A Quantum Annealing Approach for Boolean Satisfiability Problem

Juexiao Su[1], Tianheng Tu[1], and Lei He[1,2]
[1]EE Dept., Univeristy of California, Los Angeles, CA, USA
[2]Fudan State Key Lab for Application Specific Circuits and Systems, Shanghai, China
sujuexiao@g.ucla.edu, mynameisprost@g.ucla.edu, lhe@ee.ucla.edu

ABSTRACT

Quantum annealing device has shown a great potential in solving discrete problems that are theoretically and empirically hard. Boolean Satisfiability (SAT) problem, determining if there is an assignment of variables that satisfies a given Boolean function, is the first proven NP-complete problem widely used in various domains. Here, we present a novel mapping of the SAT problem to the quadratic unconstrained binary optimization problem (QUBO), and further develop a tool flow embedding the proposed QUBO to the architecture of the commercialized quantum computer D-Wave. By leveraging electronic design automation techniques including synthesis, placement and routing, this is not only the first work providing the detail flow that embeds the QUBO, but also a technique scalable for real world applications and some hard SAT problems with over 6000 variables in QUBO. Based on our results, we discuss the challenges in solving SAT using the current generation of annealing device, and explore the problem solving capability of future quantum annealing computers.

1. INTRODUCTION

Quantum computing has been proven in theory to be more efficient in solving certain classes of optimization problems [1]. In addition, [2] experimentally showed that quantum computer has a much smaller scaling factor when dealing with the problem that has the same complexity on classical computer. Among all the existing quantum heuristics, quantum annealing (QA) has drawn significant attention in recent years due to the realization of the first commercialized QA computer D-Wave. However, designing and implementing algorithms that manage to harness the enormous computational power from QA computer remains a challenging task. Several researches [3] [4] [5] have proposed approaches utilizing the D-Wave annealer; nevertheless, they are far from touching the real world problems. Solving an optimization problem over QA device often requires two procedures: mapping the given optimization problem to the quadratic unconstrained binary optimization (QUBO) problem and embedding the subsequent QUBO to the physical architecture of a quantum annealer.

QA was first proposed as an extension of simulated annealing that is able to avoid local minimal [6]. Later, practical computation framework of QA has been proposed and developed by [7]. It works by starting a physical system in a ground state and gradually varying the system until a solution of the target optimization problem is obtained. The mapping procedure is to describe the problem using the physical system. In recent years, D-Wave quantum annealer has become available [8]. However, its physical architecture creates another obstacle from using the QA. It requires a transformation that embeds the mapped problem to its architecture.

As studied previously in [9], embedding a QUBO to a practical annealer is equivalent to finding the graph minor of the annealer architecture. The same study also proposed a practical heuristic for finding graph minors over a sparse graph, while the best known exact algorithm of finding the graph minor is proposed by [10]. However, these approaches can only handle up to a few hundreds of QUBO variables and are not scalable to large QUBO problems.

Mining the computational power of quantum annealing to solve Boolean Satisfiability (SAT) and its variant problems is always a popular topic. [11] and [12] have discussed the relationship between QUBO and Weighted Maximum 2 Satifiability. [13] proposed QUBO formulation of general SAT problem based on 3-SAT. However, these researches failed to consider the embedding issue for practical architecture.

In this paper, we describe a quantum annealing approach to solve the SAT problem. Our major contributions are threefold.

1. The proposed SAT to QUBO mapping technique does not need to reduce an arbitrary SAT problem to a 3-SAT problem, and it greatly alleviates the pain in the embedding procedure by introducing the Ising primitives.
2. Based on D-Wave architecture, we convert the embedding problem from finding a graph minor to a place-and-route problem, which is similar to integrated circuit design automation. The proposed technique scales well in dealing with real world applications and some hard SAT problems.
3. We present the experimental results for large scale testcases. We discuss how to use the potential computational power of future QA devices.

DAC '16, June 05-09, 2016, Austin, TX, USA

DOI: http://dx.doi.org/10.1145/2897937.2897973

The rest of the paper is organized as follows: Section 2 introduces the computation model of QA computer. Section 3 proposes how to map SAT to QUBO. Section 4 elaborates embed the QUBO to D-Wave architecture. Section 5 presents the experimental results and section 6 concludes.

