Assessing Solution Quality of 3SAT on a Quantum Annealing Platform

Thomas Gabor[1], Sebastian Zielinski[1], Sebastian Feld[1], Christoph Roch[1], Christian Seidel[2], Florian Neukart[3], Isabella Galter[2], Wolfgang Mauerer[4], and Claudia Linnhoff-Popien[1]

[1] LMU Munich
[2] Volkswagen Data:Lab
[3] Volkswagen Group of America
[4] OTH Regensburg

Abstract. When solving propositional logic satisfiability (3SAT) instances using a quantum annealer, we analyze the effect the difficulty of every instance has on the distribution of correct and incorrect solutions returned by the quantum annealer. We show that the phase transition known for 3SAT on classical machines persists in some form (but possibly to a lesser extent) for quantum annealing.

Keywords: Quantum Computing · Quantum Annealing · 3SAT · Boolean satisfiabiliy · NP · phase transition.

1 Introduction

The benefit of the technology of quantum computing essentially lies in the quantum speedup. More generally, if quantum computing is to succeed, it needs to be faster or better or cheaper than classical computing hardware at at least one useful task.

Research in that area has cast an eye on the complexity class NP: It contains problems that are traditionally (and at the current state of knowledge regarding the P vs. NP problem) conjectured to produce instances too hard for classical computers to solve within practical time constraints. However, the problems in NP are also known to be quite susceptible to the benefits of parallelization.

As quantum mechanics has an interpretation suggesting infinite parallelism of possibilities contained within a superposition, we may have a match here. For a rather concrete implementation of quantum mechanics we focus on the technological platform of quantum annealing. Quantum annealers focus on solving optimization problems only. As a trade-off they can as of now work rather large amounts of qubits in a useful way and are one of the few technological platforms in the area of quantum technology that are easily available to the general public.

In this paper we want to evaluate the performance of quantum annealing (or more specifically a D-Wave 2000Q machine) on the canonical problem of the class NP, i.e., propositional logic satisfiability for 3-literal clauses (3SAT) [12].

As we note that there is still a remarkable gap between 3SAT instances that can be put on a current D-Wave chip and 3SAT instances that even remotely pose a challenge to classical solvers, there is little sense in comparing the quantum annealing method to classical algorithms in this case. Instead, we are most interested in the scaling behavior with respect to problem difficulty. Or more precisely: We analyze if and to what extent quantum annealing's performance suffers under hard problem instances (like classical algorithms do).

To this end, we present a quick run-down of 3SAT and the phenomenon of phase transitions in Section 2 and continue to discuss further related work in Section 3. In Section 4 we describe our experimental setup and then present the corresponding results in Section 5. We conclude with Section 6.

2 Preliminaries

The K-Satisfiability Problem (hereinafter called *K-SAT*) belongs to the class of NP-complete problems. K-SAT (for K $\geq$ 3) is a central problem in combinatorial optimization and was the first problem for which NP-completeness could be shown [26].

According to [26] the problem can be formulated as follows: Given is a set $\mathcal{B}$ consisting of N Boolean *variables*. K variables are selected from the set $\mathcal{B}$ and they (or their negations) are then combined by (K-1) OR-Operators to form a *clause*. In this way M clauses are formed. These clauses are joined by applying (M-1) AND-Operators such that a K-SAT instance is created.
The K-SAT problem asks the question, whether there is an allocation of the boolean variables from $\mathcal{B}$, so that a given K-SAT instance can be fulfilled.

In this paper only randomly generated K-SAT instances for K = 3 are considered. These problems are then called 3-SAT and are NP-complete after [12].

Example 3-SAT instance. *As explained in the previous problem formulation, a 3-SAT instance consists of 3 variables per clause. An instance of a 3-SAT problem is for example:* $\Psi = (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_3)$.

2.1 Phase transitions in SAT solving

Despite the fact that for NP-complete problems in general no algorithm is known, which can solve all problem instances of a problem efficiently, i.e. in polynomial time, it is within the scope of knowledge, that certain problem instances for many NP-complete problems, including K-SAT, are easy to solve [7]. In [28] this characteristic is described with a phase transition. The boundary of this phase transition divides the problem space into two regions. In one region a solution can be found relatively easily, because the solution density for these problems is high, in the other region it is very unlikely that problems can contain a correct solution at all. Problems that are very difficult to solve are located directly at this phase boundary [7].

It can be observed that, with randomly generated K-SAT instances, the probability of finding a correct solution decreases abruptly when a critical value of α_c (formed from the ratio of number of clauses to number of variables) is exceeded [27]. According to [26] this critical point is $\alpha_c \approx 4.267$ for randomly generated 3-SAT instances. In the surrounding area of the critical point the solution finding (here not only a concrete allocation is meant, if the instance is solvable, but also the realization that this instance cannot be solved) is algorithmically complex. Fig. 1 illustrates this phenomenon visually.

