAT is not captured well by these approximations.

In this paper we numerically investigate the random ensemble of AdSAT problems. Since the problem is quite resilient to numerical analysis, we present two heuristic algorithms and study their performances. By these means, we are able to study the crossover of the SAT-UNSAT transition at varying N. For the largest system size systematically explored ($N = 15$) we observe an $\alpha_c = 1.6$ and a tendency of α_c to decrease with increasing N. We support this picture with investigations of considerably larger system sizes ($N = 100$) where we can assert that $\alpha_c < 2.70$. To reconcile these numbers with the

[*]marco.bardoscia@ictp.it
[†]dnagaj@gmail.com
[‡]ascardic@ictp.it

analytical results of Ref. [13], one concludes that the $1/N$ and $1/N^2$ corrections required here are really large [the coefficient of $1/N$ should be of $O(10^2)$].

The paper is organized as follows. In Sec. II we formally introduce the AdSAT problem, and we briefly discuss its importance from the perspective of complexity theory, showing an efficient approach to 2-AdSAT in Appendix A. In Sec. III we present a simple complete algorithm to solve the AdSAT problem. Due to the complexity of the problem we resort to a stochastic algorithm based on simulated annealing, which is introduced in Sec. IV, and in Sec. V we discuss its much improved variant, investigating the SAT-UNSAT transition for 3-AdSAT. Our conclusions are summarized in Sec. VI.

II. PRELIMINARIES

An AdSAT formula ϕ_G is a Boolean function of N Boolean variables $\{x_i\}_{i=1,\dots,N}$, which will be referred to as *bits* and KM Boolean variables $\{J_{aj}\}_{\substack{a=1,\dots,M\\ j=1,\dots,K}}$, which we call *negations*. It can be written in terms of M clauses $\{C_a\}_{a=1,\dots,M}$, each clause having the structure

$$C_a = \bigvee_{j=1}^{K} x_{aj} \oplus J_{aj}, \tag{1}$$

where x_{aj} is the bit (taken from $\{x_i\}_{i=1,\dots,N}$) occupying the jth place in the ath clause. The negation $J_{aj} \in \{0,1\}$ decides whether the variable x_{aj} or its negation $\overline{x}_{aj}$ appears in the final form of the clause C_a. For example, the clause

$$C_b = (x_3 \oplus 0) \vee (x_5 \oplus 1) \vee (x_8 \oplus 0) = x_3 \vee \overline{x}_5 \vee x_8 \tag{2}$$

has $x_{b1} = x_3, x_{b2} = x_5, x_{b3} = x_8$, and $J_{b3} = 0, J_{b5} = 1, J_{b8} = 0$. The AdSAT formula is then a conjunction of all the clauses:

$$\phi_G(x,\mathcal{J}) = \bigwedge_{a=1}^{M} C_a, \tag{3}$$

where $\mathcal{J} = \{J_{aj}\}$. Each bit can appear at most once in each clause, and each clause can appear at most once in the formula.

An AdSAT formula ϕ_G can be conveniently represented by a bipartite graph G as in Fig. 1 in which the nodes are of two kinds: variable nodes and clause nodes. A variable node is associated to each bit, and a clause node to each clause. If the bit x_i appears in the clause C_a, there is an edge between the

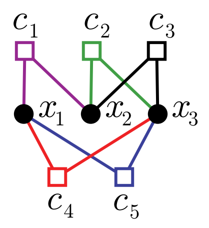

FIG. 1. (Color online) A graph of an AdSAT instance with $K = 2$, $M = 5$, and $N = 3$. Each square corresponds to a disjunctive clause (1) involving the bits (circles) or their negations. AdSAT is a game in which an adversary fixes the negations $\mathcal{J}$ on the edges and challenges a player to set the bits x so that the formula (3) is true. When the adversary has a winning strategy, the AdSAT formula ϕ_G, and its graph G is called UNSAT.

corresponding variable and clause nodes. The total number of edges is KM, and we can imagine a negation as an attribute of each edge.

An AdSAT formula is said to be SAT if

$$\phi_G \in L_{\mathrm{SAT}} \Leftrightarrow \forall \mathcal{J}, \exists x : \phi_G(x,\mathcal{J}) = 1; \tag{4}$$

i.e., for any choice of the negations $\mathcal{J}$, there is a way to assign the bits x so that the formula $\phi_G(x,\mathcal{J})$ is true. On the other hand, we call an AdSAT formula UNSAT if

$$\phi_G \in L_{\mathrm{UNSAT}} \Leftrightarrow \exists \mathcal{J}, \forall x : \phi_G(x,\mathcal{J}) = 0; \tag{5}$$

i.e., there exists an assignment of the negations $\mathcal{J}$ such that no choice of the bits x can make the formula $\phi_G(x,\mathcal{J})$ true. Since the property of being either SAT or UNSAT depends on the geometric structure of the clauses (i.e., on the bipartite graph G associated to the formula), we could also say that G is either SAT or UNSAT. Once the configuration of negations in an AdSAT formula is fixed, we are left with a standard K-SAT formula that will be denoted with $\phi_G(\cdot,\mathcal{J})$. In the following we will use the expressions SAT and UNSAT with a different meaning that will depend on the context with reference both to an AdSAT formula and to a standard K-SAT formula. In the former case their meaning is defined by (4) and (5). In the latter case a formula is SAT if an assignment of bits exists such that the formula is true; vice versa, the formula is UNSAT if it is false for all possible assignments of bits.

The name "adversarial" SAT comes naturally when we view the problem as a game between two players: a positive player controls the N bits, while the adversary player controls the KM negations. First, the adversary chooses the negations. The positive player wins if he is now able to find a configuration of bits such that the formula $\phi_G(x,\mathcal{J})$ is true. Conversely, the adversarial player wins if he is able to find a configuration of negations such that no matter which configuration of bits the positive player chooses, the formula $\phi_G(x,\mathcal{J})$ is false. For a fixed $\mathcal{J}$, the positive player has 2^N possible configurations to choose from, and he is facing a standard K-SAT problem [he wins if he is able to prove that the formula $\phi_G(\cdot,\mathcal{J})$ is SAT]. On the other hand, the adversarial player has 2^{KM} configurations available in principle. Still, at least for N negations his choice is straightforward. Suppose, for example, that x_5 appears for the first time in the second clause in the third position ($x_{23} = x_5$), then without loss of generality J_{23} (corresponding to the first appearance of bit x_5) can be immediately set to zero. We can do this for every bit, fixing N of the negations, so that the number of effective configurations for the second player is 2^{KM-N}.

