Complexity of the predecessor problem in Kauffman networks

S. N. Coppersmith
Department of Physics, University of Wisconsin-Madison, 1150 University Avenue, Madison, Wisconsin 53706, USA
(Received 31 October 2005; revised manuscript received 5 December 2006; published 8 May 2007)

Kauffman nets, also known as *N-K* models, have been studied extensively because their dynamics can be used to model a variety of interesting dynamical processes. This paper investigates the properties of the problem of determining whether or not a given configuration of a Kauffman net has a predecessor. Here it is shown that when the parameter K that governs the number of connections grows as $\ln(N)$, where N is the number of elements, the problem of finding a solution is extremely sensitive to small changes in the problem statement. This result has implications for studies of the physics of random systems and also may have applications for questions in computational complexity theory.

DOI: 10.1103/PhysRevE.75.051108 PACS number(s): 02.50.−r, 75.10.Nr, 02.60.Pn, 89.20.Ff

I. INTRODUCTION

A Kauffman net (also called a Boolean network or an *N-K* model) [1–3] has N elements $\{\sigma_1,\sigma_2,\ldots,\sigma_N\}$, each of which is a Boolean variable $\sigma_i \in \{0,1\}$, $i=1,2,\ldots,N$. The value of the ith element σ_i at time $t+1$ is determined by the value of its K inputs $j_1(i),j_2(i),\ldots,j_K(i)$ at time t, $\sigma_{j_1(i)}(t)$, $\sigma_{j_2(i)}(t),\ldots,\sigma_{j_K(i)}(t)$, via

$$\sigma_i(t+1)=f_i(\sigma_{j_1(i)}(t),\sigma_{j_2(i)}(t),\ldots,\sigma_{j_K(i)}(t)), \tag{1}$$

where each f_i is a randomly chosen Boolean function with K arguments. The K inputs for each element and the Boolean functions f_i are all chosen randomly before beginning and then fixed throughout the computation. We will denote the N Eqs. (1) for all the elements as $\{\sigma(t+1)\}=f(\{\sigma(t)\})$.

Kauffman nets have been studied because of their relevance to physics [4–8], social sciences [9,10], and biology (Kauffman's original motivation was to study gene regulation and control [1,2,11–15]). The model exhibits a phase transition as K is varied; $K<2$ is a "frozen" phase, while $K>2$ exhibits chaotic dynamics.

This paper focuses on the question of determining whether a given configuration of a Kauffman net, $\{\sigma\}$, has a predecessor configuration $\{\tau\}$ such that $\{\sigma\}=f(\{\tau\})$ [16,17]. Specifically, it is demonstrated that when K is proportional to $\ln N$, the solution is extremely sensitive to the change of a single variable in the configuration whose predecessor is to be determined. It is argued that this sensitivity may yield new insight into some problems in computational complexity, which is the study of how the resources needed to solve different computational problems depend on the size of the problem specification.

For the problem of determining whether a given configuration $\{\sigma\}$ of a Kauffman net has a predecessor, a natural choice for the energy of a given configuration $\{\tau\}$ is the number of bits in the successor configuration $\{\sigma'\}=f(\{\tau\})$ that differ from the corresponding bit in $\{\sigma\}$ [18]. The configuration $\{\sigma\}$ has a predecessor if there is a configuration $\{\tau\}$ for which this energy is zero. The results here demonstrate that the local properties of this energy landscape for the predecessor problem evolve systematically as K is increased.

For any $K>2$, local search algorithms for solving the predecessor problem that work by decreasing the number of wrong bits by changing a small number of bits in the current "guess" of the predecessor typically find only a local minimum and not the global one. The new ingredient discussed here is that for $K=A\log_2 N$, finding a solution is hard even if one starts off with a configuration whose successor has only one bit in error. In contrast, if one has found a configuration whose successor differs from the target by one bit for a random problem instance with $K=3$, there is a substantial probability that the error can be corrected via a small number of single bit flips. In addition, when $K=A\log_2 N$ with A large enough, perturbing a target configuration that has a predecessor with a single bit flip yields a new target, and, with very high probability, no configuration that differs from the original predecessor by fewer than a number of bits that grows as a power of N is the predecessor of the perturbed target. In contrast, when $K=3$, the perturbed target is reasonably likely to have a predecessor that differs from the original one by $O(1)$ bits. The systematic evolution of the error energy landscape of the predecessor problem as K is varied yields insight into the behavior of a particular random system and may also yield insight into the question of whether or not computational problems whose solutions can be verified efficiently can also be solved efficiently.

