Postmortem Computation of Pagerank on Temporal Graphs

Md Maruf Hossain, Erik Saule

University of North Carolina at Charlotte

{mhossa10,esaule}@uncc.edu

This work is supported by the National Science Foundation CCF-1652442

Outline

Traditional graph analysis

Static graph analysis

Given a graph compute some analysis

- PageRank
- Centrality
- Clustering
- ...

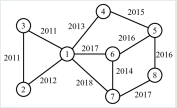
Streaming graph analysis

- The graph is revealed over time
- As new events come in, update an existing analysis to the most current graph.

Postmortem Analysis on Temporal Graph

Temporal Graph

- A graph that changes over time.
- Events at a particular time induce the creation and deletion times of edges or vertices.



Postmortem analysis

- Provided the entire temporal graph
  - all data are available at the beginning
- Perform a particular analysis of the graph
  - Pagerank
  - Centrality
  - Partitioning
- At different points in time
  - possibly sampled uniformly
  - sampled based on activity in the graph

Why study the postmortem setting?

It is the setting of your typical data analyst.
It is understudied.

Can't we use streaming analysis techniques in the postmortem setting?

We can! And they serve as prior work.
But techniques for the streaming setting spend effort managing the uncertainty about the future.
Postmortem techniques know the future.

Can't we use static techniques in the postmortem setting?

We can! And they serve as prior work.
But static techniques assume the graph never changes.

We show for a particular problem that Postmortem analysis can be much faster than both static and streaming

Outline

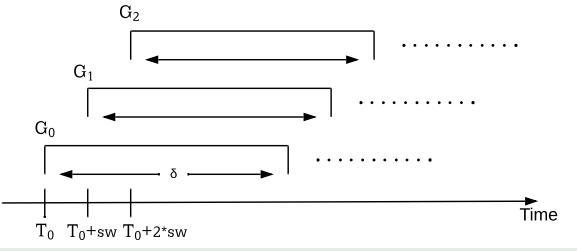
The sliding-window event model

Event induced graphs

- The graph is defined by event that induces vertices and edges.
- At a particular time, the graph is defined by the vertices and edges induced by the events that happened recently, within the last δ time units known as **window size**.

Regularly sampled temporal graph

The temporal graph is sampled at a regular interval *sw*, known as **sliding offset**.



Postmortem Graph Analysis to Compute Pagerank!

- Pagerank:
  - Sort out valuable vertices or ranking them based on the popularity on the graph.
- Why does it matter:
  - Google's link authority algorithm is basically Pagerank.
  - To find a central node or sub-graph within a larger graph.
  - Sports: find the best teams and athletes.
  - Predicting road and foot traffic in urban spaces.
  - Neuroscience: identify parts of the brain that change together as subjects aged.

$$T = \begin{pmatrix} 0 & \frac{5}{23} & \frac{4}{19} & 0 \\ \frac{5}{16} & 0 & \frac{8}{19} & 0 \\ \frac{7}{16} & \frac{15}{23} & 0 & \frac{6}{6} \\ \frac{4}{16} & \frac{3}{23} & \frac{7}{19} & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0.22 & 0.21 & 0 \\ 0.31 & 0 & 0.42 & 0 \\ 0.44 & 0.65 & 0 & 1 \\ 0.25 & 0.13 & 0.37 & 0 \end{pmatrix}$$

$$v_0 = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix} \begin{matrix} \text{(Oxlade-Chamberlain)} \\ \text{(Cazorla)} \\ \text{(Özil)} \\ \text{(Giroud)} \end{matrix} \quad \xrightarrow{\text{After one pass}}$$

$$v^{(1)} = Tv = \begin{pmatrix} 0 & 0.22 & 0.21 & 0 \\ 0.31 & 0 & 0.42 & 0 \\ 0.44 & 0.65 & 0 & 1 \\ 0.25 & 0.13 & 0.37 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix} = \begin{pmatrix} 0.11 \\ 0.18 \\ 0.52 \\ 0.19 \end{pmatrix} \begin{matrix} \text{(Oxlade-Chamberlain)} \\ \text{(Cazorla)} \\ \text{(Özil)} \\ \text{(Giroud)} \end{matrix}$$

Page Rank algorithm to find the most crucial players in soccer match.

