SCARA: Scalable Graph Neural Networks with Feature-Oriented Optimization

VLDB 2022

Ningyi Liao, Dingheng Mo, Siqiang Luo, Xiang Li, Pengcheng Yin

Presented by: Ningyi Liao

Google Research

Scalable Graph Neural Networks (GNNs)

- GNNs, such as Graph Convolutional Networks (GCNs), achieve strong performance on many graph understanding tasks
- There are increasing demands on studying modern real-world large-scale datasets (million- or billion-scale graphs)
- GCNs are resource-demanding and difficult to apply on large-scale graphs: iterative propagation, memory overhead
- Existing approaches are not scalable enough: GBP[1] uses 10^4 sec. and >192 GB RAM on Papers100M (111M nodes, 1.6B edges)

[1] Hanzhi Wang, Mingguo He, Zhewei Wei, Sibo Wang, Ye Yuan, Xiaoyong Du, and Ji Rong Wen. 2021. Approximate Graph Propagation. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Vol. 1. Association for Computing Machinery, 1686-1696.

Highlights & Contributions

- **SCARA Framework**: Pre-Propagation Decoupling Model, optimized propagation with fast GPU batch training and inference
- **FEATURE-PUSH**: feature-oriented fast vector-based propagation, sub-linear precomputation complexity
- **FEATURE-REUSE**: efficient reuse among multiple features, further saves computation with guaranteed precision
- **Performance Evaluation**: up to 100× faster precomputation, able to process billion-scale Papers100M in 100 seconds

Complexity of Scalable GNN

- **Vanilla / Sampling Model**: GCN[2]

  1. Iterative Aggregation & Propagation:

$$H^{(0)} = X$$

$$H^{(l+1)} = \sigma\left(\tilde{A}H^{(l)}W^{(l)}\right), \quad l = 0, 1, \cdots, L-1$$

Node Feature

Layer Representation

Graph Adjacency

Weight

- **Pre-Propagation Model**: SCARA (ours)

  1. Propagation Precomputation:

$$H^{(0)} = P = \sum_{l=0}^{\infty} \alpha(1-\alpha)^l \tilde{A}^l_{(r)} \cdot X$$

Graph Embedding

Graph Adjacency

Node Feature

  2. Feature Transformation:

$$H^{(l+1)} = \sigma\left(H^{(l)}W^{(l)}\right), \quad l = 0, 1, \cdots, L-1$$

Layer Representation

Weight

[2] Thomas N Kipf & Max Welling. “Semi-supervised classification with graph convolutional networks”. ICLR 2017.

Complexity of Scalable GNN

- **Vanilla / Sampling Model**: GCN[2]

  1. Iterative Aggregation & Propagation: $O(ILmF + ILnF^2)$

     ☹ *Not scalable to large m and n*

     ☹ *Not optimized for batch processing*

- **Pre-Propagation Model**: SCARA (ours)

  1. Propagation Precomputation: $O(F\sqrt{m \log n}/\lambda)$

     ☺ *Only sublinear to n*

  2. Feature Transformation: $O(ILnF^2)$

     ☺ *Efficient GPU batch training*

[2] Thomas N Kipf & Max Welling. “Semi-supervised classification with graph convolutional networks”. ICLR 2017.

