Univerza *v Ljubljani*
Fakulteta *za matematiko in fiziko*

Adiabatic Quantum Computing

Author: Vid Eržen
Advisor: Prof. Dr. Dragan Mihailović
Co-advisor: Dr. Jaka Vodeb

Ljubljana, March 24, 2022

1 Abstract

In this seminar we present adiabatic quantum computing, an algorithm that was first developed for solving combinatorial optimization problems, but is becoming increasingly important as a quantum simulator. We begin with a brief history of the algorithm and continue with a review of the adiabatic theorem that enables it. In the next section we construct the most commonly used initial and problem Hamiltonians. A short review of an experimental realization by D-Wave Systems, Burnaby, Canada is then presented. We conclude with an example of the use of adiabatic quantum computing in simulating physics, concretely the Ising spin glasses.

Contents

1 Abstract **1**

2 Introduction **1**

3 The algorithm **2**
3.1 Adiabatic theorem 2

4 Hamiltonians **4**
4.1 Problem Hamiltonian 4
4.1.1 Example: Graph partitioning 4
4.2 Initial Hamiltonian 5
4.3 Time complexity of adiabatic computation 5

5 Experimental realization **6**

6 AQC in physics **6**

7 Conclusion **9**

2 Introduction

Quantum computing (QC) originated from Feynman noticing the difficulty of simulating quantum physics on a classical computer in the early 80's [1]. He imagined a quantum simulator might be able solve this problem and it was Deutsch in 1989 [2] who formalized this idea with a proposal for gate/circuit model of quantum computer. While it uses a series of quantum gates to drive qubits into any desired state (uses the full Hilbert space), adiabatic quantum computing (AQC) is different: the system is first prepared in the ground state of some initial Hamiltonian and then slowly evolve the system towards the ground state of a final Hamiltonian which must somehow encode your problem. The ground state then represents the solution to the computational problem. The adiabatic theorem guarantees that an isolated system at zero temperature will remain in the instantaneous ground state if the Hamiltonian is varied sufficiently slowly. Conditions like this of course cannot be realized experimentally, but this is still the idea on which AQC is built.

The first versions of the algorithm were named quantum annealing (QA), because they were executed on classical computers and used simulated quantum fluctuations (tunneling) to escape the local minima of the energy landscape and find the global minimum (ground state) of optimization problems [3]. This resembles the use of thermal fluctuations in simulated annealing, hence the name.

A different approach was taken by the experimentalists, who started to build dedicated devices that exploited the adiabatic evolution to find the ground state of the optimization problems. It was appropriately named adiabatic quantum computing. Even though it only uses the ground state for computation, compared to the whole Hilbert state used in the gate model, it was shown that they are equivalent in the sense that the problems solved on one model can be mapped into a form that can be solved by the other in at most polynomial time (as a function of the problem size) [4]. This means that all problems reside in the same complexity classes for gate model and AQC.

3 The algorithm

Adiabatic quantum computing was initially developed in order to solve optimization problems, which in physics means searching for ground states. Let's first state how the algorithm will work and postpone the discussion about why it is defined this way for the next section:

1. Initialize a system in the ground state of some Hamiltonian $H(0) = H_0$.
2. Vary the Hamiltonian for a period T. The final Hamiltonian $H(T) = H_p$ should encode the optimization problem.
3. Read out the state of the system. Its configuration represents the solution of the encoded problem.

The quantum mechanical principle that enables this to work is called the adiabatic theorem and it shows how the Hamiltonian should be varied in order to solve an optimization problem.

3.1 Adiabatic theorem

The dynamics of a quantum system are governed by its (time-dependent) Hamiltonian $H(t)$. The adiabatic theorem states that if you vary the Hamiltonian slowly enough, the system will stay in the same instantaneous state. In this section we will quantify what "slowly enough" means. 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.

Before we state the results of the adiabatic theorem, let's make a few assumptions. We introduce a parameter $s = t/T \in [0, 1]$, which is going to be the only parameter that our Hamiltonian will depend on. We also assume that the energy gap g between the lower two energy eigenstates is always greater than 0 (the ground state and the first excited state mustn't cross when we vary parameter s) and denote the minimum energy gap as

$$g_{min} = \min_{0 \leq s \leq 1} E_1(s) - E_0(s).$$

The energy levels typically do not cross, except in the case when the Hamiltonian has some symmetry (operator that commutes with it). This is called avoided crossing [5] for two-state systems or level repulsion in general. We know from quantum mechanics that the symmetries are reflected in conserved quantities of the system. If the system starts in a state that is symmetric to, for example, particle exchange and the Hamiltonian also possesses this symmetry, then the final state will also be symmetric to particle exchange. We can say that parts of the Hilbert space are decoupled. Two decoupled eigenvalues might cross, but because of the conserved quantities the transition between them can never happen [4]. We can see an example of level repulsion of a 4-qubit chain in Figure 1. For the rest of the seminar we will simply assume that the energy levels do not cross during the evolution.