2. QUANTUM ANNEALING

Quantum annealing offers potential computational power by taking advantage of quantum mechanical effects, including superposition and entanglement. Introduced by [7] , the annealing procedure can be represented as a time-dependent Hamiltonian, as shown in Eq. 1.

$$H(\tau) = A(\tau)H_b + B(\tau)H_p \qquad \tau = t/t_a \tag{1}$$

Initially, the system stays in the ground state that minimizes the system energy specified by the initial Hamiltonian H_b. As long as the system is changed sufficiently slowly, the system will stay in the ground state, which is also known as adiabatic process. We gradually vary the system until it reaches a new ground state minimizing the system energy specified by the final Hamiltonian H_p. Then, the final ground state encodes the solution to minimize the system energy function.

The D-Wave and other practical QA devices comprise a superconducting physical system [8], where the value of each qubit indicates the state of the system. Therefore, the system energy $E(s)$ can be represented by the Ising model.

$$\begin{aligned} E(s) = &\sum_{1\le i\le N} h_i s_i + \sum_{1\le i\le j\le N} J_{ij} s_i s_j \\ &\mid h_i \mid \le 2, \mid J_{ij} \mid \le 1, and\, s_i \in \{+1, -1\} \end{aligned} \tag{2}$$

As shown in Eq. 2, s_i is the state of the qubit, h_i is the bias and J_{ij} is the interaction weight between two qubits. Therefore, any function minimization problem can be solved by a quantum annealer, as long as it can be represented using Ising model. In order to solve the Boolean SAT, we need to describe the problem in the form of minimizing the Ising function, which is referred as mapping in this paper and discussed in the next section. Since J_{ij} is not available between every qubits in practical QA devices, we need to put some extra effort, which is referred as embedding and discussed in Section 4.

Compared with Ising model, QUBO has a slightly different representation. In the Ising model, the variable s_i is defined by $\{-1, +1\}$, whereas in QUBO, the variable x_i is defined by $\{0, 1\}$. The two representations can be interchanged by replacing the variable in the Ising model with $s_i = 1 - 2x_i$ and altering the coefficient accordingly. The energy function in Ising model has the same form as the cost function in QUBO. So Ising model and QUBO are equivalent, we do not distinguish the two representations in the following discussions.

A few researches have discussed the error rate and the trustiness of the solution returned by quantum annealer [2] [4]. We, in this paper, confine our discussion in a scenario that the annealer always returns a correct ground state that minimizes system energy.

3. MAPPING

SAT problem is stated as: given a Boolean function defined over n variables, find an assignment of variables such that the Boolean function is true. Our mapping approach can be divided into 3 steps:

1. Convert the Boolean function in SAT into an acyclic Boolean network with only basic logic operations.
2. Convert the Boolean network to the Ising model by introducing Ising primitives and chains.
3. Set constraints so that Ising model output is logic one. Then find the corresponding assignments.

Since the Boolean function is directly related to Boolean network, it can be easily converted using a naive algorithm. Hence our focus is on the last two steps of the mapping.

3.1 Ising Model for Boolean Function

[14] has proposed expressions of the Ising model for three logic operators. In this paper, we extend the discussion to variant of these three operators. Subsequently, we propose the Ising primitives and chains to represent the given Boolean function in the Ising model.

Table 1: Basic Ising Primitives

N	Logic function	Ising function
1	$z = 0$	$2S_z$
2	$z = x \wedge y$	$-S_x - S_y + 2S_z - 2S_xS_z - 2S_yS_z + S_xS_y$
3	$z = x \wedge \neg y$	$-S_x + S_y + 2S_z - 2S_xS_z + 2S_yS_z - S_xS_y$
4	$z = x$	$-2S_xS_z$
5	$z = \neg x \wedge y$	$+S_x - S_y + 2S_z + 2S_xS_z - 2S_yS_z - S_xS_y$
6	$z = y$	$-2S_yS_z$
7	$z = x \oplus y$	None
8	$z = x \vee y$	$S_x + S_y - 2S_z - 2S_xS_z - 2S_yS_z + S_xS_y$
9	$z = \neg x \wedge \neg y$	$S_x + S_y + 2S_z + 2S_xS_z + 2S_yS_z + S_xS_y$
10	$z = x \leftrightarrow y$	None
11	$z = \neg y$	$2S_yS_z$
12	$z = x \vee \neg y$	$S_x - S_y - 2S_z - 2S_xS_z + 2S_yS_z - S_xS_y$
13	$z = \neg x$	$2S_xS_z$
14	$z = \neg x \vee y$	$-S_x + S_y - 2S_z + 2S_xS_z - 2S_yS_z - S_xS_y$
15	$z = \neg x \vee \neg y$	$-S_x - S_y - 2S_z + 2S_xS_z + 2S_yS_z + S_xS_y$
16	$z = 1$	$-2S_z$

