Lightsolver challenges a leading deep learning solver for Max-2-SAT problems

Hod Wirzberger[1], Assaf Kalinski[1], Idan Meirzada[1], Harel Primack[1], Yaniv Romano[1,2], Chene Tradonsky[1], and Ruti Ben Shlomi[1]

[1]LightSolver LTD., Tel Aviv, Israel
[2]Departments of Electrical and Computer Engineering and Computer Science, Technion IIT, Israel

Abstract

Maximum 2-satisfiability (MAX-2-SAT) is a type of combinatorial decision problem that is known to be NP-hard. In this paper, we compare LightSolver's quantum-inspired algorithm to a leading deep-learning solver for the MAX-2-SAT problem. Experiments on benchmark data sets show that LightSolver achieves significantly smaller time-to-optimal-solution compared to a state-of-the-art deep-learning algorithm, where the gain in performance tends to increase with the problem size.

1 Introduction

A wide range of real-world problems with substantial societal, economic, and scientific implications can be posed as combinatorial optimization tasks. Advances in combinatorial optimization have led to more efficient transportation systems, supply chains, resource management, and more [1, 2, 3, 4, 5]. In this work, we consider the classic maximum 2-satisfiability (`MAX-2-SAT`) problem [6], which is ubiquitous in scheduling or resource allocation tasks, to name a few applications [7].

Suppose we are given a set of N binary variables $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)$ and a set of C constraints (or clauses) with two variables per clause that form a Boolean formula $F(\boldsymbol{x})$. Our goal is to assign a binary value to each variable $\boldsymbol{x}_i$ such that the maximal number of the clauses are satisfied. The Boolean formula $F(\boldsymbol{x})$ we consider takes a conjunctive normal form, consisting of a conjunction (logical AND) of clauses, where each clause is a disjunction (logical OR) of literals. For example, the formula

$$F(\boldsymbol{x}) = (\boldsymbol{x}_1 \vee \boldsymbol{x}_2) \wedge (\neg \boldsymbol{x}_1 \vee \boldsymbol{x}_2) \wedge (\neg \boldsymbol{x}_1 \vee \neg \boldsymbol{x}_2) \tag{1}$$

has $N = 2$ variables and $C = 3$ clauses. The notations $\wedge$, $\vee$, and $\neg$ stand for the logical AND, OR, and negation operators, respectively. While there is no assignment of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ that satisfy the specific formula given above, we can make two out of three clauses true by assigning $\boldsymbol{x}_1 = 0$ and $\boldsymbol{x}_2 = 1$. Hence, the optimal `MAX-2-SAT` solution for this example is 2 clauses.

The `MAX-2-SAT` problem is known to be NP-hard [8], implying that exact solutions can be merely obtained for relatively small problems; the computational complexity rapidly increases with the number of variables [9, 10]. Yet, there exist computationally efficient approximation algorithms that are capable of finding useful solutions that are not necessarily the optimal ones. More traditional methods for tackling constraint satisfaction problems include combinatorial constraint propagation algorithms, logic programming techniques, and semi-definite programming [11, 12, 13, 9, 14]. A different line of work harnesses modern machine and deep learning algorithms, such as graph neural networks, to achieve high-quality solutions in amendable time complexity [15, 16, 17, 18, 19], leveraging the impressive computing power of graphical processing units (GPUs). Another approach to tackle the `MAX-2-SAT` problem is via quantum or quantum-inspired computing platforms [20, 21, 22]. A typical approach here is to formulate `MAX-2-SAT` as a quadratic unconstrained binary optimization (QUBO) problem [23, 24], which, in principle, can be efficiently solved on quantum computers, annealers, or simulators [25, 26, 22].

In this paper, we compare the performance of a leading off-the-shelf deep learning solver for the `MAX-2-SAT` problem, called RUN-CSP [15], to the QUBO solver of LightSolver [27, 28]: a quantum-inspired digital computing platform that simulates its all-optical devices. In our experiments, we make two

important design choices to ensure a fair comparison. First, both RUN-CSP and LightSolver algorithms are executed on the very same GPU computing resources. Second, we use the `MAX-2-SAT` benchmark data sets included in the official software package of RUN-CSP [15].[1] This experimental protocol allows us to apply RUN-CSP with the training scheme and set of hyper-parameters that the authors chose to achieve the best performance. Importantly, we evaluate the performance of the two methods by comparing their time-to-exact-solution, where the runtime does not include the training phase of RUN-CSP or the preprocessing required to formulate the QUBO problem in the case of LightSolver's algorithm. Our experiments reveal that (i) the average time-to-exact-solution obtained by LightSolver's algorithm is 2X to 1000X faster than that of RUN-CSP across all the data sets examined; and (ii) the gain in performance tends to increase with the size of the problem, which is defined by the number of variables N.