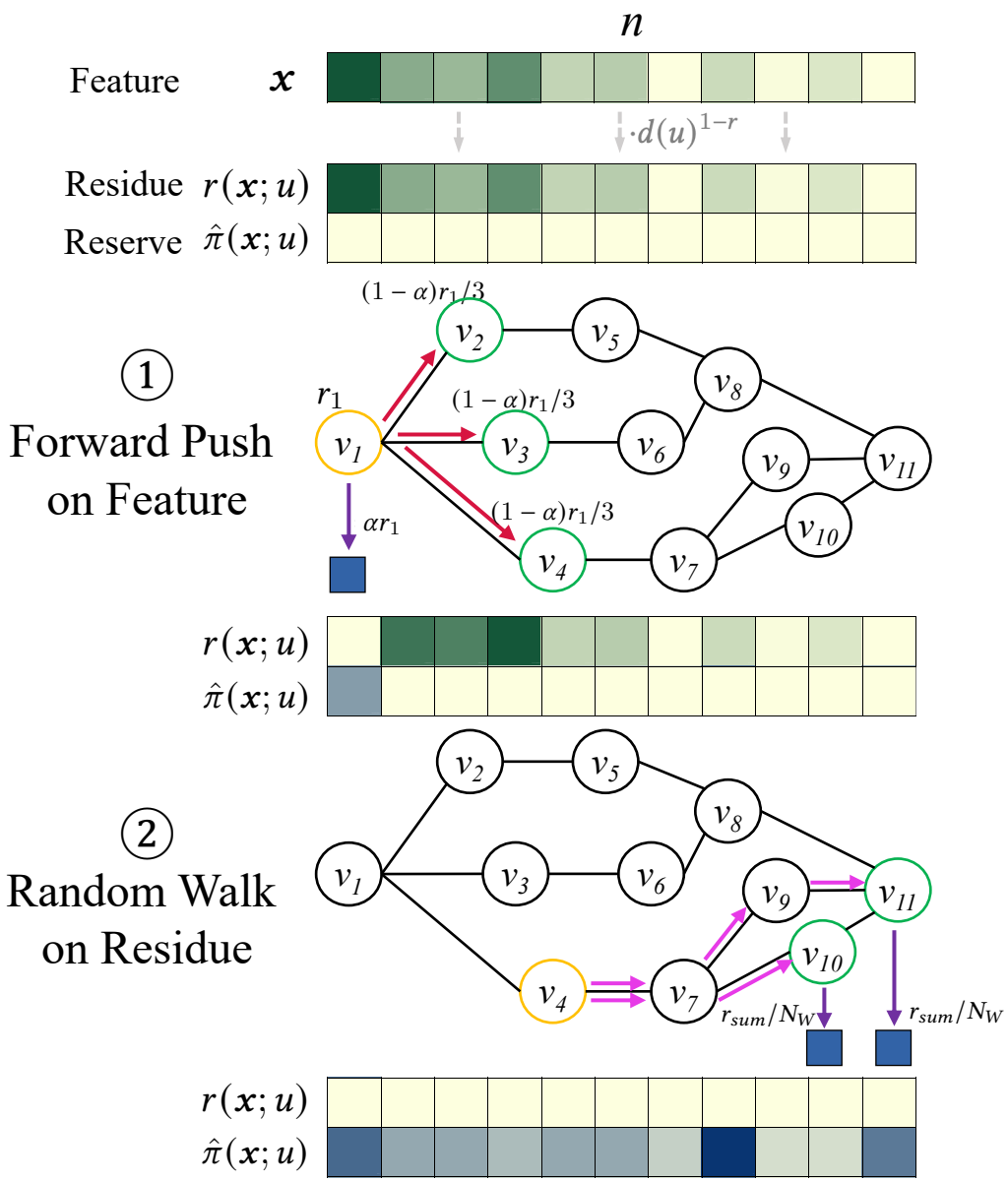
SCARA: FEATURE-PUSH

- Forward Push on Feature Value
  - Initialized by normalized feature vector

$$\boldsymbol{P} = \boldsymbol{D}^{r-1}\left(\boldsymbol{A}\boldsymbol{D}^{-1}\right)^{l}\boldsymbol{D}^{1-r}\boldsymbol{X} \tag{1}$$