Firstly, we convert the 16 basic logic operations to the Ising function in Table 1, where variable -1 represents the logic zero and +1 represents the logic one. For each Ising primitive, the system energy is minimized when the state of three variables satisfies the corresponding logic relation. Using AND primitive as an example, the Ising function is minimized to -3 if and only if $(s_1, s_2, s_3) \in \{(-1, -1, -1), (-1, 1, -1), (1, -1, -1), (1, 1, 1)\}$.

Secondly, we introduce a chain with strong negative interaction between the two qubits from connecting primitives so that the qubit states in output-input pair are aligned, as shown in Figure 1. Then, we can convert the Boolean network to an Ising model such that the energy of the Ising model is minimized if and only if the state of the qubits is one of the possible states in the Boolean network, as shown in Figure 1. In other words, we are constructing an Ising model that the total energy is minimized when all energies of primitives and chains are minimized.

THEOREM 1. *The total energy of the converted Ising model is minimized if and only if the qubit states of the Ising model satisfy all the logic operations in the acyclic Boolean network.*

PROOF. If one takes possible state from Boolean network and assign it to the Ising model, the total energy is minimized because the energies of all Ising primitives and chains are minimized.

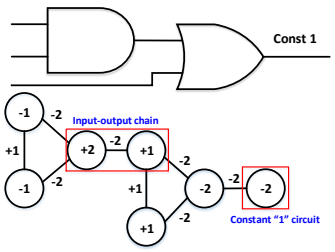

Figure 1: Ising Model for Boolean Network

To prove by contradiction, we assume that the total energy is minimized and one primitive or chain violates the logic relation. Due to the acyclicity, there are always possible states of the Boolean network satisfying all logic operations. By comparing the violated and satisfied scenarios, one can conclude that the violated primitive or chain will have the same energy with the counterpart in the satisfied scenario. At this point, we arrived at contradiction that the energy of a chain or primitive is minimized if and only if the states satisfy the logic relation. □

3.2 Solving the SAT Problem

An intuitive interpretation of the proposed Ising model for Boolean function is that the solutions returned by a quantum annealer encode all the possible states. However, for SAT problem, screening for all possible states of a Boolean network is not enough. We need to further narrow it down to the states that make the Boolean function evaluate to being true. Although it is impossible to preset a logic circuit output to one, to manipulate the output of the Ising model is relatively easy. An intuitive way is to connect the output qubit with an Ising primitive that permanently equals to one, as shown in Figure 1, where the Ising model is in graph representation.

We can determine the satisfiability by implementing the Ising model onto a quantum annealer. If the given Boolean function is satisfiable, the returned state minimizes all the primitives and chains. Otherwise, the annealer still returns a state that minimizes the energy of the entire converted Ising model, where the energy of each individual primitive or chain is not minimized.

The greatest innovation of the proposed QUBO formulation is constructing the Ising model using small Ising primitives and connecting them by chains. Compared to other QUBO formulations that also employ chains [14] [4], our method is simplified because the energies in primitives and chains are separated so that the weight of chains stays constant. Besides, we formulated a gate-level netlist, where each gate is represented by an Ising primitive. Therefore, when embedding it to a physical quantum annealer, we are able to leverage integrated circuit design automation techniques, such as logic optimization, placement and routing.