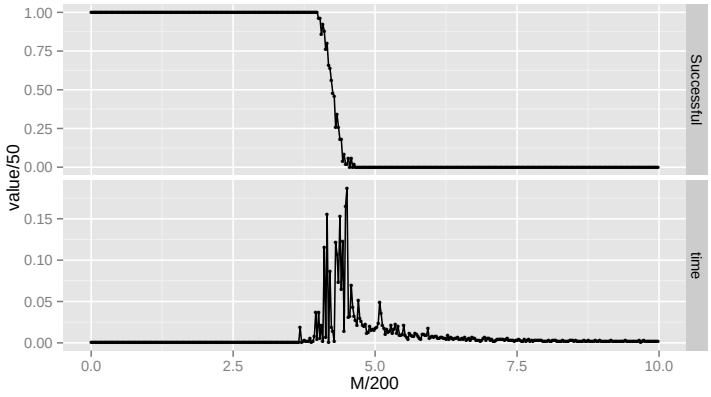

Fig. 1. Phase transition of SAT solving. In the lower figure the computational time for finding a solution for 3-SAT instances with a specific clauses-to-variables ratio (c/v-ratio) can be seen. The critical point is around $\alpha_c \approx 4.267$. In this regard, the upper figure shows the propability, whether these problem instances are solvable. So for the critical point it's hard to determine, whether a problem instance can be fulfilled with a concrete allocation or not. TODO Achsenbeschriftung value/50 M/200 erklären

In order to assess the solution quality of randomly generated 3-SAT instances we take the phase transition into account and generate instances of every complexity region. More information about that in section Experimental Setup.

2.2 Reduction from 3-SAT to Weighted Maximum Independent Set

Since it can be shown that the Weighted Maximum Independent Set (WMIS) problem is equivalent to the Ising model [9], the 3-SAT problem is first transformed into a WMIS problem, which can then be solved on an adiabatic quantum computer based on this equivalence.

The WMIS can be formulated as follows: Given is an undirected graph $\mathcal{G} = (V, E)$, where V is the set of vertices and E is the set of edges of $\mathcal{G}$, in which each vertice v ∈ V is provided with a positive real weight c_v.

The problem is then to find a set S $\subset$ V, so that S is an independent set and the total weight of S, given by $\sum_{v \in S} c_v$, is a maximum. The optimal solution is also called WMIS($\mathcal{G}$).

The reduction from 3-SAT to WMIS is well known after [10] and is for completeness recalled below.

Reduction (3-SAT $\leq_P$ WMIS). Given a 3-SAT instance $\Psi(x_1, \ldots, x_n) = C_1 \wedge ... \wedge C_m$ consisting of m clauses *($C_1, ..., C_m$)* and $n = 3m$ variables. Construct a graph G_{SAT} as follows:

Step 1:
For each clause $C_i = y_{i_1} \vee y_{i_2} \vee y_{i_3}$, $y_j \in \{x_j, \overline{x_j}\}$, a triangle is constructed, with the corners of the triangle being vertices labeled with the literals of this clause $(y_{i_1}, y_{i_2}, y_{i_3})$. Therefore the graph G_{SAT} consists of 3m vertices.

Step 2:
Edges are added between vertices of different triangles if their labels conflict. That is, for two different triangles i and j, $i \neq j$, which represent clause i and j, an edge is added, if there are variables y_{i_s} and y_{j_t}, so that $y_{i_s} = \overline{y_{j_t}}$.

By doing this reduction the theorem after [10] states, that an arbitrary 3-SAT formula Φ, consisting of m clauses, can be fulfilled, if the graph G_{SAT}, which was created by reducing Φ to a WMIS problem, has a WMIS of the size m. This assertion represents an important relation between the satisfiability of a 3-SAT problem and the solution of the WMIS problem, which we take into account in this work.

For the sake of completeness we have to mention, that D-Wave's quantum annealer is designed to find the lowest energy state of a spin glass system, described by an Ising Hamiltonian (see equation 1). Here h_i is the on-site energy of qubit i, J_{ij} are the interaction energies of two qubits i and j, and s_i representing the spin (-1, +1) of the i-th qubit. So in order to execute the WMIS problem on the quantum annealing hardware , it has to be mapped to the Ising model. However, in [9] proof is given, that the WMIS problem is equivalent to the Ising model and no more modifications have to be made.

$$\mathcal{H}(s) = \sum_i h_i s_i + \sum_{i<j} J_{ij} s_i s_j \tag{1}$$

3 Related Work

It is one of the cornerstones of complexity theory that solving NP-complete or even NP-hard decision problems is not efficiently possible (e.g., [12, 31]). Any NP-complete problem can also be cast as an optimisation problem, which allows for employing well-known optimisation algorithms to find approximate solutions – typical methods include tabu search (see, e.g., [16, **?**]) and simulated annealing (see, e.g., [19, **?**]). Countless other efficient approximation methods, together

with an elaborate taxonomy on approximation quality (how much does a given solution differ from a known global optimum?) and computational effort (how many time steps are required until an approximate solution that satisfies given quality goals is available?), have been devised (see, e.g., [**?**]).

