Faster Local Solvers for Graph Diffusion Equations

Jiahe Bai[1], Baojian Zhou[1,2], Deqing Yang [1,2], Yanghua Xiao [2]

[1]the School of Data Science, Fudan University,
[2]Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University

November 12, 2024

Background: Graph Diffusion Equations

Definition Given a propagation matrix $\boldsymbol{M}$ associated with an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, a general graph diffusion equation is defined as

$$\boldsymbol{f} \triangleq \sum_{k=0}^{\infty} c_k \boldsymbol{M}^k \boldsymbol{s},$$

where $\boldsymbol{f}$ is the diffusion vector of a source vector $\boldsymbol{s}$, and $c_k \geq 0$.

Background: Graph Diffusion Equations

Definition Given a propagation matrix $\boldsymbol{M}$ associated with an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, a general graph diffusion equation is defined as

$$\boldsymbol{f} \triangleq \sum_{k=0}^{\infty} c_k \boldsymbol{M}^k \boldsymbol{s},$$

where $\boldsymbol{f}$ is the diffusion vector of a source vector $\boldsymbol{s}$, and $c_k \geq 0$.

Many graph learning tools can be represented as diffusion vectors. What's more, to compute $\boldsymbol{f}_{\text{PPR}}$ and $\boldsymbol{f}_{\text{KATZ}}$, it is equivalent to solving the linear system $\boldsymbol{Qx} = \boldsymbol{s}$.

Equ.	$\boldsymbol{M}$	c_k	$\boldsymbol{s}$
PPR	$\boldsymbol{AD}^{-1}$	$\alpha(1-\alpha)^k$	$\boldsymbol{e}_s$
Katz	$\boldsymbol{A}$	α^k	$\boldsymbol{e}_s$
HK	$\boldsymbol{AD}^{-1}$	$e^{-\tau}\tau^k/k!$	$\boldsymbol{e}_s$
IPR	$\boldsymbol{AD}^{-1}$	$\frac{\theta^k}{(\theta^k+\theta^{10})^2}$	$\boldsymbol{e}_s$
APPNP	$\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{AD}^{-\frac{1}{2}}$	$\alpha(1-\alpha)^k$	$\boldsymbol{x}$

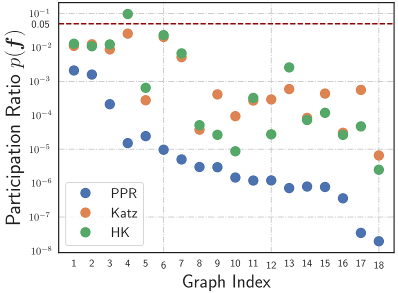
Background: Graph Diffusion Equations

Definition Given a propagation matrix $\boldsymbol{M}$ associated with an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, a general graph diffusion equation is defined as

$$\boldsymbol{f} \triangleq \sum_{k=0}^{\infty} c_k \boldsymbol{M}^k \boldsymbol{s},$$

where $\boldsymbol{f}$ is the diffusion vector of a source vector $\boldsymbol{s}$, and $c_k \geq 0$.

A key property of $\boldsymbol{f}$ is the high localization of its entry magnitudes. We measure the localization by participation ratio $p(\boldsymbol{f}) = (\sum_{i=1}^{n} |f_i|^2)^2 / (n \sum_{i=1}^{n} |f_i|^4)$.



Faster Solvers via Local Diffusion Process

We propose a novel graph diffusion framework via a *local diffusion process* for efficiently approximating GDEs.

Faster Solvers via Local Diffusion Process

We propose a novel graph diffusion framework via a *local diffusion process* for efficiently approximating GDEs.