The paper is organized as follows. Section II demonstrates that if one starts with a configuration whose successor differs by the target by just one element, then an algorithm that attempts to go "downhill" on the energy landscape will succeed with a probability that is substantial when K is finite and is very small when $K\propto\ln N$. Section III shows that when $K\propto\ln N$, if one is given a predecessor-target pair, then the probability is extremely small that a configuration that differs from the original predecessor by a small number of bits is the predecessor of a perturbed target that differs from the original target by one bit. Section IV argues that these results reflect the increasing roughness of the error energy landscape as K increases. Section V discusses why the predecessor problem may be viewed as exhibiting "self-organized criticality" [19] and discusses the possible relevance of the results to problems in computational complexity. Section VI is a summary.

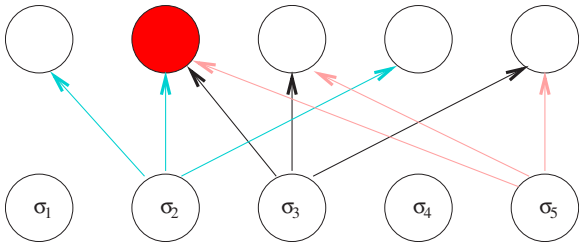

FIG. 1. (Color online) Schematic diagram illustrating why increasing K decreases the probability that a local algorithm finds a solution to the predecessor problem, starting from a configuration whose output has one element in error. In this sketch, $K=3$, there are five elements $\sigma_1,\ldots,\sigma_5$, and one imagines changing the target output of σ_2, which is shaded (red online). All the elements are shown twice, so that it is easier to see the inputs and outputs. Only the connections of σ_2, σ_2, and σ_5, the elements that have inputs to σ_2, are shown. Changing an input of σ_2 could also affect $K-1$ other outputs, and the target is reached only if all the affected outputs have the desired value.

II. DEMONSTRATION THAT A DOWNHILL ALGORITHM ALMOST ALWAYS FAILS TO FIND A SOLUTION AFTER ONE BIT FLIP WHEN $K \propto \ln N$

To demonstrate the increasing sensitivity of the solution of the predecessor problem to single-bit changes in the target as K is increased, one assumes that one is given a configuration $\{\tau\}$ such that $f(\{\tau\})\equiv\{\eta\}$ differs from the target configuration $\{\sigma\}$ by exactly one bit. One then attempts to find the state $\{\nu\}$ such that $f(\{\nu\})=\{\sigma\}$ as follows: (1) For each of the K inputs of the wrong element i, find the configuration that results when a given input is flipped, and (2) flip the input of i that minimizes the number of wrong elements in the output. This "downhill" algorithm succeeds if single-element changes of $\{\tau\}$ yield no errors in the output instead of one error.

This algorithm is characterized here using methods similar to those in Refs. [4,20]. The key point can be illustrated by considering a model in which each element is constrained to have exactly K outputs as well as K inputs. Given a configuration $\{\tau\}$ whose successor has only one element σ_{j^*} that differs from the target, one knows that the true predecessor, if it does exist, must have the property that at least one of the inputs to σ_{j^*} must be different than it is in $\{\tau\}$, since the only way to change the output σ_{j^*} is to change at least one of its inputs. Flipping a given input to σ_{j^*} affects not only σ_{j^*} itself but also $K-1$ other elements that depend on that input (see Fig. 1). The likelihood of success of the single-flip algorithm decreases strongly with K because when one input is changed, the probability that all K of its outputs are correct is $(1/2)^K$, so the probability that one of the K choices of the perturbed inputs yields the target is $\sim K(1/2)^K$, which is nonzero for finite K but vanishes algebraically with N when $K=A\log_2 N$.

The probability that the "downhill" algorithm succeeds for the Kauffman model in which only the number of inputs per element is fixed is obtained by noting that if each element has K different randomly chosen inputs, then the probability that a given element has $\mathcal{L}$ outputs is $K^{\mathcal{L}}\exp(-K)/\mathcal{L}!$. Since the total number of connections is KN and the number of connections originating from elements with $\mathcal{L}$ outputs is $\mathcal{L}NK^{\mathcal{L}}\exp(-K)/\mathcal{L}!$, the probability that a given input to the perturbed element has $\mathcal{L}$ outputs is $K^{(\mathcal{L}-1)}\exp(-K)/(\mathcal{L}-1)!$. Each of the $\mathcal{L}$ elements affected by a given element is correct with probability $1/2$, so starting with a configuration that yields a successor that differs from the target by one bit, the probability that flipping one given input of the wrong element causes the output result to have no errors is

$$P_K(0)=\sum_{\mathcal{L}=1}^{\infty}\left(\frac{1}{2}\right)^{\mathcal{L}}\frac{K^{\mathcal{L}-1}e^{-K}}{(\mathcal{L}-1)!}=\frac{1}{2}e^{-K/2}. \tag{2}$$

Similarly, the probability that flipping one input of the perturbed element yields an output configuration that differs from the target by a single element is

$$P_K(1)=\sum_{\mathcal{L}=1}^{\infty}\left(\frac{1}{2}\right)^{\mathcal{L}}\mathcal{L}\frac{K^{\mathcal{L}-1}e^{-K}}{(\mathcal{L}-1)!}=\frac{1}{4}(K+2)e^{-K/2}. \tag{3}$$