Some problems, for instance knapsack, exhibit favourable properties when cast as an optimisation problem. The latter is a member of the complexity class FPTAS (fully polynomial-time approximation scheme), which means that a solution with distance $1 + \epsilon$ (of course, $\epsilon > 0$) from an optimal solution can be determined in polynomial time in both, input size n and $1/\epsilon$ [8].

An intriguing connection that has received substantial attraction exists between (computational) NP-complete problems and the (physical) concept of phase transitions, as detailled in Sec. **??**. First investigations of the phenomenon have been performed by Kirkpatrick et al. [20], TODO [**?**] and TODO [**?**]; more recent investigations include [**?**,**?**,**?**]. Our work benefits from these insights by selecting the "most interesting", that is, computationally hardest scenarios as investigation target.

The idea of obtaining solutions for NPO (NP optimisation) problems by finding the energy ground state (or states) of a quantum mechanical system appeared around 1988 [**?**,1] in the context of efforts by Apolloni et al. [3, 4] to solve combinatorial optimisation problems. The idea of quantum annealing has been independently re-discovered multiple times [1, 15, 2, 18].

Quantum annealing techniques are usually applied to solving NP-complete or NP-hard decision problems, or optimisation problems from class NPO. Lucas [22] reviews how to formulate a set of key NP problems in the language of adiabatic quantum computing respectively quadratic unconstrained binary optimisation. In particular, problems of the types "travelling salesman" or "binary satisfiability" that are expected to have a major impact on practical computational applications if they can be solved advantageously on quantum annealers have undergone a considerable amount of research [17, 35, 30, 32, 5].

Comparing the computational capabilities of classical and quantum computers is an intriguing and complex task, since the deployed ressources are typically very dissimilar. For instance, the amount of instructions required to execute a particular algorithm is one of the main measures of efficiency or practicability on a classical machine, whereas the notion of a discrete computational "step" is hard to define on a quantum annealing device.

Interest in quantum computing has therefor spawned definitions of new complexity classes (e.g.,[21, 29]), whose relations to traditional complexity classes have been and are still subject to ongoing research (see, e.g.,[6, 24]).

These questions hold regardless of any specific physical or conceptual implementation of quantum computing since they are known to be largely interchangeable in regards to their overall computational capabilities; see, for instance, Ref. [25] for a discussion on the equivalence of gate-based and adiabatic quantum computing. Consequently, our work focuses not on comparing quantum and classical aspects of solving particular problems, but concentrates on under-

standing peculiarities of solving one particular problem (3-SAT, in our case) in-depth.

Formulating 3-SAT problems on a quantum annealing hardware has been previously considered [11, 10, 14], and we rely on the encoding techniques presented there. Van [34] and Farhi [13] have worked on analysing the complexity of solving general 3-SAT problems.

4 Experimental Setup

Quantum annealing is an optimization process that can be implemented in hardware. It is built on the adiabatic theorem which in theory guarantees the evolution of an initial configuration of the system to a configuration minimizing a specific user-defined energy function [25]. As in the real world the required conditions for the theorem can only be approximated, the results of quantum annealing are usually not exact but show a probabilistic distribution, ideally covering the desired optimal value as well.

D-Wave's quantum annealer is the first commercial machine to implement quantum annealing. Its interface is built on two equivalent mathematical models for optimization problems called Ising and QUBO, the latter of which will be used for the work of this paper. Quadratic Unconstrained Binary Optimization (QUBO) problems can be formulated as a quadratic matrix Q_{ij}. Quantum annealing then searches for a vector $x \in \{0,1\}^n$ so that $\sum_i \sum_{j<i} Q_{ij} x_i x_j + \sum_i Q_i x_i$ is minimal. The promise of quantum annealing is that–using quantum effects–specialized hardware architectures are able to solve these optimization problems much faster than classical computers in the future.

The main goal of this paper is to analyze the inherently probabilistic distribution of return values generated by quantum annealing when trying to solve hard optimization problems. We choose to demonstrate such an analysis on 3SAT because it is the canonical problem of the class NP, which is a prime target for research on performance improvements via quantum technology with respect to classical computers [].