Postmortem Graph Analysis to Compute Pagerank!

Pagerank Equation

$$PR(v) = \frac{\alpha}{|V|} + (1 - \alpha) \sum_{u \in \Gamma^-(v)} \frac{PR(u)}{|\Gamma^+(u)|}$$

Computed with a converging iterative process.

Pagerank on Temporal Graph

Input: $Events, sw, \delta, T_0, m$

1: $i \leftarrow 0$
2: **while** $i \leq m$ **do**
3: $\quad PAGERANK_i \leftarrow$ PagerankAlgorithm$(G(T_i,\ T_i + \delta))$
4: $\quad i \leftarrow i + 1$
5: $\quad T_i \leftarrow T_{i-1} + sw$
6: **end while**

Outline

Graph storage system

Edges		Edge Arrival Time	Time Interval		
v_1	v_2		T1	T2	T3
1	2	06/21/2021	✓	×	×
3	5	06/25/2021	✓	×	×
4	6	07/11/2021	✓	✓	×
2	3	08/01/2021	✓	✓	✓
2	4	08/11/2021	✓	✓	✓
5	6	09/13/2021	✓	✓	✓
2	7	10/02/2021	×	✓	✓
4	7	10/05/2021	×	✓	✓
5	7	10/06/2021	×	✓	✓
6	7	10/09/2021	×	✓	✓
1	2	11/05/2021	×	×	✓
1	3	11/06/2021	×	×	✓
2	5	11/09/2021	×	×	✓
3	5	11/12/2021	×	×	✓

Figure: Temporal edge list[Time interval T1 = (6/1/2021-9/15/2021), T2 = (7/1/2021-10/15/2021) and T3 = (8/1/2021-1/15/2022)]

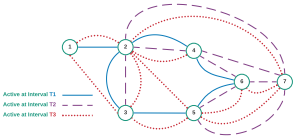


Figure: Temporal Graph

rowA = [0, 3, 9, 12, 16, 21, 24, 28]
colA = [2, 2, 3, 1, 1, 3, 4, 5, 7, 1, 2, 5, 5, 2, 6, 7, 2, 3, 3, 6, 7, 4, 5, 7, 2, 4, 5, 6]
timeA = [06/21/2021, 11/05/2021, 11/06/2021, 06/21/2021, 11/05/2021, 08/01/2021, 08/11/2021, 11/09/2021, 10/02/2021, 11/06/2021, 08/01/2021, 06/25/2021, 11/12/2021, 08/11/2021, 07/11/2021, 10/05/2021, 11/09/2021, 06/25/2021, 11/12/2021, 09/13/2021, 10/06/2021, 07/11/2021, 09/13/2021, 10/09/2021, 10/02/2021, 10/05/2021, 10/06/2021, 10/09/2021]

Figure: Temporal CSR Representation

Where do we stand?

Offline

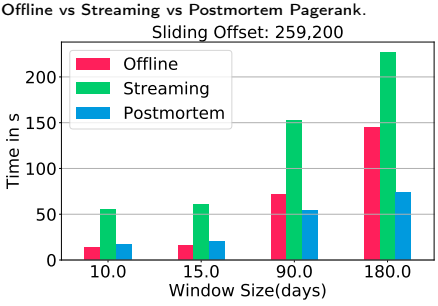
Edge list loaded once.
Build each window graphs in CSR.
Converge Pagerank with a vertex parallel algorithm.

Streaming

STINGER implementation of streaming Pagerank.
Modified to have vertex/edge batches that match the sliding window definition.

Postmortem

One graph representation with timestamp on edges.
Windows are processed one at a time with a vertex parallel algorithm.



Partial initialization

Pagerank is a converging iterative algorithm

- Static usually initializes Pagerank by $\frac{1}{|V|}$.
- The closer the initialization is to the Pagerank, the shorter the convergence.