We study the typical behavior of the random AdSAT problem, i.e., the problem in which the graph associated with the formula is a random graph. In particular, we will focus on the ensemble in which N, M, and K are fixed and each position in a clause can be occupied with equal probability by each bit, given the aforementioned constraints. Extensive numerical [3,4] and analytical [1,2,4] evidence suggests that for the random K-SAT problem [15] in the limit $N,M \to \infty$ with finite $\alpha = M/N$, there is a critical value α_c^{SAT} such that for $\alpha < \alpha_c^{\mathrm{SAT}}$ a formula is almost surely satisfiable, while for $\alpha > \alpha_c^{\mathrm{SAT}}$ it is almost surely unsatisfiable. If a similar critical value existed also for AdSAT, it would clearly be $\alpha_c^{\mathrm{AdSAT}} < \alpha_c^{\mathrm{SAT}}$. In fact, as already noted, a K-SAT formula is simply an AdSAT formula with frozen negations, and if it is not possible to make an AdSAT formula UNSAT, it means that all the K-SAT formulas with frozen negations must be SAT. It has been shown in Ref. [9] that $\alpha_c^{K-\text{QSAT}} \leqslant \alpha_c^{\text{AdSAT}}$.

Beyond the connection with QSAT, AdSAT is also relevant on its own and should find its place in the perspective of complexity theory [16]. Complexity theory classifies decision problems according to their algorithmic difficulty using a whole hierarchy of classes. Roughly speaking, a problem is considered "easy" (and it is said to belong to the class **P**) if, in the worst case, it can be solved in a time scaling polynomially with the size of the problem. On the other hand we define the class **NP**, which contains the problems for which it is possible to verify in a polynomial time if a candidate assignment is a solution of the problem. The K-SAT problem is in **NP** (we can directly check if an assignment of bits makes a Boolean expression true), while it is not believed to be in **P** as this would imply (with $K \geqslant 3$) that $\mathbf{P} = \mathbf{NP}$.

A natural increase in the complexity of the problem gives us the class $\mathbf{\Sigma}_2^p$ (**NP** with a **coNP** oracle). A problem is in $\mathbf{\Sigma}_2^p$ if its variables can be divided into two groups u and v, so that given a candidate assignment u^*, for all possible assignments of v it is possible to verify in a polynomial time that u^* is actually a solution of the problem. Since there are 2^N possible assignments of v (hence not scaling polynomially with the size of the problem) we can only assert that $\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{\Sigma}_2^p$. Establishing if those inclusions are proper or not is probably the most important problem of contemporary computer science.

AdSAT is in $\mathbf{\Sigma}_2^p$ because given a candidate adversarial assignment of negations (this corresponds to u^* in the definition of $\mathbf{\Sigma}_2^p$), the problem reduces to certifying that the formula $\phi_G(\cdot,\mathcal{J})$ is UNSAT. We can do this with a **coNP** oracle, which implies that AdSAT belongs to $\mathbf{\Sigma}_2^p$. It would be of great importance to understand if AdSAT, despite the appearance, is in **NP** or not [17]. Note that 2-SAT is in **P**. Therefore, 2-AdSAT (AdSAT for $K = 2$) is naturally contained in **NP**, as it has a short verification procedure. The witness is an adversarial assignment $\mathcal{J}^*$ of the negations. Given these, we can efficiently assert that the 2-SAT formula $\phi_G(\cdot,\mathcal{J}^*)$ is UNSAT, proving that the original adversarial SAT formula ϕ_G is UNSAT. However, the 2-AdSAT problem is even easier. We can calculate whether a 2-AdSAT formula is SAT/UNSAT by investigating the properties of the graph G, as we show in Appendix A. For the rest of the paper we will focus on the case $K = 3$.

III. A COMPLETE ALGORITHM

A complete algorithm for AdSAT must be able to determine with certainty if a given AdSAT formula ϕ_G is SAT or UNSAT. A possible straightforward algorithm consists in solving the 3-SAT formulas $\phi_G(\cdot,\mathcal{J})$ for all the possible configurations of negations $\mathcal{J}$. As soon as one configuration of negations such that $\phi_G(\cdot,\mathcal{J})$ is UNSAT is found, then the algorithm can stop as ϕ_G is also UNSAT. On the other hand, if $\phi_G(\cdot,\mathcal{J})$ is SAT on all the possible configurations of negations, then ϕ_G is SAT. In order to solve the 3-SAT formulas one must use a complete algorithm for 3-SAT such as DPLL [18,19] (we have chosen the MINISAT implementation) [20]. The corresponding pseudocode is shown in Algorithm 1.

In order to establish if the formula is SAT, the algorithm must try all the different $2^{K\alpha N}$ configurations of negations.

Algorithm 1 A complete algorithm for AdSAT, implemented as a function returning "SAT" or "UNSAT." The loop is over the allowed configurations of negations in the sense that N negations are fixed, following the argument in Sec. II.

```
function COMPLETE-ADSAT ($\phi_G$)
  for all allowed $\mathcal{J}$ do
    run DPLL on $\phi_G(\cdot,\mathcal{J})$
    if $\phi_G(\cdot,\mathcal{J})$ is UNSAT then
      return UNSAT
    end if
  end for
  return SAT
end function
```

Even taking into account the freedom to fix N negations (see Sec. II), the effective number of configurations of negations is still $2^{(K\alpha-1)N}$. As a consequence, such an algorithm is not a viable option unless N is very small. In Fig. 2 we show the fraction of UNSAT graphs Φ_{UNSAT} found with the complete algorithm vs α, for $N = 7, 8$. Moving toward increasing values of α, a transition seems to occur between a phase in which a graph is almost surely SAT, and a phase in which it is almost surely UNSAT. In fact, both for $N = 7$ and for $N = 8$ there is only a single value of α such that Φ_{UNSAT} is not equal either to zero or to one. Even if one has to take into account that the spacing between two different viable values of α is of order 10^{-1} for such small values of N, it is also true that the transition window is expected to shrink for larger values of N. Overall, the presence of a sharp transition seems more likely than a smooth crossover between the two phases. However, given the smallness of N, nothing conclusive can be said about the value of α at which the transition occurs, as sizable corrections could be expected for larger values of N.

IV. SIMULATED ANNEALING FOR ADSAT

Given the impossibility to explore all the configuration space of negations one could think of moving in such a space

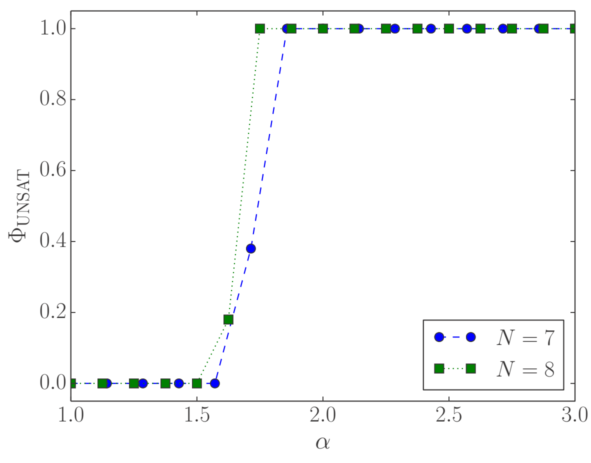

FIG. 2. (Color online) Fraction of UNSAT graphs Φ_{UNSAT} with respect to α obtained with the complete algorithm. The total number of probed graphs for each value of α is 100.

