Quantum search heuristics

Tad Hogg*
Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, California 94304
(Received 19 October 1999; revised manuscript received 12 January 2000; published 14 April 2000)

An alternative quantum algorithm for combinatorial search, adjusting amplitudes based on number of conflicts in search states, performs well, on average, for hard random satisfiability problems near a phase transition in search difficulty. The algorithm exploits correlations among problem properties more effectively than some current heuristics, and improves on prior quantum algorithms that ignore these correlations.

PACS number(s): 03.67.Lx, 89.70.+c, 02.70.−c

Shor's polynomial-time factoring algorithm [1,2] showed quantum computers [3–6] efficiently solve an important problem thought to require exponential time on our current ''classical,'' machines. Can quantum computers significantly improve other apparently intractable problems? At first sight, combinatorial searches, such as arise in scheduling, theorem proving, cryptography, genetics, and statistical physics, are one possibility. This is because many such searches are ''nondeterministic polynomial'' (NP) problems [7], which have a rapid test of whether a candidate solution is in fact a solution and an exponential growth in the number of candidates with the size of the problem. Quantum computers can test all candidates in superposition with about as many operations as a classical machine uses to test just one, suggesting large improvements are possible. Unfortunately, the difficulty of extracting a solution from the superposition appears to preclude rapid solution of at least some NP problems [8], though, as is the case classically, this remains an open question.

Lacking a definitive result on the power of quantum computers, a practical fallback is how well they perform for *typical* searches encountered in practice. This distinction is important because classical heuristics, using readily computed problem properties to suggest candidates to test, solve many NP problems much more rapidly than worst-case analyses predict. For instance, constraint satisfaction problems [9], such as arise in scheduling, consist of a number of constraints on the values various combinations of variables can take. A candidate solution for such problems can not only be evaluated in terms of *whether* it satisfies all the constraints, but also in terms of *how many* constraints it violates. This additional information is often a useful guide to finding solutions, providing the basis for heuristic searches.

Some heuristics consist of independent trials, each succeeding with a small probability p. A simple example is hill climbing: starting from a random initial state, small changes are made as long as the state appears to improve, e.g., reducing the number of violated constraints. This process continues until a solution is found or the program reaches a ''local minimum,'' at which point none of the available changes give any further improvement. In the latter case, another trial is started from a new initial state. For use with quantum computers, these independent trials are modified to run for a prespecified number of steps, rather than allowing them to terminate early when a solution is found or continue indefinitely looking for a local minimum. For such heuristics, amplitude amplification [10] reduces the average cost from $1/p$ trials to only about [11] $1/\sqrt{p}$ by repeatedly testing all candidates in superposition. This quadratic speedup, which can also apply when constructing solutions incrementally [12], is the best possible for quantum methods based only on the test of whether candidates are solutions [8].

The trials of many heuristics are not independent (e.g., they use information gained from prior, unsuccessful trials), so amplitude amplification does not apply. An example in the context of hill climbing is emphasizing those changes to the current state that remove violations for constraints that remained unsatisfied at the end of many prior trials. One way to do this associates a weight with each constraint, which is incremented at the end of any trial for which the constraint remains unsatisfied. In each trial, the hill climbing operates with respect to the weighted sum of constraint violations rather than just their total number. A more complex heuristic involves caching new constraints inferred during the trials as in so-called ''truth maintenance systems'' [13]. These additional constraints can improve subsequent searches, but also incur additional overhead leading to additional mechanisms that erase those inferred constraints judged to be no longer useful.

Finally, some heuristics do not consist of separate trials at all but rather involve, for any particular problem instance, an exponentially long computation before a solution is found. Examples are methods that construct solutions incrementally and involve backtracking to prior decision points when conflicts are found. Such methods have an exponentially large variation in the solution time among different problem instances, so arbitrarily stopping the search after a prespecified number of steps will give zero probability to obtain a solution for some instances, and probability 1 for others, thus giving no opportunity for improvement with amplitude amplification.

Any possibility of achieving greater than quadratic speedup of independent-trial heuristics, or utilizing the capabilities of other heuristics with quantum computers, requires using additional problem properties directly in the quantum algorithm. For some small or relatively easy problems such techniques are known to have high performance [14,15]. More generally, with precise information on states' distances

*URL: http://www.parc.xerox.com/hogg

to a solution, quantum methods perform well [16], but such information is not readily available for hard searches.

Heuristics often introduce complex dependencies among successive search choices, preventing a theoretical analysis of their performance; instead, they are usually evaluated empirically on a sample of typical problems. To be a useful test, this requires a class of problem instances with a high concentration of hard cases. Fortunately, such classes have been identified for a variety of NP-complete search problems [17–19]. In particular, these classes correspond to problems with an intermediate number of constraints, while those with fewer or more constraints tend to be fairly easy. Significantly, such classes of hard problems, associated with abrupt ''phase transitions'' in behavior, are found to be particularly difficult for a variety of heuristics, making them good test cases for evaluating new methods. Hence a particularly good indication of the practical utility of quantum computers for search is to study their capabilities for precisely these classes of problems that are difficult for a wide range of heuristic methods.