The adiabatic theorem tells us how slowly we should vary the Hamiltonian in order to stay in the ground state [6]:

$$T \gg \max_{0 \leq s \leq 1} \frac{\left| \langle E_1; s | \frac{\partial H}{\partial s} | E_0; s \rangle \right|}{g_{min}^2}. \tag{1}$$

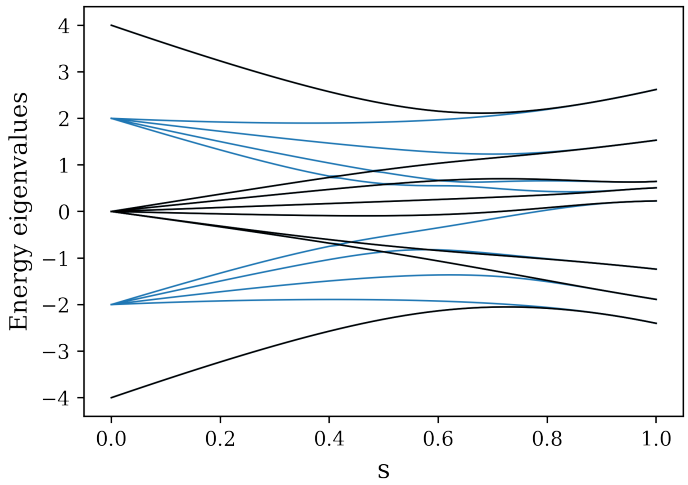

Figure 1: Typical eigenvalue flow during the evolution of a 4-qubit Hamiltonian, which possessed bit-flip symmetry $\sigma_z \longrightarrow -\sigma_z$ at all times: $H(s) = (1-s)\sum_{i,\,j>i}\sigma_i^x + s\sum_i J_{ij}\sigma_i^z\sigma_j^z$. Black curves represent the part of the spectrum with magnetization greater than 0 in computational basis (more spins pointing up than down in the final Hamiltonian), the blues are the eigenvalues of the block that corresponds to magnetization less than 0. Each eigenvalue of the final Hamiltonian is twice degenerated because of the symmetry, so we can ignore half of the spectrum. We also see that only energy levels from different sectors cross, while the eigenvalues from the same sector repel.

In order to evaulate the numerator, we restrict ourselves to interpolating Hamiltonians $H(s) = A(s)H_0 + B(s)H_p$ and we further assume that the schedules $A(s)$ and $B(s)$ are quadratic in s, were $A(s)$ is monotonically decreasing and $B(s)$ monotonically increasing. This is portrayed in Figure 3. Schedule like that is used in practice by D-Wave Systems machines [7]) and that is the reason for our choice. Using these assumptions, we can evaluate $\frac{\partial H}{\partial s} \sim s\,(H_p - H_0)$, which gives us

$$\max_{0\leq s\leq 1}\left|\langle E_1;s|\,\frac{\partial H}{\partial s}\,|E_0;s\rangle\right| \leq \max_{0\leq s\leq 1}\left|E_p^1 - E_0^0\right|, \tag{2}$$

where E_p^1 and E_0^0 denote the first excited state of the problem Hamiltonian and ground state of the initial Hamiltonian. Both of these values represent the energy of the system, which means their difference is also proportional to, at most, the energy of the system.

If we are talking about simulating physical systems, then the energy has to be an extensive function of the system size, so the nominator scales linearly with system size. By simulating a physical system, we mean determining its ground state properties since we only work in the instantaneous ground state of the Hamiltonian. An example of that is simulation of condensed matter systems, where interactions are often just between nearest neighbours or they decay quickly enough with the distance between particles. One of the first systems that was simulated on AQC was the Ising spin glass and the final section is mostly devoted to this problem.

However, if we simply want to solve an optimization problem, the energy (usually referred to as the cost function) of an abstract mathematical problem doesn't have to be extensive. In this case the denominator doesn't necessarily scale linearly with the problem size, but it never scales faster than polynomially, as we will see in the next section when we define the initial and problem Hamiltonians. We will also look at an example which will make things clearer.