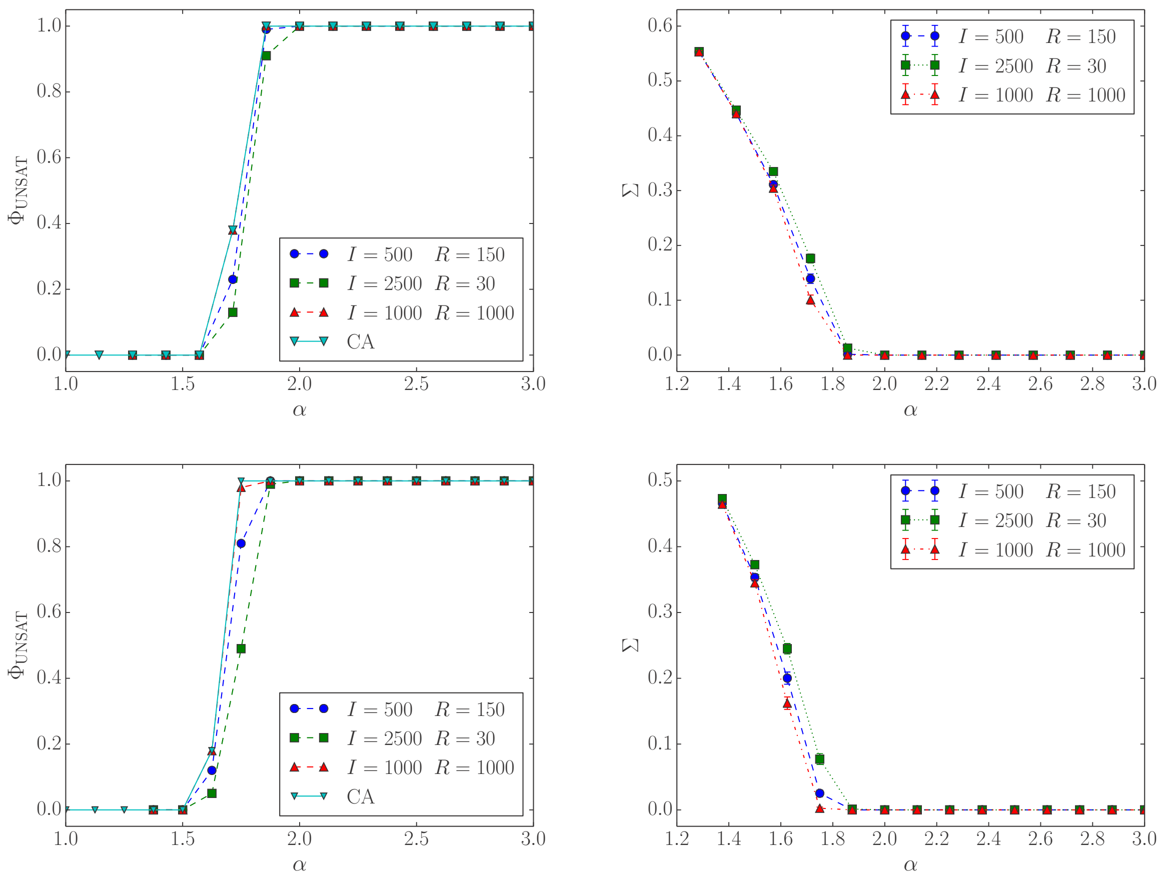

FIG. 3. (Color online) Comparison between the complete algorithm (CA) and simulated annealing for $N = 7$ (top panels) and $N = 8$ (bottom panels). Fraction of UNSAT graphs Φ_{UNSAT} (left panels) and average minimal cost function Σ (right panels) with respect to α. Error bars are one standard deviation below and above the average, and within the symbol size. The total number of probed graphs for each value of α is 100.

along some preferred directions, until an UNSAT solution is found or a minimum of the number of SAT solutions is found. In particular one can introduce a cost function measuring the number of solutions and move in a direction in which the cost function decreases.

Keeping in mind that the number of solutions is typically exponential in N (for $\alpha < \alpha_c^{\text{SAT}}$), a possible extensive cost function is the "complexity":

$$\Sigma_G(\mathcal{J}) = \frac{1}{N}\log_2[S_G(\mathcal{J}) + 1], \tag{6}$$

where $S_G(\mathcal{J})$ is the number of solutions (we used RELSAT [21,22] to count the number of solutions) of the corresponding 3-SAT problem with fixed negations $\mathcal{J}$ relative to the graph G. Note that $\Sigma_G(\mathcal{J}) = 0$ when the corresponding 3-SAT problem is UNSAT. In this context, a move is simply the flip of a randomly selected negation. As noted in Sec. II, the number of effective negations is $3M - N$, meaning that the negations involving the first occurrences of each bit in the formula are fixed.

In order to avoid getting stuck in a local minimum of the cost function one can use a Metropolis algorithm with a simulated annealing schedule. This is a stochastic process in which the move $\mathcal{J} \to \mathcal{J}'$ is always accepted if $\Sigma_G(\mathcal{J}') < \Sigma_G(\mathcal{J})$. On the other hand, if $\Sigma_G(\mathcal{J}') > \Sigma_G(\mathcal{J})$, the move can be still accepted if $e^{-\beta[\Sigma_G(\mathcal{J}') - \Sigma_G(\mathcal{J})]} > \eta$, where η is drawn uniformly between zero and one. Here β plays the role of an inverse "temperature." In simulated annealing [23], the "temperature" progressively decreases. After testing some temperature schedules, we found that $\beta = 2\sqrt{i}$ provides the best results, where $i = 1,\dots,I$ is the iteration number, and I is the total number of iterations. We opt to start from a balanced configuration of negations, i.e., a configuration such that each bit is negated the same number of times, because these are much more constrained (they have typically much fewer solutions) than the typical random SAT formula. In Algorithm 2 (see Appendix B) we show the pseudocode for simulated annealing for AdSAT.

A common extension of stochastic algorithms that is known to improve their efficiency is based on the concept of restarts. The basic idea is that the explored portion of the configuration space is not increased by increasing I, i.e., the length of a single path in such a space, but by increasing the number of paths, i.e., by restarting the algorithm from the beginning a certain number of times R. It can be naively expected that this extension is especially appropriate for simulated annealing, as a move implying $\Sigma_G(\mathcal{J}') > \Sigma_G(\mathcal{J})$ is less and less likely for lower and lower temperatures, and the chances to be stuck in a local minimum get larger and larger. In Algorithm 3 (see Appendix B) we show the corresponding pseudocode.

