

Simulations of Error in Quantum Adiabatic Computations of Random 2-SAT Instances

by

Jay S. Gill

Submitted to the Department of Physics
in partial fulfillment of the requirements for the degree of

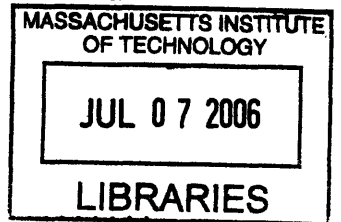
Bachelor of Science in Physics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

[June 2006]
May 2006

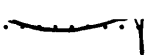
© Massachusetts Institute of Technology 2006. All rights reserved.



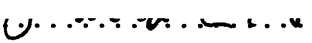
Author 

Department of Physics **ARCHIVES**
May 12, 2006



Certified by 

Professor Edward Farhi
Department of Physics
Thesis Supervisor

Accepted by 

Professor David E. Pritchard
Senior Thesis Coordinator, Department of Physics

Simulations of Error in Quantum Adiabatic Computations of Random 2-SAT Instances

by

Jay S. Gill

Submitted to the Department of Physics
on May 12, 2006, in partial fulfillment of the
requirements for the degree of
Bachelor of Science in Physics

Abstract

This thesis presents a series of simulations of quantum computations using the adiabatic algorithm. The goal is to explore the effect of error, using a perturbative approach that models 1-local errors to the Hamiltonian and estimates transition probabilities out of the ground state. The data show that a perturbation in the z -direction, parallel to the alignment of the computational basis, causes a greater error than one in the x -direction, and show with good confidence ($\chi^2_\nu = 1.7$) that the variation is as $\sin \theta$, where θ is the angle of the error term. An attempt to explore the change in error with the number of qubits was inconclusive—there was no measurable variation.

Thesis Supervisor: Professor Edward Farhi

Title: Department of Physics

Acknowledgments

I would like to thank Prof. Edward Farhi for my initial introduction to this topic, and for his supervision of this project. I would also like to thank Prof. Peter Shor, Stephen Jordan, Shay Mozes, and Edward Platt for their participation in the discussions that led to the completion of the work contained herein.

Contents

1	Introduction	11
2	Adiabatic Quantum Computation	15
2.1	Outline of Quantum Computation	15
2.1.1	Mathematical Framework	16
2.2	Satisfiability	17
2.3	The Problem Hamiltonian	18
2.4	The Adiabatic Theorem	19
2.5	The Computation	20
2.6	Example of a 2-SAT Computation	21
3	Computational Errors	25
3.1	Results of a Computation	25
3.2	Modeling the Error	28
4	Simulations	33
4.1	Generating Random Instances of 2-SAT and Random 1-Local Errors .	33
4.2	Locating the Minimum Gap	34
4.2.1	The Quadratic Method	34
4.2.2	Results	36
4.3	The Angle of the Perturbation	36
4.4	Dependence on n	37
4.5	Conclusions	39

A Simulation Code	41
A.1 Generating the Start Matrix	41
A.2 Generating the 2-SAT Instance	42
A.3 The Error Term	43
A.4 Finding the Minimum Gap	43

List of Figures

2-1	Energy levels in a 3-qubit example. The x-axis is s , the parameter that runs from 0 to 1 over the course of the computation. Note the lack of level crossings.	23
2-2	The gap in the 3-qubit example.	24
3-1	Energy levels in a 3-qubit computation corresponding to the logical expression $(x_1 \wedge x_2 \wedge x_3)$. The middle two levels are both triply degenerate. At the end of the computation, the ground state is the correct state, $ 000\rangle$ and the three states with energy 1 are $ 001\rangle$, $ 010\rangle$, and $ 100\rangle$	28
3-2	Energy levels in a 3-qubit computation of a Grover search.	29
3-3	Simulation of the effect of an error at various times in an random 8-qubit instance of 2-SAT. The dashed line shows the energy gap over the course of the computation. The solid line shows the effect of an error at each specific point in the computation. Specifically, the error is a z -spin term (as described in Eq. 3.2), applied to a particular qubit. $\epsilon = \frac{1}{2}$. The overlap between the ground states of the perturbed and unperturbed Hamiltonians is plotted.	30
4-1	5 iterations of the quadratic method to find the minimum gap. $n=8$. The circles show intermediate steps in the iterative process, and the cross shows the final result.	35
4-2	Distribution of minimum gap locations in 2000 randomly generated 10-qubit instances of 2-SAT. Median $s = 0.54$	36

4-3	The overlaps resulting from errors at various angles in 6-qubit instances of 2-SAT. θ runs from 0 (σ_z) to $\frac{\pi}{2}$ (σ_x). Each data point consists of 2000 trials. The data are fit to a sine curve $y = A + B \sin(\theta)$ with a χ^2_ν value of 1.7.	37
4-4	The overlaps resulting from errors of various magnitudes, at qubit numbers ranging from 5 to 11. No attempt is made to distinguish the 7 plotted lines, as they are not visually discernable. They are also not resolvable from each other by an amount greater than the error bars. Each data point represents 1000 trials.	38
4-5	An expansion of the $\epsilon = 1$ data set from Fig. 4-4. Note the y -axis scale.	39

Chapter 1

Introduction

Quantum computation is known to be more powerful than classical computation. Though attempts to build functional quantum computers are still at early stages, much progress has been made in the theoretical work necessary to eventually program them. If a large-scale quantum computer is ever to be realized, however, it does not suffice to only consider ideal behavior. The possibility of errors in computation must be evaluated and specifically accounted for, in order to construct models of computation which can scale to large sizes without the errors growing prohibitively large. This thesis contains a discussion of errors in a particular type of quantum computation that relies on adiabatically evolving Hamiltonians [2]. This method of computation has been shown to be computationally equivalent to the more well-known type that uses discrete unitary transformations [7].

We will examine the functionality of the adiabatic algorithm in the context of Satisfiability, the quintessential NP-complete computational problem. One of the biggest open questions about quantum computation is whether it will enable the solution of NP-complete problems in polynomial time. It has already been shown to solve a classically unsolvable problem, factorization of large numbers, in polynomial time, a result which has wide implications in many aspects of computation [1]. Though no algorithm has yet been found which demonstrates a similar speedup for NP-complete problems, it remains an important area of research, and the adiabatic algorithm is one approach being explored.

Conventional computing relies on the concept of fault-tolerance, the idea that, so long as the errors in an individual component lie within certain bounds, errors will not compound as the size of the computer increases. For example, in the design of digital logic gates, this is accomplished by specifying that components accept electrical signals across a designated input range, which is specifically larger than the corresponding output range of the previous component. In this way, even if there is noise along the wire between the two components, the signal lies within the valid input range and is interpreted as a 0 or 1. Since minor errors due to electrical fluctuation get suppressed at every step, an arbitrary number of gates can be chained together without allowing any errors to propagate down the line. Though this exact approach, which relies on the digital nature of the components involved, does not apply to quantum computation, the goal is still the same—design a system of quantum computation in which errors do not compound as the system get larger. In other words, the goal is to hold the error rate below a certain upper bound only by specifying certain noise tolerances, regardless of the size of the computation.

This problem has not been heavily explored for the adiabatic algorithm, and it is unknown whether it is possible. Here, we begin exploring some of the properties of this algorithm with regards to errors. The next chapter will introduce adiabatic quantum computation in more detail. It sets up the mathematical framework for computation in general, and then outlines the adiabaticity condition and the limitations of the algorithm. We will show how to construct the starting and ending Hamiltonians used to solve 2-SAT. Though 2-SAT is not an NP-complete problem, it is very closely related to SAT and Max 2-SAT, both of which are.