4. EMBEDDING

Mapping the original problem to QUBO does not guarantee that practical quantum annealer, such as D-Wave, will solve it. One obvious constraint is the limited connectivity on the physical architecture. Since our QUBO comprises Ising primitives and chains, we propose an embedding flow that is similar to FPGA implementation framework. In this section, we demonstrate the embedding flow.

4.1 D-Wave Architecture

The D-Wave quantum annealer arranges its qubits according to the Chimera graph [4]. A Chimera graph consists of M by N cells, each of which consists a $K_{4,4}$ bipartite graph, as shown in Figure 2. Each circle indicates a qubit, whereas, edges between circles represent the available interactions between qubits.

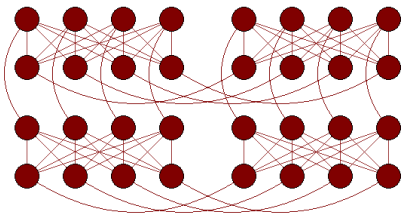

Figure 2: The D-Wave Chimera Architecture

In fact, the $K_{4,4}$ can be converted to a complete K_4 graph, where each logic qubit is represented by two physical qubits combined. This is achieved by imposing a chain between the two physical qubits; hence the two physical qubits are aligned and behave as a single qubit. To equivalently convert back to the physical $K_{4,4}$, we equally assign the bias on the logic qubits over the two physical qubits and distribute the weight on the logic interaction with two physical interactions, as shown in Figure 3.

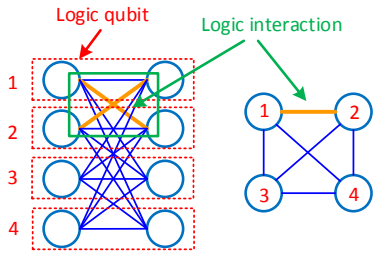

Figure 3: Logic Cell

Recall that the proposed Ising primitives consist only three qubits, we can directly embed the primitives on to a complete K_4 graph.

4.2 Placement

When each Ising primitive is assigned to a cell, a good placement substantially decides the quality and efficiency of the routing result. The objectives of placement are to place the chained primitives close to each other and to avoid future routing congestion.

We adopted the simulated annealing framework to place Ising primitives, as shown in Algorithm 1. According to [15], simulated annealing makes the largest placement improvement when succeed swap rate α is around 0.44. In our placement, the update temperature function is designed so that temperature drops more slowly when α is around 0.44.

The cost function used in our placement is illustrated by Eq. 3. Similar to the placement algorithm for FPGA [16], we used the summation of half perimeter wire length (HPWL) as our cost function, where the compensation coefficient $q(i)$ for multi-fanout net is based on research [17]. To estimate the local congestion within bounding box of each wire, we consider the routing supply by counting the available routing resources. For each mapped cell, only one qubit is available to connect with other qubits by constructing chains. That

Algorithm 1 Simulated Annealing Placement

```
initializePlacement();
initializeTemperature();
while (!shouldExit()) {
    while(!isMoveLimit()) {
        trySwapPlacedCell();
        ΔCost = cost(P_new) - cost(P_old);
        r = random(0, 1);
        if ( r < e^{-ΔCost/T} ) {
            acceptMove();
        } else {
            revertMove();
        }
    }
    T = updateT(α);
}
```

leaves each mapped cell only have one vertical and one horizontal routing resource supply. As for an unmapped cell, each has four vertical and four horizontal routing resource supplies. The vertical routing supply and the horizontal supply are represented by S_v and S_h respectively. By introducing routing supply, the cost function will have smaller value if wires are placed in the area that have more routing resource supply.

$$Cost = \sum_{i \in Nets} q(i)[\frac{bb_x(i)}{S_h(i)} + \frac{bb_y(i)}{S_v(i)}] \quad (3)$$

4.3 Routing

After finishing placed all Ising primitives, the remaining task is to connect them. However, for a large Ising model that consists of many primitives, it can be very hard to successfully route all the wire to the designated location. For example, a previously routed wire may block other wire later. Therefore an efficient routing algorithm is the key to successfully embedded the Ising model.

4.3.1 Routing Graph

Figure 4 shows the mapped and unmapped cell structures to build the routing graph. To handle the symmetric issue for assigning qubits within a cell, we employed virtual MUX to separate the Ising primitives and qubits. Once the routing is finished, we examine the usage of virtual MUX and hence determine the corresponding Ising primitive assignment. Cells not mapped with primitives are called unmapped cells. They are solely used as routing resources.