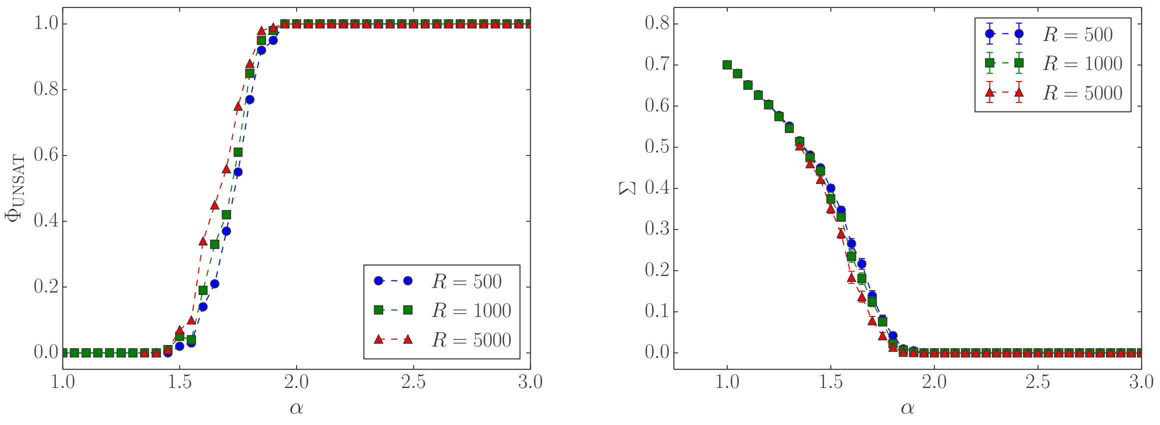

FIG. 4. (Color online) Fraction of UNSAT graphs Φ_{UNSAT} (left panel) and average minimal cost function Σ (right panel) with respect to α given by simulated annealing for $N = 20$. Error bars are one standard deviation below and above the average, and within the symbol size. The total number of probed graphs for each value of α is 100 and $I = 500$.

In order to check for the relevance of introducing restarts we tested several combinations of I and R using both the fraction of UNSAT graphs Φ_{UNSAT} and the minimal cost function averaged over the graphs:

$$\Sigma = \overline{\min_{\mathcal{J}} \Sigma_G(\mathcal{J})}, \tag{7}$$

where the minimum is obviously over the configuration of negations explored by the algorithm. From Fig. 3 it is evident that keeping the total number of steps IR constant, a better performance is achieved by increasing R rather than I. The fact that it is necessary to use very large values of both I and R to obtain results comparable to those given by the complete algorithm, even for such small values of N, is a clear sign of the difficulty of the problem. This observation is confirmed by probing larger values of N. In fact, from Fig. 4 we see that for $N = 20$ there is not a value of R reachable under our computational limits such that either Φ_{UNSAT} or Σ start to saturate.

V. AN IMPROVED STOCHASTIC ALGORITHM

We now propose a more efficient way to explore the space of configurations for values of α close to the transition. Starting from a formula ϕ_G composed of M clauses and a positive integer ΔM we build a sequence of formulas $\phi_1, \ldots, \phi_{\Delta M}$; defining for convenience $\phi_0 \equiv \phi_G$, ϕ_{s+1} is obtained simply by adding one (new) clause to ϕ_s. For ΔM sufficiently large $\phi_{\Delta M}$ will be almost surely in the UNSAT phase and will be easy to make it UNSAT using simulated annealing. Let $\mathcal{J}_{\Delta M}$ be the configuration of negations that makes $\phi_{\Delta M}$ UNSAT. Assuming that the configuration of negations that makes $\phi_{\Delta M-1}$ UNSAT (or for which its number of solutions of is minimal) is not "far" from $\mathcal{J}_{\Delta M}$, we try to make $\phi_{\Delta M-1}$ UNSAT using simulated annealing and starting from the configuration of negation $\mathcal{J}_{\Delta M}$, but with the last clause removed. Clearly the goodness of such an approach has to be evaluated *a posteriori*, and in particular by comparing it with the simulated annealing introduced in Sec. IV. This procedure is recursively iterated proceeding from $\phi_{\Delta M}$ to ϕ_0 and, while passing from ϕ_{s+1} to ϕ_s, keeping the configuration of negations that has made ϕ_{s+1} UNSAT or, if ϕ_{s+1} has not been made UNSAT, the last accepted configuration in the annealing of ϕ_{s+1} (which might not necessarily be the configuration with the minimum number of solutions for ϕ_{s+1}). Once at ϕ_0 we run the simulated annealing normally to minimize the number of solutions. In such a way the full annealing is divided in $\Delta M + 1$ chunks each of (maximum) I steps, for a total of $(\Delta M + 1)I$ steps; for each chunk, if ϕ_s is made UNSAT during the annealing, the effective number of steps will be lower. For consistency we use alternately balanced formulas, i.e., in which each bit is alternately negated and not negated, so that the balance holds even after removing the clauses added to ϕ_0. A subtle point is how to handle the temperature schedule of the annealing while passing from ϕ_{s+1} to ϕ_s. The schedule used has the same endpoints of the schedule described in Sec. IV, in the sense that the first chunk starts from $\beta = 0$ and the last chunk ends at $\beta = 2\sqrt{I}$. We explicitly note that in the final chunk (the annealing starting from ϕ_0) the temperature is lower than it would be if the final chunk were the only chunk (as in Sec. IV). The corresponding pseudocode is shown in Algorithm 4 (see Appendix B).

In order to compare the performance of the simple simulated annealing with that of its improved variant we look precisely at the value of the restart r_G after which the formula has been made UNSAT; clearly $r_G \leqslant R$. In Fig. 5 we plot $\overline{\log_2 r_G/N}$ (the over line stands for the average over the graphs that have been made UNSAT) vs α. It is clear that the improved algorithm is able to provide an equal or better performance than simulated annealing, and such a gain is more and more noticeable passing from $N = 15$ to $N = 20$, especially for $\alpha < 1.8$.

Subsequently, we analyze how the critical value of α changes with respect to the number of restarts in the improved algorithm. It is convenient to define

$$\gamma = N/\log_2 R, \tag{8}$$

which contains the information on the number of restarts in an appropriate way. In fact, one expects the complexity of the problem, indicated by the value of R necessary to obtain convergence, to scale exponentially with N, in analogy with what happens with the WALKSAT and other local algorithms for the K-SAT problem [24–26]. Therefore one should observe a critical value of γ below which the results do not depend on γ.