Chapter 3 explores the possibility of errors in more detail. It discusses some various ways to model errors and some ways to evaluate their effects on a computation. We will choose a criterion for successful computation, and a way to evaluate the success probability of a specified type of error. The section concludes with some motivation for the thought that noise might have the worst effect on a computation during the point of minimum gap.

The final chapter presents the result of a series of simulations. The most concrete

result is an analysis of the effect of error angle (angle of a hypothetical stray field, relative to the reference axis of the computation), where we fit the dependence of error magnitude on angle to a sine function. This is followed by an attempt to observe the variation of error magnitude on the size of the computation, but it will be evident from the data that much higher numbers of qubits must be explored for a complete conclusion on the subject to be reached.

Chapter 2

Adiabatic Quantum Computation

In 2000, Farhi et al. posited an algorithm for solving combinatorial problems on a quantum computer [2]. This algorithm relied not on a series of discrete unitary operators applied to the quantum basis, but rather on a continuous evolution of Hamiltonians. Using the adiabatic theorem of quantum mechanics, one can conclude that if the Hamiltonian is varied slowly enough, the system can be maintained in a ground state. We will now outline how this can be used to solve problems.

2.1 Outline of Quantum Computation

A classical computer relies on *bits*, components which can occupy one of two states (commonly called 0 and 1). A computer using n bits can exist in one of 2^n possible states at any one time. Computations are performed by discrete manipulations of the bits, and the length of a computation is given by the number of such manipulations required [1].

In contrast, a quantum computer relies on *qubits*, which are quantum 2-state systems, such as spin- $\frac{1}{2}$ particles. A single qubit may occupy an uncountably infinite number of states, which are various superpositions of its two eigenstates, and we usually represent it as a vector in a two-dimensional space. An ensemble of n such systems is an n -qubit quantum computer. The computer moves in an abstract space which is a direct product of the vector spaces representing each of the bits. Hence

our computer has 2^n basis states, corresponding to the valid states of the analogous classical computer, and can exist in an arbitrary complex superposition of these states.

The quantum computer can be manipulated by two types of algorithms. The first is by application of discrete physical inputs that correspond to unitary operators in the vector space that represents the computer. Standard quantum computation is based on “circuits” of these operators, which are each presumably to be implemented by physical coupling between the qubits. As with classical computation, the complexity of a problem relies on the number of such manipulations inherent in its solution. The second method of quantum computation relies on slow, continuous evolution of the Hamiltonian governing the qubits. This method will be further explicated in this chapter.

2.1.1 Mathematical Framework

We will put aside some of the physical details of the construction of such a computer, and cast it into an abstract mathematical setting. Here we outline the mathematical conventions used throughout this paper.

We think of the qubits as electrons or other arbitrary spin- $\frac{1}{2}$ systems. Our basis states for purposes of computation are the eigenstates of these particles with respect to the spin operator in the z -direction, S_z . We can call the eigenstates $|z\rangle$, where $|z = 0\rangle = |S_z = \frac{\hbar}{2}\rangle$ and $|z = 1\rangle = |S_z = -\frac{\hbar}{2}\rangle$. An n -qubit computer has 2^n eigenstates of the operator $S_z \otimes S_z \otimes \dots \otimes S_z$, where each S_z is the spin- z operator being applied to the corresponding electron. These basis elements are of the form $|z_1\rangle |z_2\rangle \dots |z_n\rangle$, where each $|z_i\rangle$ represents the spin- z state of the i^{th} electron. An n -vector \vec{z} over the set $\{0, 1\}$ can represent an eigenstate, or a choice of bits z_1, z_2, \dots, z_n .

Note that the *logical* z operator,

$$L_z = \frac{1}{2}(I - \sigma_z), \tag{2.1}$$

where σ_z is the Pauli spin matrix

$$\sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad (2.2)$$

can be used to define our convention of states $|0\rangle$ and $|1\rangle$ in the computational basis. Hence, a “measurement” of our quantum computer generally refers to an observation of the operator $L_z \otimes L_z \otimes \dots \otimes L_z$, returning a sequence of 0’s and 1’s.

The price we pay for the seemingly infinite precision allowed by the continuous milieu of quantum computation is unavoidable uncertainty. At the end of a computation, the system is in some state

$$|\psi\rangle = \sum_{\vec{z}} c_{\vec{z}} |\vec{z}\rangle. \quad (2.3)$$

Unfortunately, the postulates of quantum mechanics dictate that final measurement of the system does not give us complete information about the state $|\psi\rangle$, but rather yields exactly one of the basis states $|\vec{z}\rangle$, with a probability given by

$$P(\vec{z}) = |c_{\vec{z}}|^2. \quad (2.4)$$

At the end of any quantum computation, we desire $|c_{\vec{w}}|^2$ to be as large as possible, for \vec{w} the correct solution, expressed as a sequence of 0’s and 1’s.

2.2 Satisfiability

A classic NP-complete problem is *satisfiability*: does a given logical expression in conjunctive normal form have at least one satisfying variable assignment?

Given a set of logical variables x_i , we take a clause to be an arbitrary disjunction (“or”) of any number of elements or their negations, such as $(x_1 \vee x_2 \vee \bar{x}_3)$. A logical statement in conjunctive normal form is an arbitrary conjunction (“and”) of such clauses. We are concerned in this paper with 2-SAT, the problem of satisfiability

where each clause contains exactly two variables, for example:

$$(x_1 \vee x_2) \wedge (x_3 \vee \bar{x}_4) \wedge (\bar{x}_2 \vee x_1) \tag{2.5}$$

2.3 The Problem Hamiltonian

Imagine that we concerned not only with whether or not a variable assignment satisfies all of the clauses in a logical statement, but more specifically with the number of clauses it violates (which may be zero). In addition, every clause has an energy cost associated with violating it (which may all be 1, or some can be weighted more than others). This allows us to cast the problem into quantum mechanics, by mapping the logical variables onto qubits.

Take a 2-qubit case with one clause $(x_1 \vee x_2)$. The complete information about this problem can be expressed as a Hamiltonian:

$$H_P = |11\rangle\langle 11|, \tag{2.6}$$

or, as a matrix,

$$H_P = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.7}$$

First of all, note that this Hamiltonian is diagonal in the computational basis, because all computational basis states correspond to logical variable assignments that have defined truth values for the clause. Secondly, the energy eigenvalue of any basis state gives the number of clauses that it violates. As a consequence of this, the ground states of the system, if they have energy 0, correspond to all possible satisfying assignments.

A similar procedure can be carried for any 2-SAT problem (and for any satisfiability or cost-minimization problem). The diagonal of this Hamiltonian matrix will encode the complete information about the existence of satisfying assignments. In

fact, given the matrix elements themselves, the existence of a satisfying assignment can be determined at once, by checking for zeros on the main diagonal. Unfortunately, we cannot explicitly compute these elements in less than exponential time, for it requires counting the clauses violated by each of the 2^n basis states. The only way we can build the Hamiltonian is as a sum on 2-local (equal to the identity in all but a 2-qubit subspace) terms corresponding to the clauses. The trick, then, will be to find the ground state of this “problem Hamiltonian.”

2.4 The Adiabatic Theorem

What we shall see is that, unlike in other forms of computation, finding the ground state will not require specific ingenuity on our part—we can set up a situation in which Nature will hunt it out for us. This is accomplished by relying on the Adiabatic Theorem.