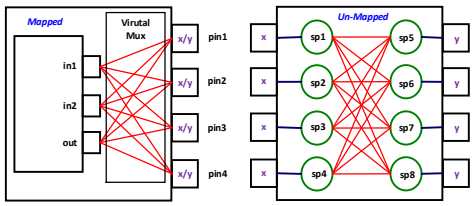

Figure 4: Mapped/Unmapped Cell Structure

A corresponding routing resource graph (RR graph), which facilitates routing, is built based on the post placed Ising model. All pins, spins, inner-cell interactions and inter-cell interactions are represented by nodes in the RR graph and

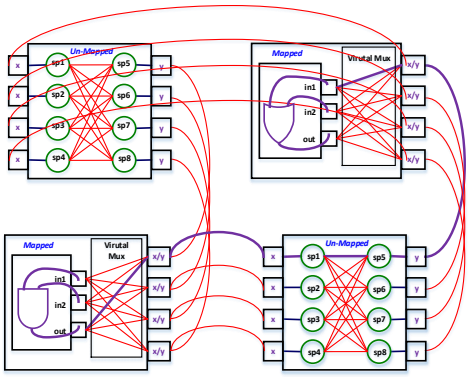

Figure 5: Post Routed Cell

their connections by edges. Determined by the nature of the D-Wave architecture, all edges have no direction except those edges associate with Ising primitive inputs and outputs.

4.3.2 Routing Algorithm

Algorithm 2 Routing Algorithm

```
while ( overused resources exist) { /* illegal routing*/
    annotateAllSlackRatio();
    sortNetsSlackRatio();
    foreach (net i) {
        ripUp(net i);
        foreach (sink j in net i) {
            pushRoutedResources();
            findShortestPath(sink j);
        }
        updateCongestionCost();
    }
    updateHistoricalCost();
}
```

The routing algorithm scheme is described in Algorithm 2. Similar to other negotiation-based routing algorithm like PathFinder [18], the cost function of each node is calculated by Eq. 4, where C_b is the base cost, C_h is the historical cost, C_c is the current congestion cost and sr is the slack ratio of a wire. The base cost is a constant. Current congestion cost is calculated based on the current usage of the routing resource. The calculation of historical cost takes into account the historical congestion, in order to avoid swing between routing iterations. The slack ratio indicates the importance of the wire. The longer the wire, the more important it is. In calculating the cost, slack ratio plays trade-off between the base cost and other cost. The rationale behind is to route the important wires with shorter path while detour those less important ones.

$$Cost_n = (1 - \sqrt{sr}) * C_b + \sqrt{sr} * C_b * C_h * C_c \quad (4)$$

4.3.3 Routing Tune-up

To save routing runtime, we limit our routing exploration to bounding box and its vicinity. By limiting the exploration space, the router maintains a good quality routing result and save unnecessary exploration runtime.

Furthermore, we encourage reuse the partial tree when routing multi-fanout wire, because it saves routing resources.

This is achieved by adding a push routed resources before the shortest path algorithm, ensuring the costs of routed resources are set to zero.

4.4 Configuration

Based on the placement and routing result, the final step is to generate the configuration to program the weight of qubits and interactions. This is achieved by first assigning Ising primitive configurations to each cell with logic qubits and interactions. Then, configuration of physical Ising model is generated by converting the logic cells to physical cells and combining the routing result. Figure 6 shows a configuration map of a majority voter.