4.1 Defining 3SAT as a QUBO matrix

However, 3SAT is not even usually formulated as an optimization problem (see Section 2) or a QUBO matrix specifically. Thus, we require a (polynomial-time) translation of any 3SAT instance into a QUBO matrix so that the solutions generated by the quantum annealer can be translated back to solution of the initial 3SAT instance.

For this purpose, we use the approach given in [10, 11] by translating 3SAT into the Weak Maximum Independent Set (WMIS) problem first and then translating the WMIS instance into a QUBO matrix. We omit the details of this process and instead refer to literature [10, 11, 22]. However, we shall briefly discuss the implications of that translation process.

In the end for a 3SAT instance, i.e., a formula, with m clauses for n variables we end up with a QUBO matrix of size $3m \times 3m$ with the solution vector $x \in \{0,1\}^{3m}$. The solution can be thought of as using a qubit for each literal in the initial formula and thus consisting of a triplet of qubits for each 3SAT clause. This usually means that we have much more qubits than variables in the formula. Nonetheless, a QUBO solution is mapped to a value assignment for the variables in the 3SAT formula. Thus, when running successfully, the quantum annealer will output a satisfying assignment for a given 3SAT formula. We can check if the assignment really is correct (i.e., each variable has a value assigned and the whole formula reduces to *True*) using few instructions of classical computation. Obviously, if among several experimental runs the quantum annealer does return a correct assignment, the corresponding 3SAT formula is satisfiable. If the quantum annealer only return incorrect assignments, we will regard the formula as unsatisfiable (although the prove of that is only probabilistic).

There are some aspects to note about how the QUBO solution vectors are mapped to variable assignments. Given a QUBO solution vector $(x_i)_{0 \leq i \leq 3m-1}$ for a 3SAT formula with literals $(l_i)_{0 \leq i \leq 3m-1}$, a variable v is assigned the value *True* if it occurs in a literal $l_i = v$ and $x_i = 1$. Likewise, a variable v is assigned the value *False* if it occurs in a literal $l_i = \neg v$ and $x_i = 1$. It is important to note that $x_i = 0$ has *no implication* on the value of the variable in l_i.

Intuitively, we can interpret $x_i = 1$ to mean "use the value of l_i to prove the satisfaction of clause $c_{(i \mod 3)}$". From our QUBO optimization, we expect to find one (and only one) suitable l_i for every clause in the 3SAT formula.[5]

This is important as it opens up a wide range of different QUBO solutions which may just encode the exact same variable assignment at the 3SAT level. However, it also means that seemingly suboptimal QUBO solutions may encode correct 3SAT assignments. For example, consider the (a bit redundant) 3SAT formula $(v_0 \vee v_1 \vee v_2) \wedge (v_0 \vee v_1 \vee v_2)$: The QUBO solution $x = 100001$ would imply the assignment of $v_0 = \textit{True}$ and $v_2 = \textit{True}$, which indeed is theoretically sufficient to prove the formula satisfiable. The exact same assignment would be implied by $x = 001100$. However, note that none of these imply a full assignment of every variable in the 3SAT instance since none say anything about the value of v_1. Still, we can trivially set v_1 to any arbitrary value and end up with a correct assignment. Also not that while the QUBO is built in such a way to opt for one single value 1 per triplet in the bitstring, even bitstrings violating this property can encode correct solution. In our example, the suboptimal QUBO solution $x = 100000$ still encodes all necessary information to prove satisfiability.

4.2 Evaluating Postprocessing

As can be seen from the last example, postprocessing is an integral part of solving problem with quantum annealing. As discussed in the previous section, we

[5] Perhaps interestingly, this intuition matches the concept of constructivism in logic and mathematics. We are not only looking for the correct answer, but are looking for a correct and complete proof of an answer, giving us a single argument for each part of the formula.

consider a QUBO solution correct not only if its matches the expected structure for minimizing the QUBO energy function but instead iff it directly implies a correct assignment in the definition given above. Thus, where the expected structure for QUBO optimizes so that the amount bits assigned 1 equals the amount of clauses m, we also consider less full answers correct.

On top of that, there are solutions that cannot be mapped to an assignment immediately, but still with next to no effort. We want to regard these as well and implemented a postprocessing step we call *logical postprocessing*. It is applied whenever none of the qubits corresponding to a single clause c_k are set to 1 by the quantum annealer and the respective QUBO solution is not already correct. In that case, we iterate through all literals l_i in that clause c_k and check if we could set $x_i = 1$ without contradicting any other assignment made within x. If we find such an l_i, we set $x_i = 1$ and return the altered bitstring x.

The software platform provided by D-Wave to use the quantum annealer already offers integrated postprocessing methods as well, which we will also empirically prove to be more powerful than logical postprocessing in the following Section 5. Again, for greater detail we refer to the D-Wave documentation on that matter [33]. At a glance, the employed postprocessing method splits the QUBO matrix into several subproblems, tries to optimize these locally, and then integrates that local solution into the complete solution if it yields an improvement. We call this method *D-Wave postprocessing*.