The *local diffusion process* is defined as a process of updates $\{(\boldsymbol{x}^{(t)}, \boldsymbol{r}^{(t)}, \mathcal{S}_t)\}_{0 \le t \le T}$.

$$\left(\boldsymbol{x}^{(t+1)}, \boldsymbol{r}^{(t+1)}, \mathcal{S}_{t+1}\right) = \phi\left(\boldsymbol{x}^{(t)}, \boldsymbol{r}^{(t)}, \mathcal{S}_t; \boldsymbol{s}, \epsilon, \mathcal{G}, \mathcal{A}_\theta\right), \quad 0 \le t \le T.$$

We say this process *converges* when $\mathcal{S}_T = \emptyset$ if there exists such T; the generated sequence of active nodes are $\mathcal{S}_t$. The total number of operations of local solver $\mathcal{A}_\theta$ is

$$\mathcal{T}_{\mathcal{A}_\theta} = \sum_{t=0}^{T-1} \operatorname{vol}(\mathcal{S}_t) = T \cdot \overline{\operatorname{vol}}(\mathcal{S}_T).$$

Faster Solvers via Local Diffusion Process

The standard Gauss-Seidel iterative method with Successive Overrelaxation can be localized via *local diffusion process* as

$$\text{LocalSOR} : \boldsymbol{x}^{(t+t_{i+1})} = \boldsymbol{x}^{(t+t_i)} + \omega \cdot \tilde{\boldsymbol{e}}_{u_i}^{(t+t_i)}, \quad \boldsymbol{r}^{(t+t_{i+1})} = \boldsymbol{r}^{(t+t_i)} - \omega \cdot \boldsymbol{Q} \cdot \tilde{\boldsymbol{e}}_{u_i}^{(t+t_i)},$$

Faster Solvers via Local Diffusion Process

The standard Gauss-Seidel iterative method with Successive Overrelaxation can be localized via *local diffusion process* as the following

$$\text{LocalSOR}: \boldsymbol{x}^{(t+t_{i+1})} = \boldsymbol{x}^{(t+t_i)} + \omega \cdot \tilde{\boldsymbol{e}}_{u_i}^{(t+t_i)}, \quad \boldsymbol{r}^{(t+t_{i+1})} = \boldsymbol{r}^{(t+t_i)} - \omega \cdot \boldsymbol{Q} \cdot \tilde{\boldsymbol{e}}_{u_i}^{(t+t_i)},$$

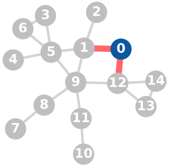
If we reformulate $\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{s}$ as $\boldsymbol{x}_t^* = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}) \triangleq \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{Q}\boldsymbol{x} - \boldsymbol{s}^\top \boldsymbol{x}$. The standard gradient descent and Chebyshev methods can be localized as the following

$$\text{LocalGD}: \qquad \boldsymbol{x}^{(t+1)} = \boldsymbol{x}^{(t)} + \boldsymbol{r}_{\mathcal{S}_t}^{(t)}, \qquad \boldsymbol{r}^{(t+1)} = \boldsymbol{r}^{(t)} - \boldsymbol{Q}\boldsymbol{r}_{\mathcal{S}_t}^{(t)},$$

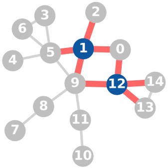
$$\text{LocalCH}: \boldsymbol{\pi}^{(t+1)} = \boldsymbol{\pi}^{(t)} + \frac{2\delta_{t+1}}{1-\alpha}\boldsymbol{D}^{1/2}\boldsymbol{r}_{\mathcal{S}_t}^{(t)} + \delta_{t:t+1}(\boldsymbol{\pi}^{(t)} - \boldsymbol{\pi}^{(t-1)})_{\mathcal{S}_t}, \delta_{t+1} = \left(\frac{2}{1-\alpha} - \delta_t\right)^{-1}$$

$$\boldsymbol{D}^{1/2}\boldsymbol{r}^{(t+1)} = \boldsymbol{D}^{1/2}\boldsymbol{r}^{(t)} - (\boldsymbol{\pi}^{(t+1)} - \boldsymbol{\pi}^{(t)}) + (1-\alpha)\boldsymbol{A}\boldsymbol{D}^{-1}(\boldsymbol{\pi}^{(t+1)} - \boldsymbol{\pi}^{(t)}).$$