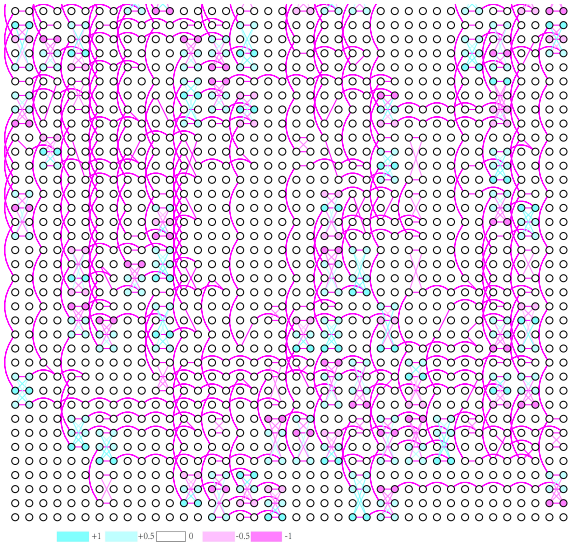

Figure 6: The Configuration of a Majority Voter

5. EXPERIMENTS

In this section, we report results of our experiments using the proposed QUBO mapping approach and tool flow. We chose the SAT problem testcases from SAT competition and logic circuit benchmark, which have sophisticated Boolean network used in real applications.

Our flow starts with converting SAT testcases in CNF and circuit benchmarks to BLIF format, which is a Boolean network format used in academic world. We used ABC synthesis tool to optimize the converted BLIF ensuring that every logic operator is binary. Then we applied the proposed flow to map and embed the optimized Boolean network.

Our experiments were ran on a server with Xeon E3-1245L CPU and 8-Gigabyte memory. Figure 7 and 8 show the succeeded testcases embedded to the 12 x 12 cells architecture same to the capacity of the latest D-Wave 2X system. Some of these testcases are in the large circuit class in the MCNC benchmark suite showing that D-Wave has the potential in solving real world problems. The proposed technique is extremely efficient for the current generation device, as all testcases are finished in less than a second. Still, the capacity of qubits and their interactions limit the capability of D-Wave to solve larger problems.

To evaluate the scalability of the proposed approach, we imagined a D-Wave architecture annealer device with 100 x 100 cells, which is almost hundred times of the latest D-Wave system. We successfully managed to embed some hard SAT problems and complicated logic circuits into this device. The results presented in Table 2 are the 20 largest ones among all succeeded testcases. Runtime indicates the proposed technique scales well for future devices.

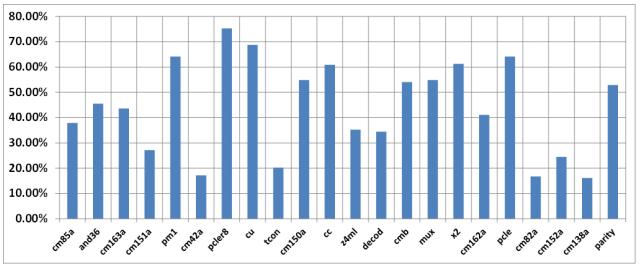

Figure 7: Utilization Rate on 12 x 12 Cells

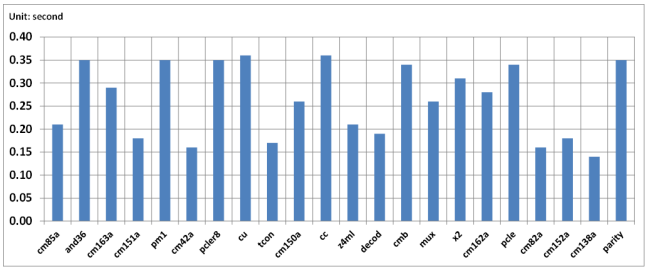

Figure 8: Runtime for 12 x 12 Cells

We separately analyzed the runtime increase caused by placement and routing. For placement, one necessary step is to update the cost, which is calculated based on the HPWL and the number of used cells within the bounding boxes, requiring information update from all the nets. Therefore, increase either in device size or the update frequency would increase placement runtime. For routing procedure, as the device grows larger, the router tends to explore more nodes in a large routing graph. Moreover, for more complicated designs, the router needs more iterations to find valid path leading to cost inflation, further slowing down the router. Our solution to decrease the placement runtime is reduce the update frequency, while for routing, we can adjust exploration range and historical cost coefficient.

The succeeded testcases also give us a glimpse of the problems can be solved by future devices. *sgen6* series testcases are originated from SAT Competition 2014 [19], which are not solved in the SAT competition. Other than that, many applications such as equivalent checking and test generation can be derived from logic testcases. For these applications, people have been long searching for efficient methods providing high quality results. The current generation of D-Wave is able to return a valid result in millisecond level, [2] also showed that quantum computer is much more scalable compare to classical computer, our experimental results further validating the great potential of quantum annealer to become an efficient tool in solving SAT related problems.