In order to evaluate the solution quality regarding 3SAT we employ both methods. The goal is to assess the expected quality on a 3SAT-to-3SAT level, i.e., we measure how well we can solve the given 3SAT instance and regard the translation to and from QUBO as a mere tool chain but not that interesting on its own for the scope of this paper.

5 Evaluation

To assess the solution quality of 3SAT on a quantum annealing platform, using the previously discussed method of encoding 3SAT problems, we ran several experiments on a D-Wave 2000Q system. Using ToughSAT[6] we generated 3SAT instances of various difficulty (i.e., with various values for α). However, as discussed in Section 2, for $|\alpha - 4.2| \gg 0$ problem instances become very easy to solve. We observed that effect on the quantum annealer as well, i.e., all of these instances were trivially solved on the D-Wave machine. Thus, for the remainder of this section, we focus on hard instances (with $\alpha = 4.2$) to assess the quality in the really hard cases.

Experiments have shown that using the standard embedding tools delivered with the D-Wave platform, we can only reliably find a working embedding on the D-Wave 2000Q chip for 3SAT instances with at most 42 clauses. In order to maintain $\alpha = 4.2$, it follows that the generated 3SAT instance contain 10 different variables. We only assess solution quality for 3SAT instances that are satisfiable (but of course we are not telling the solver that).

[6] https://toughsat.appspot.com/

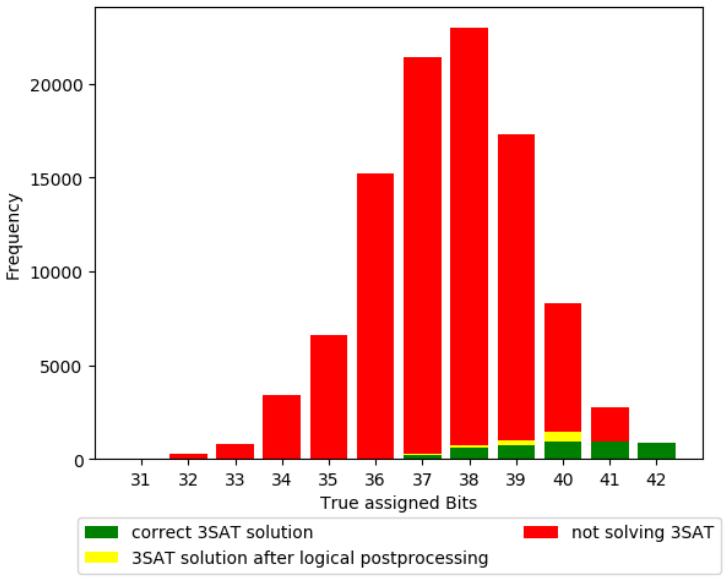

Fig. 2. Distribution of correct (green) and incorrect (red) answers returned by the quantum annealer *without D-WAVE postprocessing*. Answers that can trivially be transformed into valid answers using logical postprocessing are marked in yellow. The plot shows 100,000 answers in total for 100 different hard 3SAT instances ($\alpha \approx 4.2$).

Figure 2 shows the result distribution of these runs on the D-Wave machine. In the x-axis, we sorted the returned results according to the bits that have been assigned the value 1 or *True*. As discussed in Section **??** the optimal solution is supposed to set one bit for each clause, i.e., is supposed to contain 42 bits set to *true*. However, as there are only 10 different variables, there are answers that only set 10 bits that still map to complete and valid solution for 3SAT. From Figure 2 we can see that some of these solution are found for bitcounts starting from 37 through 41. Interestingly, the complete range of answers gathered seems to follow a distribution centered around 37 or 38 and no answer with more than 42 bits are returned. This means that the constraint of never setting multiple bits per clause is fully respected in the evaluation of our QUBO matrix. It is important to note that although there is some amount of correct solutions, these are only distributed across 24 of the 100 randomly generated problem instances. Thus, most of them have not been solved at all.

Furthermore, we applied the logical postprocessing described in Section **??** to the incorrect answers in Figure 2. However, it shows little improvement on the total amount of correct answers collected. As discussed, we expect the postprocessing method delivered with the D-Wave software package to be more powerful. So we ran the complete evaluation experiment again, only this time turning on the integrated postprocessing. The results are shown in Figure 3.