A system evolves according to the Schrödinger equation:

$$i \frac{d}{dt} |\psi(t)\rangle = H(t) |\psi(t)\rangle. \quad (2.8)$$

Let the starting state $|\psi(0)\rangle$ be the ground state of the starting Hamiltonian $H(0)$. The relevant part of the Adiabatic Theorem tells us that the evolving state $|\psi(t)\rangle$ will always be the ground state of $H(t)$, so long as the change of $H(t)$ is “slow enough,” and so long as there are no *level crossings*, degeneracies between the bottom two eigenstates of H .

We can use this fact to perform our computation. If the physical Hamiltonian is varied, slowly and without level crossings, from any Hamiltonian whatsoever (so long as the system is in the ground state to begin with) to our problem Hamiltonian as discussed above, a measurement of the ground state at the end will, with high probability, yield the correct solution.

How slow is slow enough? This depends on the size of the gap. Specifically, let us reparameterize the system in terms of s , a variable that runs from 0 to 1 as t runs

from 0 to T , the endpoint of the computation:

$$i\frac{d}{ds}|\psi(s)\rangle = \frac{dt}{ds}H(s)|\psi(s)\rangle \quad (2.9)$$

Throughout the rest of this paper, s will be used to measure the progress of the computation. The Adiabatic Theorem tells us that the overlap between $|\psi(s=1)\rangle$ and the desired final ground state $|\psi_0(1)\rangle$ of H_P can be kept greater than $1 - \epsilon^2$, where ϵ depends on the square of the gap g between the two lowest eigenvalues of H at any point in the computation, and on the local computation speed $\frac{ds}{dt}$. In other words, the computation speed must vary like g^2 , in order to maintain large overlap in the final state. Assuming a constant computation speed (which much be used unless the location of the minimum gap is known in advance), the overall time of computation goes like g_{min}^{-2} [3] [5].

2.5 The Computation

As shown in the preceding sections, if we can construct a starting Hamiltonian which can be continuously varied to the problem Hamiltonian, we can perform our computation. Construction of the starting Hamiltonian, for our problem, will be simple. We want a starting state which is uniform superposition of all the computational basis states, to avoid biases in the results of the computation. In our physical model of a spin system, this is accomplished by starting with all the magnetic alignments in a horizontal direction (perpendicular to the z -axis). This corresponds to a Hamiltonian made up of spin operators in the x -direction (to keep our computations real) on every qubit.

To remain within our mathematical framework of 0 and 1 eigenvalues, we build the starting Hamiltonian H_S as follows:

$$H_S = \sum_i \frac{1}{2}(I - \sigma_x)_i \quad (2.10)$$

(i over all qubits), where

$$\frac{1}{2}(I - \sigma_x) = \frac{1}{2} \left(I - \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \right) = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix}. \quad (2.11)$$

Also, since our starting Hamiltonian is made up of σ_x terms and our final Hamiltonian is made up of σ_z terms, we will generally avoid symmetries that can lead to level crossings.

For the course of this discussion, the rate of variation in our computation will be constant:

$$H(s) = (1 - s)H_S + sH_P, \quad (2.12)$$

where

$$s = \frac{t}{T}, \quad (2.13)$$

T being the total time required. We do not know, *a priori*, where on the scale of $s = 0$ to $s = 1$ the minimum gap will reside in a particular computation. Without that information, there is nothing to be gained from a more complex variation of the speed. For an interesting example of where knowledge of the gap over time can improve computation speed, see the adiabatic solution to the Grover search problem [4] [6].

2.6 Example of a 2-SAT Computation

Here we demonstrate the computation of a typical 2-SAT problem. We will use a randomly generated 3-qubit instance of 2-SAT:

$$(\bar{x}_2 \vee x_3) \wedge (\bar{x}_2 \vee x_3) \wedge (\bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2) \wedge (x_1 \vee \bar{x}_3) \wedge (x_1 \vee \bar{x}_3) \wedge (x_1 \vee \bar{x}_3) \wedge (x_1 \vee x_2) \wedge (x_1 \vee x_2) \quad (2.14)$$

The duplicate terms effectively serve to weight some clauses more strongly than others. This has no bearing on the presence of a solution, but does alter the evolution of the system.

Using a convention where 0 corresponds to a “true” value of a particular variable, and 1 to “false,” we can construct the problem Hamiltonian

$$H_P = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}. \quad (2.15)$$

The starting matrix consists of the logical x -operator (Eq. 2.11) acting on every qubit. To act with a one-qubit operator on a particular qubit, we form a direct product with an identity in every other place:

$$L_{x1} = \frac{1}{2}(I - \sigma_x) \otimes I \otimes I. \quad (2.16)$$

$$H_S = \sum_i L_{xi} = \frac{1}{2} \begin{bmatrix} 3 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\ -1 & 3 & 0 & -1 & 0 & -1 & 0 & 0 \\ -1 & 0 & 3 & -1 & 0 & 0 & -1 & 0 \\ 0 & -1 & -1 & 3 & 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 & 3 & -1 & -1 & 0 \\ 0 & -1 & 0 & 0 & -1 & 3 & 0 & -1 \\ 0 & 0 & -1 & 0 & -1 & 0 & 3 & -1 \\ 0 & 0 & 0 & -1 & 0 & -1 & -1 & 3 \end{bmatrix}. \quad (2.17)$$

The computation proceeds, parameterized by s , according to Eq. 2.12. We see in Fig. 2-1 how the energy levels evolve in time. At $s = 0$, we have the uniformly superposed eigenstates of $H(0) = H_S$. At $s = 1$, the eigenstates of the Hamiltonian are computational basis states, and each energy level corresponds to an entry in the problem Hamiltonian, and hence to the “cost” (degree of unsatisfaction) of one of the

variable assignments in the initial problem. Since, at $s = 1$, there is a unique ground state, our initial problem had a unique solution, and since there are no level crossings, an adiabatic computation would find this solution.

Of interest to us is the gap between the first two energy levels, as the computation proceeds. As discussed above, the size of the gap determines the chance of a “level jump” which causes an error in the final result. This gap is displayed in Fig. 2-2. It, in general, is smallest near the middle of the computation, but this is not always the case. Later on, the location of the minimum gap in a variety of cases will be of interest to us, as this is the most likely location for an error to occur.

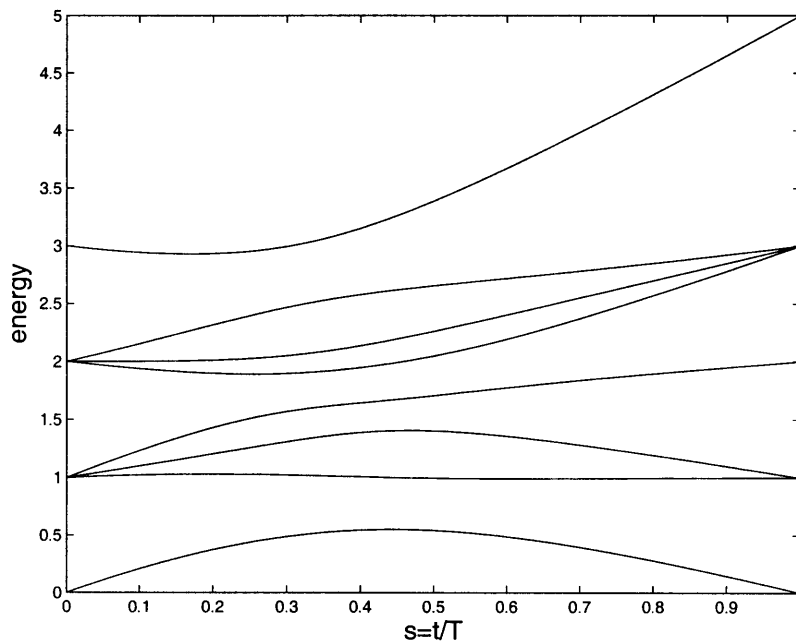


Figure 2-1: Energy levels in a 3-qubit example. The x-axis is s , the parameter that runs from 0 to 1 over the course of the computation. Note the lack of level crossings.

