GNNs as Parametrized Primal-dual Algorithms*

Jan Hůla[1][2]

[1] Czech Technical University in Prague, Prague, Czech Republic
[2] University of Ostrava, Ostrava, Czech Republic

1 Introduction

Graph neural networks (GNNs) have been recently applied to many combinatorial optimization and feasibility problems, very often by making ad hoc architectural decisions. In this contribution, we explore the possibility of obtaining a GNN architecture in a more principled fashion.

We assume that the GNN in question is recurrent which means that the same message-passing equations are used in each time-step. The equations of such a GNN describe a dynamical system. The question is, therefore, how to design a dynamical system whose fixed points are solutions to the problem at hand. We show that the equations of a GNN can be systematically derived by starting with an ILP formulation of the problem, deriving a primal-dual algorithm for it, and then relaxing the discrete constraints by lifting the variables into a high dimensional space. The dynamics therefore evolves the high-dimensional vectors corresponding to primal and dual variables and rounding is done by standard methods used in SDP relaxations. The resulting equations can then be parametrized and fine-tuned to a desired problem distribution. In our contribution, we will show how to derive such equations for the Boolean satisfiability problem.

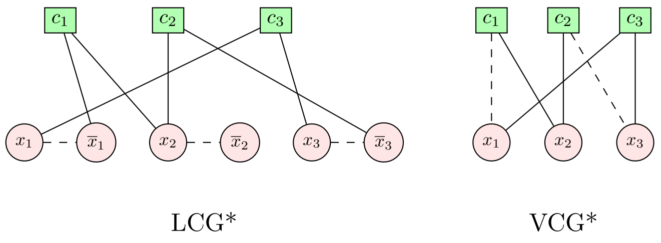

Figure 1: LCG and VCG of the CNF formula $(\overline{x}_1 \vee x_2) \wedge (x_2 \vee \overline{x}_3) \wedge (x_1 \vee x_3)$.

Selsam et al. [3] demonstrated that a GNN can be used to decide satisfiability of Boolean formulas and that it is possible to extract a satisfying assignment from the literal embeddings. Their GNN (called NeuroSAT) operates on bipartite literal-clause graphs, where each literal is connected to the clauses that contain it and also to the literal with opposite polarity as seen in Figure 1 (left). The NeuroSAT architecture is recurrent in the sense that the embeddings of the literals and clauses are updated iteratively, in an alternating fashion, by two LSTM update functions. This feature allows to scale to number of node updates during test time.

Hula et al. [1] empirically demonstrated that NeuroSAT is able to decide satisfiability by acting as a MaxSAT solver, which tries to greedily maximize the number of satisfied clauses

*The results were supported by the Ministry of Education, Youth and Sports within the program ERC CZ under the project POSTMAN no. LL1902.

(Figure 2). In a subsequent work Mojzisek et al. [2] explored this direction further and provided a simplified architecture which operated on a variable-clause graph (Figure 1 (right)) and uses a simple RNN instead of the LSTM update function.

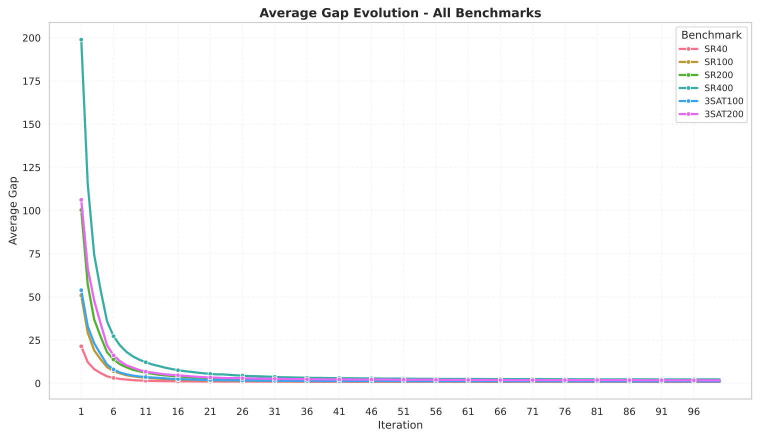

Figure 2: The gap between the average number of unsatisfied clauses for the optimal assignment and the assignment found by the GNN, visualized over the iterations. Individual curves correspond to different benchmark sets; the numbers in the legend denote the number of variables within a formula.

We show that the equations of the GNN can be further simplified and moreover can be systematically derived by starting with a definition of the problem and deriving a primal-dual algorithm for it. We start with the equations provided by [2] and show them in a matrix form where variable embeddings at time step k are represented as rows of the matrix X^k (same for clause embeddings in Λ^k) :

$$X^{k+1} = \text{proj}_{\|X_i\|=1}(\text{RNN}_1(X^k, (A^{+T} f_1(\Lambda^k) + A^{-T} f_2(\Lambda^k))) \tag{1}$$

$$\Lambda^{k+1} = \text{proj}_{\|\Lambda_i\|=1}(\text{RNN}_2(\Lambda^k, A^+ f_3(X) + A^- f_4(X^{k+1}))). \tag{2}$$

Guided by the primal-dual derivation, we manage to simplify the equations to the following form:

$$X^{k+1} = \text{proj}_{\|X_i\|=1}(X^k + (A^{+T} f_1(\Lambda^k) + A^{-T} f_2(\Lambda^k))) \tag{3}$$

$$\Lambda^{k+1} = \text{proj}_{\Lambda\geq 0}(\Lambda^k + (A^+ f_3(X^{k+1}) + A^- f_4(X^{k+1})) + B). \tag{4}$$

We use $\text{proj}_{\|X_i\|=1}$ to denote a normalization layer which normalizes the rows of X^k to unit norm. The functions f_i (also applied row-wise) are 2-layer MLPs and create the messages which are then aggregated over the ”positive” and ”negative” edges (by matrix multiplications with A^+ and A^- respectively).

This elegantly explains the mechanism by which a GNN is able to find a satisfying assignment for the CNF formula and opens a door for transfer of many techniques used in numerical optimization to the GNN framework because such equations can be derived for arbitrary optimization or feasibility problems.

References

[1] Jan Hůla, David Mojžíšek, and Mikoláš Janota. Understanding GNNs for Boolean Satisfiability through Approximation Algorithms.

[2] David Mojžíšek, Jan Hůla, Ziwei Li, Ziyu Zhou, and Mikoláš Janota. Neural Approaches to SAT Solving: Design Choices and Interpretability.

[3] Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, given=Leonardo family=Moura, and David L. Dill. Learning a SAT Solver from Single-Bit Supervision.