Overall, we have successfully embedded over 100 testcases, showing that our approach scales properly for the current generation and even for the future devices.

6. CONCLUSIONS

We have presented a complete tool flow that first maps the Boolean SAT problem to QUBO problem and then embeds it into the practical quantum annealer. By leveraging

Table 2: Experiments on Architecture with 100 x 100 Cells

Name	Primitive#	Chain#	RouteIter#	WireLength	CellUsage	InterUsage	TotalUsage	RunTime (s)
C6288	2370	2432	22	68366	11.88%	54.26%	66.14%	704.31
C5315	1775	2141	10	42438	8.89%	32.92%	41.82%	376.51
pair	1574	1918	10	37268	7.89%	28.91%	36.80%	344.62
dalu	1361	1509	16	41730	6.82%	33.29%	40.11%	309.39
frg2	1131	1421	10	28422	5.66%	22.09%	27.75%	202.16
C3540	1031	1129	15	31944	5.16%	25.51%	30.67%	222.75
i7	865	1261	9	13032	4.33%	9.40%	13.73%	119.66
sgen6-960-5-1	889	1207	41	37200	4.44%	29.69%	34.13%	2391.32
edges-072-3-7923777-13	782	1206	12	21366	3.75%	15.74%	19.49%	1077.08
x3	857	1125	8	18098	4.29%	13.77%	18.06%	106.61
edges-070-3-1250111-33	760	1172	10	20036	3.64%	14.71%	18.36%	1613.07
apex6	786	1054	7	15678	3.93%	11.79%	15.72%	98.5
sgen6-840-5-1	775	1053	51	28074	3.87%	22.24%	26.11%	1672.47
i9	736	910	9	17166	3.68%	13.24%	16.92%	92.56
i6	683	957	6	10744	3.42%	7.84%	11.26%	64.43
rot	661	931	9	11894	3.31%	8.69%	11.99%	77.79
too_large	726	800	10	22766	3.63%	18.21%	21.84%	99.86
alu4	717	743	21	26076	3.59%	21.07%	24.65%	186.44
edges-025-4-10062999-1-00	519	719	8	12106	2.48%	8.89%	11.37%	835.36
i2	418	818	8	4778	2.09%	2.97%	5.05%	36.23

and adapting the integrated circuit design automation techniques, we have successfully converted the embedding problem into a place-and-route problem. We have tested over 100 testcases that can be successfully implemented onto a D-Wave architecture quantum annealer in reasonable amount of time. Our approach also enables us to implement some hard SAT problems that there is no existing solver can solve. In conclusion, we greatly enhanced the SAT problem solving ability of quantum annealer by making the breakthrough beyond architecture limits.

As for future work, we would like to further reduce the number of used qubits by developing a logic optimization algorithm that works exclusively for D-Wave architecture. Furthermore, we would also like to investigate the use of analytical placement algorithm to reduce runtime and improve placement quality.

7. REFERENCES

[1] A. M. Childs and W. van Dam, "Quantum algorithms for algebraic problems," *Reviews of Modern Physics*, vol. 82, no. 1, pp. 1–52, Jan. 2010.

[2] S. Boixo, T. F. RÃÿnnow, S. V. Isakov, Z. Wang, D. Wecker, D. A. Lidar, J. M. Martinis, and M. Troyer, "Quantum annealing with more than one hundred qubits," *Nature Physics*, vol. 10, no. 3, pp. 218–224, Feb. 2014, arXiv: 1304.4595.

[3] A. Perdomo-Ortiz, J. Fluegemann, S. Narasimhan, R. Biswas, and V. N. Smelyanskiy, "A Quantum Annealing Approach for Fault Detection and Diagnosis of Graph-Based Systems," *The European Physical Journal Special Topics*, vol. 224, no. 1, pp. 131–148, Feb. 2015, arXiv: 1406.7601.

[4] Z. Bian, F. Chudak, R. Israel, B. Lackey, W. G. Macready, and A. Roy, "Discrete optimization using quantum annealing on sparse Ising models," *Interdisciplinary Physics*, vol. 2, p. 56, 2014.