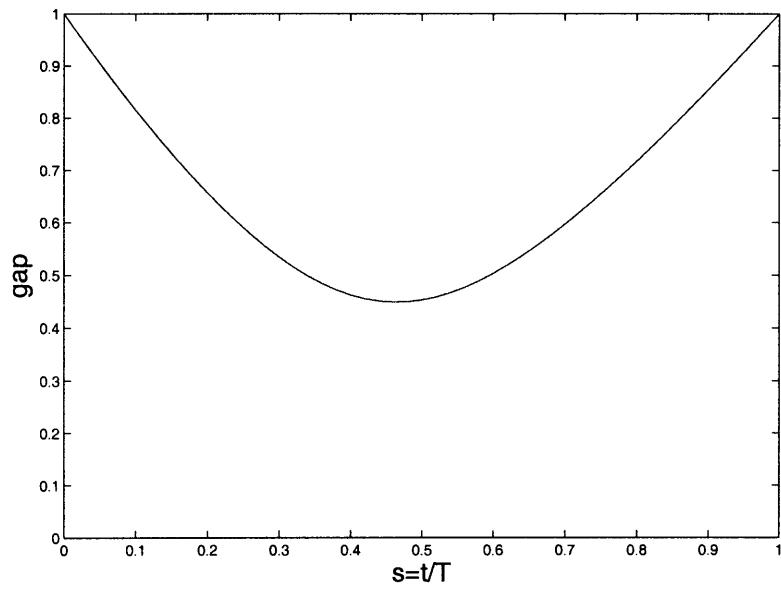


Figure 2-2: The gap in the 3-qubit example.

Chapter 3

Computational Errors

Adiabatic quantum computation has been proven to be computationally equivalent to discrete quantum computation. Specifically, there exists a method to convert any discrete quantum computation (implemented as a sequence of unitary gates) into an adiabatic form [7]. What is relatively unexplored, however, is the fault-tolerance properties of adiabatic computation. If this method of computation is ever to be realized, some understanding will have to be gained about its behavior in non-ideal conditions.

3.1 Results of a Computation

Essentially, an adiabatic computation can fail in one of two ways. Either the final Hamiltonian produced by the computer is not precisely identical to the desired problem Hamiltonian, and so its ground state is not the solution state, or something causes the quantum system to make a level transition at an intermediate point in the computation.

In general, a minor variation in the “path” of the computation should have no effect, so long as it does not shrink the gap by a factor significant enough to defeat the adiabaticity condition. This is an important point—the continuous nature of the computation might lead one to believe that it would be susceptible to infinitesimal errors, but the adiabatic nature of it means that this is not necessarily the case. In

many cases, an error of nontrivial size may perturb the path in such a way as to have no effect on the final result. Unfortunately, any detailed exploration of this assertion would be extremely difficult, as it would require a way to predict the effect of a random perturbation on the energy gap.

An important question to ask is: how much error in the final result of the computation is acceptable? In other words, we want to know what our criterion is for declaring a computation successful. Since a computation can be repeated a large number of times, we have to consider what methods we might have for extracting the correct solution out of a list of nonidentical outputs. For example, consider the two computations presented in Figs. 3-1 and 3-2. The first is a case of satisfiability where every clause contains one variable, such that there is a unique correct solution, specified one bit at a time. Here, the three lowest-lying incorrect eigenstates of the problem Hamiltonian all differ from the correct solution by exactly one bit. Even in the case of a severe error in which the probabilities of all four of these outcomes (the correct solution $|000\rangle$ and the three lying nearest it $|001\rangle$, $|010\rangle$, and $|100\rangle$) were equal, we would be able to discern the correct answer. For, in each bit, we would find a $0 \frac{3}{4}$ of the time, and a $1 \frac{1}{4}$ of the time. As such, a sort of “vote” at each bit location would clearly yield an answer of 000, even though the state $|000\rangle$ was not a favored outcome.

The second example, however, does not afford us this advantage. It is a 3-bit Grover search. This is an unstructured search, in which there is one correct solution, 7 incorrect solutions, and no further information. It is not technically an instance of satisfiability, but is a related problem that can be solved adiabatically [4]. The

problem Hamiltonian is

$$H_P = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.1)$$

As seen in this matrix and in Fig. 3-2, all incorrect states are equally likely. It is possible that some type of error would cause the computation to return any of the incorrect states with equal probability. In this case, a “voting” approach like the one mentioned above would quickly be stymied. The difference is that in this case, incorrect states which are erroneous in most or all of the bits are still possible outcomes of simple errors, so we cannot rely on the assumption that any particular bit will be correct in the majority of trials.

In order to achieve a correct solution in the Grover problem, the final state of the computation must have a large overlap with the correct solution state. No weaker criterion will suffice. In the case of a general satisfiability problem, there is no easy way to improve upon this requirement. There is always the possibility, for example, that in a particular instance the state with every bit incorrect will violate only a small number of clauses and be a likely outcome of an erroneous computation. In some cases, knowledge of the problem’s clause structure can be of some use. For example, if the qubits can be partitioned into two subsets, and all the clauses only involve qubits from one of the two sets, then a form of analysis in which the outcomes of the two halves of the problem are evaluated independently (like the individual qubits in the first example above) will yield a correct result more frequently than one which simply searches for the most common complete output.

Since the instances examined in this paper are generated randomly, we will not

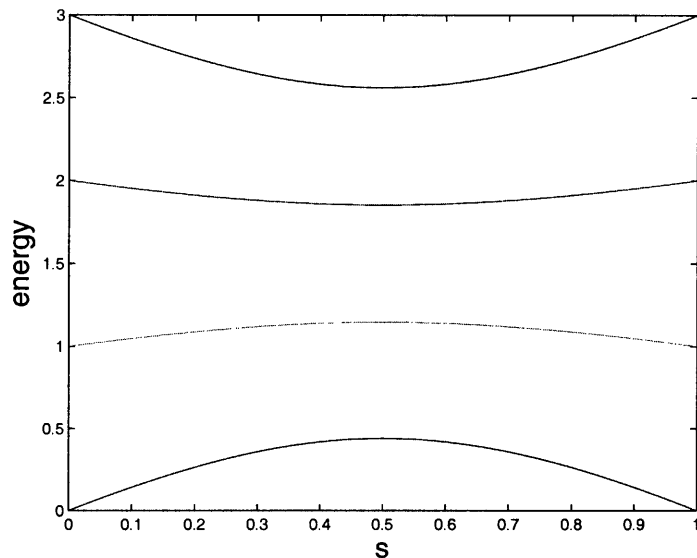


Figure 3-1: Energy levels in a 3-qubit computation corresponding to the logical expression $(x_1 \wedge x_2 \wedge x_3)$. The middle two levels are both triply degenerate. At the end of the computation, the ground state is the correct state, $|000\rangle$ and the three states with energy 1 are $|001\rangle$, $|010\rangle$, and $|100\rangle$.

have specific knowledge of the clause structure of any particular instance. As such, the metric we will use to evaluate the effect of an error will be overlap between the erroneous and the correct state. This should correlate to the fraction of computation trials which return the correct result.