2 Methods

2.1 Benchmark: RUN-CSP

The RUN-CSP algorithm [15] is a generic deep-learning-based solver for maximum binary constraint satisfaction problems. At a high level, the idea is to train a recurrent graph neural network on a set of instances of the constraint satisfaction problem of interest by minimizing a loss function that rewards solutions that satisfy a large number of constraints. The update of the network parameters can be thought of as a message-passing algorithm. Importantly, once the training is complete, the fitted model can be applied to new test instances of arbitrary size as the network parameters are shared across all variables. Interestingly, RUN-CSP was shown to outperform state-of-the-art heuristic algorithms for solving the `MAX-2-SAT` problem, motivating our desire to compare LightSolver's algorithm to RUN-CSP.

2.2 Proposed: LightSolver

To solve the `MAX-2-SAT` problem on LightSolver's platform we formulate this task as a QUBO problem, which, in general, is given by [24]

$$\min_{\boldsymbol{x}\in\{0,1\}^N} \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}. \tag{2}$$

Above, $\boldsymbol{Q}\in\mathbb{R}^{N\times N}$ is a QUBO matrix and $\boldsymbol{x}$ is the vector of binary unknowns—the literals. To translate the Boolean function of interest $F(\boldsymbol{x})$ into the above quadratic form, we observe that, in general, there are three possible clause types that should be mapped to their corresponding quadratic forms, as summarized below [24].

Clause	Quadradic Mapping
$\boldsymbol{x}_i \vee \boldsymbol{x}_j$	$1-\boldsymbol{x}_i-\boldsymbol{x}_j+\boldsymbol{x}_i\boldsymbol{x}_j$
$\neg\boldsymbol{x}_i \vee \boldsymbol{x}_j$	$\boldsymbol{x}_i-\boldsymbol{x}_i\boldsymbol{x}_j$
$\neg\boldsymbol{x}_i \vee \neg\boldsymbol{x}_j$	$\boldsymbol{x}_i\boldsymbol{x}_j$

Notably, the mapping is defined such that when a clause is satisfied the value of the quadratic term is 0. By contrast, when the clause is not met the quadratic mapping is equal to 1. Lastly, we sum the quadratic mappings of all clause terms and get the final QUBO problem. In our running example, the formula (1) can be mapped into the following objective:

$$\min_{\boldsymbol{x}_1,\boldsymbol{x}_2\in\{0,1\}} (1-\boldsymbol{x}_1-\boldsymbol{x}_2+\boldsymbol{x}_1\boldsymbol{x}_2)+(\boldsymbol{x}_1-\boldsymbol{x}_1\boldsymbol{x}_2)+(\boldsymbol{x}_1\boldsymbol{x}_2).$$

Observe that $\boldsymbol{x}_i=\boldsymbol{x}_i\boldsymbol{x}_i$ since $\boldsymbol{x}_i$ is binary, and therefore the above objective perfectly fits into (2)—it is nothing but a sum of quadratic elements. One attractive feature of the above formulation is that the matrix $\boldsymbol{Q}$ is of size $N\times N$, regardless of the number of clauses [24].

2.3 Experiments

We follow [15] and compare RUN-CSP to LightSolver on the benchmark instances from the unweighted track of the Max-Sat Evaluation 2016 [29], for which the optimal solution of each problem is known. The number of variables and clauses of the different benchmark instances are summarized in Table 1.

[1] Code is available online at https://github.com/RUNCSP/RUN-CSP

Instance name	Variables N	Clauses C
t3pm3	27	162
t4pm3	64	384
t5pm3	125	750
t6pm3	216	1269
t7pm3	343	2058

Table 1: Properties of the benchmark instances of the unweighted track of the Max-Sat Evaluation 2016.

The evaluation metric we used is 'time-to-optimal-solution' defined as [27, 30]

$$\text{Time-to-optimal-solution} = \arg\min_{t>0} \; t \cdot \frac{\ln(1-0.99)}{\ln(1-p_i(t))},$$

where t is the simulation time, and $p_i(t)$ is the empirical probability of reaching an optimal solution. In plain words, we report the smallest simulation time required to obtain an optimal solution at least once, with 99% probability. For RUN-CSP, we swept over a set of possible hyper-parameters and reported the choice that achieved the best time-to-optimal-solution.