4 Hamiltonians

4.1 Problem Hamiltonian

It is the Ising Hamiltonian

$$H_p = \sum_{i<j} J_{ij}\sigma_i^z\sigma_j^z + \sum_i h_i\sigma_i^z \tag{3}$$

that is used in practice by D-Wave Systems as the problem Hamiltonian. Expression (3) is an abstract Hamiltonian and it doesn't have the units of energy: parameters J_{ij} and h_i are dimensionless with magnitudes around 1. We will add the energy scales in the next section, when we look at experimental realization of it.

In order to solve an optimization problem, you therefore need to translate it into the Ising form. 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 on a graph with random edge weights (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 (3) 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.

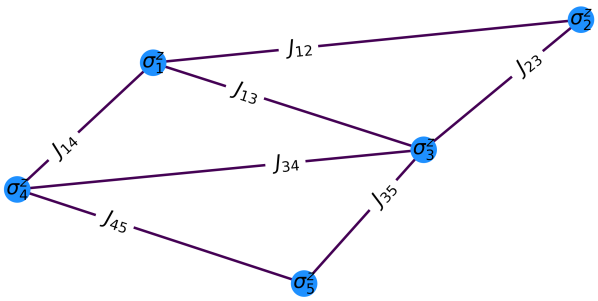

Figure 2: We can visualize the problem Hamiltonian as a graph with each vertex corresponding to a decision variable σ_i^z and edges between them represent the interactions J_{ij}.

4.1.1 Example: Graph partitioning

We chose graph partitioning as a toy example of translating a NP problem into the Ising form. Let's consider an undirected graph $G = (V, E)$ with vertices V and edges E. We would like to know how to cut the graph into two parts of the same size, such that the number of edges between both parts will be minimal. We then place Ising spin $\sigma_i^z = \pm 1$ at each vertex of the graph. The orientation of each spin tells us to which part of the graph the corresponding vertex should belong. The solution can be obtained by finding the ground state of

$$H = \left(\sum_i \sigma_i^z\right)^2 + \alpha \sum_{(i,j)\in E} \sigma_i^z\sigma_j^z = 2\sum_{i,\,j>i} \sigma_i^z\sigma_j^z + \alpha \sum_{(i,j)\in E} \sigma_i^z\sigma_j^z. \tag{4}$$

The first term is a penalty for the parts not being equal size and the second term is the penalty for each bond between partitions of the graph, and parameter α represents the ratio between the two penalties. Increasing α corresponds to enforces the minimization of the number of edges between the partitions and loosening the demand for partitions being the same size. In the equality we used that fact that $\sigma_i^{z\,2} = 1$ and omitted the sum over the diagonal because it would only add a constant offset and would therefore be irrelevant to the optimization problem. We can identify the Hamiltonians (3) and (4) by defining

$$J_{ij} = \begin{cases} 2 + \alpha; & (i,j) \in E \\ 2; & (i,j) \notin E \end{cases}$$

We can view this simply as a construction of the cost function that is quadratic in all of its variables, which is probably a bit less daunting than translating a problem into the Ising form. Some other well known problems, which can solved this way are the graph coloring problem and the traveling salesman.

4.2 Initial Hamiltonian

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_i^z 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

$$H_0 = \sum_i \sigma_i^x, \tag{5}$$

which is also the one that is actually used.The energy scale will be added in the next section.

4.3 Time complexity of adiabatic computation

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 ($J_{ij} = \pm 1$), 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 [11].

5 Experimental realization

In this section we are going to look how adiabatic quantum computing is realized in practice. In particular, we are describing how D-Wave Systems, Burnaby, Canada, implemented the algorithm [7]. The full one-parameter Hamiltonian running on their quantum processing unit (QPU) is

$$H(s) = -A(s)\sum_i \sigma_i^x + B(s)\left[\sum_i h_i\sigma_i^z + \sum_{i,\,j>i} J_{ij}\sigma_i^z\sigma_j^z\right], \tag{6}$$

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 and one between the two qubits) and two non-tunable compound-compound Josephson junction (ccjj) couplers between qubits and global current I_{ccjj} that control the coupling with the transverse field. 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 (we could say that something like a quantum version of Moore's law holds). 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.