3.2 Modeling the Error

There are a variety of ways to model the sorts of error that can arise in a computation. For example, one could try to model a thermal error by considering the whole computer as coupled to an external environment. These sorts of analyses can easily get rather complicated. And since, in the general case, we know little about the gap and other factors might be affected by arbitrary perturbations, more progress is to be made by considering the effects of specific types of errors than can be modeled more easily.

One simple error to imagine, as mentioned above, is that the final Hamiltonian

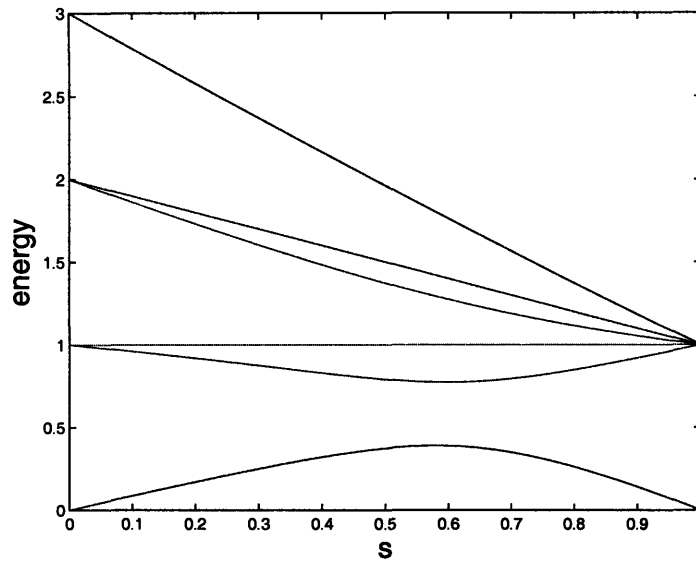


Figure 3-2: Energy levels in a 3-qubit computation of a Grover search.

is not precisely the desired one. In actuality, this will always be the case, due to uncertainties in the physical setup of the computer. If the rest of the computation goes as planned, however, the system should still reach the ground state of the actual Hamiltonian. The probability of success in this case would be equal to the overlap between the ground states of the correct Hamiltonian and the actual one. This provides a good first approach to analysis of errors, as it encapsulates the notion that an adiabatically evolving system will wind up in the correct ground state, even in the event of small fluctuations along the way.

However, there is still the possibility that, even in an adiabatically evolving system, a perturbation of some kind will “bump” the system up to an excited state. Presumably, the time when the computation is most vulnerable to this sort of error is at the point where the gap is smallest. These errors are the focus of this paper.

A simple way to imagine the error is as a stray magnetic field affecting one of the spin systems. This will add a term to the Hamiltonian which consists of spin operators on a single qubit. Specifically, imagine that a qubit is exposed to a field of strength ϵ , and that this perturbation occurs over a time scale which is very short compared to time scale of the whole computation. We can then consider the unperturbed

Hamiltonian to be constant, equal to $(1 - s)H_S + sH_P$. The sudden, non-adiabatic addition of the error changes the Hamiltonian to be equal to:

$$H' = (1 - s)H_S + sH_P + \epsilon\sigma_{zi}, \quad (3.2)$$

for some qubit i (for a field in the z -direction). We are concerned with transition probability from the ground state up to any other state. In the case of an instantaneous perturbation to an otherwise constant Hamiltonian, the state of the system will remain the same (the ground state of the unperturbed Hamiltonian) at the moment of the perturbation. It will now be a linear combination of states of the perturbed Hamiltonian. If the evolution of the system at all other times is adiabatic, then the probability that the system will remain in the ground state is equal to the overlap between the perturbed and unperturbed Hamiltonians at the moment of the error.

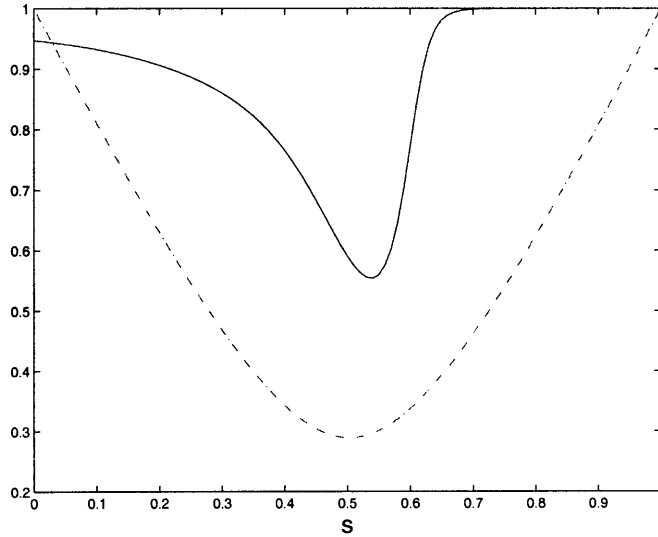


Figure 3-3: Simulation of the effect of an error at various times in an random 8-qubit instance of 2-SAT. The dashed line shows the energy gap over the course of the computation. The solid line shows the effect of an error at each specific point in the computation. Specifically, the error is a z -spin term (as described in Eq. 3.2), applied to a particular qubit. $\epsilon = \frac{1}{2}$. The overlap between the ground states of the perturbed and unperturbed Hamiltonians is plotted.

This is the method we will explore in this paper. For an examination of the

behavior of this model in a typical, randomly chosen instance, see Fig. 3-3. This graph shows the effect of a constant error, of the sort described above, when applied at any point in this computation. For reference, the gap is also shown. Though there is not a specific relation between the two, we see that the error is worst near the middle of the computation, where the gap tends to be smallest. The specific location of both of these minima varies between instances. As we approach the end of the computation, the Hamiltonian $H(s)$ is very similar to the problem Hamiltonian H_P , and small perturbations have less of an effect.

For consistency, when we compare overlaps and success probabilities in different conditions, we will apply the error at a constant value of s . This should give a more uniform basis for comparison than trying to locate the minimum gap (or the point of minimum success probability) for each particular trial. The value of s used be determined by finding the average point of minimum gap over a large number of trials.

Chapter 4

Simulations

Here we present a series of simulations which use the methods described in the preceding section. There are three sets of experiments. The first is to determine the average location of the minimum gap, the second is to examine the effect of the angle of the perturbation on the error, and the third is to compare error magnitudes at different qubit numbers.

4.1 Generating Random Instances of 2-SAT and Random 1-Local Errors

For each trial, we must generate H_S , H_P , and an error term.

H_S is constant among all trials will the same number of qubits. It is easily generated recursively. $H_{S1} = L_x$ (see Eq. 2.11). For $n > 1$:

$$H_{S_n} = H_{S_{(n-1)}} \otimes I + I^{\otimes(n-1)} \otimes L_x, \quad (4.1)$$

where the \otimes are tensor products.

H_P is diagonal, and is produced by adding clauses to an empty matrix until the number of zeros on the main diagonal is equal to exactly 1 (this is how we generate uniquely satisfiable instances). If the number of zeros on the main diagonal ever reaches 0, the entire process is restarted from an empty matrix.

A clause is generated by choosing two qubits at random, and applying to them a matrix of the form

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4.2)$$

with the 1 at any location on the diagonal. This is analagous to a 2-bit OR clause.