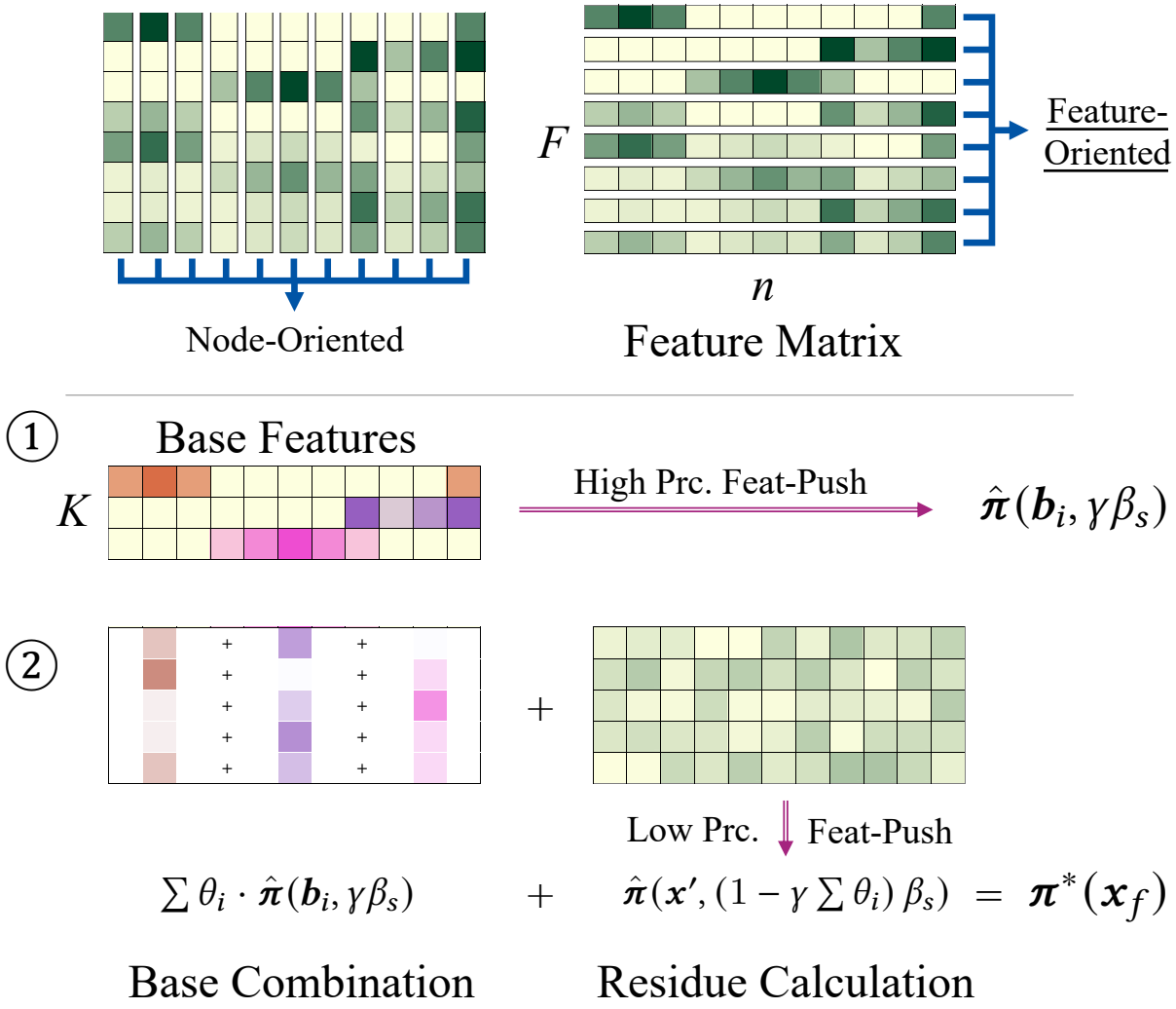
  - Complexity: $O\left(\|\boldsymbol{x}\|_1 / r_{max}\right)$
- Random Walk on Feature Residue
  - Reduced number of Random Walks
  - Complexity: $O\left(m \cdot r_{max}/\beta\right)$
- Combination: Push Coefficient β
  - Overall complexity: $O(\sqrt{\frac{m\|\boldsymbol{x}\|_1}{\beta}})$

$$\Longrightarrow O(\sqrt{m \log n}/\lambda) \tag{2}$$



SCARA: Feature-Reuse

- Base Features
  - Select $K \ll F$ base features
  - High Prc. Feat-Push with $\beta = \gamma\beta_s$
- Non-Base Features
  - Combination: linear combination of base calculation results
  - Low Prc. Feat-Push: sparse vectors thus faster
  - Complexity: $O\left(\sqrt{\frac{m(1-\theta_{sum})}{\beta_s(1-\gamma\theta_{sum})}}\right)$



Experimental Evaluation

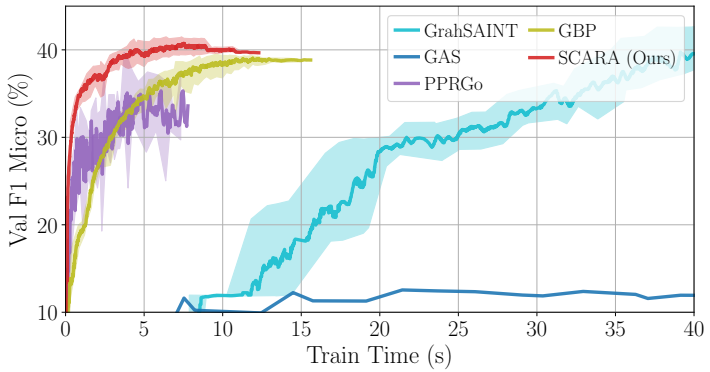
- **Time Efficiency**: 10-100× faster precomputation, comparable or better training and inference clock time
- **Memory Efficiency**: Paper100M with 64GB without OOM
- **Effectiveness**: similar or better F1-score, fast convergence