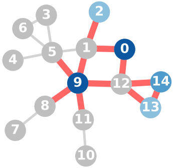
Local Diffusion Process



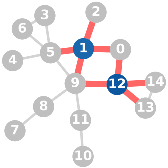
$\mathcal{S}_0 = \{0\}$



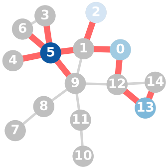
$\mathcal{S}_1 = \{1, 12\}$



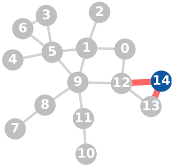
$\mathcal{S}_2 = \{0, 2, 9, 13, 14\}$



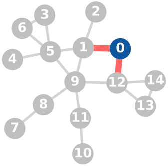
$\mathcal{S}_3 = \{1, 12\}$



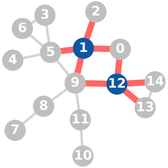
$\mathcal{S}_4 = \{0, 2, 5, 13\}$



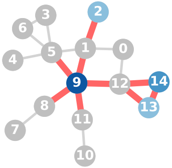
$\mathcal{S}_5 = \{14\}$



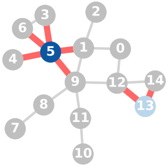
$\mathcal{S}_0 = \{0\}$



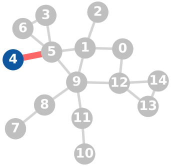
$\mathcal{S}_1 = \{0, 1, 12\}$



$\mathcal{S}_2 = \{1, 2, 9, 12, 14\}$



$\mathcal{S}_3 = \{5, 13\}$



$\mathcal{S}_4 = \{4, 12\}$

Properties of LocalSOR

Let $\boldsymbol{Q} \triangleq \boldsymbol{I} - \beta \boldsymbol{P}$ where $P_{uv} \neq 0$ if $(u, v) \in \mathcal{E}$; 0 otherwise, and $\boldsymbol{P} \geq \boldsymbol{0}_{n \times n}$. Define maximal value $P_{\max} = \max_{u \in \mathcal{V}} \|\boldsymbol{P}\boldsymbol{e}_u\|_1$. Assume that $\boldsymbol{r}^{(0)} \geq \boldsymbol{0}$ is nonnegative and $P_{\max}$, β are such that $\beta P_{\max} < 1$, then the local diffusion process via LocalSOR with $\omega \in (0, 1)$ has the following properties

Properties of LocalSOR

Let $\boldsymbol{Q} \triangleq \boldsymbol{I} - \beta \boldsymbol{P}$ where $P_{uv} \neq 0$ if $(u, v) \in \mathcal{E}$; 0 otherwise, and $\boldsymbol{P} \geq \mathbf{0}_{n \times n}$. Define maximal value $P_{\max} = \max_{u \in \mathcal{V}} \|\boldsymbol{P}\boldsymbol{e}_u\|_1$. Assume that $\boldsymbol{r}^{(0)} \geq \mathbf{0}$ is nonnegative and $P_{\max}$, β are such that $\beta P_{\max} < 1$, then the local diffusion process via LocalSOR with $\omega \in (0, 1)$ has the following properties

1. **Nonnegativity.** $\boldsymbol{r}^{(t+t_i)} \geq \mathbf{0}$ for all $t \geq 0$ and $t_i = (i-1)/|\mathcal{S}_t|$.

Properties of LocalSOR

Let $\boldsymbol{Q} \triangleq \boldsymbol{I} - \beta \boldsymbol{P}$ where $P_{uv} \neq 0$ if $(u, v) \in \mathcal{E}$; 0 otherwise, and $\boldsymbol{P} \geq \mathbf{0}_{n \times n}$. Define maximal value $P_{\max} = \max_{u \in \mathcal{V}} \|\boldsymbol{P}\boldsymbol{e}_u\|_1$. Assume that $\boldsymbol{r}^{(0)} \geq \mathbf{0}$ is nonnegative and $P_{\max}$, β are such that $\beta P_{\max} < 1$, then the local diffusion process via LocalSOR with $\omega \in (0, 1)$ has the following properties

1. **Nonnegativity.** $\boldsymbol{r}^{(t+t_i)} \geq \mathbf{0}$ for all $t \geq 0$ and $t_i = (i-1)/|\mathcal{S}_t|$.
2. **Monotonicity property.** $\|\boldsymbol{r}^{(0)}\|_1 \geq \cdots \|\boldsymbol{r}^{(t+t_i)}\|_1 \geq \|\boldsymbol{r}^{(t+t_{i+1})}\|_1 \cdots$.