The results are summarized in Figure 1, where both RUN-CSP and LightSolver are evaluated on NVIDIA T4 GPU on the AWS platform. As portrayed, LightSolver consistently achieves the optimal solution, with significantly smaller execution times for larger problem sizes. For the largest problem, RUN-CSP did not manage to reach the optimal solution after 400 seconds, converging to the first excited state at best (white marker).

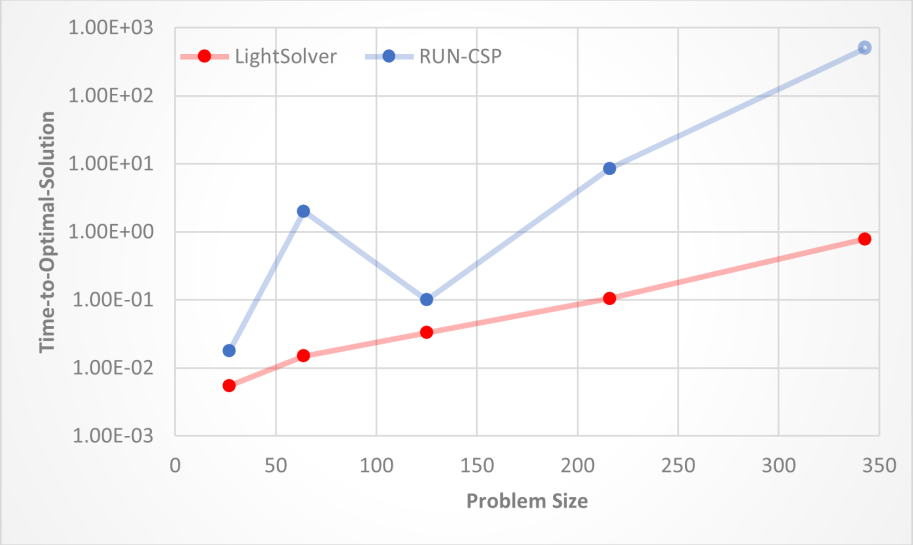

Figure 1: Comparison between the time-to-optimal-solution of LightSolver and RUN-CSP algorithms, evaluated on benchmark `MAX-2-SAT` problems.

Conclusions

In this work, we compared LightSolver quantum-inspired algorithm to a leading deep learning approach for solving the NP-hard `MAX-2-SAT` problem. Deep learning networks are a popular strategy for facing complex optimization problems due to their wide application potential. However, they require a costly training step prior to providing solutions. Quantum and quantum-inspired solvers, on the other hand, do not require a training stage and thus can function as an on-demand resource. Our experiments show that the LightSolver simulator achieves 2X-1000X faster time-to-optimal-solution for solving the Max-2-SAT problems compared to the deep learning solver, for problems of small to medium scale. This result further supports the immense potential of quantum computers and quantum-inspired platforms to yield better solutions for challenging combinatorial optimization problems.

References

[1] Vangelis Th Paschos. *Applications of combinatorial optimization*, volume 3. John Wiley & Sons, 2014.

[2] Gita Naseri and Mattheos AG Koffas. Application of combinatorial optimization strategies in synthetic biology. *Nature communications*, 11(1):1–14, 2020.

[3] Michael Fortun and Silvan S Schweber. Scientists and the legacy of world war II: The case of operations research (OR). *Social studies of science*, 23(4):595–642, 1993.

[4] Thomas L Magnanti. Combinatorial optimization and vehicle fleet planning: Perspectives and prospects. *Networks*, 11(2):179–213, 1981.

[5] Harvey J Greenberg, William E Hart, and Giuseppe Lancia. Opportunities for combinatorial optimization in computational biology. *INFORMS Journal on Computing*, 16(3):211–231, 2004.

[6] Marc Mezard and Andrea Montanari. *Information, physics, and computation*. Oxford University Press, 2009.

[7] Dingzhu Du, Jun Gu, and Panos M Pardalos. *Satisfiability problem: theory and applications*, volume 35. American Mathematical Soc., 1997.

[8] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[9] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable CSPs? *SIAM Journal on Computing*, 37(1):319–357, 2007.

