Incremental and Accuracy-aware Personalized Pagerank through Scheduled Approximation

Fanwei Zhu, Yuan Fang, Kevin Chang, Jing Ying

Zhejiang University
University of Illinois at Urbana-Champaign
Motivation: Useful for ranking, expensive to compute

- Graphs are everywhere, calling for graph-based ranking algorithm

- Personalized Pagerank (PPV)
  - Effective for ranking
  - Expensive to compute
Focus: Efficiency aspect of PPV computation

- Exact computation
- Partial personalization
- Approximation

Prohibitive space/time cost

Arbitrary query

Accuracy vs. Efficiency
Key insight: Scheduled approximation

- Partitioning by importance

- Prioritizing computation

<table>
<thead>
<tr>
<th></th>
<th>PPV(^{(0)})</th>
<th>PPV(^{(1)})</th>
<th>PPV(^{(2)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>0.098</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>0.056</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>0.063</td>
<td></td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>0.021</td>
<td></td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>0.016</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Novelty: Incremental & accuracy-aware

- Incremental query processing

Tour set:

Estimated:

- Accuracy aware

\[ Err^{(i)} = \sum_{q} |PPV(p) - PPV^{(i)}(p)| = 1 - \sum_{q} PPV^{(i)}(p) \]
Challenges: Efficient implementation

- **Challenge 1**: How to effectively partition tours?

- **Challenge 2**: How to efficiently compute each PPV increment?

$$PPV^{(4)} = inc^0 + inc^1 + inc^2 + inc^3$$

importance of tours w.r.t query?
Solution: Hub-based realization

- **Hub nodes**
  - Discriminating: high out-degree decaying reachability
  - Sharing: popularity segments shared by tours

![Diagram of hub nodes](image)

- \( R = 0.054 \)
- \( R = 0.027 \)

\[ R(a \rightarrow b \rightarrow d) = \frac{1}{2} \times 0.85^2 \times 0.15 = 0.054 \]
\[ R(a \rightarrow c \rightarrow d) = \frac{1}{2} \times 0.85^2 \times \frac{1}{2} \times 0.15 = 0.027 \]

\( H = \{a, c\} \)
Solution: Hub-based realization

- **Hub nodes**
  - Discriminating: high out-degree decaying reachability
  - Sharing: popular segments shared by tours

\[ H = \{a, c\} \]

\[ c \rightarrow d \quad \text{shared by} \quad g \rightarrow a \rightarrow c \rightarrow d \]
Challenge 1:
Discriminating provides partition metric

- More hubs, less important
- Partition tours by hub length (# of hubs)

\[ H = \{ a, c \} \]
Challenge 2: Sharing enables reusing overlaps

- Reuse “prefix” among iterations
- Precompute “building blocks”
Results: Fast with accuracy control

- More iterations render better accuracy

- Faster online/offline computation
  - (a) DBLP
  - (b) LiveJournal

Online query time:
- 2~7x faster than H
- 2~5x faster than M

Offline precomputation time:
- 4~11x faster than H
- 3~14x faster than M
Conclusion and future work

○ Conclusion
  • a scheduled approximation strategy to approximate PPVs
  • an efficient hub-based realization
  • up to 7x faster with accuracy control

○ Future work
  • automatic parameter configuration
  • tackling dynamic, evolving graph
  • generalizing to other graph algorithms
Thank you!