Toward this end, we present an alternative quantum algorithm that performs well, on average, for hard k satisfiability (k-SAT) problems, which consist of n Boolean variables and m clauses. A clause is a logical OR of k variables, each of which may be negated. A solution is an assignment, i.e., a value, true or false, for each variable, that satisfies all the clauses. An assignment is said to conflict with any clause it does not satisfy. An example 2-SAT problem with three variables and two clauses is v_1 OR (NOT v_2) and v_2 OR v_3, which has four solutions, e.g., v_1=false, v_2=false, and v_3=true. For $k \geqslant 3$, k-SAT is NP-complete [7], i.e., is among the most difficult of NP problems.

A well-studied class of such problems is random k-SAT, in which the m clauses are selected uniformly at random. Specifically, for each clause, a set of k variables is selected randomly from among the $\binom{n}{k}$ possibilities. Then each of the selected variables is negated with probability 1/2 to produce the clause. Thus each of the m clauses is selected, with replacement, uniformly from among the $\binom{n}{k}2^k$ possible clauses. The difficulty of solving such randomly generated problems varies greatly from one instance to the next. This class has a high concentration of hard instances when $\mu \equiv m/n$ is near a phase transition in search difficulty [17–19]. For random 3-SAT this transition is near $\mu = 4.25$, the value used for the results presented here as well as extensive prior studies of classical heuristics for SAT. When μ is at least somewhat smaller or larger than this value, the problems are typically much easier for both classical and quantum methods, including the quantum algorithm presented here. Thus as with evaluating classical heuristics, testing quantum algorithms using a value of μ near the transition gives a particularly stringent test on their average effectiveness.

The quantum algorithm consists of a series of steps, with amplitude adjustments varying linearly with the step and the number of conflicts [14], a property commonly used in classical heuristics. Specifically, four real-valued ''phase parameters'' R_0, R_1, T_0, and T_1 define

$$\rho_h = \frac{1}{j} R\left(\frac{h-1}{j}\right), \qquad \tau_h = \frac{1}{j} T\left(\frac{h-1}{j}\right) \tag{1}$$

for steps $h = 1, \ldots, j$ with $R(\lambda) = R_0 + R_1(1-\lambda)$, $T(\lambda) = T_0 + T_1(1-\lambda)$. These values are used as follows.

Superpositions are described by a vector with an amplitude for each assignment. Starting with an equal superposition of all 2^n assignments, i.e., $\psi_s^{(0)} = 2^{-n/2}$, the superposition $\psi^{(j)}$ after j steps is

$$\psi^{(j)} = U^{(j)} P^{(j)} \cdots U^{(1)} P^{(1)} \psi^{(0)}. \tag{2}$$

The algorithm involves two types of matrices: the diagonal phase adjustments $P^{(h)}$, depending on the particular problem instance being solved, and the matrix $U^{(h)}$, mixing amplitudes from different states without regard to the particular problem.

Specifically, $P^{(h)}$ is diagonal with $P_{ss}^{(h)} = e^{i\pi\rho_h c(s)}$ where $c(s)$ is the number of conflicts in assignment s. Since $c(s)$ itself is efficiently evaluated (by comparing the state with each of the m clauses) and has only $m+1 = O(n)$ possible values, $0, \ldots, m$, this matrix operation can be implemented efficiently on quantum computers [22] as a generalization of the technique used for amplitude amplification. To see this, let C be the operator reversibly evaluating $c(s)$ that takes $|s,a\rangle$ to $|s, a \oplus c(s)\rangle$ where $\oplus$ denotes the bitwise exclusive-OR operation and a is an additional register capable of representing integers up to m. Let $\hat{P}^{(h)}$ be the $(m+1)\times(m+1)$ diagonal matrix with entries $e^{i\pi\rho_h c}$ for c from 0 to m. Starting with the superposition of states $\Sigma_s \psi_s |s,0\rangle$, applying C gives $\Sigma_s \psi_s |s, c(s)\rangle$. We then apply $I \otimes \hat{P}^{(h)}$, giving $\Sigma_s \psi_s e^{i\pi\rho_h c(s)} |s, c(s)\rangle$. This operation involves only the polynomial-sized diagonal matrix $\hat{P}^{(h)}$ acting on the extra register, and so can itself be implemented efficiently. A second application of C then disentangles the additional register, $\Sigma_s \psi_s e^{i\pi\rho_h c(s)} |s,0\rangle$, which is the required operation involving the exponentially large matrix $P^{(h)}$.

Viewing assignments as strings of n bits, the mixing matrix is defined as $U^{(h)} = W T^{(h)} W$, where $T^{(h)}$ is diagonal with $T_{ss}^{(h)} = e^{i\pi\tau_h |s|}$, $|s|$ denotes the number of 1-bits in s, and W is the Walsh transform, $W_{rs} = 2^{-n/2}(-1)^{|r \wedge s|}$, where $|r \wedge s|$ is the number of 1's the two assignments have in common. Thus $U_{rs}^{(h)} = 2^{-n} \Sigma_t e^{i\pi\tau_h |t|} (-1)^{|r\wedge t| + |s \wedge t|}$. Each 1-bit of t contributes 0, 1, or 2 to $|r \wedge t| + |s \wedge t|$ when the corresponding positions of r and s are both 0, have exactly a single 1-bit, or are both 1, respectively. Thus $(-1)^{|r\wedge t| + |s\wedge t|} = (-1)^z$ where z is the number of 1-bits in t that are in exactly one of r and s: such bits of t can be selected only from positions where r and s have different values. The number of such positions equals $d(r,s)$, the Hamming distance between r and s or, equivalently, the number of variables assigned different values in r and s. This gives $U_{rs}^{(h)} = 2^{-n} \Sigma_t e^{i\pi\tau_h |t|} (-1)^z$. Among the states t with x 1-bits, there are $\binom{d}{z}\binom{n-d}{x-z}$ with a given value of z, so the sum can be written as

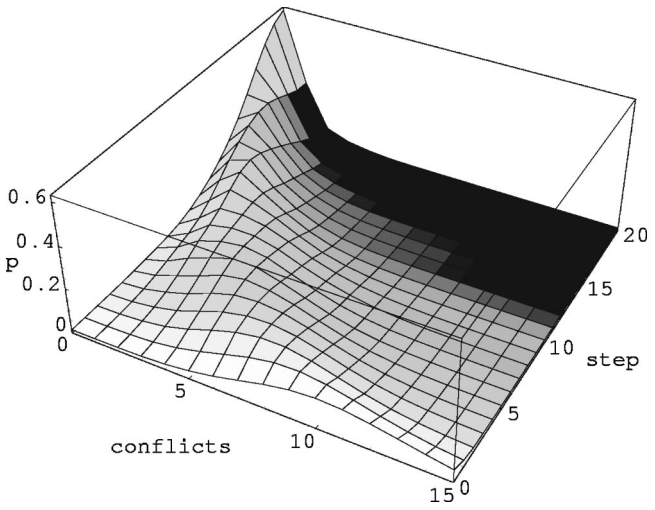

FIG. 1. Search behavior for a randomly generated 3-SAT problem with $n=20$ and $\mu=4.25$. For each step h, the figure shows the probability $p^{(h)}(c)$ in assignments with each number of conflicts. Shading is based on the relative deviations of the amplitudes, described in the text. The small contributions for assignments with $c>15$ are not included in the figure.

$$U_{rs}^{(h)}=2^{-n}\sum_{xz} e^{i\pi\tau_h x}(-1)^z\binom{d}{z}\binom{n-d}{x-z}$$
$$=2^{-n}(1-e^{i\pi\tau_h})^d(1+e^{i\pi\tau_h})^{n-d}. \quad (3)$$

Thus $U_{rs}^{(h)}$ is [20], up to an overall phase and normalization constant, $(-iv_h)^{d(r,s)}$, where $v_h=\tan(\pi\tau_h/2)$.

With these definitions, Eq. (2) can be evaluated efficiently by a quantum computer [10,21,22]. Observing the final superposition gives an assignment having c conflicts with probability $p^{(j)}(c)=\Sigma_{s|c(s)=c}|\psi_s^{(j)}|^2$, with the sum over all assignments with c conflicts.

The best choices for the phase parameters and number of steps depend on the problem instance. In practice, these will not be known. Instead, we select parameters that work well on average for random 3-SAT with a given value of μ. For $\mu=4.25$, the approximate analysis given below predicts good average performance if j grows at least as fast as n (for definiteness we take $j=n$) and $R_0=4.86\,376$, $R_1=-4.18\,118$, $T_0=1.2$, and $T_1=3.1$. Figure 1 shows the behavior for one problem instance: each step shifts the peak in the probability distribution toward assignments with fewer conflicts, until a large probability builds up in the solutions. This shift is also seen for other problem instances (as well as when averaged over many samples), but with differing final probabilities. This behavior contrasts with amplitude amplification where the probability in solutions increases but all other amplitudes decrease uniformly.

In this algorithm, amplitudes depend on the problem's conflict distribution, precluding an exact analytic evaluation of the algorithm's asymptotic behavior. Instead, as for most classical heuristics, one must rely on empirical evaluation or approximate analyses. Figure 2 shows the growth of the search cost, measured by the expected number of steps, $j/p^{(j)}(0)$, based on collections of randomly generated problems. The exponential fit gives the cost growing as $e^{0.10n}$. The observations for this range of variables are also close to a power-law fit, growing as $n^{1.7}$.

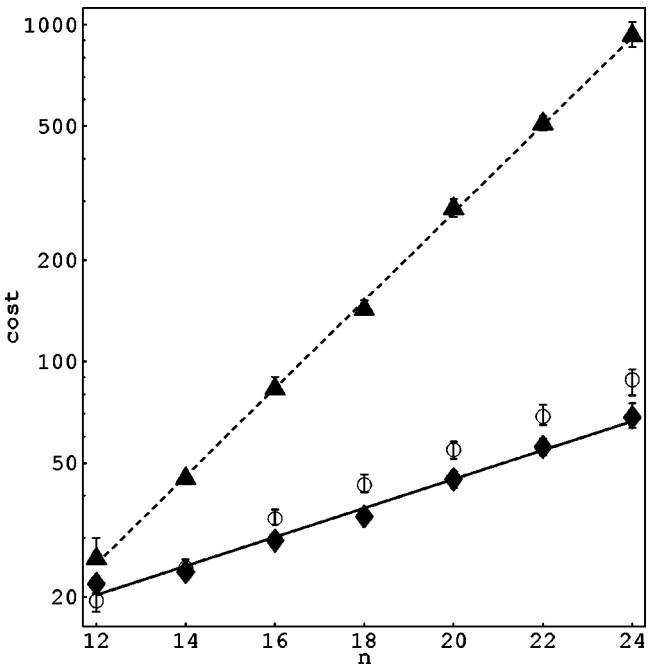

FIG. 2. Log plot of median search cost vs n for the quantum heuristic (diamond), amplitude amplification *assuming the number of solutions is known* (triangle), and GSAT [23] with restarts after $2n$ steps (circle). For each n, the same 1000 soluble random 3-SAT problems with $\mu=4.25$ were solved with each method (except only 500 samples for $n=24$). For those n not divisible by 4, half the samples had $m=\lfloor 4.25n\rfloor$ and half had m larger by one. Error bars show the 95% confidence intervals ([40], p. 124). The curves show exponential fits to the quantum heuristic (solid) and amplitude amplification (dashed).

Provided the number of solutions S is known, the cost for the simplest amplitude amplification is [21] $(\pi/4)\sqrt{2^n/S}$, also shown in Fig. 2. The values grow as $e^{0.30n}$. In practice, S is not known *a priori*, but even so the expected cost is less than four times larger [21], so it does not affect the exponential growth rate.

Average costs for even the best known classical heuristics also grow exponentially, though more slowly. For instance, Fig. 2 shows that a good classical heuristic, GSAT [23], grows somewhat faster than this quantum heuristic. The GSAT algorithm starts from a random assignment and, for each step, examines the number of conflicts in the assignment's neighbors (i.e., assignments obtained by changing the value for a single variable) and moves to a neighbor with the fewest conflicts. If a solution is not found after a prespecified number of steps, e.g., because the current assignment is a local minimum, the search is tried again from a new random assignment. The most significant comparison between GSAT and the quantum heuristic is the relative growth rates in the search costs, as measured by the number of steps. This is because actual search times will depend on detailed implementations of the steps. Although the number of elementary computational steps involving evaluating the number of con-

flicts in an assignment (and, in the case of GSAT, its neighbors) is similar for both techniques, differences in the extent to which operations can be optimized away (e.g., as is possible in some cases for NMR-based quantum implementations [35]) and the relative clock rates of classical and quantum machines remain to be seen. At any rate, Fig. 2 shows that including the number of conflicts in the phase adjustments reduces, on average, the number of steps required for the quantum algorithm below that required for GSAT. This is true even though GSAT in fact makes use of somewhat more problem structure than this quantum algorithm, namely the difference in numbers of conflicts between a state and its neighbors. Because the trials are independent, both the quantum heuristic introduced here and GSAT can be quadratically improved with amplitude amplification [11].

As another comparison, a good classical backtracking algorithm scales [24] as $2^{n/19.5}=e^{0.036n}$. This technique uses additional problem structure, namely conflicts in partial assignments, and more complicated processing at each step precluding a simple cost comparison for the small problem sizes accessible to classical simulation of the quantum heuristic. Moreover, unlike GSAT and the quantum heuristic, the backtracking algorithm is a complete search method, i.e., it can not only find a solution if one exists but can also determine that a problem has no solutions. Nevertheless, it is interesting to note that, even without making use of this additional structure, the quantum heuristic, when combined with amplitude amplification, has an exponential growth rate of $\sqrt{e^{0.10n}}=e^{0.05n}$, only a bit larger than this classical backtracking method. In studies using problems with hundreds or thousands of variables, techniques such as GSAT, based on hill climbing, significantly outperform backtracking algorithms [23] for soluble problems. Thus comparing the quantum heuristic to GSAT is a more important evaluation than using a backtracking algorithm.

From this discussion, the quantum algorithm appears to improve on some classical heuristics for a well-studied class of hard NP searches, but definitive statements cannot be made based only on such small problems. Unfortunately, classical simulations of quantum machines incur an exponential slowdown, preventing evaluation with larger problems.

This leaves the option of an approximate ''mean-field'' analysis using average properties of random *k*-SAT, which successfully helps understand and improve classical heuristics [25,17,26,27]. Empirical observation of how the algorithm changes the amplitudes for assignments with c conflicts, illustrated schematically in Fig. 3, indicates that for the dominant values of c at each step, as n increases the amplitudes are adequately characterized by average values $\Phi_c^{(h)} \equiv \langle \psi_r^{(h)} \rangle_c$, where $\langle\ \rangle_c$ denotes the average over assignments with c conflicts. With this approximation, step h of Eq. (2) becomes

$$\psi_r^{(h)} \propto \sum_{dc} (-iv_h)^d e^{i\pi\rho_h c} \Phi_c^{(h-1)} \nu(r;d,c), \tag{4}$$

where $\nu(r;d,c)$ is the number of assignments with c conflicts at distance d from assignment r. Equation (4) gives a mean-field approximation for $\Phi_{c'}^{(h)}$ by replacing $\nu(r;d,c)$ with its average value $\langle \nu(r;d,c) \rangle_{c'}$, which equals $\binom{n}{d} P(c|c',d)$ where $P(c|c',d)$ is the probability an assignment has c conflicts when at distance d from another with c' conflicts. This conditional probability is $P(c|c',d) = P(c,c'|d)/P(c')$ where $P(c') = \binom{m}{c'} 2^{-km} (2^k-1)^{m-c'}$ is the probability an assignment has c' conflicts, and $P(c,c'|d) = \Sigma_B P(B, c-B, c'-B|d)$ is the joint probability two assignments separated by distance d have, respectively, c and c' conflicts. Finally, $P(B,b,b'|d)$ is the probability the assignments have B conflicts in common and, respectively, b and b' unique conflicts. The explicit form for this probability distribution depends on the class of problems. For random *k*-SAT, $P(B,b,b'|d)$ is a multinomial distribution:

$$\binom{m}{B,b,b',m-B-b-b'} P_{\text{both}}^{B} P_{\text{unique}}^{b+b'} P_{\text{rest}}^{m-B-b-b'}, \tag{5}$$

where $P_{\text{both}} = 2^{-k}\binom{n-d}{k}/\binom{n}{k}$, $P_{\text{unique}} = 2^{-k} - P_{\text{both}}$, and $P_{\text{rest}} = 1 - 2P_{\text{unique}} - P_{\text{both}}$ are the probabilities a randomly selected clause conflicts with both assignments, with r but not s, or with neither assignment, respectively. For instance, $n-d$ variables have the same assigned value in both r and s, hence $\binom{n-d}{k}$ choices for the k variables appearing in a clause will involve only these commonly assigned variables. For such a choice, the clause conflicts with both r and s, with probability 2^{-k}, or with neither. Combining these factors gives the value of P_{both} since in random *k*-SAT each clause is selected uniformly at random. Similarly, the expression for P_{unique} makes use of the fact that a random clause has probability 2^{-k} to conflict with a given assignment. With these expressions, Eq. (4) relates the average amplitudes after step h to those of the prior step.

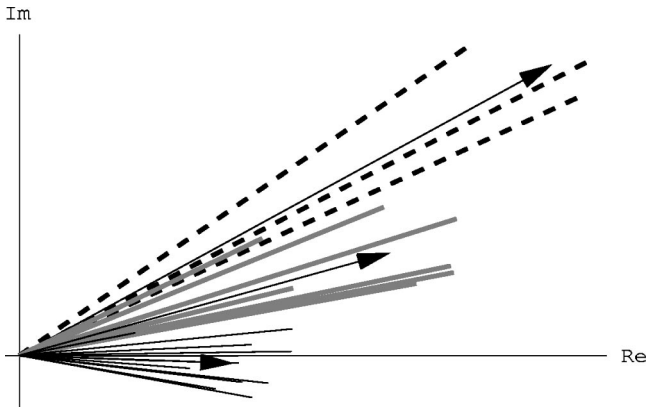

FIG. 3. Schematic illustration of amplitude clustering for an intermediate step of the algorithm. The lines represent amplitudes as vectors in the complex plane. The groups are for assignments with $c=0$, i.e., solutions (dashed), $c=1$ (gray), and $c=2$ (solid). The arrows show the average values $\Phi_c^{(h)}$. Values for $c>2$ are not shown.

Figure 4 compares this approximation of $\Phi_c^{(h)}$ with actual values for a sample of random problems halfway through the algorithm, i.e., after step $h=n/2$. Since a constant shift in the

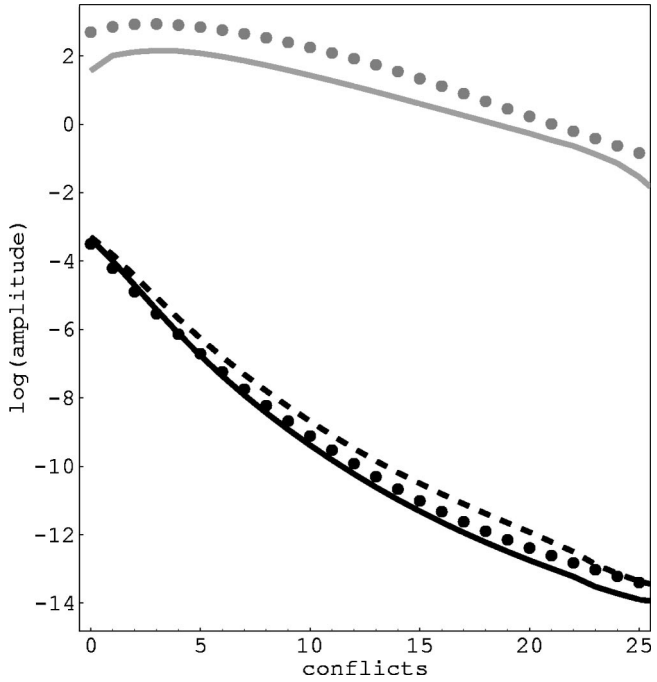

FIG. 4. Behavior of amplitude average and spread. Real (black) and imaginary (gray) parts of $\ln \Phi_c^{(10)}$ vs c. The curves show empirical values, averaged over 100 random 3-SAT problems with $n=20$, $\mu=4.25$. The points show the predictions from Eq. (4). The dashed curve shows the empirical values of $\ln \Psi_c^{(10)}$.

imaginary values is an irrelevant overall phase, the approximation is fairly good. To indicate how well $|\Phi_c^{(h)}|$ characterizes amplitude sizes, $\Psi_c^{(h)}=\sqrt{\langle|\psi_r^{(h)}|^2\rangle_c}$ is also shown. Particularly for the dominant c values, below $c=5$ in Fig. 4, the difference in behavior is relatively small. For the behavior vs h, Fig. 1 shows the relative deviation $\sqrt{\langle|\psi_r^{(h)}-\Phi_c^{(h)}|^2\rangle_c}/|\Phi_c^{(h)}|$ by the shading, ranging from white, when this ratio is 0, to black, when it is larger than 3. The largest ratios occur only in the last few steps and for values of c for which $p^{(h)}(c)$ is quite small, limiting their effect.

As illustrated in Fig. 1, at each step amplitude concentrates in a narrow range of conflicts. This concentration becomes more pronounced as n increases, so the smooth variation of $\ln \Phi_c^{(h)}$ in Fig. 4 allows a linear expansion to capture the main behavior as $n\to\infty$, i.e., taking $\Phi_c^{(h)} \propto e^{(-f_h/2+i\pi a_h)c}$, with f_h and a_h real-valued parameters. The constant of proportionality provides normalization and an irrelevant overall phase. The initial superposition, with equal amplitudes, has $f_0=0$, $a_0=0$. The c values dominating the amplitudes will be those near the average $\Sigma_c c 2^n P(c)|\Phi_c^{(h)}|^2$, which, with this expansion, is $c_{\rm avg}(f_h)=m/F(f_h)$ where $F(f)\equiv(2^k-1)e^f+1$. For hard random k-SAT problems $m \propto n$ so significant amplitude is in solutions whenever e^f is at least of order n.

Using this linear form for $\Phi_c^{(h)}$ and Eq. (5) in Eq. (4) and expanding around $c_{\rm avg}(f_{h-1})$ give a recursion for f_h and a_h in terms of f_{h-1} and a_{h-1}. For large n, the changes at each step are small so the recursion is approximated by a differential equation by writing $f_h\equiv f(\lambda)$, $a_h\equiv a(\lambda)$ with $\lambda=h/j$, giving

$$\frac{df}{d\lambda}=\pi A e^{-f/2}F(f)\sin(\pi a),$$

$$\frac{da}{d\lambda}=R(\lambda)-\frac{A}{2}[e^{-f/2}(F(f)-2)\cos(\pi a)-2^k+2], \tag{6}$$

with initial conditions $f(0)=a(0)=0$, with

$$A=T(\lambda)\frac{k}{2^k-1}\exp\left(-k\mu\frac{e^f-2e^{f/2}\cos(\pi a)+1}{F(f)}\right) \tag{7}$$

and R and T defined with Eq. (1).

For most phase parameters, the solution is well behaved, giving $O(1)$ values for f over the full range of λ as $n\to\infty$. In such cases, $c_{\rm avg}$ remains proportional to m and amplitudes are not significantly concentrated in solutions. However, Eq. (6) can also develop logarithmic singularities, e.g., with the limiting form $f(\lambda)\sim -2\ln(1-\lambda)$ and $a(\lambda)\sim 1/2$ as $\lambda\to 1$. In general such solutions will not also satisfy the initial conditions $f(0)=a(0)=0$. Instead, requiring the solution to satisfy both sets of conditions imposes two constraints on the four phase parameters. That is, for given choices of, say, T_0 and T_1, such solutions exist only for specific choices of R_0, R_1 that can be found numerically. One such choice, given above, is used in the figures. Small changes in T_0, T_1, with R_0, R_1 adjusted to maintain the singularity in Eq. (6), give similar behavior. This technique also applies to other classes of random k-SAT, e.g., with different μ.

Since the recursion for f and a remains finite for all h, the growth in f_h deviates from the singular solution of Eq. (6) for the last few steps. This deviation is significant when the next term in the Taylor series expansion of the recursion, $f''(\lambda)/j^2$, is comparable to $f'(\lambda)/j$, i.e., when $1-\lambda=O(1/j)$. For large n, this limits e^{f_j} to be of order n^2, more than enough to give significant probability in solutions. Thus with suitable choices for the phase parameters, this analysis predicts the average number of search steps to find that a solution grows only linearly in n. Based on small problem sizes, Fig. 2 shows that the analysis correctly identifies phase parameters giving good average performance for hard random 3-SAT. In particular, the actual cost grows more slowly than classical heuristics using similar problem information. However, the costs in Fig. 2 grow faster than linearly. If this continues for larger n, identifying the correct scaling requires improving the analysis, e.g., accounting for the spread in amplitudes among states with c conflicts arising from the variance in $\nu(r;d,c)$ values in Eq. (4). The remaining freedom to select T_0, T_1 and the number of steps, and introduce some nonlinearity in $R(\lambda)$ and $T(\lambda)$, could help minimize the spread.

A number of extensions are possible. First, the amplitude shift of Fig. 1 also occurs in problems with no solutions: amplitude is enhanced in states with few conflicts. Thus, like local classical search methods such as GSAT but unlike amplitude amplification, the algorithm applies directly to combinatorial optimization, i.e., finding a minimal conflict state [28].

Second, the mean-field analysis also applies to other classes of search problems, provided, as with Eq. (5), the probabilities relating problem properties can be determined. This is possible for a variety of commonly studied random search classes. More realistic classes lack analytically known probability distributions, but sampling representative instances allows estimating the $P(c|c',d)$. Such estimates may even be useful for random ensembles, allowing some tuning of phase parameters for a particular problem instance.

Third, in common with amplitude amplification [21] and some classical methods [29], the growth of $p^{(h)}(0)$, as seen in Fig. 1, means stopping a bit before the largest probability reduces the cost. Furthermore, the wide performance distribution makes this algorithm suitable for improvement via portfolios [30,31]. This is especially true due to the small correlation between the costs of the quantum heuristic and GSAT among problems with the same number of solutions, as shown in Fig. 5. More specifically, among random 3-SAT problems, much of the variation in costs for both methods is due to the differing numbers of solutions. For the remaining cost variation among those with the same number of solutions, the most difficult cases for the quantum method are not also the most difficult for classical methods such as GSAT, and vice versa. Hence applying both GSAT and this quantum heuristic to a set of problems, halting when either finds a solution, can further reduce the median cost, particularly when this portfolio is itself combined with amplitude amplification.

This heuristic relies on the correlation between number of conflicts and distance to a solution. Other properties classical heuristics exploit may also give useful phase adjustments. Examples include how an assignment's conflicts compare to those of its neighbors, conflicts in partial assignments used in backtracking searches, how an assignment to one variable affects others, and identifying new constraints during search. Conversely, for which searches are these correlations too weak for quantum methods to use effectively? This question is particularly important for cryptography, which relies on easily finding hard searches that are readily solved with additional information (i.e., the key) [32].

These results indicate additional problem properties allow quantum searches to perform better than previously thought, but one must keep in mind their limitations: as with studies of classical heuristics, they do not provide rigorous bounds on the average search cost and, even if the algorithm performs well on average, they give no guarantee for specific instances. Nevertheless, restricting consideration to algorithms whose behavior is analytically simple is likely to underestimate the potential of quantum computers for typical searches. With ongoing developments in error correction [33,34] and implementation [35–39], quantum machines with even a modest number of bits and limited coherence time could help address these issues by evaluating heuristics beyond the range of classical simulation. This will be particularly useful for more complicated heuristics, using additional problem properties, whose theoretical analysis is even more difficult. Exploring their behavior will identify opportunities quantum computers have for using information available in combinatorial searches to significantly improve performance.

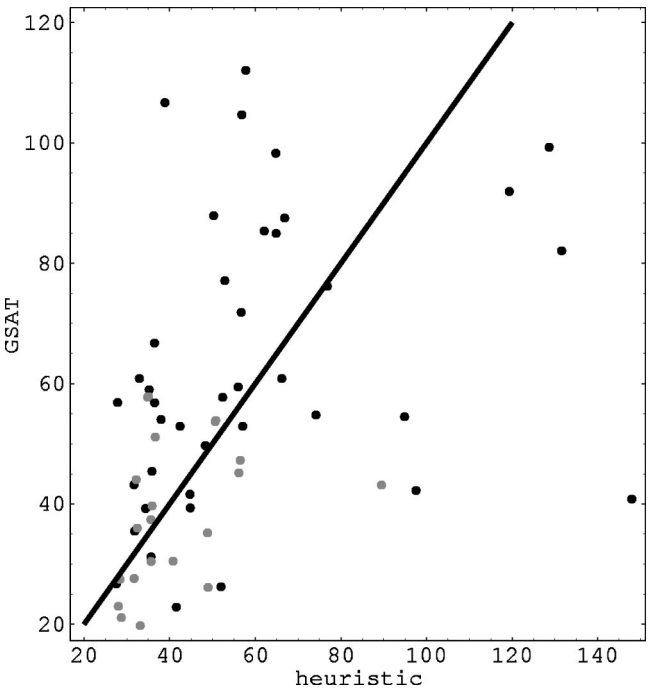

FIG. 5. Comparison of quantum heuristic and GSAT costs for some of the random 3-SAT problems with $n=20$, $\mu=4.25$ used in Fig. 2, whose median number of solutions is 8. The black and gray points correspond to problems with 8 and 15 solutions, respectively. Within each group, the correlation coefficient between the two methods is about 30%. Those points above the line are more costly to solve with GSAT than the quantum heuristic, and vice versa.

I have benefited from discussions with Matt Franklin, Wolf Polak, Eleanor Rieffel, and Christof Zalka.

[1] P. W. Shor, in *Proceedings of the 35th Symposium on Foundations of Computer Science*, edited by S. Goldwasser (IEEE Press, Los Alamitos, CA, 1994), pp. 124–134.
[2] I. L. Chuang, R. Laflamme, P. W. Shor, and W. H. Zurek, Science **270**, 1633 (1995).
[3] D. Deutsch, Proc. R. Soc. London, Ser. A **400**, 97 (1985).
[4] D. P. DiVincenzo, Science **270**, 255 (1995).
[5] R. P. Feynman, in *Feynman Lectures on Computation*, edited by J. G. Hey and R. W. Allen (Addison-Wesley, Reading, MA, 1996).
[6] A. Steane, Rep. Prog. Phys. **61**, 117 (1998).
[7] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W. H. Freeman, San Francisco, 1979).
[8] C. H. Bennett, E. Bernstein, G. Brassard, and U. V. Vazirani, SIAM J. Comput. **26**, 1510 (1997).
[9] A. Mackworth, in *Encyclopedia of Artificial Intelligence*, edited by S. Shapiro (Wiley, New York, 1992), pp. 285–293.

[10] L. K. Grover, Phys. Rev. Lett. **78**, 325 (1997); Los Alamos e-print quant-ph/9706033.
[11] G. Brassard, P. Hoyer, and A. Tapp, in *Proceedings of the 25th International Colloquium on Automata, Languages, and Programming (ICALP98)*, edited by K. Larsen (Springer, Berlin, 1998), pp. 820–831; Los Alamos e-print quant-ph/9805082.
[12] N. J. Cerf, L. K. Grover, and C. P. Williams, *Applicable Algebra in Engineering, Communication and Computing* (Springer, Berlin, 1998); Los Alamos e-print quant-ph/9806078.
[13] J. de Kleer, Artif. Intel. **28**, 127 (1986).
[14] T. Hogg, Phys. Rev. Lett. **80**, 2473 (1998); e-print at publish.aps.org/eprint/gateway/eplist/aps1997oct30_002.
[15] L. Spector, H. Barnum, H. J. Bernstein, and N. Swamy, *Proceedings of the 1999 Congress on Evolutionary Computing*, edited by P. Angeline (IEEE, Washington, D.C., 1999).
[16] L. K. Grover, *Proceedings of the 1st NASA International Conference on Quantum Computing and Quantum Communications*, edited by C. Williams and S. Gulati (Springer, Berlin, 1998); Los Alamos e-print quant-ph/9802035.
[17] P. Cheeseman, B. Kanefsky, and W. M. Taylor, in *Proceedings of IJCAI91*, edited by J. Mylopoulos and R. Reiter (Morgan Kaufmann, San Mateo, CA, 1991), pp. 331–337.
[18] S. Kirkpatrick and B. Selman, Science **264**, 1297 (1994).
[19] *Frontiers in Problem Solving: Phase Transitions and Complexity*, edited by T. Hogg, B. A. Huberman, and C. P. Williams, special issue of Artif. Intel. **81** (1996).
[20] T. Hogg, Los Alamos e-print quant-ph/9812049 (unpublished).
[21] M. Boyer, G. Brassard, P. Hoyer, and A. Tapp, in *Proceedings of the Workshop on Physics and Computation (PhysComp96)*, edited by T. Toffoli *et al.* (New England Complex Systems Institute, Cambridge, MA, 1996), pp. 36–43.
[22] T. Hogg, C. Mochon, E. Rieffel, and W. Polak, Int. J. Mod. Phys. C **10**, 1347 (1999); Los Alamos e-print quant-ph/9811073.
[23] B. Selman, H. Levesque, and D. Mitchell, in *Proceedings of the 10th National Conference on Artificial Intelligence (AAAI92)*, edited by P. Rosenbloom and P. Szolovits (AAAI Press, Menlo Park, CA, 1992), pp. 440–446.
[24] J. M. Crawford and L. D. Auton, Artif. Intel. **81**, 31 (1996).
[25] S. Edelkamp and R. E. Korf, in *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI98)*, edited by J. Mostow and C. Rich (AAAI Press, Menlo Park, CA, 1998), pp. 299–304.
[26] T. Hogg, Int. J. Mod. Phys. C **9**, 13 (1998).
[27] I. P. Gent, E. MacIntyre, P. Prosser, and T. Walsh, in *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI96)*, edited by W. Clancey and D. Weld (AAAI Press, Menlo Park, CA, 1996), pp. 246–252.
[28] E. C. Freuder and R. J. Wallace, Artif. Intel. **58**, 21 (1992).
[29] M. Luby, A. Sinclair, and D. Zuckerman, International Computer Science Institute Report No. TR-93-010 (unpublished).
[30] B. A. Huberman, R. M. Lukose, and T. Hogg, Science **275**, 51 (1997).
[31] C. P. Gomes and B. Selman, in *Proceedings of the 13th Conference on Uncertainty in AI (UAI-97)*, edited by D. Geiger and P. Shenoy (Morgan Kaufmann, Los Altos, CA, 1997), pp. 190–197.
[32] N. Koblitz, *Algebraic Aspects of Cryptography* (Springer, Berlin, 1998).
[33] P. Shor, Phys. Rev. A **52**, 2493 (1995).
[34] E. Knill, R. Laflamme, and W. H. Zurek, Science **279**, 342 (1998).
[35] I. L. Chuang *et al.*, Nature (London) **393**, 143 (1998); Los Alamos e-print quant-ph/9801037.
[36] I. L. Chuang, N. Gershenfeld, and M. Kubinec, Phys. Rev. Lett. **80**, 3408 (1998).
[37] B. E. Kane, Nature (London) **393**, 133 (1998).
[38] P. M. Platzman and M. I. Dykman, Science **284**, 1967 (1999).
[39] J. E. Mooij *et al.*, Science **285**, 1036 (1999).
[40] G. W. Snedecor and W. G. Cochran, *Statistical Methods*, 6th ed. (Iowa State University, Ames, IA, 1967).