Properties of LocalSOR

Let $\boldsymbol{Q} \triangleq \boldsymbol{I} - \beta \boldsymbol{P}$ where $P_{uv} \neq 0$ if $(u, v) \in \mathcal{E}$; 0 otherwise, and $\boldsymbol{P} \geq \boldsymbol{0}_{n \times n}$. Define maximal value $P_{\max} = \max_{u \in \mathcal{V}} \|\boldsymbol{P}\boldsymbol{e}_u\|_1$. Assume that $\boldsymbol{r}^{(0)} \geq \boldsymbol{0}$ is nonnegative and $P_{\max}$, β are such that $\beta P_{\max} < 1$, then the local diffusion process via LocalSOR with $\omega \in (0, 1)$ has the following properties

1. **Nonnegativity.** $\boldsymbol{r}^{(t+t_i)} \geq \boldsymbol{0}$ for all $t \geq 0$ and $t_i = (i-1)/|\mathcal{S}_t|$.
2. **Monotonicity property.** $\|\boldsymbol{r}^{(0)}\|_1 \geq \cdots \|\boldsymbol{r}^{(t+t_i)}\|_1 \geq \|\boldsymbol{r}^{(t+t_{i+1})}\|_1 \cdots$.

If the local diffusion process converges (i.e., $\mathcal{S}_T = \emptyset$), then T is bounded by

$$T \leq \frac{1}{\omega \overline{\gamma}_T (1 - \beta P_{\max})} \ln \frac{\|\boldsymbol{r}^{(0)}\|_1}{\|\boldsymbol{r}^{(T)}\|_1}, \text{ where } \overline{\gamma}_T \triangleq \frac{1}{T} \sum_{t=0}^{T-1} \left\{ \gamma_t \triangleq \frac{\sum_{i=1}^{|\mathcal{S}_t|} r_{u_i}^{(t+t_i)}}{\|\boldsymbol{r}^{(t)}\|_1} \right\}.$$

Properties of LocalSOR

- For $\alpha \in (0,1), \omega = 1$, the run time of LocalSOR for solving PPR with the stop condition $\|\boldsymbol{D}^{-1}\boldsymbol{r}^{(T)}\|_\infty \leq \alpha\epsilon$ and initials $\boldsymbol{x}^{(0)} = \boldsymbol{0}$, $\boldsymbol{r}^{(0)} = \alpha\boldsymbol{e}_s$ is bounded as

$$\mathcal{T}_{\text{LocalSOR, PPR}} \leq \min\left\{\frac{1}{\epsilon\alpha}, \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\alpha\bar{\gamma}_T} \ln\frac{C_{\text{PPR}}}{\epsilon}\right\}, \qquad \text{where } \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\bar{\gamma}_T} \leq \frac{1}{\epsilon}.$$

  The estimate $\boldsymbol{x}^{(T)}$ satisfies $\|\boldsymbol{D}^{-1}(\boldsymbol{x}^{(T)} - \boldsymbol{f}_{\text{PPR}})\|_\infty \leq \epsilon$.