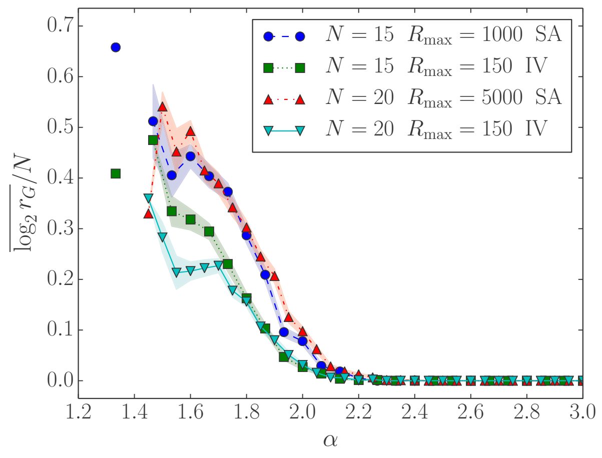

FIG. 5. (Color online) Comparison between simulated annealing (SA) and its improved variant (IV) for $N = 15$ and $N = 20$. Averaged logarithm of the number of restarts after which a graph is made UNSAT $\overline{\log_2 r_G}/N$ with respect to α. Semitransparent regions are one standard deviation below and above the average. The total number of probed graphs for each value of α is 100 and $I = 500$.

The result that we are most interested in is the critical value α_N^* for which the fraction of SAT and UNSAT formulas is the same ($\Phi_{\text{UNSAT}} = \Phi_{\text{SAT}} = 1/2$) [27]. We expect that as R is increased for a fixed N, the curve $\alpha_N^*(\gamma)$ should bend and a plateau will intercept the α_N^* axis at the predicted "critical" α for the given N, as is evident from the curves for $N = 7$, 8 and 10 in Fig. 6. The plateau therefore signals the reaching of the asymptote and the sufficiency of the number of restarts R. Actually for both $N = 7$ and 8 at the plateau one completely recovers the results given by the complete algorithm. Increasing N the plateau shifts towards smaller and smaller values of α_N^*. As a consequence, all the curves at finite N would need a negative shift in α_N^* to collapse on the theoretical limit curve for N going to infinity, confirming that the thermodynamic limit of α^* is at smaller values than

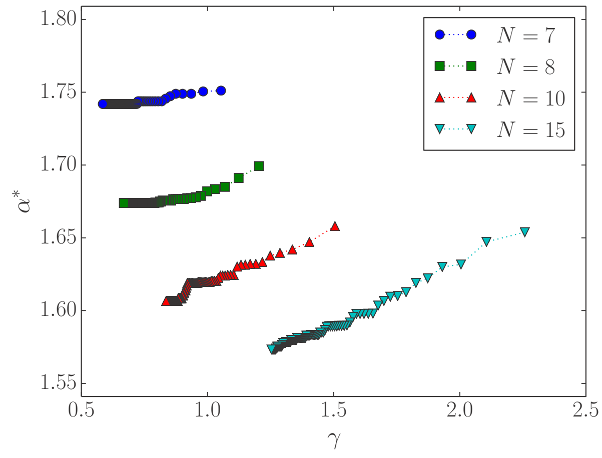

FIG. 6. (Color online) Interpolated critical value of the density of clauses α_N^* with respect to γ given by the improved algorithm for $N = 7, 8, 10, 15$. The total number of probed graphs for each value of α is 100 and $I = 2000$. $R_{\max}$ is the number of restarts corresponding to the left endpoint of the curves and is equal to 4000 for all the curves.

those observed here. Using the values of I and R practically accessible to us, the curve for $N = 15$ does not reach a plateau, but it already extends up to values of α_{15}^* well below the plateau of the curve for $N = 10$.

Actually, while the position of the plateaux depends only on N, the position of the knee depends also, although weakly, on the number of iterations I per restart. In fact for larger values of I the knee should move towards larger values of γ.

As regards $N = 20$, due to our present computational limits, we focus on the single value $\alpha = 1.6$, which is right above the transition for $N = 15$ (with 64 UNSAT graphs). With $I = 2000$ and $R = 18\,000$ we are able to make UNSAT only 62 graphs for $N = 20$. One could be tempted to consider $I = 2000$ not sufficiently large for $N = 20$, but increasing I while keeping R large enough is not a viable option due to our current computational limits. Such a result could depend either on the fact that the number of needed restart is expected to scale exponentially in the critical region or to an inversion point, in the sense that the threshold is starting to move towards larger values of α.

Since disentangling the two effects is beyond our reach at the moment, we try to gain more information by investigating graphs of larger size, but restricting to values of α close to 3. In all the investigated cases we are able to make UNSAT 100 graphs out of 100. In particular, for $\alpha = 3$ and $N = 150$ all graphs are UNSAT using $I = 2000$ after a maximum of 45 restarts, for $\alpha = 2.75$ and $N = 100$ using $I = 2000$ after a maximum of 194 restarts, and for $\alpha = 2.70$ and $N = 100$ using $I = 4000$ after a maximum of 194 restarts. Looking more closely, passing from $\alpha = 2.75$ to $\alpha = 2.70$ there is a sizable drop in the number of graphs made UNSAT for $I = 1000$ (respectively 84 and 47), which could be a hint of the beginning of the critical region. Since for larger values of I there is not much difference (for $\alpha = 2.70$ 97 graphs already are made UNSAT using $I = 2000$) it is more likely that lower values of I are not sufficiently large when we enter the region in which the graphs become "tougher." The small number of restarts is quite indicative of the fact that $\alpha = 2.70$ is still quite above the critical value (in the critical region, the number of restarts needed to decide the formula is exponential in N) [28]. As a consequence, a value of $\alpha_c > 2$ in the thermodynamic limit is conceivable only admitting the possibility of a very large $1/N$ correction (logarithmic or a small power). If the leading corrections are $1/N$ as usual in mean-field glasses, to reconcile with the theoretical expectations of $\alpha_c = 3.39$ [13], the coefficient should be $\simeq 200$, which has to be compared with one as α is dimensionless. Moreover, an even larger subleading ($1/N^2$) coefficient with a negative sign should be present, since we observe a decreasing of α_N^* with N for all the accessible values of N. At the moment, we do not know the possible reasons for the discrepancy with the result $\alpha^* = 3.39$ in Ref. [13].

The numerics performed in Ref. [13] (simulated annealing without restarts) for the ensemble of random regular graphs (an L-regular graph is a graph in which every variable belongs exactly to L clauses, so that $KM = LN$) suggests a smaller value of the transition threshold $\alpha_c \simeq 2.33$ ($L = 7$), but with finite-size corrections pointing towards larger values. In particular, $\Phi_{\text{UNSAT}} = 1$ for $L > 7$, while $\Phi_{\text{UNSAT}} \simeq 0.1$ for $L = 7$ and $N = 54$. For $L = 6$ ($\alpha = 2$) $\Phi_{\text{UNSAT}} \simeq 0.5$ for $N = 18$, $\Phi_{\text{UNSAT}} \simeq 0.15$ for

$N = 27$, and $\Phi_{\text{UNSAT}} = 0$ for larger values of N. $\Phi_{\text{UNSAT}} = 0$ for smaller values of L. Our algorithm is able to outperform the numerics in Ref. [13] finding $\Phi_{\text{UNSAT}} = 0.58$ for $L = 7$ and $N = 54$. For $L = 6$ we find $\Phi_{\text{UNSAT}} = 0.2$ for $N = 36$, $\Phi_{\text{UNSAT}} = 0.94$ for $N = 27$, and $\Phi_{\text{UNSAT}} = 1$ for $N = 18$. Consequently, also in this case it is impossible to distinguish if this effect is due to a genuine shift of α_c, or to the increasing difficulty to make larger and larger graphs UNSAT. In Ref. [13] the authors explain this discrepancy (the analytical value α_c is around 3.66 for regular graphs) with the presence of strong preasymptotic effects.

Using our improved algorithm to probe regular graphs we see a transition threshold which seems to shift towards smaller values of α for increasing N, again in contrast with Ref. [13]. In particular, using $I = 2000$ and $R = 4000$, for $N = 9$ the curve $\alpha_N^*(\gamma)$ reaches the plateau around 1.81, while for $N = 12$ it arrives to 1.79 without reaching the plateau. Let us point out that determining the value of α_N^* for regular graphs is certainly more problematic, due to the large granularity in the values of α. In fact the allowed values of α are $1,4/3,5/3,2,7/3,\ldots$ if N is a multiple of 3, and only the integers otherwise.

Finally, we would like to comment on the configuration of negations that minimize Σ. As was also pointed out in Ref. [13], it is reasonable to expect that such a configuration is balanced in the thermodynamic limit. However, even after requesting that a configuration is balanced, there is still a lot of freedom to set the negations and plenty of room for more complex, long-range correlations between negations to play a role. In other words, one expects the balancing to be a necessary condition for a configuration minimizing Σ, but not a sufficient one, giving then only an *upper bound* to the AdSAT threshold, as our numerics suggest. Looking at the configurations obtained at the end of our improved algorithm for small values of N (i.e., far from the thermodynamic limit) we actually find that, when the graph becomes UNSAT only after a nontrivial number of iterations, the fraction of balanced configurations could even be very small. For $N = 100,125,150$ and $\alpha = 3$ we find a larger fraction of balanced configurations (around 75%) and slightly increasing with N. Therefore, at small values of N (meaning $N \lesssim 200$) these other correlations play even a larger role than balancing, leading to best solutions that are *unbalanced*. We think that this effect will go away in the thermodynamic limit, but this is a clear indication that other conditions are playing a prominent role making the restriction to balanced instances necessary but not sufficient. Hence, it might be that the value of α_c found in Ref. [13] is an upper bound rather than the exact value of the AdSAT transition threshold.

VI. CONCLUSIONS

We have numerically investigated AdSAT, a quantified Boolean formula problem which originated from the study of the quantum analog of SAT. Previous investigations in Ref. [13] have proposed a threshold at $\alpha_c = 3.39$. On the one hand numerics on small system sizes (up to $N = 15$) suggest $\alpha_c < 1.6$. On the other hand, we are able to exclude the region $\alpha_c > 3$ ($\alpha_c > 2.70$) for systems of size up to $N = 150$ ($N = 100$) upon the assumption of regular $1/N$ corrections to the critical values of α_N^*. As already pointed out, even considering the present limits on the numerically accessible values of N, these results could be compatible with the analytical results in Ref. [13] only if unusually large finite-size corrections were present.

Moreover, our upper bound on α_c for AdSAT is also much closer to the numerical upper bound found in Ref. [9] for the threshold of 3-QSAT, suggesting that the difference between the quantum problem and the classical one might not be so striking.

Improving on the algorithms would be much desirable as AdSAT is a member of the quantified Boolean problems family, which is only recently starting to get the attention it deserves. One possibility would be to use a belief propagation supported heuristic algorithm as devised in Ref. [29] to solve AdSAT. We hope to do this in the future, together with a deep analysis of the nature of the configuration of negations minimizing Σ, which, as highlighted at the end of Sec. V, is certainly a direction worth pursuing.

ACKNOWLEDGMENTS

The authors would like to thank Lenka Zdeborová for many discussions in the early stages of this work. A.S. also thanks Roderich Moessner for many useful discussions. D.N. was supported by Frank Verstraete's ERC grant QUERG and thanks the Slovak Research and Development Agency grant APVV-0646-10 COQI. M.B. would like to thank Maria Valentina Carlucci for the helpful discussions.

APPENDIX A: 2-ADSAT IS IN P

We now show that adversarial 2-SAT (2-AdSAT) is in **P**; i.e., we prove that we can efficiently find whether for a given graph G of clauses (each involving two bits), an adversary can choose the negations so that the formula $\phi_G(x,\mathcal{J})$ (3) is UNSAT for any choice of x.

Note that any disconnected subgraphs can be analyzed by themselves and that any treelike (no loops) part of the graph can be cut off, as any 2-SAT instance on a tree is satisfiable for any value of the bit at the root of the tree.

The adversary now has two simple ways to set up the negations. First, he can use an "implication line" [see Fig. 7(a)], which works as $(x = 1) \Rightarrow (y = 1)$. It is a line of clauses with the negations set so that a clause involving two successive bits b_i and b_{i+1} on the line is $(b_i \oplus 1) \vee b_{i+1}$. When all of these clauses are true, $x = 1$ implies $y = 1$. Note that if we choose $(b_i \oplus 1) \vee (y \oplus 1)$ in the last clause, we create the implication line $(x = 1) \Rightarrow (y = 0)$. The adversary can similarly set the implication lines $(x = 0) \Rightarrow (y = 0)$ and $(x = 0) \Rightarrow (y = 1)$.

Second, the adversary can use a "bit-fixing cycle" [see Fig. 7(b)], which sets $x = 0$. It is made from an implication line $(x = 1) \Rightarrow (y = 1)$ closed by the final clause $\overline{x} \vee \overline{y}$. These

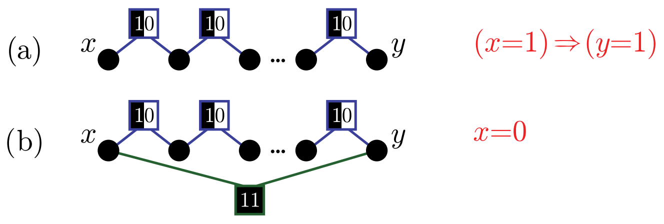

FIG. 7. (Color online) For 2-AdSAT, the adversary can choose the negations so that he (a) creates an implication and (b) fixes a bit.

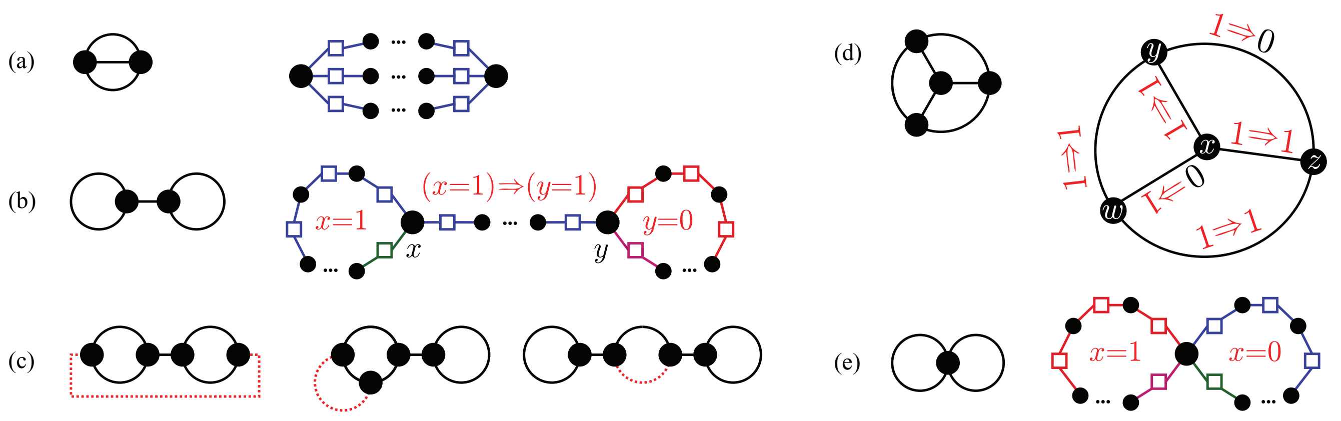

FIG. 8. (Color online) Solving 2-AdSAT divides into a few easily checkable cases (assuming a connected component with pruned treelike parts). Only the graph of the type in (a) is adversarially satisfiable (SAT for any choice of negations); the others are UNSAT.

can be simultaneously true only for $x = 0$. Analogously, the adversary can choose to fix $x = 1$.

We can now apply these tools to the remainder of the graph after dividing it into disconnected components and pruning the treelike parts. If there is only a single loop (no node has degree more than 2), the instance is trivially SAT, no matter how hard the adversary tries. Let us then look at the five remaining interesting cases that can happen within a connected component.

(1) The largest node degree is 3. If there are exactly two nodes with degree 3, connected via three paths as in Fig. 8(a), the adversarial 2-SAT instance is SAT. One can rule out most of the possible solutions by setting up implication lines on the paths. However, this allows us to set only three implication lines for the two bits (nodes with degree 3), which is not enough to rule all four possible bit assignments of the two degree-3 nodes. Such an adversarial 2-SAT instance is thus SAT.

(2) The largest node degree is 3. There are exactly two nodes with degree 3, connected via a single path as in Fig. 8(b). The adversarial 2-SAT instance is now UNSAT, as we can fix the bits to $x = 1$, $y = 0$, and connect them via an implication line $(x = 1) \Rightarrow (y = 1)$ that rules out this possibility.

(3) The largest node degree is 3. There are exactly four nodes with degree 3, and the graph contains the case as in (b) as a subgraph [see Fig. 8(c)]. This can be tested by removing lines [the red dotted lines in Fig. 8(c)] and seeing what we get. The adversary then deals with the subgraph described in (b), making it UNSAT.

(4) The largest node degree is 3. There are exactly four nodes of degree 3, and when we remove any of the lines, we recover case (a). This means our graph is depicted in Fig. 8(b). We can now set up three implication lines

$$(x = 1) \Rightarrow (y = 1),$$
$$(x = 1) \Rightarrow (z = 1),$$
$$(y = 1) \Rightarrow (z = 0).$$

This fixes $x = 0$, because $x = 1$ implies $y = z = 1$, which is inconsistent with the last implication. We now set up three more implication lines

$$(x = 0) \Rightarrow (w = 1),$$
$$(w = 1) \Rightarrow (y = 1),$$
$$(w = 1) \Rightarrow (z = 1).$$

Assuming now that $x = 0$, we have $w = y = z = 1$. However, because we already know that $y = z = 1$ is inconsistent with one of the implications above, such an adversarial 2-SAT instance is UNSAT. Note that starting with more than four nodes with degree-3 in a connected component, the 2-AdSAT instance is UNSAT, as it maps either to (3) or (4) by removing lines until we end up with a connected component with exactly four degree-3 nodes.

Algorithm 2 Simulated annealing for AdSAT, implemented as a function returning both the minimum value of the complexity Σ_G^* found and the last accepted configuration of negations $\mathcal{J}$.

```
function SA-ADSAT($\phi_G$,$\mathcal{J}$,$I$)
  $\Sigma_G(\mathcal{J}) \leftarrow$ complexity of $\phi_G(\cdot,\mathcal{J})$
  $\Sigma_G^* \leftarrow \Sigma_G(\mathcal{J})$
  if $\Sigma_G(\mathcal{J}) = 0$ then
    return 0, $\mathcal{J}$
  end if
  for $t = 1,\ldots,I$ do
    $\beta \leftarrow 2\sqrt{t}$
    obtain $\mathcal{J}'$ flipping a random (allowed) negation in $\mathcal{J}$
    $\Sigma_G(\mathcal{J}') \leftarrow$ complexity of $\phi_G(\cdot,\mathcal{J}')$
    if $\Sigma_G(\mathcal{J}') = 0$ then
      return 0, $\mathcal{J}'$
    else if $0 < \Sigma_G(\mathcal{J}') < \Sigma_G(\mathcal{J})$ then
      $\mathcal{J} \leftarrow \mathcal{J}'$
      $\Sigma_G(\mathcal{J}) \leftarrow \Sigma_G(\mathcal{J}')$
      if $\Sigma_G(\mathcal{J}') < \Sigma_G^*$ then
        $\Sigma_G^* \leftarrow \Sigma_G(\mathcal{J}')$
      end if
    else
      $\eta \leftarrow$ random number in [0,1]
      if $e^{-\beta[\Sigma_G(\mathcal{J}')-\Sigma_G(\mathcal{J})]} > \eta$ then
        $\mathcal{J} \leftarrow \mathcal{J}'$
        $\Sigma_G(\mathcal{J}) \leftarrow \Sigma_G(\mathcal{J}')$
      end if
    end if
  end for
  return $\Sigma_G^*$, $\mathcal{J}$
end function
```

Algorithm 3 Simulated annealing for AdSAT with restarts, implemented as a function returning the minimum value of the complexity Σ_G^* found.

```
function RESTART-SA-ADSAT($\phi_G$, $R$, $I$)
  for $r = 1, \ldots, R$ do
    pick a random balanced config. of negations $\mathcal{J}$
    $\Sigma_G, \tilde{\mathcal{J}} \leftarrow$ SA-ADSAT($\phi_G$, $\mathcal{J}$, $I$)
    if $\Sigma_G = 0$ then
      return 0
    end if
    if $r = 1$ then
      $\Sigma_G^* \leftarrow \Sigma_G$
    else if $r > 1$ and $\Sigma_G < \Sigma_G^*$ then
      $\Sigma_G^* \leftarrow \Sigma_G$
    end if
  end for
  return $\Sigma_G^*$
end function
```

(5) Finally, we are left with the cases which contain a node with degree at least 4. The graph then contains a subgraph depicted in Fig. 8(e). The adversary uses the two loops to fix the bit x to be 0 and 1, making the instance UNSAT.

Algorithm 4 Improved variant of the simulated annealing for AdSAT with restarts, implemented as a function returning the minimum value of the complexity $\Sigma_G{}^*$ found.

```
function IV-ADSAT($\phi_G$, $\Delta M$, $R$)
  $M \leftarrow$ number of clauses in $\phi_G$
  for $r = 1, \ldots, R$ do
    $s \leftarrow M + \Delta M$
    $\phi_s \leftarrow \phi_G$ expanded with $\Delta M$ additional clauses
    pick a random alternately balanced configuration of negations $\mathcal{J}_s$
    while $s > 0$ do
      $\tilde{\Sigma}, \mathcal{J}_s \leftarrow$ SA-ADSAT($\phi_s$, $\mathcal{J}_s$, $I$)
      $\phi_{s-1} \leftarrow \phi_s$ with the last clause removed
      $\mathcal{J}_{s-1} \leftarrow \mathcal{J}_s$ with the last clause removed
      $s \leftarrow s - 1$
    end while
    $\Sigma_G, \tilde{\mathcal{J}} \leftarrow$ SA-ADSAT($\phi_0$, $\mathcal{J}_0$, $I$)
    if $\Sigma_G = 0$ then
      return 0
    end if
    if $r = 1$ then
      $\Sigma_G^* \leftarrow \Sigma_G$
    else if $r > 1$ and $\Sigma_G < \Sigma_G^*$ then
      $\Sigma_G^* \leftarrow \Sigma_G$
    end if
  end for
  return $\Sigma_G^*$
end function
```

APPENDIX B: PSEUDOCODES

Here we list the pseudocodes for simulated annealing for AdSAT (Algorithm 2), its extension with restarts (Algorithm 3), both discussed in Sec. IV, and the improved variant (Algorithm 4), discussed in Sec. V.

[1] M. Mézard, G. Parisi, and R. Zecchina, Science **297**, 812 (2002).
[2] E. Friedgut and J. Bourgain, J. Am. Math. Soc. **12**, 1017 (1999).
[3] S. Kirkpatrick and B. Selman, Science **264**, 1297 (1994).
[4] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky, Nature (London) **400**, 133 (1999).
[5] J. Kempe, A. Kitaev, and O. Regev, SIAM J. Comput. **35**, 1070 (2006).
[6] S. Bravyi, arXiv:quant-ph/0602108 (2006).
[7] D. Gosset and D. Nagaj, in *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science* (IEEE, Piscataway, NJ, 2013), pp. 756–765.
[8] To be more precise, $\mathbf{QMA}_1$ is the quantum analog of $\mathbf{MA}_1$, where the verification procedure allows false instances to be accepted with a small probability.
[9] C. R. Laumann, R. Moessner, A. Scardicchio, and S. L. Sondhi, Quantum Inf. Comput. **10**, 1 (2010).
[10] C. R. Laumann, A. M. Läuchli, R. Moessner, A. Scardicchio, and S. L. Sondhi, Phys. Rev. A **81**, 062345 (2010).
[11] C. R. Laumann, R. Moessner, A. Scardicchio, and S. L. Sondhi, in *Modern Theories of Many-Particle Systems in Condensed Matter Physics*, edited by D. C. Cabra, A. Honecker, and P. Pujol (Springer, Berlin, Heidelberg, 2012), pp. 295–332.
[12] A. Ambainis, J. Kempe, and O. Sattath, J. ACM **59**, 24 (2012).
[13] M. Castellana and L. Zdeborová, J. Stat. Mech.: Theory Exp. (2011) P03023.
[14] S. Bravyi, C. Moore, and A. Russell, in *Proceedings of Innovations in Computer Science-ICS 2010*, edited by A. Chi-Chih Yao (Tsinghua University Press, Beijing, 2010), pp. 482–489.
[15] A. K. Hartmann and M. Weigt, *Phase Transitions in Combinatorial Optimization Problems* (Wiley-VCH, Weinheim, 2005).
[16] S. Arora and B. Barak, *Computational Complexity: A Modern Approach* (Cambridge University Press, Cambridge, 2009.
[17] This might appear hopeless. However, AdSAT is a graph property, so it is imaginable that it has a compressed testing procedure. Moreover, our attempts to show that AdSAT is $\mathbf{\Sigma}_2^p$-complete were not successful.
[18] M. Davis and H. Putnam, J. ACM **7**, 201 (1960).
[19] M. Davis, G. Logemann, and D. Loveland, Comm. ACM **5**, 394 (1962).
[20] N. Sörensson and N. Eén, http://www.minisat.se.
[21] R. J. J. Bayardo and J. D. Pehousek, in *Proceedings of the 17th National Conference on Artificial Intelligence*, edited by H. Kautz and B. Porter (AAAI Press, Menlo Park, 2000), pp. 157–162.
[22] R. J. J. Bayardo, https://code.google.com/p/relsat/.

[23] S. Kirkpatrick, D. C. Gelatt, and M. P. Vecchi, Science **220**, 671 (1983).

[24] C. H. Papadimitriou, in *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science* (IEEE, Piscataway, NJ, 1991), pp. 163–169.

[25] U. Schöning, in *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science* (IEEE, Piscataway, NJ, 1999), p. 410.

[26] U. Schöning, Algorithmica **32**, 615 (2002).

[27] In practice it is determined in the following way: we compute $|\Phi_{\text{UNSAT}} - 1/2|$ for all the values of α, rank them in ascending order, and select the first five values; then we make a linear interpolation between these points and use the interpolating function to compute α^*, the value of α corresponding to $\Phi_{\text{UNSAT}} = 1/2$.

[28] Being optimistic, one could conjecture that the complexity of the problem is $\gamma \sim 1$ as seen for the low N numerics. This means that, assuming a sufficient number of iterations (probably in the millions), the number of restarts necessary to get a converged α_N^* for $N = 150$ is about 2^{150}–10^{45}.

[29] P. Zhang, A. Ramezanpour, L. Zdeborová, and R. Zecchina, J. Stat. Mech.: Theory Exp. (2012) P05025.