Dataset	Nodes n	Edges m	Features F	Dataset	Nodes n	Edges m	Features F
Reddit	232,965	114,615,892	602	Papers100M	111,059,956	1,615,685,872	128

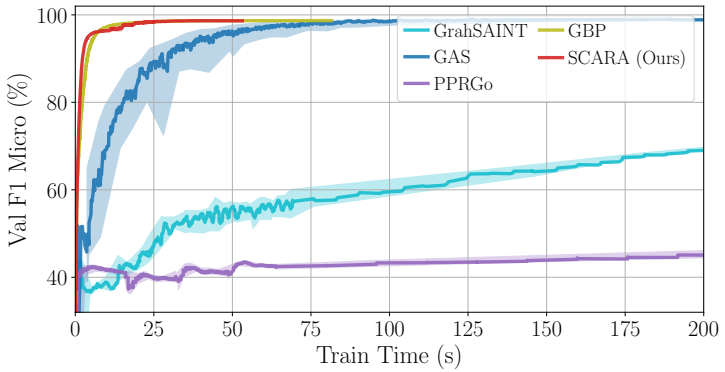
Transductive	Reddit						Papers100M					
	Learn	(Pre.	+ Train)	Infer	Mem.	F1	Learn	(Pre.	+ Train)	Infer	Mem.	F1
GraphSAINT	<u>51.5</u>	(–	51.5)	26.1	11.1	30.7 ±3.0	–	–	–	–	OOM	–
GAS	3563	(–	3563)	**0.1**	14.6	38.0 ±0.2	–	–	–	–	OOM	–
PPRGo	163	(157	+ 4.8)	74.1	<u>8.0</u>	31.0 ±1.7	–	–	–	–	OOM	–
GBP	1891	(2127	+ 16.3)	6.2	8.4	<u>39.2 ±0.3</u>	–	–	–	–	OOM	–
SCARA (ours)	**12.0**	(1.8	+ 10.6)	<u>4.8</u>	**4.7**	**40.3 ±0.7**	**1471**	(83.5	+ 1388)	**2.8**	**63.7**	**35.5 ±0.8**

Experimental Evaluation

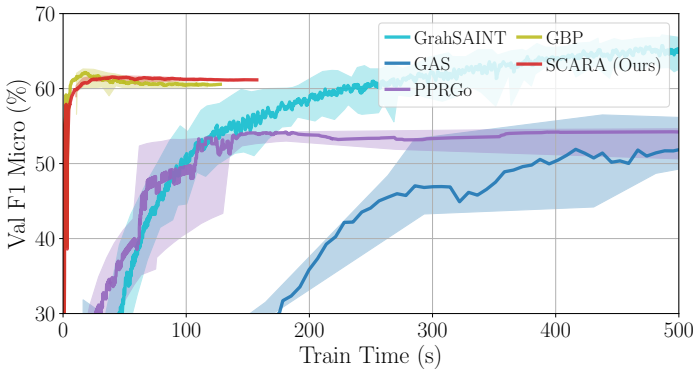
- **Time Efficiency**: 10-100× faster precomputation, comparable or better training and inference
- **Memory Efficiency**: Paper100M with 64GB without OOM
- **Effectiveness**: similar or better F1-score, fast convergence



(a) Reddit



(b) PPI



(c) Yelp

Summary and Future Work

- **SCARA framework**: FEATURE-PUSH for fast single feature computation, FEATURE-REUSE for reuse among features
- **Feature-Oriented Propagation**: sub-linear precomputation complexity with guaranteed precision
- **Performance Evaluation**: 10-100× faster precomputation, efficient feature transformation, relatively low RAM overhead
- **Future Work**: expand and generalize feature-oriented optimizations, apply to more GNN models

THANK YOU

Acknowledgments

This research is supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 1 Seed (RS05/21) and in part by NTU startup grant. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore. Xiang Li is supported by Shanghai Pujiang Talent Program (Project No. 21PJ1402900) and Shanghai Science and Technology Committee General Program (Project No. 22ZR1419900). We also thank the anonymous reviewers for their valuable feedback.

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE