ADIABATIC QUANTUM COMUTATION

INTRODUCTION

Hi, today I'm going to be talking about a model of quantum computation, called adiabatic quantum computation, which was initially developed as an optimization algorithm.

After the introduction, the algorithm will be presented and a review of the adiabatic theorem that enables it will follow. Next, we will take a look at an example that can be solved with this algorithm. After that, an experimental realization of the algorithm will be briefly described. Even though this algorithm was developed as an optimization algorithm, it is becoming increasingly important as a quantum simulator, which was Feynman's original idea for the use of quantum computers in the 80s. Specifically, we will take a look at how we can simulate the Ising spin glasses.

HISTORY

As we said, AQC was initially develop as an optimization algorithm, which in physics means finding the ground state in the energy landscape. For those of you that have taken the Model analysis course, you know how simulated annealing achieves that: it uses thermal fluctuations to escape local minima and increases the chance of finding the global ground state. The predecessor of AQC worked similarly, except in that they used quantum fluctuations to escape minima and they are appropriately called quantum annealing. It was executed on classical computers, but was shown that it is superior to simulated annealing in most cases. Experimentalists, however, started to build dedicated devices that exploited the adiabatic evolution to find the solution to the optimization problems. SHOW THE THREE EXAMPLES ON THE GRAPHS.

THE ALGORITHM

The algorithm works as follows: Initialize a system in the ground state of the Hamiltonian, vary the Hamiltonian for a period T, the final Hamiltonian should encode the optimization problem. Adiabatic theorem tells us how slowly we should vary the Hamiltonian in order to remain in the instantaneous ground state with large probability. Final state of the system is therefore the ground state / solution of an optimization problem encoded in the final Hamiltonian. SHOW ON THE GRAPH.

COMPARISON WITH GATE MODEL

The question that arises is: if this is a type of quantum computation, how does it compare to the gate model or the circuit model as it is sometimes called? To be able to do that, let's quickly review the gate model of QC. It uses a series of quantum gates that can drive two-state quantum systems - qubits into any desired state. It therefore uses the full Hilbert space to perform the calculation. AQC on the other hand only uses the instantaneous ground state of a time-dependent Hamiltonian, which is a very small subspace. However, the two models are theoretically equivalent, which means that they can simulate each other in polynomial time. However, the only commercial AQCs available run a simple Hamiltonian and that is not equivalent to the gate model. Also the price you have to pay in order to simulate the gate model on AQC is really large, so that the equivalence is mostly just theoretical.

ADIABATIC THEOREM

Let's now review the adiabatic theorem that enables the AQC, which tells us how slowly the Hamiltonian should be varied in order for the system to remain in the instantaneous ground state. Since adiabatic quantum computing was initially developed in order to search for the ground state of optimization problems, we will be interested in possible transitions from the ground to the first excited state. DEFINE STUFF. The assumption of non-crossing energy states seem restrictive, but it actually doesn't happen if the Hamiltonian only depends on a single parameter, like in the case of

AQC. This is known as avoided crossing. That is unless the Hamiltonian possesses some symmetry like we see in the figure. But symmetries lead to conserved quantities and we cannot transition from one sector to the other, therefore the only energy gaps that we care about are the one from the same sector. From now on we will simply assume that the levels don’t cross.

PROBLEM HAMILTONIAN

Ising Hamiltonian is used in practice by D-Wave, the only company that offers commercial AQC, so we will focus on that. This is where the roots of AQC as an optimization algorithm can be seen. A lot of combinatorial optimization problems have already been translated into this form [8], which is also the main reason why this Hamiltonian is used in practice. Another is that searching for ground state of an Ising Hamiltonian (3) with randomly chosen interaction strengths $\{J_{ij}\}$ (spin glass problem) is a known NP-hard problem for classical computers and we can map NP problems between each other in polynomial time. Therefore we can, by efficiently finding the ground state of the Ising Hamiltonian (3), solve any other NP problem efficiently. And AQC was initially developed for solving such optimization problem, so it is only logical to use the Ising Hamiltonian as the problem Hamiltonian.

If the Ising Hamiltonian encodes our problem, there is a nice way of visualizing it: we can view it as a graph with some vertices/nodes and edges between them (Figure 2). Each vertex represent a binary decision variable that can be either $+1$ or -1 and connections/edges between them represent the interactions J_{ij} between them. We can also say a lot about the scaling of the energy: the first part of Hamiltonian (3) is a sum of $n(n-1)/2$ terms and the second part includes n terms, if n is the problem size. This means that the full problem energy can be at most quadratic in n.