[10] Sanjeev Khanna, Madhu Sudan, Luca Trevisan, and David P Williamson. The approximability of constraint satisfaction problems. *SIAM Journal on Computing*, 30(6):1863–1920, 2001.

[11] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2-3):357–365, 2005.

[12] Chuan Luo, Shaowei Cai, Wei Wu, Zhong Jie, and Kaile Su. CCLS: an efficient local search algorithm for weighted maximum satisfiability. *IEEE Transactions on Computers*, 64(7):1830–1843, 2014.

[13] Javier Larrosa, Federico Heras, and Simon De Givry. A logical approach to efficient Max-SAT solving. *Artificial Intelligence*, 172(2-3):204–233, 2008.

[14] Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 245–254, 2008.

[15] Jan Toenshoff, Martin Ritzert, Hinrikus Wolf, and Martin Grohe. Graph neural networks for maximum constraint satisfaction. *Frontiers in artificial intelligence*, 3:580607, 2021.

[16] Henrique Lemos, Marcelo Prates, Pedro Avelar, and Luis Lamb. Graph colouring meets deep learning: Effective graph neural network models for combinatorial problems. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 879–885. IEEE, 2019.

[17] Marcelo Prates, Pedro HC Avelar, Henrique Lemos, Luis C Lamb, and Moshe Y Vardi. Learning to solve np-complete problems: A graph neural network for decision TSP. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4731–4738, 2019.

[18] Daniel Selsam, Matthew Lamm, B Benedikt, Percy Liang, Leonardo de Moura, David L Dill, et al. Learning a SAT solver from single-bit supervision. In *International Conference on Learning Representations*, 2018.

[19] Zhuwen Li, Qifeng Chen, and Vladlen Koltun. Combinatorial optimization with graph convolutional networks and guided tree search. *Advances in neural information processing systems*, 31, 2018.

[20] Siddhartha Santra, Gregory Quiroz, Greg Ver Steeg, and Daniel A Lidar. MAX 2-SAT with up to 108 qubits. *New Journal of Physics*, 16(4):045006, 2014.

[21] Sergey Bravyi. Efficient algorithm for a quantum analogue of 2-SAT. *Contemporary Mathematics*, 536:33–48, 2011.

[22] Puya Mirkarimi, Adam Callison, Lewis Light, Nicholas Chancellor, and Viv Kendon. Comparing the hardness of MAX 2-SAT problem instances for quantum and classical algorithms. *arXiv preprint arXiv:2206.06876*, 2022.

[23] Gary Kochenberger, Fred Glover, Bahram Alidaee, and Karen Lewis. Using the unconstrained quadratic program to model and solve Max 2-SAT problems. *International Journal of Operational Research*, 1(1-2):89–100, 2005.

[24] Fred Glover, Gary Kochenberger, and Yu Du. A tutorial on formulating and using QUBO models. *arXiv preprint arXiv:1811.11538*, 2018.

[25] Zhengbing Bian, Fabian Chudak, William Macready, Aidan Roy, Roberto Sebastiani, and Stefano Varotti. Solving SAT (and MaxSAT) with a quantum annealer: Foundations, encodings, and preliminary results. *Information and computation*, 275:104609, 2020.

[26] Hiroki Oshiyama and Masayuki Ohzeki. Benchmark of quantum-inspired heuristic solvers for quadratic unconstrained binary optimization. *Scientific reports*, 12(1):1–10, 2022.

[27] Idan Meirzada, Assaf Kalinski, Dov Furman, Tsafrir Armon, Talya Vaknin, Harel Primack, Chene Tradonsky, and Ruti Ben-Shlomi. Lightsolver–a new quantum-inspired solver cracks the 3-regular 3-XORSAT challenge. *arXiv preprint arXiv:2207.09517*, 2022.

[28] Yaniv Romano, Harel Primack, Talya Vaknin, Idan Meirzada, Ilan Karpas, Dov Furman, Chene Tradonsky, and Ruti Ben Shlomi. Quantum sparse coding. *arXiv preprint arXiv:2209.03788*, 2022.

[29] Fahiem Bacchus, Jeremias Berg, Matti Järvisalo, and Rubens Martins. MaxSAT evaluation 2020: Solver and benchmark descriptions. 2020.

[30] Matthew Kowalsky, Tameem Albash, Itay Hen, and Daniel A Lidar. 3-regular three-XORSAT planted solutions benchmark of classical and quantum heuristic optimizers. *Quantum Science and Technology*, 7(2):025008, 2022.