西安电子科技大学
XIDIAN UNIVERSITY
EMORY UNIVERSITY
GEORGE MASON UNIVERSITY

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

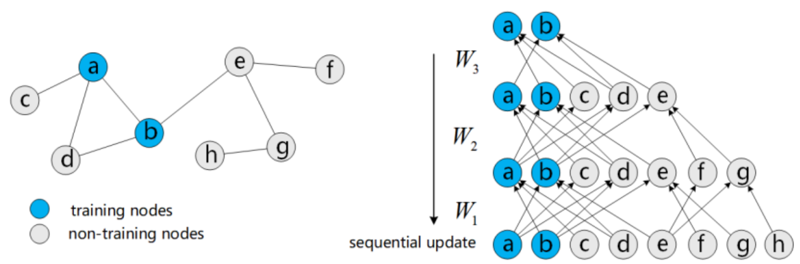
Introduction: GCN Training

Problem 1.

$$\min_{\{W\}_{l=1}^{L},\{Z\}_{l=1}^{L}} \ell(Z_L, Y)$$
$$s.t.\ Z_l = \underbrace{f_l(\tilde{A}Z_{l-1}W_l)}_{\text{nonlinear activation}},\ l < L,\ Z_L = \tilde{A}Z_{L-1}W_L.$$

Challenges of GCN training via SGD include:

- **Node dependency:** node representation depends on a large number of neighboring nodes;
- **Layer dependency:** weights in different layers are updated in order.



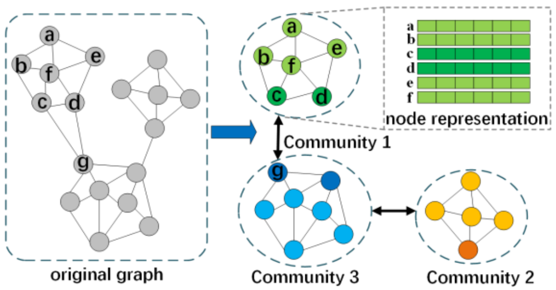
(a) original graph (b) original GCN training

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

Community Partitions of GCN Models

- Divide the original graph into M communities by METIS;
- Partition $\tilde{A}$, Z, Y based on communities.

$$\tilde{A} = \begin{bmatrix} \tilde{A}_{1,1} & \cdots & \tilde{A}_{1,M} \\ \vdots & \ddots & \vdots \\ \tilde{A}_{M,1} & \cdots & \tilde{A}_{M,M} \end{bmatrix},$$
$$Z_l = [Z_{l,1}^T, Z_{l,2}^T, \cdots Z_{l,M}^T]^T,$$
$$Y = [Y_1^T, Y_2^T, \cdots Y_M^T]^T.$$



Community-based ADMM for distributed GCN Training

We relax Problem 1 to Problem 2 via imposing nonlinear constraints as penalties:

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 transform equivalently from Problem 2 to Problem 3 based on community partitions:

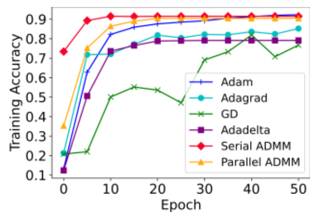
Problem 3.

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

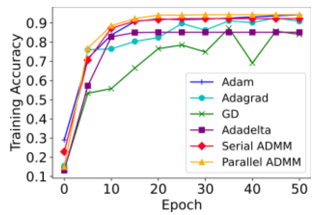
- Fixing l, $Z_{l,m}$ for m can be updated in parallel by different agents.
- In order to reduce time complexity, the quadratic approximation technique is applied to solve subproblems.

Experimental Results

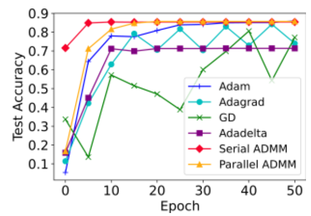
Our proposed parallel ADMM (i.e. red curve) outperforms all comparison methods.



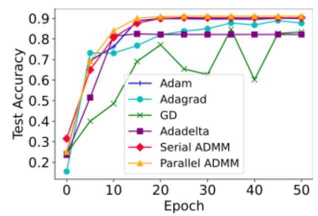
(a). Training performance for Amazon Computers.



(b). Training performance for Amazon Photos.



(c). Test performance for Amazon Computers.



(d). Test performance for Amazon Photos.

The parallel ADMM reaches 3x speedup (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.