More generally, the probability that flipping one input of the perturbed element yields $\mathfrak{K}$ bits in error is

$$P_K(\mathfrak{K})=\frac{e^{-K}}{K}\sum_{\mathcal{L}=\mathfrak{K}}^{\infty}\left(\frac{K}{2}\right)^{\mathcal{L}}\frac{\mathcal{L}}{\mathfrak{K}!(\mathcal{L}-\mathfrak{K})!}. \tag{4}$$

Now one gets to pick the input that yields the fewest incorrect outputs. Defining $Q_K(j)$ as the probability that the best output configuration differs from the target in j bits, one finds

$$Q_K(0)=1-[1-P_K(0)]^K,$$

$$Q_K(j)=\left(1-\sum_{i=0}^{j-1}P_K(i)\right)^K-\left(1-\sum_{i=0}^{j}P_K(i)\right)^K,$$

$$1\leqslant j\leqslant K-1,$$

$$Q_K(K)=\left(1-\sum_{j=0}^{K-1}P_K(j)\right)^K. \tag{5}$$

Equations (5) follow because if the smallest number of errors yielded by this process is i, no trials can yield a number of errors less than i, at least one trial must yield i errors, and $[1-\int_{j=0}^{i}P_K(j)]^K$ is the probability that only more than i errors are obtained.

This procedure fixes the error if the resulting configuration has no wrong bits. The quantity $Q_K(0)$, which bounds from below the probability that the error has been fixed, is nonzero for finite K but vanishes as $A\log_2 NN^{-A/2}$ when $K=A\log_2 N$. More sophisticated estimates accounting for the possibility that changing an input can yield a configuration with a different error which in turn can be corrected by flipping one input indicate that the probability that the algorithm corrects the error is close to 50% when $K=3$ but vanishes as $N^{-A/2}\log N$ when $K=A\log_2 N$.

III. DEMONSTRATION THAT NO CONFIGURATION THAT IS NEAR THE PREDECESSOR OF A GIVEN TARGET IS THE PREDECESSOR OF A NEW TARGET THAT DIFFERS BY ONE BIT WHEN $K \propto \ln N$

In this section we again start with a target configuration $\{\sigma\}$ that has a predecessor $\{\tau\}$, and then consider a perturbed target $\{\sigma'\}$ that is the same as $\{\sigma\}$ except for the value of one element σ_{j^*}. It is shown that when $K=A \log_2 N$ with A large enough, then the probability that a configuration $\{\tau'\}$ exists that is a predecessor to $\{\sigma'\}$ and that differs from $\{\tau\}$ by a number of bits that is less than N^x, with x strictly greater than zero, is bounded above by $e^2 N^{A/2-2}$ as $N \to \infty$. This result is plausible because if one changes a single bit of $\{\tau\}$ with the hope of changing only σ_{j^*}, then it is extremely likely that the number of bits in the successor configuration that changes is of order $\ln N$. To fix these newly erroneous output elements, one must flip at least one input of each of the output elements that were flipped in error in the first step, which is extremely likely to create still more wrong output elements, and so on. The "damage" of wrong outputs must spread until flipping an element corrects more errors than it creates, which requires that enough outputs are wrong that there is an element that is an input to many wrong outputs. If $\mathcal{M}_w$, the number of wrong outputs, is smaller than N^y with $y<1/2$, then the probability that the erroneous outputs share more than one input must be very low; this can be seen by noting that the sum of the number of inputs of all the erroneous outputs is $\mathcal{M}_i=\mathcal{M}_w K=AN^y \log_2 N$. So long as $\mathcal{M}_i \ll \sqrt{N}$, then the probability of duplication exceeding that obtained trivially by choosing individual inputs so that they are both inputs to the same element is extremely small [21].

We present a more complete analysis here for systems in which each element has exactly K inputs and K outputs, a restriction that does not affect the results but simplifies the analysis. One begins with a configuration $\{\tau\}$ such that $f(\{\tau\})$ is the target configuration $\{\sigma\}$, and now considers a new target configuration $\{\sigma'\}$ that is identical to $\{\sigma\}$ except for a single bit flip. One then asks how many bits one must flip in the configuration $\{\tau\}$ to obtain a configuration $\{\tau'\}$ such that $f(\{\tau'\})=\sigma'$. To see that $\{\tau'\}$ differs from $\{\tau\}$ by many bits, consider all configurations that differ from the original predecessor $\{\tau\}$ by up to S bits, where $S<N^x$, with $x<1/2$. Given a specific choice of $M<S$ bits of the input configuration that are flipped, let Q to be the number of elements with at least one input that has been changed. When $M \ll \sqrt{N}/K$ and K is large, Q satisfies the bound $Q>MK/2$. This is because the number of affected outputs would be MK if no output elements shared inputs, and while one can choose input elements that have a common output, the probability that two elements share more than one output is negligible when the connections are chosen randomly and $MK \ll \sqrt{N}$ [21]. Therefore, Q is typically at least $M(K-1)$, and is certainly larger than $MK/2$.

