First, we would like to thank the reviewers for their well thought out and detailed feedback on our work.

Reviewer 1: (a) We acknowledge the importance of improving the accessibility of our work to the machine learning community and thus we will include an explanatory figure (in Section 3) along with background on the Erdős probabilistic method and the method of conditional expectation. (b) In terms of correctness, both the statement of theorem 1 and the direction of the inequality at line 168 will be updated accordingly for the final version of the paper. Again, we thank the reviewer for pointing out those mistakes.

Reviewer 2: (a) We decided to focus our attention on two problems that have different types of constraints to illustrate the flexibility of our approach. As discussed in Sec. 6, certain constraints would be more complicated to handle, e.g., imposing a tree or path structure on the solutions. We agree that solving more problems is essential to demonstrate the generality of our framework and it is one of our current priorities with extending this work. (b) Regarding scalability, indeed GNNs can be computationally expensive in practice. However, this is an active field of research and recent works indicate that it is possible to scale GNN training to millions of nodes [Bojchevski et al., 2020, Rossi et al., 2020]. Another appealing possibility is that of "emergent generalization" as it has been described by Joshi et al. [2019]. That is, a GNN trained on smaller instances of a problem could be trained to perform well on larger instances at test time. We are optimistic that such engineering and conceptual developments in the field of GNNs will yield practically efficient methods over time. (c) Regarding our GNN architecture and tuning, our decisions follow standard conventions in the field: batch and graph size normalization as well as skip connections have been shown to improve performance. Furthermore, our use of GIN and GAT was based on their prominence in the GNN literature as experimentally successful modules. These observations and more interesting practical details about GNNs can be found in the work by Dwivedi et al. [2020]. (d) Regarding the lines 277-278 where we discuss solver scaling, we have indeed used the default Gurobi settings (MIPFocus=0) and therefore our statement will be adjusted to reflect that.

Reviewer 3: (a) The reviewer's summary is largely accurate, however, we would like to emphasize that our solutions are obtained through the method of conditional expectation instead of sampling. (b) We recognize that the loss needs to be derived for each particular problem. At the same time, the probabilistic penalty formulation allows for a rather general and principled way of constructing loss functions for CO problems. (c) We would also remark that the loss functions that we have derived are not generally convex/concave, and that neural-networks/SGD have been successful at providing solutions to problems with non-convex objectives. Ultimately, our view is that the ability of our method to yield good solutions will largely depend on how well the neural network will minimize the loss. This, in turn, will depend on how well the neural networks' inductive biases match the problem constraints. Finally, in practice we have observed that, if the architecture can minimize the loss efficiently, then it can also be carefully tuned to do so consistently regardless of the weight initialization (i.e., with different random seeds).

Reviewer 4: (a) It is indeed the case that for larger clique instances the greedy method is more efficient. Greedy methods strike a nice balance between accuracy and speed. Yet, on the real world datasets it is clear that the greedy methods are inferior when it comes to accuracy. The main reason why the greedy algorithm outperforms the neural methods on the hard instance dataset (RB) is the time budget limitation that we have imposed in order to have comparable time costs for all the methods. Given a sufficiently large time budget, both our method and RUN-CSP can match the accuracy of the greedy heuristic (some experiments along those lines an be found in the paper by Toenshoff et al. [2019]). Additionally, the use of a suitable greedy heuristic necessitates expert knowledge, which is something a neural approach would circumvent by fitting the data distribution. (b) For partitioning, our method is slower than the smooth relaxations that we compare against, but consistently achieves superior accuracy. In addition, as it can be seen in Table 5 of the supplementary material, smooth relaxations generally struggle to yield solutions that obey more complicated constraints, as it is the case in the maximum clique problem. The advantage of our approach is that it can consistently satisfy constraints while also achieving good performance. Currently, our sequential decoding module induces the largest computational overhead, although the number of loss function evaluations is still linear on the number of nodes in the worst case. Improving the runtime of our decoding module is a subject we are actively investigating.

References

A. Bojchevski, J. Klicpera, B. Perozzi, A. Kapoor, M. Blais, B. Rózemberczki, M. Lukasik, and S. Günnemann. Scaling graph neural networks with approximate pagerank. *arXiv preprint arXiv:2007.01570*, 2020.

V. P. Dwivedi, C. K. Joshi, T. Laurent, Y. Bengio, and X. Bresson. Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*, 2020.

C. K. Joshi, T. Laurent, and X. Bresson. On learning paradigms for the travelling salesman problem, 2019.

E. Rossi, F. Frasca, B. Chamberlain, D. Eynard, M. Bronstein, and F. Monti. Sign: Scalable inception graph neural networks. *arXiv preprint arXiv:2004.11198*, 2020.

J. Toenshoff, M. Ritzert, H. Wolf, and M. Grohe. Run-csp: Unsupervised learning of message passing networks for binary constraint satisfaction problems. *arXiv preprint arXiv:1909.08387*, 2019.