In the first part of the seminar we assumed zero temperature and no coupling to the environment, but we can never actually achieve that. Both of these effects try to destroy our results. Finite temperature enables excitations into higher energy states, so at least we can do is to keep our QPU cool. Exactly how cool depends on the energy scales of the system. Rule of thumb is that $T \ll \hbar\omega/k_B \approx 100\,mK$, where we plugged in $\omega = (A(s)+B(s))_{min}/h \approx 2.5\,GHz$ as the smallest energy scale during the anneal (see Figure 3). It is actually kept at around $15\,mK$, which certainly satisfies this condition.

6 AQC in physics

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 field (there will be σ^x-s in the Hamiltonian). Together they comprise the transverse field Ising model (TFIM)

$$H/\mathcal{J} = -\Gamma/\mathcal{J}\sum_i \sigma_i^x + \sum_{i,j>i} J_{ij}\sigma_i^z\sigma_j^z, \tag{7}$$

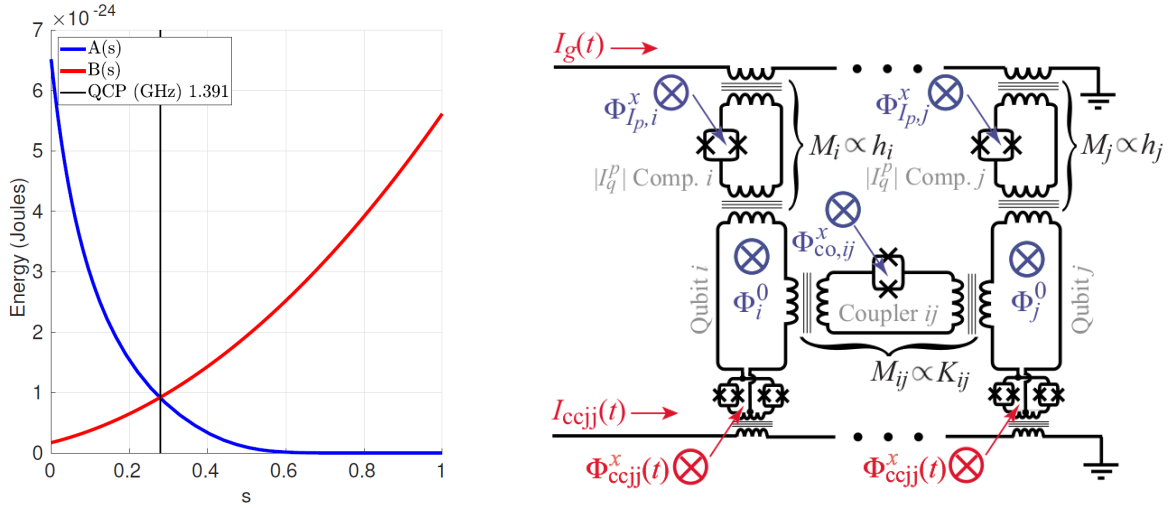

Figure 3: Left: Energy scales of initial and problem Hamiltonians. We start at $s = 0$ by exposing qubits to strong transverse field which forces them in the ground state, then evolve slowly and try to end up in the ground state of problem Hamiltonian at $s = 1$. Right: Schematic representation of two coupled qubits. Bias current I_g together with couplers, tunable by $\Phi^x_{I_{p,i}}$ control the biases h_i, tunable coupler $\Phi^x_{co,ij}$ controls the interaction between qubits and current I_{ccjj} controls qubits coupling to the trasverse field. Images taken from [7] and [12].

which is a toy model for order-disorder phase transitions in quantum systems. This is a lot more difficult to simulate it on a classical computer than the ordinary Ising model. It is usually analyzed using quantum Monte Carlo methods (QMC), which are very computationally intensive. As far as we know, people are able to simulate lattices with a few hundred lattice sites with QMC. In a recent paper [13], 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 a lot more than what classical computers are capable of, so we can talk about the so called quantum advantage [14], which simply means that a quantum computer or simulator can outperform a classical one. The next 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 [15].

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. 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 $\mathcal{J}$ of the system. We can see that by setting $\Gamma(s)/\mathcal{J} = A(s)/B(s)$ and $k_B T(s)/\mathcal{J} = k_B T_{QPU}/B(s)$, 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 spedification 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,

and then quench the dynamics by quickly increasing $s \longrightarrow 1$. This represents the measurement of our quantum system and is the biggest talking point about whether or not devices like this can really be used for simulating physics. The reason for that is the duration of the measurement. Its typical timescale is around $1\,\mu s$, while the typical energy scale in the system is a few $GHz \times h$, 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 used $1\,ns$-quenching, which is at least comparable with the energy scales, but this feature has not yet been made public [16].