The error term is generated by choosing a qubit a random and applying to it

$$E = \epsilon(\sin(\theta)L_x + \cos(\theta)L_z) = \epsilon \begin{bmatrix} \frac{1}{2} \sin(\theta) & \frac{1}{2} \sin(\theta) \\ \frac{1}{2} \sin(\theta) & \frac{1}{2} \sin(\theta) + \cos(\theta) \end{bmatrix}. \quad (4.3)$$

All matrices used in these simulations are stored in a “sparse” format, a list of nonzero values indexed by location, rather than as a conventional table. This improves computational efficiency, due to the large numbers of zero entries in these matrices.

4.2 Locating the Minimum Gap

As discussed in the preceding section, we want find one particular point in the computation (i.e. one value of s) at which we will perform the ensuing computations of the effects of error. The precise location of this point is not of overwhelming importance, but the data showed previously indicate that the effects of the error will be most noticeable near the point of minimum gap.

4.2.1 The Quadratic Method

Computing the gap at an arbitrary point in the computation requires finding the two lowest eigenvalues of a 2^n -by- 2^n matrix. In order to run a large number of trials efficiently, we want to minimize the number of times we must take these eigenvalues. To do so, we use the following iterative algorithm, commonly called the “quadratic method” for locating minima of a smooth function with a small number of evaluations:

1) Evaluate the gap at three arbitrary points s_1 , s_2 , and s_3 to obtain values y_1 , y_2 ,

and y_3 . Here we use .25, .5, and .75 as starting values of s .

- 2) The three points can be placed exactly on some parabola $y = as^2 + bs + c$.
- 3) Let s_4 be the location of the minimum of this parabola: $s_4 = -\frac{b}{2a}$.
- 4) Evaluate the gap at s_4
- 5) Out of s_1, s_2 , and s_3 , throw away the one which produced the largest gap.
- 6) Repeat from (2), using the two remaining starting values, and s_4 , as the three starting values.

After some algebra, we find that steps 2 and 3, the determination of s_4 from the first three points, can be accomplished by the following equation:

$$s_4 = \frac{y_1(s_2^2 - s_3^2) + y_2(s_3^2 - s_1^2) + y_3(s_1^2 - s_2^2)}{2(y_1(s_2 - s_3) + y_2(s_3 - s_1) + y_3(s_1 - s_2))} \quad (4.4)$$

An example of the outcome of this method is shown in Fig. 4.4. The circles, showing intermediate steps, are quite near the minimum, demonstrating the fast convergence. In fact, the last three points are almost indistinguishable from the final result.

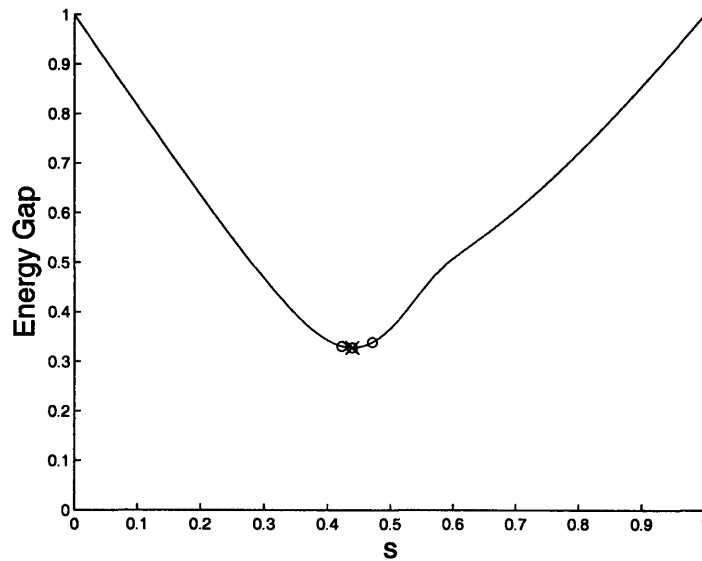


Figure 4-1: 5 iterations of the quadratic method to find the minimum gap. $n=8$. The circles show intermediate steps in the iterative process, and the cross shows the final result.

4.2.2 Results

2000 trials were run on random 10-qubit instances. In each case, 5 iterations of the above algorithm were used to locate the minimum gap. The results are displaying in Fig. 4-2. The median location for the minimum gap was found at $s = 0.54$. This is the value of s used for future trials.

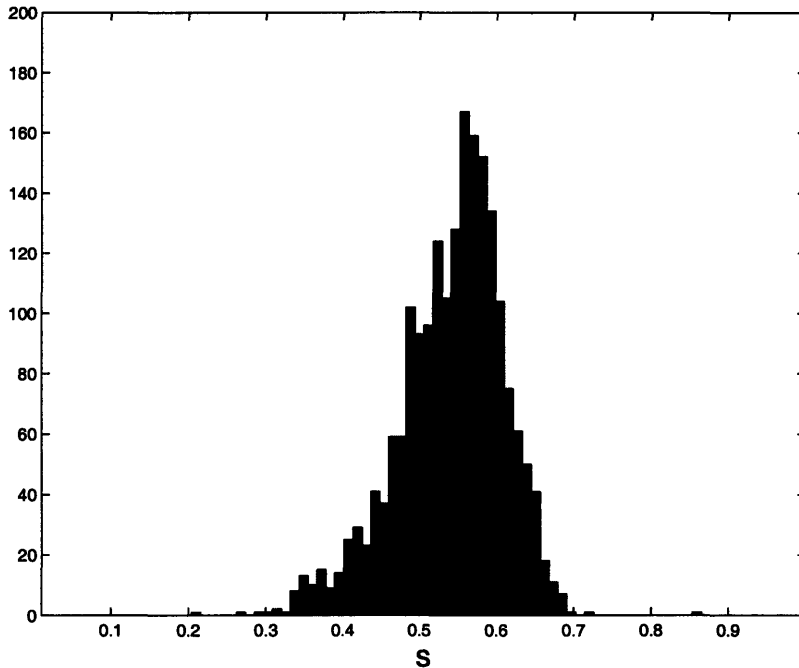


Figure 4-2: Distribution of minimum gap locations in 2000 randomly generated 10-qubit instances of 2-SAT. Median $s = 0.54$.

4.3 The Angle of the Perturbation

In our error model, based on a stray spin term in the Hamiltonian, the error can come in different directions. A σ_z term corresponds to a perturbing field parallel to the computational axis, or a σ_x term, perpendicular to it. Here we examine the effect of the angle on the final overlap (our criterion for success).