GRAPH PARTITIONING PROBLEM

FULL HAMILTONIAN

Let’s now take a look at the initial Hamiltonian of the algorithm. The first requirement that we pose on the initial Hamiltonian is that its ground state must be easily
achievable. The whole point of adiabatic computing is to evolve the system from ground state of one Hamiltonian to the ground state of another, and if we are unable to begin in the ground state, the procedure would be meaningless.

The choice of the problem Hamiltonian somewhat restricts our choice of the initial Hamiltonian. We know from quantum mechanics that symmetries of H are reflected in conserved quantities. Because our problem Hamiltonian only consists of terms that include σ_z operators, we cannot include only σ_z -s in the initial Hamiltonian, otherwise the whole Hamiltonian would commute with σ_z^i and nothing would happen during the evolution of the Hamiltonian. The final state would simply be the same as the initial state, which is not what we want. With that in mind, the simplest possible Hamiltonian with sufficient complexity is the transverse Hamiltonian. EXPLAIN THE FULL HAMILTONIAN INTUITIVELY.

COMPLEXITY

When determining the efficiency of an algorithm, one usually look at how time needed to solve a problem scale with the problem size. This is reffered to as time complexity of an algorithm. We will use two-state quantum systems - qubits to solve the problems and will refer to the number of qubits used in problem solving as the system size. The relation between the problem size and the system size is problem specific, but for most NP problems we can equate the two. That means that if we want to solve a combinatorial problem with n variables (like traveling salesman with n cities), we will usually need n (e.g. graph partitioning example) or n^2 (e.g. traveling salesman problem) qubits on the AQC to solve it [8]. We can then use the adiabatic condition (1) as a way to quantify the algorithm’s complexity. We know that the numerator of (1) scales at most quadratically with the system size, but since we generally care about whether or not our the time complexity of an algorithm scales exponentially with the system size, we can forget this term. In order to be able to tell how well the algorithm performs, we have to able to determine how the energy gap between the ground state and the first excited state scales. We are able to do that analytically only in some special cases, like the Ising spin glass problem with fixed interaction magnitudes ($J_{ij} = \pm1$), where it was shown that the energy gap is inversely proportional to the problem size: $g_{min} \sim 1/n$ [4]. Generally, we can only determine it for small enough problems numerically and then extrapolate the results [9]. There have been some proven polynomial speedups when comparing AQC to classical computing (e.g. quadratic speedup in Grover's algorithm: we can find an element in an unordered list in $\sqrt{n}$ time, compared to n for a classical computer [10]), but there has not been any evidence of an exponential speedup. Some improved methods, like diabatic and reversed adiabatic quantum computing were invented and they look more promising in this regard but are beyond the scope of this seminar.

EXPERIMENTAL REALIZATON

which is realized by coupling radio-frequency SQUIDs (superconducting quantum interference devices), portrayed schematically in Figure 3 [12]. There are three different components on the image: two qubits, three tunable couplers (two between qubits and global bias current I_g that sets the energy scale $B(s)$ of the problem Hamiltonian, and one between the two qubits) and two non-tunable ccjj (stands for compound-compund Josephson junction) couplers between qubits and global current I_{ccjj} that control the energy scale of the initial Hamiltonian $A(s)$. The tunable couplers can be tuned by applying bias magnetic flux through the superconducting loops. We are not going to go into details here, but because of the different structures of the couplers, we are able to implement the full Hamiltonian (6). The functions $A(s)$ and $B(s)$ represent the energy scales of the initial and problem Hamiltonians at any given parameter s. So far we haven't really specified where the sum over interacting pairs in (6) runs. Qubits on a machine form a graph, which is getting bigger and bigger year by year. Currently, every one of the 5700 qubits is connected to 15 other qubits, which makes for approximately 40000 tunable couplers between qubits. It is quite extraordinary that we are able to build such systems, but more often than not, problems are still to large, or the problem graphs are to dense (every variable in the problem is coupled to too many - more than 15 - other variables). In that case we have to group physical qubits into groups (logical qubits), which then form a more densely connected graph on which we can (maybe) solve our problem. This of course affects performance of such machines negatively but it is the best we've got.
Issues.

ISING SPIN GLASSES

The first system we can think of simulating on a device that evolves according to the Ising Hamiltonian is the Ising model. To make it more interesting, let's put the Ising magnet in a trasverse magnetic field (there will be σ_x-s in the Hamiltonian). Together, they comprise the transverse field Ising model (TFIM). This is a toy model for order-disorder phase transitions in quantum systems. It is known that TFIM in d dimensions is equivalent to ordinary Ising model in d+ 1 dimensions and is therefore more difficult to simulate on a classical computer [13]. It is usually analyzed using quantum Monte Carlo methods (QMC). In a recent paper [14], TFIM on a simple cubic lattice of size $8 \times 8 \times 8$ was simulated using an adiabatic quantum computer. By today this number has grown to $15 \times 15 \times 12$, which is of the same order as the classical computers are capable of: typical lattice sizes simulated with classical computers are around 153 [13]. After surpassing the capabilities of classical computers, we can talk about the so called quantum advantage [15]. The final step is to achieve quantum supremacy: the ability to perform tasks on a quantum computer that would be impossible to do on a classical in a reasonable time period (like a lifetime). Google announced quantum supremacy in 2019 for a very specific problem, but we are not there yet with quantum simulators [16].

We can make the model more interesting still by randomly sampling interactions between lattice sites: $J_{ij} = -1$ (ferromagnetic) with probability p and +1 (antiferromagnetic) with probability $1 - p$. The randomness of interaction makes this a spin glass problem. At first sight this model doesn't really seem applicable, but it is, besides being a simple disordered model in condensed matter physics, actually used extensively in the field of artificial neural networks [17]. Also half of the 2021 Nobel prize in physics was awarded to Giorgio Parisi for his study of the spin glass problem.

In order to draw a phase diagram in the parameter space (Γ, p, T), we have to compare our model (7) with (6), which actually describes our machine and extract the desired quantities. We are not in direct control of the temperature on the QPU, but we can control it effectively by setting the energy scale J of the system. We can see that by setting $\Gamma(s)/J = A(s)/B(s)$ and $k_B T(s)/J = k_B T_{QPU}/B(s)$, we obtain the desired parameters. The question that arises next is how do we simulate the system at fixed parameters, if we know that they are functions that change smoothly with the dimensionless time $s = t/T$. The solution to this issue is the specification of custom schedules s(t) that don't have to be a linear functions of time. Particularly useful is the "pause anneal schedule" depicted in Figure 4. We slowly increase s towards the desired value and then keep the system at fixed s for a long time which fixes our parameters. This is done because we want to ensure that the system is in equilibrium at given parameters. The next step is to quench the dynamics by quickly increasing $s \longrightarrow 1$. If this would be infinitely quick, it would represent a projection of the state of the system on the z-axis: a quantum measurement. It is, however, quite slow and is therefore the biggest talking point about whether or not devices like this can really be used for simulating physics. Its typical timescale is around 1 μs, while the typical energy scale in the system is a few GHz ×h, which corresponds to a sub-nanosecond regime. The system therefore evolves appreciably during the measurement and we don't really sample the system at specified parameters. This too is changing: an experiment from 2022 by D-Wave research group 7 used 1 ns-quenching, which is at least comparable with the energy scales, but this feature has not yet been made public

With that in mind, we can take a quick look at the results of the experiment. Beside the usual paramagnetic and antiferromagnetic phases, there is also a spin glass phase, which is characterized as a disordered phase with no spontaneous magnetization. However, at any given an instance of interactions $\{J_{ij}\}$, there are still some states that are preferable to others, unlike a paramagnet where all states are equally likely. If we imagine increasing p from 0 to 1/2, there will be a more and more lattice sites that have the same number of ferro and antiferromagnetic interactions. That means that such lattice sites won't really have any preference where to point and there will be many different states with equal energy, including many approximate ground states and we want to sample them by running the algorithm many times on every instance of the interactions. Because there are many ground states with similar energy, we can end up in any one of these states, and by averaging results over many instances we can determine the ground state properties of the system at given parameters. Subsequently, we can determine the whole phase diagram, which is portrayed schematically in Figure 5. The spin glass phase occurs when the disorder parameter p is larger than the critical value $p_c = 0.22 \pm 0.01$. We won't go into details of how to determine that here, but the plot of the results is shown if Figure 6 [14]. The article also determined the critical transverse field $\Gamma_c/J = 5.1 \pm 0.3$ and one critical exponent $a = 0.7 \pm 0.3$, all of which coincide with the numerical studies of the same problem reasonably well: critical disorder parameter p_c, for example differs by 0.002, which is approximately 1%. This shows that such devices can be used to simulate physics remarkably well if we consider the measurement problem.