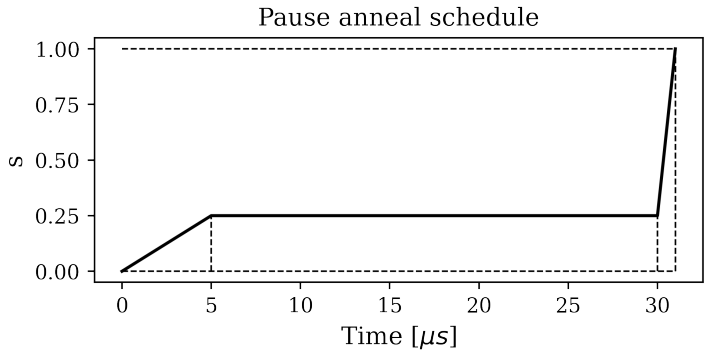

Figure 4: An example of a pause anneal schedule. We slowly increase s towards the desired value (we want to stay in the ground state), let the system evolve according to the Hamiltonian $H(s)$ and then quench the dynamics of the system by quickly increasing s to 1. This represents a quantum measurement. By repeating the procedure many times, we should be able to probe the system's ground state.

With that in mind, we can take a quick look at the results of the experiment. Beside the usual paramagnetic and ferromagnetic 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. By running the algorithm many times on every instance of the interactions, 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 and subsequently 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. The article also determined the critical transverse field $\Gamma_c/\mathcal{J} = 5.1 \pm 0.3$ 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.

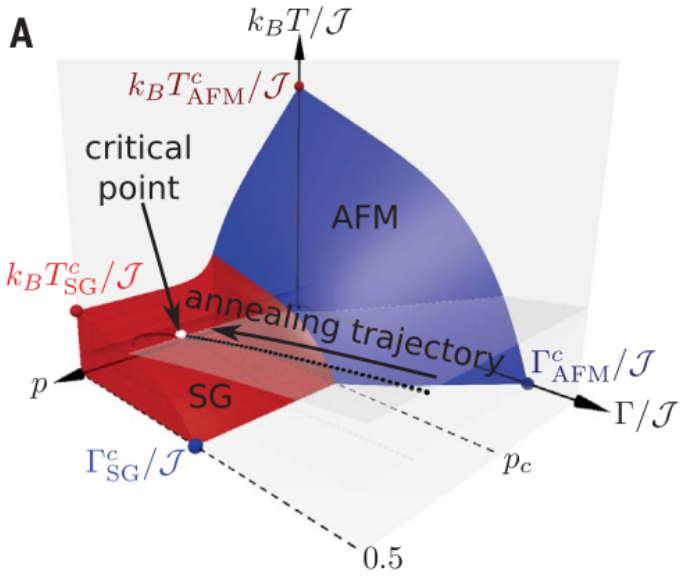

Figure 5: Schematic phase diagram of the Ising spin glass (7) in the (Γ, p, T) space. Increasing temperature and transverse field introduce thermal and quantum fluctuations into the model which results in the transition from the antiferromagnetic (AFM) to the paramagnetic phase (PM) for $0 < p < p_c$. The critical values of parameters decrease with p because of the added source of disorder. No magnetic order can occur for $p > p_c$, but this is not the PM phase, we call it the spin glass (SG) phase.