[5] A. Perdomo-Ortiz, N. Dickson, M. Drew-Brook, G. Rose, and A. Aspuru-Guzik, "Finding low-energy conformations of lattice protein models by quantum annealing," *arXiv:1204.5485 [quant-ph]*, Apr. 2012, arXiv: 1204.5485.

[6] A. B. Finnila, M. A. Gomez, C. Sebenik, C. Stenson, and J. D. Doll, "Quantum Annealing: A New Method for Minimizing Multidimensional Functions," *Chemical Physics Letters*, vol. 219, no. 5-6, pp. 343–348, Mar. 1994, arXiv: chem-ph/9404003.

[7] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, "A Quantum Adiabatic Evolution Algorithm Applied to Random Instances of an NP-Complete Problem," *Science*, vol. 292, no. 5516, pp. 472–475, Apr. 2001.

[8] R. Harris, M. W. Johnson, T. Lanting, A. J. Berkley, J. Johansson, P. Bunyk, E. Tolkacheva, E. Ladizinsky, N. Ladizinsky, T. Oh, F. Cioata, I. Perminov, P. Spear, C. Enderud, C. Rich, S. Uchaikin, M. C. Thom, E. M. Chapple, J. Wang, B. Wilson, M. H. S. Amin, N. Dickson, K. Karimi, B. Macready, C. J. S. Truncik, and G. Rose, "Experimental Investigation of an Eight Qubit Unit Cell in a Superconducting Optimization Processor," *Physical Review B*, vol. 82, no. 2, Jul. 2010, arXiv: 1004.1628.

[9] J. Cai, W. G. Macready, and A. Roy, "A practical heuristic for finding graph minors," *arXiv:1406.2741 [quant-ph]*, Jun. 2014, arXiv: 1406.2741.

[10] I. Adler, F. Dorn, F. V. Fomin, I. Sau, and D. M. Thilikos, "Faster parameterized algorithms for minor containment," *Theoretical Computer Science*, vol. 412, no. 50, pp. 7018–7028, Nov. 2011.

[11] C. C. McGeoch, "Adiabatic Quantum Computation and Quantum Annealing: Theory and Practice," *Synthesis Lectures on Quantum Computing*, vol. 5, no. 2, pp. 1–93, Jul. 2014.

[12] D. A. Battaglia, G. E. Santoro, and E. Tosatti, "Optimization by quantum annealing: Lessons from hard satisfiability problems," *Physical Review E*, vol. 71, no. 6, p. 066707, Jun. 2005.

[13] I. Hen and A. P. Young, "Exponential Complexity of the Quantum Adiabatic Algorithm for certain Satisfiability Problems," *Physical Review E*, vol. 84, no. 6, Dec. 2011, arXiv: 1109.6872.

[14] W. M. Zhengbing Bian, Fabian Chudak and G. Rose, "The ising model: teaching an old problem new tricks," *D-Wave Systems Technical Report*, Aug. 2010.

[15] J. Lam and J.-M. Delosme, "Performance of a new annealing schedule," in *, 25th ACM/IEEE Design Automation Conference, 1988. Proceedings*, Jun. 1988, pp. 306–311.

[16] A. Marquardt, V. Betz, and J. Rose, "Timing-driven Placement for FPGAs," in *Proceedings of the 2000 ACM/SIGDA Eighth International Symposium on Field Programmable Gate Arrays*, ser. FPGA '00. New York, NY, USA: ACM, 2000, pp. 203–213.

[17] C.-l. E. Cheng, "Risa: Accurate And Efficient Placement Routability Modeling," in *, IEEE/ACM International Conference on Computer-Aided Design, 1994*, Nov. 1994, pp. 690–695.

[18] L. McMurchie and C. Ebeling, "PathFinder: A Negotiation-Based Performance-Driven Router for FPGAs," in *Proceedings of the Third International ACM Symposium on Field-Programmable Gate Arrays, 1995. FPGA '95*, 1995, pp. 111–117.

[19] A. Belov, D. Diepold, M. J. H. Heule, and M. JÃd'rvisalo, "Proceedings of SAT Competition 2014," 2014.