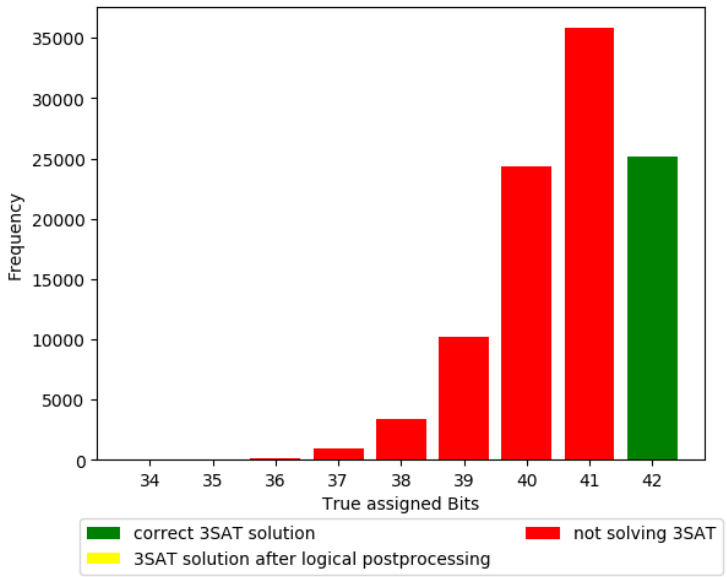

Fig. 3. Distribution of correct (green) and incorrect (red) answers returned by the quantum annealer *using D-WAVE postprocessing*. Answers that can trivially be transformed into valid answers using logical postprocessing are marked in yellow. The plot shows 100,000 answers in total for 100 different hard 3SAT instances ($\alpha \approx 4.2$).

We observed that the D-Wave postprocessing managed to optimize all correct but "incomplete" answers, mapping them to solution with 42 bits assigned the value *True*. Out of the 100,000 queries, this yielded 25,142 correct answers. Moreover, these correct answers span 99 of the 100 randomly generated 3SAT instances so that we consider the problem solved here. Effectively, this shows that quantum annealing does suffer from a breakdown in performance at the point of the phase transition in the 3SAT problem. In comparison to the immense decrease in performance seen in classical solvers (cf. Section **??**), a drop to around 25% appears rather desirable. A quick example: To achieve a $1 - 10^{-12}$ confidence of returning the correct answer our experimental setup requires around 97 queries. At a glance, that scaling factor with respect to problem difficulty is much better than what is observed for classical algorithms. It is important to note, however, that these experiments were performed for problem instances so small that their evaluation does not pose a challenge to classical processors at all, i.e., below the point of reasonable performance metrics. These results only proof relevant if they scale with future versions of quantum annealing hardware that can tackle much larger problem instances.

So far, we have not discerned between different correct solutions. We were content as long as the algorithm returned but one. However, for the user it is interesting to know if he or she will receive the same solution or an even distribution across the complete solution space. Our experiments show that when a lot of correct solutions are found for a certain problem instance, there are cases where we can see a clear bias towards a specific solution variant. Figure 4 shows the distributions of specific solutions. While some formulae seem to yield rather narrow distributions over the different possible answers, others definitely seem to have a bias towards certain solutions. However, the former also tend to have relatively smaller sample sizes as there are less solutions in total to consider. Further investigation could still reveal a distinctive distribution in these cases as well. Thus, we consider this behavior of the quantum annealer to be roughly in line with the findings of [23] who show an exponential bias in ground-state sampling of a quantum annealer.

Lastly, we performed an additional experiment to check for the contingency of the presented results. As we have considered rather small problem instances (to the current limitations of the D-Wave chip) and a rather over-specified encoding of 3SAT (which allows for things like logical postprocessing to add benefit in the first place), we posed the question how much of solving these 3SAT instances actually requires an "intelligent" solving algorithm and how much part of the solution can be attributed to the environment. To examine this, we generated "solutions" to 3SAT instances using a random generator. We first considered a simple random generator that produced bitstrings of the same length as the D-Wave machine's answers. Figure 5 (left) shows the result for *easy* 3SAT instances, i.e. $\alpha = 0.2$: Out of 100,000 queries for 100 different formulae, 19 returned answers represented a valid 3SAT solution. Interestingly, logical postprocessing was still able to yield another 2,425 solutions. When we increased the problem difficulty to the previously considered $\alpha = 4.2$, none of the randomly generated