The probability that changing one or more inputs of Q elements changes the outputs of R elements is

$$\left(\frac{1}{2}\right)^Q \frac{Q!}{R!(Q-R)!} \approx \left(\frac{1}{2}\right)^Q \frac{Q^Q}{R!(Q-R)^{(Q-R)}} \leqslant \left(\frac{1}{2}\right)^Q \frac{Q^Q}{(Q-R)^{(Q-R)}} \approx \left(\frac{1}{2}\right)^Q (Qe)^R. \tag{6}$$

In Eq. (6) the second line follows because $K \propto \log N$ and so $Q \gg 1$, and since the case of interest is $R=1$, one knows that $Q-R \gg 1$ as well. The upper bound in the third line follows because $1 \geqslant 1/R!$ for any $R \geqslant 0$, and the last line follows because $\lim_{Q\to\infty}(1-R/Q)^Q=e^{-R}$.

Now we obtain an upper bound for $\mathcal{P}_M$, the probability that a choice of the M inputs exists for which $R=1$. Because the probability that $Q>MK/2$ is essentially unity, when $K \propto \ln N$, even when M is small many elements have their inputs flipped, so that the probability that the output has only one element flipped is very small. Specifically, given a choice of the M elements in the predecessor that are flipped, when MK is large, Eq. (6) shows that the probability that the perturbed successor differs from the original one by only one bit is bounded above by $(eMK/2)2^{-MK/2}$. There are $N!/(M!(N-M)!) \leqslant (Ne)^M$ ways of choosing the M elements in the predecessor to flip (the bound being valid when N is large and $M \ll \sqrt{N}$), so when $K=A \log_2 N$, one has

$$\mathcal{P}_M \leqslant (Ne)^M \left(\frac{MKe}{2}\right) 2^{-MK/2} = (N^{-(A/2-1)}e)^M \left(\frac{MeA}{2} \log_2 N\right). \tag{7}$$

Therefore, when $MeA \log_2 N<N$, $\mathcal{P}_M$ is bounded above by

$$\mathcal{P}_M < \frac{e}{2}(N^{-(A/2-2)}e)^M. \tag{8}$$

When $A>4$, as $N \to \infty$ this bound is small for any $M>0$, and moreover, so long as $S \ll N^{1/2}$ so that all the approximations are valid, the sum $\Sigma_{M=1}^{S} \mathcal{P}_M$ is bounded above by $e^2 N^{-(A/2-2)}$ [this follows because $x/(1-x)<2x$ for all x obeying $0<x<1/2$]. This result means that for all $S \ll N^{1/2}$, the sum of the probabilities for all values of M between 1 and S yields the target is bounded above by $e^2 N^{-(A/2-2)}$.

IV. ARGUMENT THAT RESULTS INDICATE THAT THE ERROR ENERGY LANDSCAPE GETS VERY ROUGH WHEN $K \propto \ln N$

In the two preceding sections it was shown that if one starts with a target configuration that has a predecessor, then no configuration that has fewer than $S \lesssim N^{1/2}$ bits changed from the original predecessor is the predecessor of a new target that is obtained by changing one bit of the original target. Our interpretation of this result is that the error energy landscape becomes increasingly rough as K increases, in the sense that many of the new targets will have predecessors, but that these predecessors differ by many bits from the predecessors of the unperturbed targets. However, another possible explanation for the results is that the perturbed targets do not have predecessors at all. We have not been able to

prove that the second scenario does not happen, but we have performed numerical enumerations on small systems of up to 24 elements and found that for a given N, the fraction of configurations with predecessors increases monotonically as K increases. These numerical results are evidence that the failure to find nearby predecessors reflects that the error energy landscape is becoming increasingly rough, in that a reasonable fraction of the perturbed targets have predecessors, but that these predecessors differ from those of the predecessors of the unperturbed targets by many bits.

V. RELEVANCE OF RESULTS TO QUESTIONS IN COMPUTATIONAL COMPLEXITY

Computational complexity theory is the study of how the computational resources required to solve a given problem grow with the size of the input needed to specify the problem [22]. The close relationships between the physics of random systems and computational complexity theory have been explored for nearly two decades [23,24].

Whether or not P, the complexity class of problems that can be solved in a time that grows polynomially with the size of the problem specification ("polynomial time"), and NP, the class of problems for which a solution can be verified in polynomial time, are distinct is a central unanswered question in computational complexity theory [25]. The class of NP-complete problems are equivalent in that being able to solve any one of them in polynomial time implies that any problem in NP can be solved in polynomial time [26–28]. An intuitive picture believed to be appropriate for NP-complete problems is that the presence of conflicting constraints, or "frustration" [29], causes each problem to have an "energy landscape" with many local minima, and finding the global minimum is difficult because typical algorithms must explore an extremely large number of local minima to find the global one [24].

