nature machine intelligence

Matters arising

https://doi.org/10.1038/s42256-022-00589-y

Modern graph neural networks do worse than classical greedy algorithms in solving combinatorial optimization problems like maximum independent set

Received: 1 July 2022

Accepted: 8 November 2022

Published online: 30 December 2022

Check for updates

Maria Chiara Angelini [1,2] & Federico Ricci-Tersenghi[1,2,3]

ARISING FROM: Martin J. A. Schuetz et al. *Nature Machine Intelligence* https://doi.org/10.1038/s42256-022-00468-6 (2022).

The recent work by Schuetz et al.[1] 'Combinatorial optimization with physics-inspired graph neural networks' introduces a physics-inspired unsupervised graph neural network (GNN) to solve combinatorial optimization problems on sparse graphs. To test the performances of these GNNs, the authors show numerical results for two fundamental problems: the maximum cut and the maximum independent set (MIS), concluding "that the graph neural network optimizer performs on par or outperforms existing solvers, with the ability to scale beyond the state of the art to problems with millions of variables". Here we show that a simple greedy algorithm, running in almost linear time, can find solutions for the MIS problem of much better quality than the GNN in a much shorter time. In general, many claims of superiority of neural networks in solving combinatorial problems are at risk of being not solid enough, as we lack standard benchmarks based on really hard problems. We propose one of such hard benchmarks, and we hope to see future neural network optimizers tested on these problems before any claim of superiority is made.

Recent years have seen an incredible increase in the use of neural networks to solve all kinds of problems, both in applications and in fundamental science; discrete combinatorial optimization problems make no exception[2–4] and are extremely relevant, given that they are often at the basis of our understanding of the fundamental computational limits.

For these reasons, the proposal made by Schuetz et al.[1] to use physics-inspired unsupervised GNNs to solve combinatorial optimization problems defined on a graph seemed very promising and got published in a journal with high impact. The authors tested the performance of the GNN on two standard optimization problems: the maximum cut and the MIS. A very good property of this newly introduced GNN optimizer is that it can scale to problem instances much larger than what many previous deep-learning approaches could handle[5–7].

Let us focus on the MIS problem, which is defined as follows. Given an undirected random regular graph of fixed degree d (d-RRG) with n nodes, an independent set (IS) is a subset of vertices not containing any pair of nearest neighbours. The MIS problem then requires finding the largest IS, whose size is called α. The distribution of the MIS density α/n among different d-RRG concentrates for large n to a value $\rho(d)$, which depends only on the degree d. The MIS is an NP-hard problem; however, one can hope to find an algorithm that finds in polynomial time an IS whose size is as close as possible to the maximal one. Moreover, the performances of a good algorithm should not degrade for larger n.

The GNN of ref. 1 can find the IS for graphs of very large sizes ($n \leq 10^6$): the runtime is proportional to a small power of the problem size, $t \approx n^{1.7}$, and the performances are stable with n (this is highly non-trivial, looking at previous neural network approaches to optimization problems[5–7]). The size of the IS found by the GNN and the running times are reported with open symbols in Fig. 1.

Problems in ref. 1 arise when comparing the GNN performances with other available algorithms and thus claiming the superiority of the approach based on GNN. Schuetz et al.[1] consider only the Boppana–Halldorsson (BH) approximated algorithm[8] that shows a runtime scaling as $t \approx n^{2.9}$ in the range $n \leq 500$.

Comparison with simple greedy algorithms

However, there exist many other algorithms for computing ISs, that work much faster than BH, and the new GNN optimizer should be compared with these too.

The simplest possible algorithm that one can design to solve the MIS problem is the greedy algorithm (GA)[9], which takes a time to reach a solution that is linear in the problem size n. GA works as follows. It starts from an empty IS and G^0 being the original graph. At each step t, a node is chosen at random from the graph G^t and added to the IS.