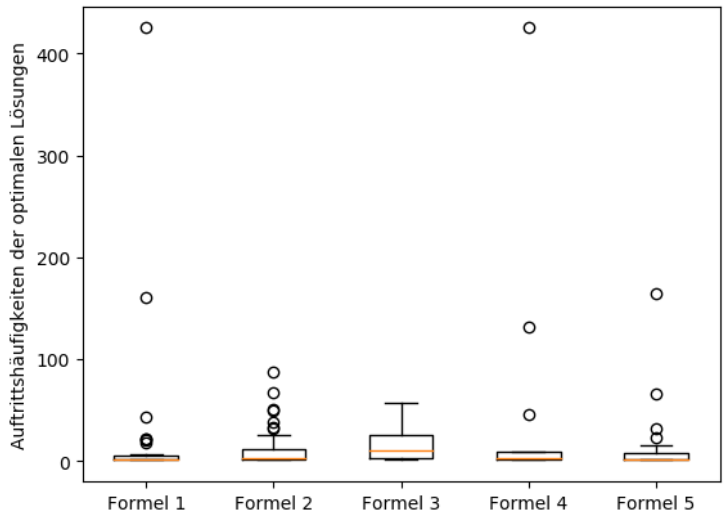

Fig. 4. Frequency of occurrence of different solutions for 5 formulae with many returned solutions. While most solutions are found once or just a few times, there are specific solutions that are found much more often.

"solutions" were correct or could be postprocessed to be correct. Futhermore, we performed this test on another random generator which return bitstrings of the form $\{001, 010, 100\}^{42}$, i.e., solutions which are always correct at setting just one bit per 3SAT clause. While this allowed us to find more (i.e., 12,043) correct solutions in the easy instances (cf. the right part of Figure 5), we still id not generate a single correct solution among 10,000 for the transition point instances.

6 Conclusion

We have shown that problem difficulty of 3SAT instances also affects the performance of quantum annealing as it does for classical algorithms. However, bound by the nature of both approaches, the effects are quite different with complete classical algorithms showing longer runtimes and quantum annealing showing less precision. A first quantification of that loss of precision suggests that it may not be too detrimental and comparatively easy to deal with. However, due to the maximum available chip size for quantum annealing hardware at the moment., no large-scale test could be performed. Thus, no real assumptions on the scaling of this phenomenon (and thus the eventual real-world benefit) can be made yet.

Our results suggest that there are cases where single solutions from a set of equally optimal solutions are much more likely to be returned than others. This observation is in line with other literature on the results of quantum annealing,

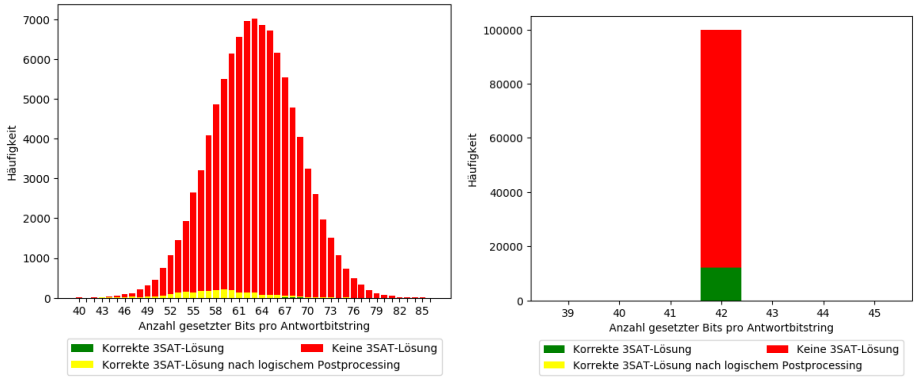

Fig. 5. Distribution of correct (green) and incorrect (red) answers returned by two different random generators. Answers that can trivially be transformed into valid answers using logical postprocessing are marked in yellow. Both plots shows 100,000 answers in total for 100 different hard 3SAT instances ($\alpha \approx 0.2$).

however, it is interesting to note that it also translates into the original problem space of 3SAT.

The observed results gain practical relevance with larger chip sizes for quantum annealers. We thus suggest to perform these and/or similar tests for future editions of quantum annealing hardware. if the effects persist, they represent a huge advantage of quantum hardware over any other known approaches for solving NP-hard problems.

References

1. Albash, T., Lidar, D.A.: Adiabatic quantum computing. arXiv preprint arXiv:1611.04471 (2016)
2. Amara, P., Hsu, D., Straub, J.E.: Global energy minimum searches using an approximate solution of the imaginary time schrödinger equation. The Journal of Physical Chemistry **97**(25), 6715–6721 (1993)
3. Apolloni, B., Carvalho, C., De Falco, D.: Quantum stochastic optimization. Stochastic Processes and their Applications **33**(2), 233–244 (1989)
4. Apolloni, B., De Falco, D., Cesa-Bianchi, N.: A numerical implementation of "quantum annealing". Tech. rep. (1988)
5. Benjamin, S.C., Zhao, L., Fitzsimons, J.F.: Measurement-driven quantum computing: Performance of a 3-sat solver. arXiv preprint arXiv:1711.02687 (2017)
6. Bernstein, E., Vazirani, U.: Quantum complexity theory. SIAM Journal on Computing **26**(5), 1411–1473 (1997)
7. Cheeseman, P.C., Kanefsky, B., Taylor, W.M.: Where the really hard problems are. In: IJCAI. vol. 91, pp. 331–340 (1991)
8. Chen, L., Aihara, K.: Chaotic simulated annealing by a neural network model with transient chaos. Neural networks **8**(6), 915–930 (1995)