- For $\alpha \in (0, 1/d_{\max}), \omega = 1$, the run time of LocalSOR for solving Katz with the stop condition $\|\boldsymbol{D}^{-1}\boldsymbol{r}^{(T)}\|_\infty \leq \epsilon$ and $\boldsymbol{x}^{(0)} = \boldsymbol{0}$ and $\boldsymbol{r}^{(0)} = \boldsymbol{e}_s$ is bounded as

$$\mathcal{T}_{\text{LocalSOR, Katz}} \leq \min\left\{\frac{1}{\epsilon(1-\alpha d_{\max})}, \frac{\overline{\text{vol}}(\mathcal{S}_T)}{(1-\alpha d_{\max})\bar{\gamma}_T} \ln\frac{C_{\text{Katz}}}{\epsilon}\right\}, \text{ where } \frac{\overline{\text{vol}}(\mathcal{S}_T)}{\bar{\gamma}_T} \leq \frac{1}{\epsilon}.$$

  The estimate $\hat{\boldsymbol{f}}_{\text{Katz}} = \boldsymbol{x}^{(T)} - \boldsymbol{e}_s$ satisfies $\|\hat{\boldsymbol{f}}_{\text{Katz}} - \boldsymbol{f}_{\text{Katz}}\|_2 \leq \|(\boldsymbol{I} - \alpha\boldsymbol{A})^{-1}\boldsymbol{D}\|_1 \cdot \epsilon$.

Applications

Our accelerated local solvers are ready for many applications such as

- Solve GDEs with discrete-time dynamic graph.
- Accelerate the message propagation in GNN networks.

Alg 1 InstantGNN(LocalGS) $(\mathcal{G}, \boldsymbol{p}, \boldsymbol{r}, \epsilon, \alpha, s, \beta)$

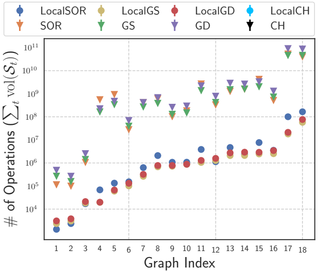
1: **while** $\max_u |r[u]| \geq \epsilon d_{\text{out}}^{1-\beta}[u]$ **do**
2: $\quad$ Push(u)
3: **return** $(\boldsymbol{p}, \boldsymbol{r})$

4: **procedure** Push(u):
5: $\quad p[u] \leftarrow p[u] + \alpha \cdot r[u]$
6: $\quad r[u] \leftarrow 0$
7: $\quad$ **for** v in $\mathcal{N}_{\text{out}}(u)$ **do**
8: $\quad\quad r_s[v] \leftarrow r[v] + \frac{(1-\alpha)\cdot r[u]}{d_{\text{out}}^{1-\beta}[u] d_{\text{out}}^{\beta}[v]}$

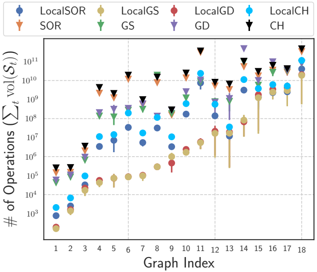
Alg 2 InstantGNN(LocalSOR)$(\mathcal{G}, \boldsymbol{p}, \boldsymbol{r}, \epsilon, \alpha, s, \beta, \omega)$

1: **while** $\max_u |r[u]| \geq \epsilon d_{\text{out}}^{1-\beta}[u]$ **do**
2: $\quad$ Push(u)
3: **return** $(\boldsymbol{p}, \boldsymbol{r})$

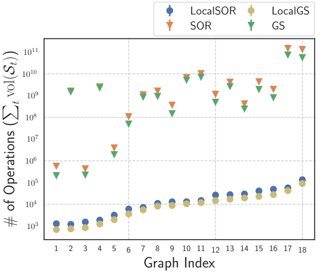
4: **procedure** Push(u):
5: $\quad p[u] \leftarrow p[u] + \alpha \cdot \omega \cdot r[u]$
6: $\quad r[u] \leftarrow r[u] - \omega \cdot r[u]$
7: $\quad$ **for** v in $\mathcal{N}_{\text{out}}(u)$ **do**
8: $\quad\quad r[v] \leftarrow r[v] + \omega \cdot \frac{(1-\alpha)\cdot r[u]}{d_{\text{out}}^{1-\beta}[u] d_{\text{out}}^{\beta}[v]}$



PPR($\alpha = 0.1, \epsilon = \frac{1}{n}$)