[1]Dipartimento di Fisica, Sapienza Università di Roma, Rome, Italy. [2]Istituto Nazionale di Fisica Nucleare, Rome, Italy. [3]Institute of Nanotechnology (NANOTEC) - CNR, Rome unit, Rome, Italy. e-mail: maria.chiara.angelini@roma1.infn.it;

https://doi.org/10.1038/s42256-022-00589-y

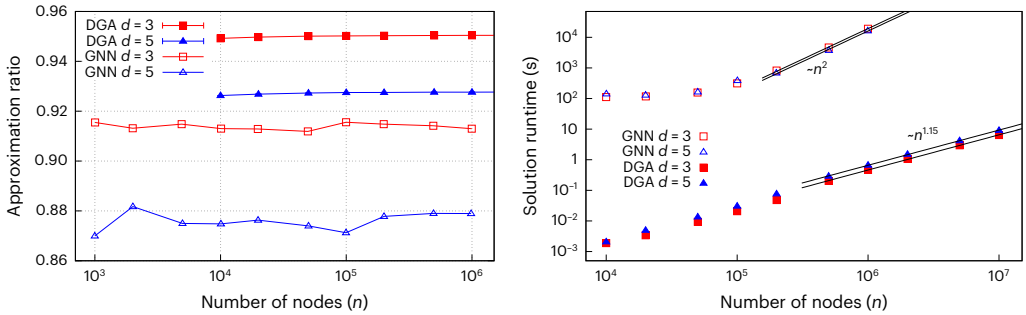

Fig. 1 | Comparison of the GNN and a simple DGA in computing ISs in *d*-RRGs. Left: the size of the IS found by the DGA (full symbols) and by the GNN of ref. 1 (empty symbols), reported as an approximation ratio with respect to the known theoretical upper bounds. The DGA finds much better solutions than the GNN. Right: running time in seconds for the DGA (full symbols) and the GNN (empty symbols). The DGA scaling is much better than the one for GNN. At $n = 10^6$, the speed-up of the DGA is larger than a factor 10^4 with respect to the GNN.

All its neighbours are removed from the graph G^t, thus getting a new graph G^{t+1}. At the time t^* such that G^{t^*} is empty, the process stops.

One can also construct an improved version of the GA by exploiting the degrees of the nodes. In the degree-based GA (DGA), at each step, one chooses the node with the smallest degree in G^t (ref. 10). This algorithm runs in a time almost linear in n if the degree-based ordering of the nodes is managed smartly.

In Fig. 1, we compare the performances of the GNN of ref. 1 (empty symbols) and the DGA (full symbols) in finding MIS on d-RRG with $d = 3, 5$. In the left panel, we report the quality of the solutions found by the two algorithms, shown as the approximation ratio (AR) to the current best upper bounds $\rho_{\rm UB}(d = 3) = 0.45537$ and $\rho_{\rm UB}(d = 5) = 0.38443$ (ref. 11). These bounds are likely to be not strict as the replica method provides an optimal AR smaller than 1, namely $\mathrm{AR}_{\rm 1RSB}(d = 3) \approx 0.990$ and $\mathrm{AR}_{\rm 1RSB}(d = 5) \approx 0.987$ (ref. 12). The DGA clearly outperforms the GNN of ref. 1, especially in the case $d = 5$, where the ISs found by the DGA are 6% larger than those found by the GNN.

In the right panel, we show the running times for both algorithms: data for the DGA have been collected by running on a 2.3 GHz MacBook Pro while data for the GNN have been extracted from Fig. 5 in ref. 1. The latter correspond to the aggregated runtime that includes the post-processing, because the authors furnish no additional information on the time needed for the different steps of the computation. However, the post-processing time to check whether the output configuration is an IS should be linear in n, so most of the time should be dedicated to the GNN computation. The scaling of the running times with the problem size is much better for the DGA than for the GNN, with the former being almost linear in n (the exponent 1.15 is probably due to pre-asymptotic effects), whereas the last data points for the GNN scale close to quadratically in n. Not only is the scaling better for the DGA but also the actual running times are orders of magnitude faster. For example, for $n = 10^6$, the DGA shows a speed-up with respect to the GNN by a factor larger than 10^4.