Specifying a Kauffman net requires time and space polynomial in N so long as K grows no faster than $\ln(N)$: this follows because specifying the K inputs for each of the N elements takes a number of bits proportional to $NK\ln(N)$, and specifying the f_i takes $N2^K$ bits (one per output for each of the 2^K possible inputs for each element). When $K=A\log_2 N$ with A a constant, the problem of determining whether a given configuration has a predecessor is in NP.

The question of whether a given configuration of a given Kauffman net has a predecessor can easily be rewritten as an instance of satisfiability (SAT), a classic NP-complete problem [22,26,28]. An instance of SAT asks one to determine whether an assignment of $\mathcal{N}$ variables exists such that a set of $\mathcal{M}$ constraints on these variables can all be satisfied simultaneously. An often-studied type of SAT is $\mathcal{K}$-SAT, in which each constraint or clause consists of the logical OR of $\mathcal{K}$ conditions or literals, each of which is satisfied if a variable is set either to one or to zero. For example, a three-SAT problem with four variables A, B, C, and D, and four clauses is

$$(A=1 \text{ or } B=0 \text{ or } D=1) \text{ and } (A=0 \text{ or } C=1 \text{ or } D=0) \text{ and } (B=1 \text{ or } C=1 \text{ or } D=1) \text{ and } (A=0 \text{ or } C=0 \text{ or } D=1). \quad (9)$$

f_1		f_2		f_3		f_4	
inputs	output	inputs	output	inputs	output	inputs	output
000	0	000	1	000	0	000	0
001	1	001	0	001	1	001	1
010	1	010	1	010	1	010	0
011	0	011	0	011	1	011	1
100	0	100	1	100	0	100	0
101	1	101	0	101	0	101	1
110	1	110	1	110	1	110	0
111	0	111	0	111	0	111	1

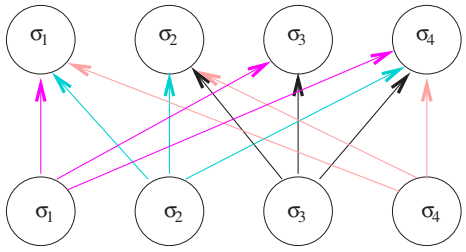

FIG. 2. (Color online) A Kauffman model with $N=4$ and $K=3$ whose equivalent satisfiability problem is discussed in the text. Each of the four elements σ_1, σ_2, σ_3, and σ_4 is shown twice, so that it is easier to see the inputs and outputs for all the elements.

This expression is satisfiable because it holds for the assignments $A=1$, $B=1$, $C=1$, $D=1$.

It is straightforward to rewrite the problem of finding a predecessor of Kauffman net configuration as a satisfiability problem. The variables are just the values of the different elements, and the constraints arise from the requirement that each element in the successor configuration be equal to the target. For example, consider the Kauffman net shown in Fig. 2, which consists of four elements $\sigma_1,\ldots,\sigma_4$. The inputs for σ_1 are σ_1, σ_2, and σ_4, the inputs for σ_2 are σ_2, σ_3, and σ_4, the inputs for σ_3 are σ_1 and σ_3, and the inputs for σ_4 are σ_1, σ_2, σ_3, and σ_4, and the Boolean functions specifying each output are shown in the figure. To write the problem of whether the configuration (1, 1, 1, 1) has a predecessor as a satisfiability formula in standard conjunctive normal form in which the formula is the conjunction (and) of disjunctions (or), it is useful to consider the configurations of the inputs that do *not* yield the desired outputs; e.g., for the functions specified in Fig. 2, the output $\sigma_1(t+1)$ is zero (and hence *not* the target) if at time t its inputs take on *none* of the values (000, 011, 100, and 111), which means the inputs satisfy

$$[\sigma_1(t)=1 \text{ or } \sigma_2(t)=1 \text{ or } \sigma_4(t)=1] \text{ and } [\sigma_1(t)=1 \text{ or } \sigma_2(t)=0 \text{ or } \sigma_4(t)=0] \text{ and } [\sigma_1(t)=0 \text{ or } \sigma_2(t)=1 \text{ or } \sigma_4(t)=1] \text{ and } [\sigma_1(t)=0 \text{ or } \sigma_2(t)=0 \text{ or } \sigma_4(t)=0]. \quad (10)$$

The satisfiability expression encoding the constraints on all the elements is just the conjunction of the subformulas describing the analogous constraints on the inputs for all the individual elements. The resulting satisfiability formula has N variables and approximately $N2^{K-1}$ clauses, each with K literals. The number of clauses follows because for randomly chosen unbiased functions the output specifying a given el-

ement is nonzero for approximately half of the 2^K different configurations of the inputs. The actual number and length of clauses is actually somewhat less because of simplifications arising when a function does not change when an input is changed—if, for fixed values of $\sigma_2, \ldots, \sigma_K$, the output is the same for $\sigma_1=0$ and $\sigma_1=1$, then the formula is equivalent to one in which two clauses each with K literals are replaced by one clause with $K-1$ literals. For instance, requiring the function f_2 in Fig. 2 to yield the output 0 can be simplified to ($\sigma_2=1$ or $\sigma_4=1$) and ($\sigma_2=0$ or $\sigma_4=1$). Ignoring this complication, the resulting instance is a K-SAT formula with N variables and approximately $N2^{K-1}$ clauses.

