Notes on “A Fast Machine Learning Algorithm for the MaxCut Problem”

I Introduction:

The Max-Cut problem is a graph partitioning problem that involves finding the best way to divide a graph's vertices into two sets so that the number of edges between the two groups is maximized.

It is an NP hard problem (nondeterministic-polynomial time). Basically means it is a complex problem to solve/come up with the most efficient solution for. A ML algorithm can be useful for this purpose.

If we assign each node to a binary value (+1) or (-1), the cut-size of the graph can be written as:

$$Cut(G) = \frac{1}{2} \sum_{(i,j)\in E} (1 - \sigma(i)\sigma(j))$$

Although there are no known polynomial-time methods for the Max-Cut problem, there are heuristic algorithms that provide more practical approaches, such as the Greedy or simulated annealing algorithms. However, they do not provide the most efficient solution(s) that exist. The implementation of a ML algorithm in a similar problem, as well as an explanation of its Max-Cut application is in the rest of the paper.

II Ising-Spin Glass Model

The Ising-Spin Glass Model is a statistical model that describes the behavior of spin glasses, which are systems where atoms have randomly oriented spins.

The Hamiltonian of the spin system (aids understanding the correspondence of ISG Model and Max-Cut):

$$H(\sigma) = -\sum_{i<j} w(ij)\sigma(i)\sigma(j)$$

Since i and j are both connected in the graph for this problem, w can be assigned a value of -1, so the Hamiltonian can be reduced to this:

$$H(\sigma) = \sum_{(i,j)\in E} \sigma(i)\sigma(j)$$

And using this equivalency:

$$\begin{aligned} \text{Cut}(G) &= \frac{1}{2} \sum_{(i,j)\in E} (1 - \sigma_i\sigma_j) \\ &= \frac{1}{2} \sum_{(i,j)\in E} 1 - \frac{1}{2} \sum_{(i,j)\in E} \sigma_i\sigma_j \\ &= \frac{1}{2}|E| - \frac{1}{2}H(\boldsymbol{\sigma}) \end{aligned}$$

We get that maximizing the cut size of graph G is equivalent to minimizing the Hamilton of the spin system. For the spin system, there are already established insights from statistical mechanics (Boltzmann Distribution) to help solve it, unlike the Max-Cut problem. So, using the Ising-Spin Glass Model as a well established reference, we can refer to it for the Max-Cut ML algorithm.

III Classical Methods

There are some classical algorithms that can be applied to the Max Cut problem, however, for the purpose of finding the most efficient result, they have shortcomings.

The Greedy Algorithm: Each node, denoted by x, will be assigned a value of 0 or 1, post-cut. To find the next node: x^(k+1) from the current: x^k, we just look for the neighboring node with the lowest Hamiltonian from the current node. However, it has a shortsightedness in that it is vulnerable to local minimums.

Simulated Annealing Algorithm: Similar to the Greedy Algorithm, it finds a neighboring node with the lowest Hamiltonian, however with an added hyperparameter/Metropolis factor, which is temperature.

The temperature, denoted by T divides the exponent in the given equation:

$$P = \min(1, e^{-(H(\boldsymbol{x}')-H(\boldsymbol{x}))/T}),$$

The temperature decreases at each step, following some annealing schedule. This helps to avoid getting trapped in local minima prematurely, however it is still subject to them.

The Variational Classical Annealing Algorithm: This method uses neural networks to model the probability distribution of different graph cuts. It minimizes the KL divergence between the target distribution (Boltzmann distribution) and a learned distribution. VCA aims to approximate the equilibrium state of the Ising system, but its approach can be slow due to the need to capture correlations between individual spins.

IV Transition Probability in Ground State

The goal is to learn how a system moves from an initial state x(0)x^{(0)}x(0) to the ground state (optimal solution).
This is done by learning a **transition probability distribution** p(x∣x(0))p(x∣x^{(0)})p(x∣x(0)), which models how states change.
The **Hamiltonian** (energy) is used as the loss function—minimizing it gets you closer to the optimal cut.

The loss function is the expected Hamiltonian:
$H(x|x^{0})$

Basically, they calculate the average energy of the system using Monte Carlo sampling.

The structure of the neural network uses a Long Short-Term Memory (LSTM) network to model the transition probabilities. The LSTM takes the initial state and predicts the transitions in steps, which is used to generate the final state.

Unlike Variational Classical Annealing, this approach doesn't use Shannon entropy as regularization, which speeds up the process by focusing on transitions between whole configurations rather than individual spins, which speeds up the process.

V Experiment Results

TABLE I
RESULTS FOR GRAPH MAXCUT ON SYNTHETIC INSTANCES

Nodes	Gurobi	L2T	Improvement	Speedup
20	67 (5s)	71 (36s)	+5.97%	0.139×
30	132 (10s)	135 (93s)	+2.27%	0.108×
100	1408 (2000s)	1415 (33s)	+0.49%	60.6×
1000	128508 (4400s)	129714 (119s)	+0.94%	36.97×
5000	- (> 8h)	3175813 (202s)	-	–

TABLE II
RESULTS ON THE GSET DATASET. THE COMPARED METHODS ARE BLS, DSDP, KGLWG, RUN-CSP, PI-GNN.

Graph	Nodes	Edges	BLS	DSDP	KHLWG	RUN-CSP	PI-GNN	L2T (**Ours**)
G14	800	4694	**3064**	-	2922	3061	2943	3029
G15	800	4661	**3050**	2938	**3050**	2928	2990	2995
G22	2000	19990	**13359**	12960	**13359**	13028	13181	13167
G49	3000	6000	**6000**	**6000**	**6000**	**6000**	5918	5790
G50	3000	6000	**5880**	**5880**	**5880**	**5880**	5820	5720
G55	5000	12468	**10294**	9960	10236	10116	10138	10017
G70	10000	9999	**9541**	9456	9458	-	9421	9358

After running the L2T algorithm on a Linux server with one GPU A100, with different parameters, it was found that the learning rate is $3x10^{-5}\sim2x10^{-3}$

Compared to the popular optimization solver Gurobi, L2T has outperformed it on all tested graphs, in terms of time and quality of solution.