Community-based Distributed Training of Graph Convolutional Networks via ADMM

Hongyi Li, Junxiang Wang, Yongchao Wang, Yue Cheng and Liang Zhao

NEURAL INFORMATION PROCESSING SYSTEMS
OPT 2021 Workshop

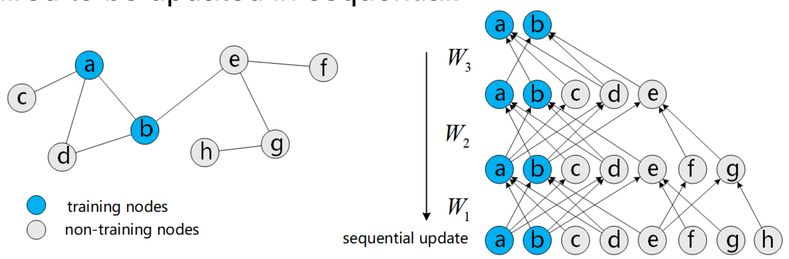
Introduction: GCN Training

Problem 1. $\min_{\{W\}_{l=1}^{L},\{Z\}_{l=1}^{L}} \ell(Z_L, Y),$

$$s.t.\ Z_l = \underbrace{f_l(\tilde{A}Z_{l-1}W_l)}_{\text{activation function}}, l < L, Z_L = \tilde{A}Z_{L-1}W_L.$$

Cons for original GCN training via SGD include:

- **Node dependency:** the loss for each node depends on a large number of neighboring nodes;
- **Layer dependency:** node representations in different layers are required to be updated in sequential.



(a) original graph (b) original GCN training

Motivation of this paper:
Can we design an algorithm to update variables for different nodes and layers in parallel?

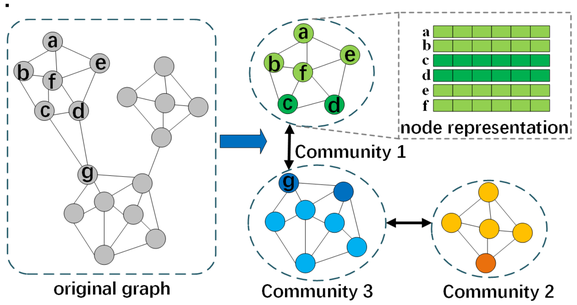
Community-based GCN Training

- Divide the original graph into M communities by METIS;
- Assign each community to a single agent;
- Partition $\tilde{A}$, Z, Y accordingly:

$$\tilde{A} = \begin{bmatrix} \tilde{A}_{1,1} & \cdots & \tilde{A}_{1,M} \\ \vdots & \ddots & \vdots \\ \tilde{A}_{M,1} & \cdots & \tilde{A}_{M,M} \end{bmatrix},$$

$$Z_l = \left[Z_{l,1}^T, Z_{l,2}^T, \cdots Z_{l,M}^T\right]^T,$$

$$Y = [Y_1^T, Y_2^T, \cdots Y_M^T]^T.$$



Community-based ADMM for distributed GCN Training

We relax non-linear constraints in Problem 1 to get Problem 2:

Problem 2. $\min_{\{W\}_{l=1}^{L},\{Z\}_{l=1}^{L}} \ell(Z_L, Y) + \underbrace{\frac{\nu}{2}\sum_{l=1}^{L-1}\left\|Z_l - f_l(\tilde{A}Z_{l-1}W_l)\right\|_F^2}_{\text{relaxation}},\ s.t.\ Z_L = \tilde{A}Z_{L-1}W_L.$

- When $\nu \to \infty$, Problem 2 approximates Problem 1;
- Z_l, W_l for different l can be updated in parallel.

We further transfer Problem 2 to Problem 3:

Problem 3.

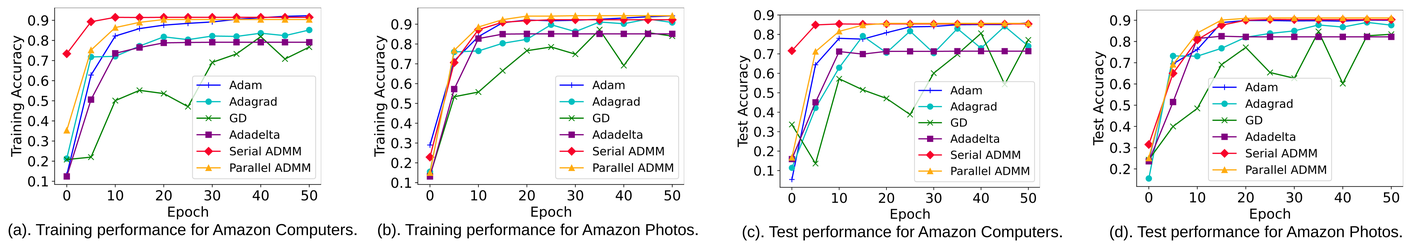
$$\min_{\{W\}_{l=1}^{L},\{Z\}_{l=1}^{L}} \sum_{m=1}^{M} \ell\left(Z_{L,m}, Y_m\right) + \frac{\nu}{2}\sum_{l=1}^{L-1}\sum_{m=1}^{M}\left\|Z_{l,m} - f_l((\tilde{A}_{m,m}Z_{l-1,m} + \sum_{r\in\mathcal{N}_m}\tilde{A}_{m,r}Z_{l-1,r})W_l)\right\|_F^2$$

$$s.t.\ Z_{L,m} = \left(\tilde{A}_{m,m}Z_{L-1,m} + \sum_{r\in\mathcal{N}_m}\tilde{A}_{m,r}Z_{L-1,r}\right)W_L, m = 1, \cdots M.$$

- Fixing l, $Z_{l,m}$ for m can be updated in parallel by different agents.
- To reduce computation complexity, we apply quadratic approximation technique to solve subproblems.

Experimental Results

Our proposed Serial ADMM and Parallel ADMM outperform most of comparison methods with fastest convergence.



(a). Training performance for Amazon Computers. (b). Training performance for Amazon Photos. (c). Test performance for Amazon Computers. (d). Test performance for Amazon Photos.

Parallel ADMM is 2x faster than Serial ADMM (2 layers, 3 communities).

	Serial ADMM (sec)	Parallel ADMM (sec)			
Dataset	Total	Training	Communication	Total	Speedup
Amazon Computers	80.82	14.94	9.54	24.48	3.30
Amazon Photo	50.81	8.80	8.27	17.07	2.98

Comparison of training and communication time on two datasets.