2000 trials were run at each of 41 different angles, evenly spaced between $\theta = 0$ (σ_z) and $\theta = \frac{\pi}{2}$ (σ_x). A new random 6-qubit instance of 2-SAT and a new error term,

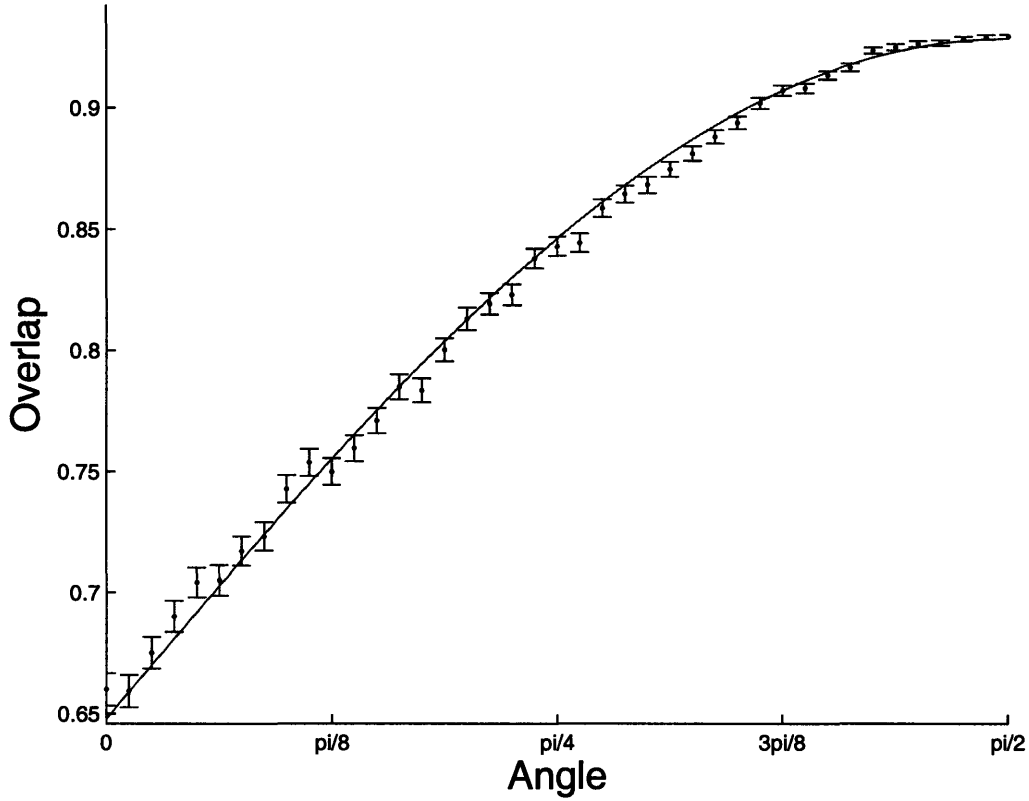


Figure 4-3: The overlaps resulting from errors at various angles in 6-qubit instances of 2-SAT. θ runs from 0 (σ_z) to $\frac{\pi}{2}$ (σ_x). Each data point consists of 2000 trials. The data are fit to a sine curve $y = A + B \sin(\theta)$ with a χ^2_ν value of 1.7.

with $\epsilon = 1$, were generated for each trial. The error term was used to perturb the Hamiltonian at $s = .54$, and the overlap taken between the original and erroneous Hamiltonians.

The results are shown in Fig. 4-3. The data fit extremely well to a sine curve with an arbitrary vertical offset. In general, we can conclude that z -errors are significantly worse than x -errors.

4.4 Dependence on n

An overarching point of interest is how the effect of an error changes with the size of the computation. Unfortunately, simulating large computations is extremely difficult,

as the number of entries in the Hamiltonian matrix quadruples with each qubit added. Here we present an attempt to detect any variation in error magnitude at low numbers of qubits.

A series of simulations was run at qubit numbers from 5 to 11. 20 different error magnitudes were used, ranging from $\epsilon = 0.1$ to $\epsilon = 2$. $\epsilon = 2$ is a rather large “perturbation,” even larger than the magnitude of a single 2-SAT clause, but extending the measurements out this far may have amplified some variations between qubit numbers. 1000 trials were run at each value of n and ϵ .

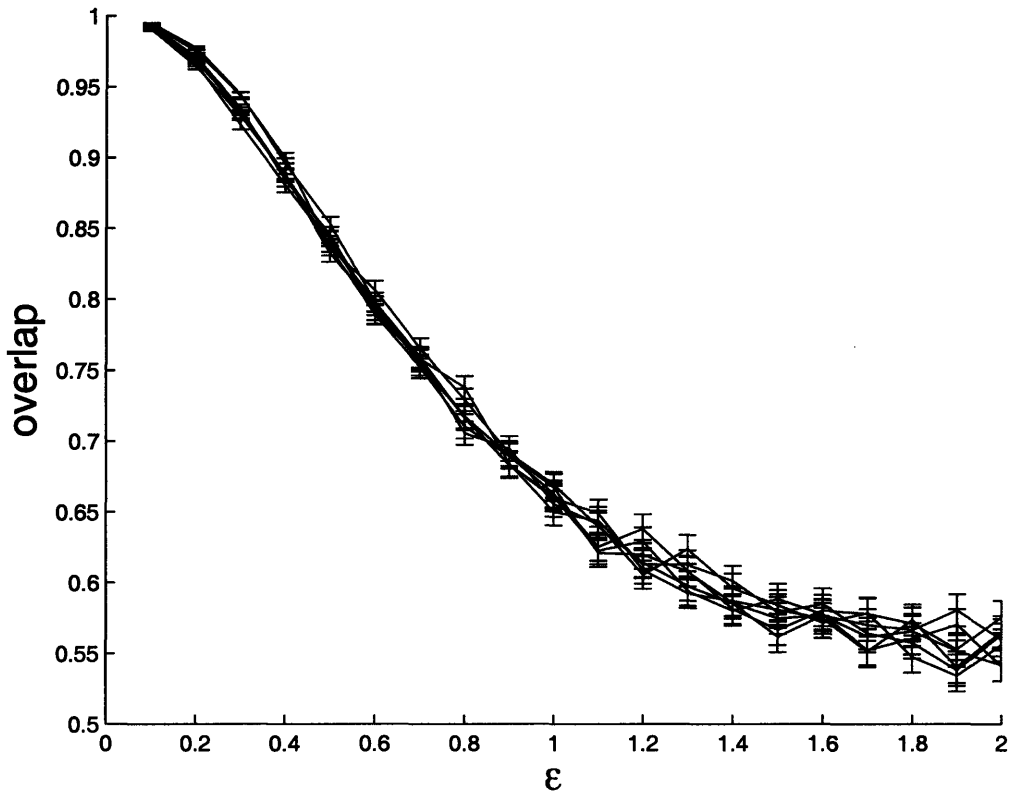


Figure 4-4: The overlaps resulting from errors of various magnitudes, at qubit numbers ranging from 5 to 11. No attempt is made to distinguish the 7 plotted lines, as they are not visually discernable. They are also not resolvable from each other by an amount greater than the error bars. Each data point represents 1000 trials.

As visible in Fig. 4-4, there is no clear variation among the 7 data sets—they are almost exactly superimposed. To make this even more apparent, the 7 points from

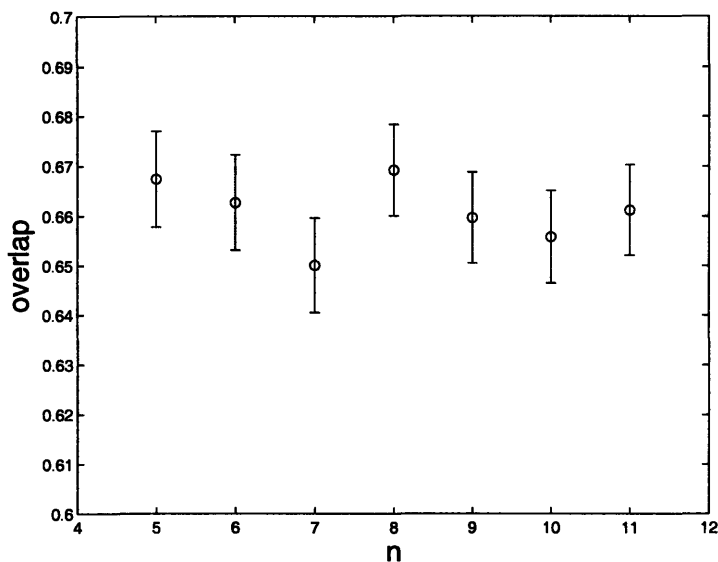


Figure 4-5: An expansion of the $\epsilon = 1$ data set from Fig. 4-4. Note the y -axis scale.

one particular ϵ value are juxtaposed clearly in Fig. 4-5. There is no visible trend in the effect of the error. It is constant, to the extent that we can observe.