We thus think that the claim in ref. 1 "We find that the graph neural network optimizer performs on par or outperforms existing solvers, with the ability to scale beyond the state of the art to problems with millions of variables." is not supported by the data shown here and should be modified.

Discussion

We have reported in detail the performances of the DGA because we believe that such a simple GA should be considered as a minimal benchmark and any new algorithm must perform at least better than the DGA to be taken into serious consideration.

But the DGA is not the end of the story. There exist many other standard algorithms that do better than the DGA. A thorough study of the performances of these algorithms to solve the MIS problem can be found in ref. 13. For completeness, we just report here the AR achieved by some of them (addressing the reader to ref. 13 for further details). Both simulated annealing and parallel tempering, two very effective algorithms based on Markov chain Monte Carlo (MCMC), reach $\mathrm{AR}_{\rm MCMC}(d = 3) \approx 0.984$ and $\mathrm{AR}_{\rm MCMC}(d = 5) \approx 0.981$. While belief propagation with reinforcement, a very effective message passing algorithm, can reach $\mathrm{AR}_{\rm BPR}(d = 3) \approx 0.987$ and $\mathrm{AR}_{\rm BPR}(d = 5) \approx 0.981$.

Not only are the performances of these standard algorithms overwhelming better than the GNN of ref. 1 but also they approach closely the supposedly optimal AR computed via the replica method, $\mathrm{AR}_{\rm 1RSB}(d = 3) \approx 0.990$ and $\mathrm{AR}_{\rm 1RSB}(d = 5) \approx 0.987$ (ref. 12). This observation suggests that finding the optimal MIS in d-RRG with $d = 3, 5$ is not a really hard problem and the optimum can be well approximated by algorithms running in polynomial time in n. Indeed, a statistical physics study investigating the structure of IS in d-RRG[12] found that only for $d > 16$, increasing the IS size, the space of IS undergoes a clustering transition, which is usually related to hardness in sampling. For $d < 16$, the structure of IS is such that the MIS is likely to be easy to approximate.

A fundamental question at the time of evaluating a new optimization algorithm is the following: 'What are the really hard problems that should be used as a benchmark to test algorithmic performances?'.

We have argued that for the MIS using d-RRG with $d < 16$ is likely to be an easy problem and the test would be not very selective. However, for larger d, we expect the optimization to become much more demanding because the clustering of the IS of large size is likely to create relevant barriers that affect any algorithm searching for the MIS. This picture is supported by analytical results in the large d limit, where no algorithm is known to find ISs of density larger than $\rho_{\rm alg}(d) = \log(d)/d$, even if the MIS is known to have density $\rho_{\max}(d) = 2\log(d)/d$ (ref. 14): that is, no algorithm achieves an AR better than 0.5 in this limit.

So, a possible answer to the fundamental question above is to study MIS on d-RRG with $d > 16$. And start by comparing with the results presented in ref. 13 for $d = 20$ and $d = 100$. Obviously, a good optimization algorithm should run in a time polynomial (better if linear) in n, and the quality of solutions found should be better than simple existing algorithms and should not degrade with increasing n.

In our opinion, at present, optimizers based on neural networks (like the one presented in ref. 1) do not satisfy the above requirements and are not able to compete with simple standard algorithms to solve hard optimization problems. We showed that this is true for the GNN introduced in ref. 1 applied to the MIS problem and the same conclusion holds also in the case of the maximum-cut problem on sparse graphs, as shown in the concurrent comment by Boettcher[15].