9. Choi, V.: Minor-embedding in adiabatic quantum computation: I. the parameter setting problem. Quantum Information Processing **7**(5), 193–209 (2008)
10. Choi, V.: Adiabatic quantum algorithms for the np-complete maximum-weight independent set, exact cover and 3sat problems. arXiv preprint arXiv:1004.2226 (2010)
11. Choi, V.: Different adiabatic quantum optimization algorithms for the np-complete exact cover and 3sat problems. Quantum Information & Computation **11**(7-8), 638–648 (2011)
12. Cook, S.A.: The complexity of theorem-proving procedures. In: Proceedings of the third annual ACM symposium on Theory of computing. pp. 151–158. ACM (1971)
13. Farhi, E., Goldstone, J., Gosset, D., Gutmann, S., Meyer, H.B., Shor, P.: Quantum adiabatic algorithms, small gaps, and different paths. arXiv preprint arXiv:0909.4766 (2009)
14. Farhi, E., Goldstone, J., Gutmann, S., Sipser, M.: Quantum computation by adiabatic evolution. arXiv preprint quant-ph/0001106 (2000)
15. Finnila, A., Gomez, M., Sebenik, C., Stenson, C., Doll, J.: Quantum annealing: a new method for minimizing multidimensional functions. Chemical physics letters **219**(5-6), 343–348 (1994)
16. Glover, F., Laguna, M.: Tabu search*. In: Handbook of combinatorial optimization, pp. 3261–3362. Springer (2013)
17. Heim, B., Brown, E.W., Wecker, D., Troyer, M.: Designing adiabatic quantum optimization: A case study for the traveling salesman problem. arXiv preprint arXiv:1702.06248 (2017)
18. Kadowaki, T., Nishimori, H.: Quantum annealing in the transverse ising model. Physical Review E **58**(5), 5355 (1998)
19. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. science **220**(4598), 671–680 (1983)
20. Kirkpatrick, S., Selman, B.: Critical behavior in the satisfiability of random boolean expressions. Science **264**(5163), 1297–1301 (1994)
21. Klauck, H.: The complexity of quantum disjointness. In: LIPIcs-Leibniz International Proceedings in Informatics. vol. 83. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2017)
22. Lucas, A.: Ising formulations of many np problems. Frontiers in Physics **2**, 5 (2014)
23. Mandra, S., Zhu, Z., Katzgraber, H.G.: Exponentially biased ground-state sampling of quantum annealing machines with transverse-field driving hamiltonians. Physical review letters **118**(7), 070502 (2017)
24. Marriott, C., Watrous, J.: Quantum arthur–merlin games. Computational Complexity **14**(2), 122–152 (2005)
25. McGeoch, C.C.: Adiabatic quantum computation and quantum annealing: Theory and practice. Synthesis Lectures on Quantum Computing **5**(2), 1–93 (2014)
26. Mézard, M., Zecchina, R.: Random k-satisfiability problem: From an analytic solution to an efficient algorithm. Physical Review E **66**(5), 056126 (2002)
27. Monasson, R., Zecchina, R.: Entropy of the k-satisfiability problem. Physical review letters **76**(21), 3881 (1996)
28. Monasson, R., Zecchina, R., Kirkpatrick, S., Selman, B., Troyansky, L.: Determining computational complexity from characteristic 'phase transitions'. Nature **400**(6740), 133 (1999)
29. Morimae, T., Nishimura, H.: Merlinization of complexity classes above bqp. arXiv preprint arXiv:1704.01514 (2017)

30. Moylett, D.J., Linden, N., Montanaro, A.: Quantum speedup of the traveling-salesman problem for bounded-degree graphs. Physical Review A **95**(3), 032323 (2017)
31. Murty, K.G., Kabadi, S.N.: Some np-complete problems in quadratic and nonlinear programming. Mathematical programming **39**(2), 117–129 (1987)
32. Strand, J., Przybysz, A., Ferguson, D., Zick, K.: Zzz coupler for native embedding of max-3sat problem instances in quantum annealing hardware. In: APS Meeting Abstracts (2017)
33. Systems, D.W.: Postprocessing Methods on D-Wave Systems (2016)
34. Van Dam, W., Mosca, M., Vazirani, U.: How powerful is adiabatic quantum computation? In: Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on. pp. 279–287. IEEE (2001)
35. Warren, R.H.: Small traveling salesman problems. Journal of Advances in Applied Mathematics **2**(2) (2017)