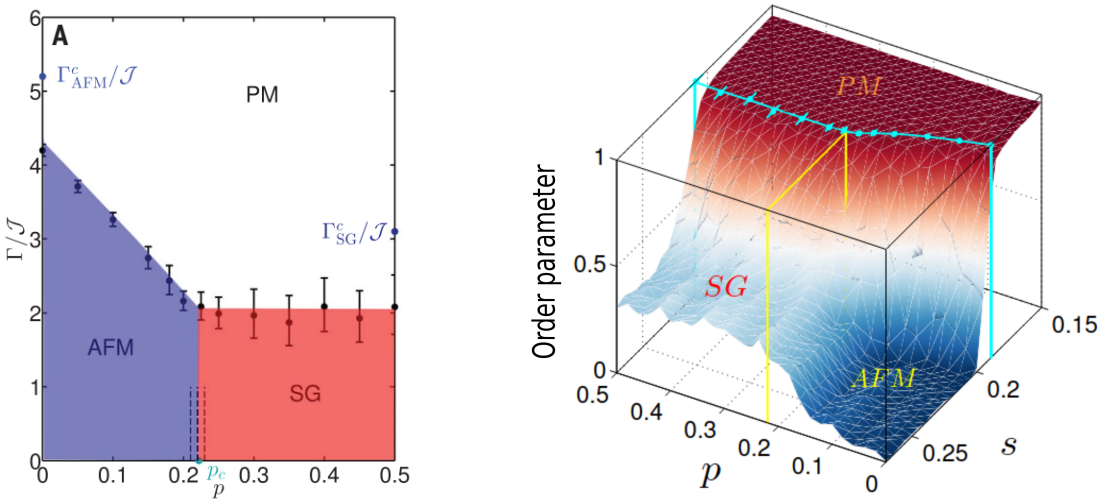

Figure 6: Experimental results of simulating the Ising spin glass on a D-Wave machine, taken from [13]. They used the Edwards-Anderson order parameter to distinguish the spin glass phase from the rest and linear susceptibility to determine the transition points between anttiferromagnetic (AFM) amd paramagnetic (PM) phase. We are not going to go into more details about what those are here. The critical values agree to around a few percent with those obtained by classical simulations.

7 Conclusion

We have seen how a simple idea, like adiabatic evolution, can be used to solve some real world problems. The rate of change of the Hamiltonian in order to solve the optimization problem with high success probability was quantified. But we must not forget that the adiabatic condition is still only an

estimate and that the outcome of the algorithm is probabilistic. It is therefore necessary to run the algorithm many times and pick the best solution, but that still doesn't guarantee that it is optimal. This becomes increasingly important for large problems because of the decreasing energy gaps between different states. This is also the reason why it is usually used alongside a classical computer to solve problems today.

This algorithm is also becoming increasingly important as a quantum simulator. Because gate model quantum computers are still relatively small, this is currently the best option (among quantum devices) for simulating physics, like condensed matter systems. We are currently in the regime where the capabilities of AQC are comparable to those of classical computers, but it is not going to be long before become superior.

Even if the gate model might some day take over, the simplicity of the idea underlying AQC is so appealing that it alone deserves our attention.

References

[1] Richard P Feynman, "Simulating physics with computers", in: *Feynman and computation*, CRC Press, 2018, pp. 133–153.

[2] David Elieser Deutsch, "Quantum computational networks", *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 425, no. 1868, pp. 73–90 (1989).

[3] Tadashi Kadowaki and Hidetoshi Nishimori, "Quantum annealing in the transverse Ising model", *Physical Review E*, vol. 58, no. 5, p. 5355 (1998).

[4] Edward Farhi et al., "Quantum computation by adiabatic evolution", *arXiv preprint quant-ph/0001106* (2000).

[5] Wikipedia, *Avoided crossing*, URL: `https://en.wikipedia.org/wiki/Avoided_crossing`, (accessed: 20.03.2022).

[6] Tameem Albash and Daniel A Lidar, "Adiabatic quantum computation", *Reviews of Modern Physics*, vol. 90, no. 1, p. 015002 (2018).

[7] D-Wave Systems, *D-Wave Annealing implementations and controls*, URL: `https://docs.dwavesys.com/docs/latest/c%5C_qpu%5C_annealing.html`, (accessed: 20.03.2022).

[8] Andrew Lucas, "Ising formulations of many NP problems", *Frontiers in physics*, p. 5 (2014).

[9] Edward Farhi et al., "A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem", *Science*, vol. 292, no. 5516, pp. 472–475 (2001).

[10] Jérémie Roland and Nicolas J Cerf, "Quantum search by local adiabatic evolution", *Physical Review A*, vol. 65, no. 4, p. 042308 (2002).

[11] EJ Crosson and DA Lidar, "Prospects for quantum enhancement with diabatic quantum annealing", *Nature Reviews Physics*, vol. 3, no. 7, pp. 466–489 (2021).

[12] Richard Harris et al., "Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor", *Physical Review B*, vol. 82, no. 2, p. 024511 (2010).

[13] R Harris et al., "Phase transitions in a programmable quantum spin glass simulator", *Science*, vol. 361, no. 6398, pp. 162–165 (2018).

[14] Andrew D King et al., "Scaling advantage over path-integral Monte Carlo in quantum simulation of geometrically frustrated magnets", *Nature communications*, vol. 12, no. 1, pp. 1–6 (2021).

[15] Frank Arute et al., "Quantum supremacy using a programmable superconducting processor", *Nature*, vol. 574, no. 7779, pp. 505–510 (2019).

[16] Andrew D King et al., "Coherent quantum annealing in a programmable 2000-qubit Ising chain", *arXiv preprint arXiv:2202.05847* (2022).