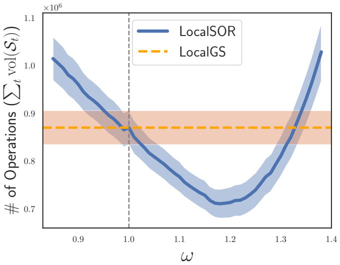
Katz($\alpha = \frac{1}{1+\|\boldsymbol{A}\|_2}, \epsilon = \frac{1}{m}$)

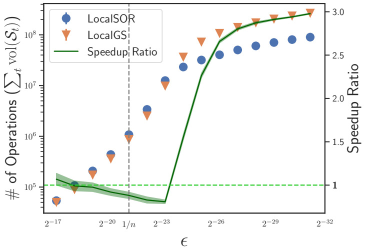


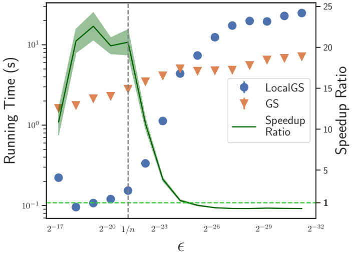
HK($\tau = 10, \epsilon = \frac{1}{\sqrt{n}}$)

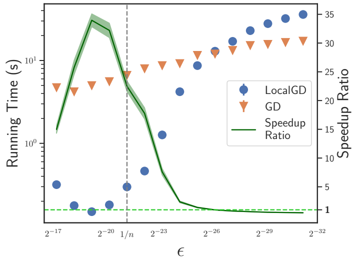
All local methods significantly speed up their global counterparts on all datasets.

Experiments

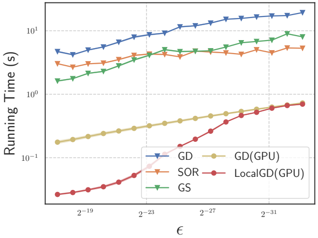


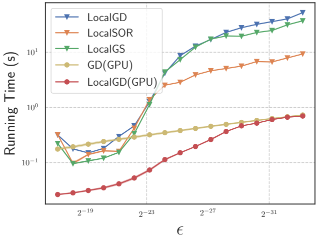


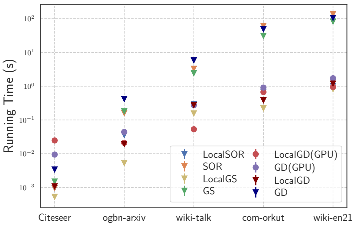




Experiments

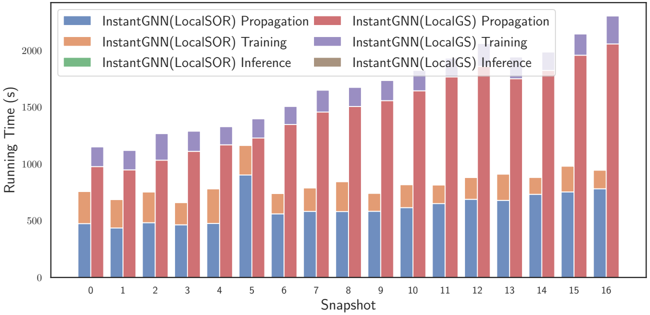


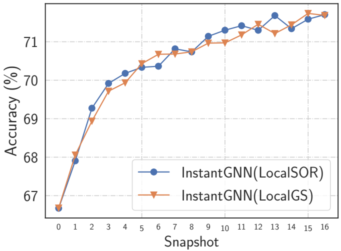


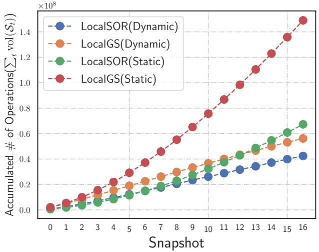


LocalGD (GPU) can be much faster than GD (GPU) and other methods based on CPUs.

Experiments







Thanks!

- Our code is publicly available at `https://github.com/JiaheBai/Faster-Local-Solver-for-GDEs`.
- If you have any questions, contact us:
  Jiahe Bai, jhbai20@fudan.edu.cn;
  Baojian Zhou, bjzhou@fudan.edu.cn.