The predecessor problem with $K \propto \log N$ may be a particularly useful one to study because it may be a "self-organized critical" [19] version of satisfiability. The critical point in satisfiability that is usually studied occurs when one examines an ensemble of random satisfiability instances in which the $\mathcal{M}$ clauses are chosen uniformly at random (each clause consisting of $\mathcal{K}$ literals, with the variables in each occurring with the same probability, and equally likely to be negated or un-negated). If one fixes $\mathcal{K}$ and $\mathcal{N}$, as the number of clauses $\mathcal{M}$ is increased, then there is a transition between a SAT phase in which almost all instances are satisfiable and an unSAT phase in which almost all instance are unsatisfiable [30,31]. It is known that instances that are at the SAT-unSAT critical point are the most difficult to solve [30,31].

When $\mathcal{K}$ is large, the SAT-unSAT transition for problems in which the clauses are chosen uniformly at random (each variable occurring with the same probability, and either negated or un-negated with equal probability) occurs when α, the ratio of the number of clauses to the number of variables, is at a critical value α_{cSAT} that in the limit $K \to \infty$ satisfies: $\alpha_c \to 2^{\mathcal{K}} \ln 2 \approx (0.69)2^{\mathcal{K}}$ [32]. A simple estimate for $\alpha_{\mathrm{Kauffman}}$, the ratio of the number of clauses to the number of variables for a Kauffman net predecessor problem, is obtained by noting that the Kauffman net predecessor problem has N variables and approximately $N2^{K-1}$ clauses (this estimate ignores the fact that the satisfiability instance corresponding to a given Kauffman net predecessor problem has some clauses with lengths less than K), yielding $\alpha_{\mathrm{Kauffman}}=(0.5)2^K$. We compare α_{cSAT} and $\alpha_{\mathrm{Kauffman}}$ by ignoring the fact that for the Kauffman net predecessor problem the choices of the variables in different clauses are correlated (they divide up until N sets of roughly 2^{K-1} clauses that each involve the same K variables) and compute a dimensionless quantity to estimate how far the predecessor problem is from the SAT-unSAT critical point, $\delta\alpha \equiv (\alpha_{\mathrm{cKauffman}}-\alpha_{\mathrm{cSAT}})/\alpha_{\mathrm{cSAT}} \approx -0.28$. The most naive interpretation of this value is that the predecessor problem for a Kauffman net is close to the SAT-unSAT transition but still within the SAT phase, but this is unlikely because adding additional constraints by requiring an output to take on two different values causes the corresponding satisfiability formula to be surely unsatisfiable. The most natural interpretation is that the actual critical value of α in the predecessor problem is shifted slightly because of correlations between clauses, and that the predecessor problem is indeed tuned to be at a critical point.

It is easy to show that the predecessor problem is indeed "self-tuned" to be critical if $K=N$, when each configuration is a truly random function of its predecessor [33–35]. When $K=N$, the successors of any two different configurations are completely uncorrelated, and a given configuration has no predecessor with probability $1/e$ [16]. This result in itself indicates that the problem is self-tuned to be critical, because as $N \to \infty$ a randomly chosen formula is satisfiable with probability 1 in the SAT phase and 0 in the unSAT phase. The model can be generalized in different ways that make its criticality explicit. One way is to add constraints by multiply specifying the values of some outputs. The probability that a predecessor configuration exists is $1/e$ if the new constraints are compatible with the previously existing ones and 0 if they are not. Since doubly specifying an output yields an incompatible constraint with probability 1/2, the probability that a Boolean net with N elements and $N\delta$ doubly-specified outputs is satisfiable is $(1/e)(1/2)^{N\delta}$, which tends to zero as $N \to \infty$ for any $\delta > 0$. Conversely, the probability that at least one configuration has a successor that has $N(1-\rho)$ elements consistent with a given target set is $1-(1/e)^{(2^{N\rho})}$ (this can be seen by noting that no suitable predecessor configuration exists only if all the successor configurations consistent with the specifications have no predecessor). Therefore, if all but ρN output elements are specified, then the probability that at least one configuration satisfies the constraints approaches unity for any positive ρ as $N \to \infty$. Another way to generalize the model to exhibit explicitly its criticality is to choose Boolean functions from a probability distribution in which functions are chosen with probability p and $1-p$ if their outputs are zero and one, respectively. The probability that the target configuration that is all 1's does not have a predecessor is $(1-p^N)^{2^N}$, which for large N approaches $\exp[-(2p)^N]$. As $N \to \infty$, this probability approaches 0 when $2p > 1$ and 1 when $2p < 1$.

When K is finite, the predecessor problem is in the unSAT phase, because the probability that a randomly chosen configuration has a predecessor is bounded above by the value $(1-2^{-2^K})^N$. This bound follows because with probability $2^{-(2^K-1)}$ the function determining the value of any single element is independent of the values of all its inputs (this follows because there are 2^{2^K} Boolean functions with K inputs, 2 of which are independent of all the inputs), and for such a function, with probability 1/2 the output value will be inconsistent with the target [36]. For fixed K this upper bound on the probability that a randomly chosen configuration has a predecessor vanishes as $N \to \infty$, while when $K = A \log_2(N)$, this bound approaches unity as $N \to \infty$. These simple bounds lead to our conjecture that the Kauffman net predecessor problem with $K \propto \ln N$ is actually self-tuned so that it is at the critical point for a SAT-unSAT transition with clauses whose randomness is correlated.

From the point of view of computational complexity, Kauffman nets with $K \geqslant 3$ do not have a fundamental distinction from those with $K \propto \ln N$, because Kauffman nets with any $K \geqslant 3$ and $K \leqslant A \ln N$ can be specified using a number of bits that grows as a polynomial of N, and correspond to satisfiability instances with $\mathcal{K} \geqslant 3$, a regime for which randomly chosen satisfiability instances appear to require an exponentially long time to solve [17,37]. Nonetheless, this paper demonstrates that the local properties of the energy landscape exhibit systematic evolution as K is increased even

within this regime. Increasing K even further is known to make the predecessor problem harder. Indeed, when $K=N$, although specifying the model requires space that grows exponentially with N, one can still ask how many evaluations of the Boolean functions are required to determine whether such a predecessor exists. A candidate solution can be verified with a single evaluation of each Kauffman net function, but because in this case each configuration is a truly random function of its predecessor [33–35], the only way to determine whether a predecessor exists is to check all of the exponentially many candidates [16]. This result demonstrates that the Kauffman net with $K=N$ can act as an oracle relative to which P and NP are not equal [16].

The results in this paper demonstrate that a Kauffman net predecessor problem with $K \propto \ln N$ has properties that seriously constrain the computational strategies that could be used to solve in polynomial time. First, the algorithm must yield the exact answer, since the local search algorithm cannot correct even single-bit errors. Second, the algorithm must explicitly depend on the specification of every bit of every input function as well as every bit of the target configuration. The sensitivity to the choice of functions is because if one realization of the functions yields the target output, then a second function realization that differs from the first by a single bit change could yield a configuration that differs from the target by one bit, and the arguments given above then demonstrate that the predecessor for the second function realization, if it exists, has a large number of bits that are different than for the first function realization.

Demonstrating that *no* algorithm can solve the predecessor problem in time that grows no faster than polynomially with N would solve the famous P versus NP problem [25–27], and we do not purport to do that here. The properties of the Kauffman net predecessor problem investigated here may be useful in slightly different contexts, however. We have argued elsewhere [38] that Boolean functions of N variables have different phases, as defined by the behavior of the sequence of functions that is obtained when one eliminates individual variables via the transformation

$$\begin{aligned} h_i(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N) &= T[f(x_1, \ldots, x_N)] \\ &= f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_N) \\ &\quad \oplus f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_N), \end{aligned} \tag{11}$$

where $\oplus$ denotes addition modulo 2. This transformation is potentially interesting because (1) generic Boolean functions (those for which the output for a given input is chosen to be either one or zero with equal probability) are in a "generic" phase that has the property that applying the transformation yields another generic Boolean and (2) many well-known efficiently computable functions can be shown to be in a non-generic phase. The phases defined using Eq. (11) do not correspond to complexity classes as defined in computer science [22], though Ref. [38] speculates on possible nontrivial relationships. It may be useful to characterize the different phases of Boolean functions because the phase classification procedure is possible computationally and may yield a deeper understanding of other ways to characterize the complexity of a given function.

We argue here that it is very plausible that the successor and predecessor functions of a Kauffman net are in different phases. Because each output element only depends on K input elements, it is easy to see that the successor problem for a Kauffman net with $K \leqslant A \ln N$ is not in the generic phase, because the result of one renormalization step of Eq. (11) is nonzero only if changing a given input element changes the output. Conversely, since the predecessor problem is nonlocal, it is extremely plausible that whether or not changing one element of the target configuration affects whether a predecessor exists depends on the values of a large number of other elements. Whether it is possible to prove that this property persists for many levels of renormalization is an interesting open problem.

VI. SUMMARY

This paper shows that the problem of finding a predecessor configuration of a random Kauffman net is extremely sensitive to single-bit changes in the target configuration when the number of inputs to each element grows logarithmically with the number of elements. If one bit of a target configuration that has a predecessor is changed, then no configuration that differs from the original predecessor by a number of bits that grows as N^x for an x that is greater than zero is the predecessor of the perturbed target. We also show that the predecessor problem has some features of "self-organized criticality" in that it is naturally tuned to be near a critical point of an equivalent satisfiability problem. These properties may be useful in the quest to develop new strategies to characterize the computational resources needed to solve problems whose solutions can be verified in a number of computational steps that grows as a polynomial of the size of the problem specification.

ACKNOWLEDGMENTS

The author gratefully acknowledges financial support from Grant Nos. NSF-DMR 0209630 and NSF-EMT 0523680, and useful conversations with Eric Bach, Robert Joynt, and Dieter von Melkebeek. The hospitality of the Aspen Center for Physics, where some of this work was done, is greatly appreciated.

[1] S. Kauffman, Nature (London) **244**, 177 (1969).
[2] S. Kauffman, *The Origins of Order: Self-organization and Selection in Evolution* (Oxford University Press, Oxford, 1993).
[3] M. Aldana, S. Coppersmith, and L. Kadanoff, in *Perspectives and Problems in Nonlinear Science*, edited by E. Kaplan, J. Marsden, and K. Sreenivasan (Springer, Berlin, 2003).
[4] U. Bastolla and G. Parisi, Physica D **98**, 1 (1996).
[5] U. Bastolla and G. Parisi, Physica D **115**, 203 (1998).
[6] U. Bastolla and G. Parisi, Physica D **115**, 203218 (1998).
[7] A. Bhattacharjya and S. Liang, Phys. Rev. Lett. **77**, 1644 (1996).
[8] X. Qu, M. Aldana, and L. Kadanoff, J. Stat. Phys. **109**, 967 (2002).
[9] J. Hurford, IEEE Trans. Evol. Comput. **5**, 111 (2001).
[10] R. Axelrod, in *Simulating Social Phenomena*, edited by R. Conte, R. Hegselmann, and P. Terna (Springer, Berlin, 1997), pp. 21–40.
[11] S. Kauffman, J. Theor. Biol. **22**, 437 (1969).
[12] S. Kauffman, J. Theor. Biol. **44**, 167 (1974).
[13] S. Kauffman, Physica D **10**, 145 (1984).
[14] S. Kauffman, *At Home in the Universe: The Search for Laws of Self-Organization and Complexity* (Oxford University Press, Oxford, 1995).
[15] J. E. S. Socolar and S. A. Kauffman, Phys. Rev. Lett. **90**, 068702 (2003).
[16] C. Bennett and J. Gill, SIAM J. Comput. **10**, 96 (1981).
[17] O. Goldreich (unpublished).
[18] G. Baskaran and D. L. Stein, Phys. Rev. Lett. **59**, 373 (1987).
[19] P. Bak, C. Tang, and K. Wiesenfeld, Phys. Rev. Lett. **59**, 381 (1987).
[20] U. Bastolla and G. Parisi, Physica D **115**, 203 (1998).
[21] P. Diaconis and F. Mosteller, J. Am. Stat. Assoc. **84**, 853 (1989).
[22] C. Papadimitriou, *Computational Complexity* (Addison-Wesley, New York, 1994).
[23] Y. Fu and P. Anderson, J. Phys. A **19**, 1605 (1986).
[24] M. Mézard, G. Parisi, and M. Virasoro, *Spin Glass Theory and Beyond* (World Scientific, Singapore, 1987).
[25] C. M. Institute, http://www.claymath.org/millenium/
[26] S. Cook, in *Third Annual ACM Symposium on the Theory of Computing* (ACM, New York, 1971), pp. 151–158.
[27] L. Levin, SIAM J. Comput. **15**, 285 (1986).
[28] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, New York, 1979).
[29] G. Toulouse, Commun. Phys. (London) **2**, 115 (1977).
[30] P. Cheeseman, B. Kanefsky, and W. M. Taylor (unpublished).
[31] S. Kirkpatrick and B. Selman, Science **264**, 1297 (1994).
[32] D. Achlioptas, A. Naor, and Y. Peres, Nature (London) **435**, 759 (2005).
[33] B. Derrida, Phys. Rev. B **24**, 2613 (1981).
[34] B. Derrida and G. Weisbuch, J. Phys. (Paris) **47**, 1297 (1986).
[35] B. Derrida and H. Flyvbjerg, J. Phys. (Paris) **48**, 971 (1987).
[36] S. Coppersmith, L. P. Kadanoff, and Z. Zhang, Physica D **149**, 11 (2001).
[37] M. Mézard, G. Parisi, and R. Zecchina, Science **297**, 812 (2002).
[38] S. Coppersmith, eprint arXiv:cs.CC/0608053 (unpublished).