4.5 Conclusions

The first major result of these experiments, the strong variation of error magnitude with the angle of the perturbation, fits with what one would expect for this sort of computation. Since each qubit settles into either a spin-up or spin-down orientation over the course of the computation, a magnetic disturbance in one of those two directions has a 50% chance of forcing it in the wrong direction, causing a significant error. An perturbation in the perpendicular (x) direction, on the other hand, puts exactly the same type influence on a qubit as does the starting Hamiltonian H_S which is already acting on it with some strength, so it seems likely that this error would not be as destructive.

The second portion of the results leaves many questions unanswered. Is it possible that the error caused by a constant-sized perturbation is, in fact, constant as the number of qubits increases? This not inconsistent with our data, but is highly unlikely.

Among other reasons, since the minimum gap size gets smaller as n increases, a smaller perturbation should be necessary in order to effect a transition. The experiments here raise the question of how far one must go in order to see a significant variation. Here, we more than doubled the number of qubits without seeing a measurable change.

The first logical extension of this experiment would to extend it to as high a dimension as possible, something which is only achieved through a large increase in computational power. A conclusive result might be a long way away, however. Even if a larger number of qubits revealed an increase in the effect of errors, it might require a yet far larger number to identify the behavior and growth of the error. Without that information, good estimates as to the large-scale success of the computation model would be difficult. Other follow-ups might involve simulations of various ways of countering the error, but even these run into computational constraints. Attempting to actively introduce robustness against error will generally involve adding qubits, which further shrinks the size limits within which we can collect data. Nevertheless, even at these small qubit numbers, such analysis may begin giving insight into future possibilities for error corrections.

We have only scratched the surface of the deep topics which will need to be addressed if large-scale computation is ever to be realized. The most significant limiting factor is the ability to simulate the behavior of the large quantum systems using the computational tools of today. Progress is being made, however, and these models of computation still hold much promise for successful applications in the future.

Appendix A

Simulation Code

A collection of the some of the code described earlier in the thesis.

A.1 Generating the Start Matrix

This recursive function quickly builds the starting matrix in sparse format.

```
function[H_S] = BuildStartMatrix2(k)
if k==1
    H_S = sparse([1/2,-1/2;-1/2,1/2]);           %base case
else
    H_S = kron(BuildStartMatrix2(k-1),speye(2)); %recursive call
    M = [1];                                     % builds the new term to be added
    for i = 1:k
        if i==k
            M=kron(M,sparse([1/2,-1/2;-1/2,1/2]));
        else
            M=kron(M,speye(2)); %speye is the identity matrix
        end
    end
    end
    H_S=H_S+M;
end
```

A.2 Generating the 2-SAT Instance

These functions generate H_P for a random 2-SAT instance on a given number of bits k . It returns a vector of length 2^k which contains the diagonal of the problem Hamiltonian.

```
function[V] = UniquelySatisfiableInstance(k) %V is the diagonal of H_P
V=zeros(1, 2^k);
while NumberOfSatisfyingAssignments(V)~=1 %Stop if unique sat. assignment
    if NumberOfSatisfyingAssignments(V)==0
        V=zeros(1, 2^k);           %Restart
    else
        V=V+RandomClause(k);      %Add another clause
    end
end

function[n] = NumberOfSatisfyingAssignments(vec) %number of zeros in a vector
n=0;
for i = 1:length(vec)
    if vec(i)==0
        n=n+1;
    end
end

function[V] = RandomClause(k) %generates a random, 2-bit OR clause
i=ceil(rand*k);           %choose one random qubit
j=i;
while j==i               %choose a second, distinct qubit
    j=ceil(rand*k);
end
V=[1];
```

```

for q = 1:k
    if (q==i | q==j)          %if one of these two qubits
        if rand<.5
            V=kron(V,[0,1]); %multiply in part of a clause
        else
            V=kron(V,[1,0]);
        end
    else
        V=kron(V,[1,1]);      %multiply in the identity
    end
end
end

```

A.3 The Error Term

This function generates the error term, as described in Eq. 4.3

```

function[M] = ErrorTermTheta(k,mag,theta)
M = sparse([mag]);
errmat=(sin(theta)*[1/2,-1/2;-1/2,1/2]+cos(theta)*[1,0;0,0]);
j=ceil(rand()*k);          %choose a bit at random
for i = 1:k
    if i==j
        M=kron(M,sparse(errmat));      %apply the error to this bit
    else
        M=kron(M,speye(2));
    end
end
end

```

A.4 Finding the Minimum Gap

This is an implementation of the method described in section 4.2.1.

```

function [smin,gapmin,recordx,recordy] = MinGap2(iter,startH,finalH)
%returns the final point (smin, gapmin) as well as record of intermediate points
x=zeros(4,2);      %main array.  Col 1 is x, col 2 is y
x(1,1)=.25;
x(2,1)=.5;        %initial x-values
x(3,1)=.75;
recordx=zeros(1,iter);
recordy=zeros(1,iter);
for i = 1:3
    x(i,2)=Gap2(x(i,1),startH,finalH); %evaluate the first three points
end
for i = 1:iter
    x(1:3,1:2)=sortrows(x(1:3,1:2),-2); %sort the three points by y-value
    x(4,1)=(x(1,2)*((x(2,1))^2-(x(3,1))^2)+x(2,2)*((x(3,1))^2-(x(1,1))^2)
        +x(3,2)*((x(1,1))^2-(x(2,1))^2))/(2*(x(1,2)*(x(2,1)-x(3,1))
        +x(2,2)*(x(3,1)-x(1,1))+x(3,2)*(x(1,1)-x(2,1))));
        %locate the minimum
    x(4,1)=min(max(x(4,1),0),1);      %truncate to within [0,1]
    x(4,2)=Gap2(x(4,1),startH,finalH); %evaluate the new point
    recordx(i)=x(4,1);
    recordy(i)=x(4,2);
    for j = 1:3          %store new values in rows 1-3 for next iteration
        x(j,1)=x(j+1,1);
        x(j,2)=x(j+1,2);
    end
end
smin = x(4,1);
gapmin = x(4,2);

```

Bibliography

- [1] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, UK, 2000.
- [2] E. Farhi, J. Goldstone, et al, “Quantum Computation by Adiabatic Evolution,” quant-ph/0001106
- [3] J. Roland and N. Cerf, “Quantum Search by Local Adiabatic Evolution,” quant-ph/0107015
- [4] E. Farhi and S. Gutmann, “An Analog Analogue of a Digital Quantum Computation,” quant-ph/9612026
- [5] W. van Dam, M. Mosca, U. Vazirani, “How Powerful is Adiabatic Quantum Computation?” quant-ph/0206003
- [6] L.K. Grover, “A Fast Quantum Mechanical Algorithm for Database Search,” quant-ph/9605043
- [7] D. Aharonov, W. van Dam, et al, “Adiabatic Quantum Computation is Equivalent to Standard Quantum Computation,” quant-ph/0405098