One could argue that both these examples are analysing problems on sparse graphs, and neural networks can be more effective on denser graphs. However, there are already results allowing for such a comparison in dense combinatorial problems: for example, in the

problem of recovering a planted clique in a dense graph, the neural networks introduced in ref. 16 do not reach the performances of message passing algorithms[17] or those of the Monte Carlo parallel tempering method[18].

In conclusion, we believe that it is of primary importance to understand whether and when neural networks can become competitive in solving hard problems or whether there is any deeper reason for their failure.

References

1. Schuetz, M. J. A., Brubaker, J. K. & Katzgraber, H. G. Combinatorial optimization with physics-inspired graph neural networks. *Nat. Mach. Intell.* https://doi.org/10.1038/s42256-022-00468-6 (2022).
2. Cappart, Q. et al. Combinatorial optimization and reasoning with graph neural networks. Preprint at https://arxiv.org/abs/2102.09544 (2021).
3. Kotary, J., Fioretto, F., van Hentenryck, P., & Wilder, B. End-to-end constrained optimization learning: a survey. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence* (Ed. Zhou, Z-H.) 4475–4482 (IJCAI, 2021).
4. Bengio, Y., Lodi, A. & Prouvost, A. Machine learning for combinatorial optimization: a methodological tour d'horizon. *Eur. J. Oper. Res.* **290**, 405–421 (2021).
5. Selsam, D. et al. Learning a sat solver from single-bit supervision. In *Proc. 7th International Conference on Learning Representations* (ICLR, 2019).
6. Cameron, C., Chen, R., Hartford, J. & Leyton-Brown, K. Predicting propositional satisfiability via end-to-end learning. In *Proc. AAAI Conference on Artificial Intelligence* Vol. 34, 3324–3331 (AAAI, 2020).
7. Toenshoff, J., Ritzert, M., Wolf, H. & Grohe, M. Graph neural networks for maximum constraint satisfaction. *Front Artif. Intell.* **3**, 98 (2021).
8. Boppana, R. & Halldórsson, M. M. Approximating maximum independent sets by excluding subgraphs. *BIT Numer. Math.* **32**, 180–196 (1992).
9. Karp, R. M. & Sipser, M. Maximum matching in sparse random graphs. In *22nd Annual Symposium on Foundations of Computer Science (sfcs 1981)* 364–375 (IEEE, 1981).
10. Wormald, N. C. Differential equations for random processes and random graphs. *Ann. Appl. Probab.* **5**, 1217–1235 (1995).
11. McKay, B. D. Independent sets in regular graphs of high girth. *Ars Combinatoria* **23**, 179–185 (1987).
12. Barbier, J., Krzakala, F., Zdeborová, L. & Zhang, P. The hard-core model on random graphs revisited. *J. Phys. Conf.Ser.* **473**, 012021 (2013).
13. Angelini, M. C. & Ricci-Tersenghi, F. Monte Carlo algorithms are very effective in finding the largest independent set in sparse random graphs. *Phys. Rev. E* **100**, 013302 (2019).
14. Coja-Oghlan, A. & Efthymiou, C. On independent sets in random graphs. *Random Struct. Algorithms* **47**, 436–486 (2015).
15. Boettcher, S. *Nat. Mach. Intell.* (2022).
16. Levinas, I. & Louzoun, Y. Planted dense subgraphs in dense random graphs can be recovered using graph-based machine learning. *J. Artif. Intell. Res.* **75**, 541–568 (2022).
17. Deshpande, Y. & Montanari, A. Finding hidden cliques of size $\sqrt{N/e}$ in nearly linear time. *Found. Comput. Math.* **15**, 1069–1128 (2015).
18. Angelini, M. C. Parallel tempering for the planted clique problem. *J. Stat. Mech.* **2018**, 073404 (2018).

Author contributions

All authors contributed equally to this manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to Maria Chiara Angelini.

Peer review information *Nature Machine Intelligence* thanks the anonymous reviewers for their contribution to the peer review of this work.