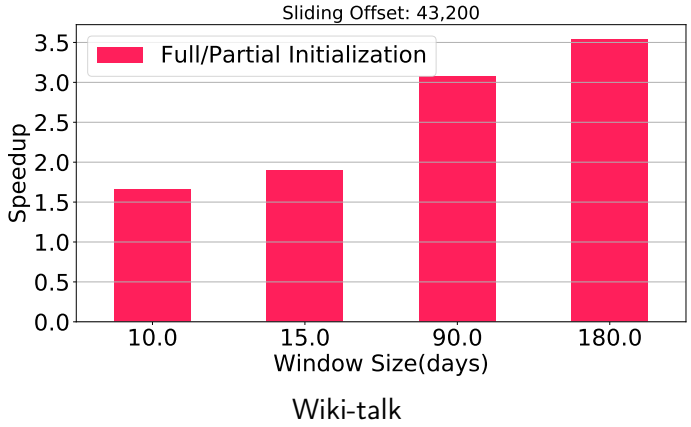
Proportional initialization

- In a temporal analysis, most vertices co-exist in consecutive intervals.
- Custom partial initialization for the existing vertices.

$$PR_i[u] = \frac{|V_i \cap V_{i-1}|}{|V_i|} * \frac{PR_{i-1}[u]}{\sum_{v \in V_i \cap V_{i-1}} PR_{i-1}[v]}$$

- $\frac{1}{|V|}$ for the new vertices.

Partial initialization improves performance significantly



Multi-windowing

Problem

Traversing the graph for a particular window-graph requires traversing every edge.

Solution

Split the temporal graph in multiple chunks: multi-window graphs.
Traversing one timestamp graph only requires to traverse one multi-window.
Some edges need to appear in two multi-window graphs.

Parallelism

Levels of parallelism

- Window-Level parallelization.
- Application-Level parallelization.
- Nested parallelization.

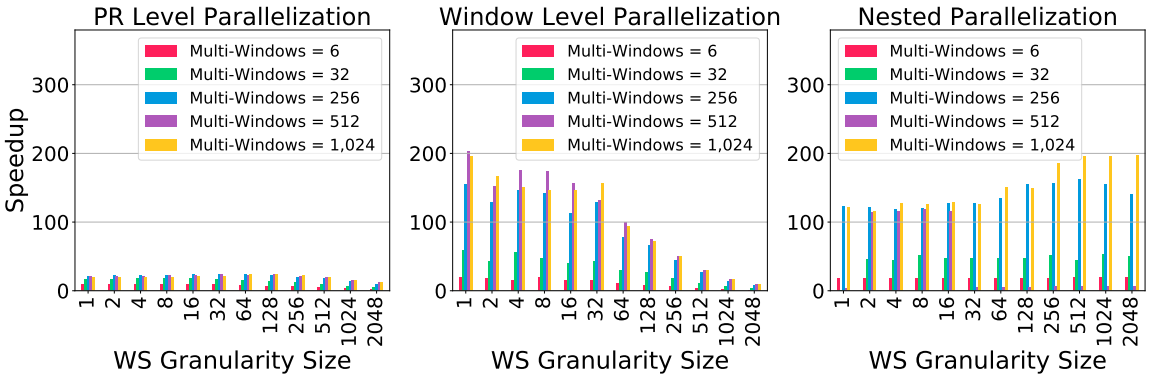
Concerns

- Nested parallelism can bring potential load imbalance because of different size of tasks.
- Processing window-graph t and t+1 at the same time prevents partial initialization.

Work-stealing (implementation with Thread Building Block)

- The threads will be originally allocated a chunk of contiguous work.
- Contiguous chunk will only be broken when the other threads are running out of work.
- Opportunistically use partial initialization.

Impact of window size on the postmortem graph analysis. (wiki-talk)



(Speedups are computed relative to the streaming implementation.)

Computing multiple window-graph at the same time (SpMM)

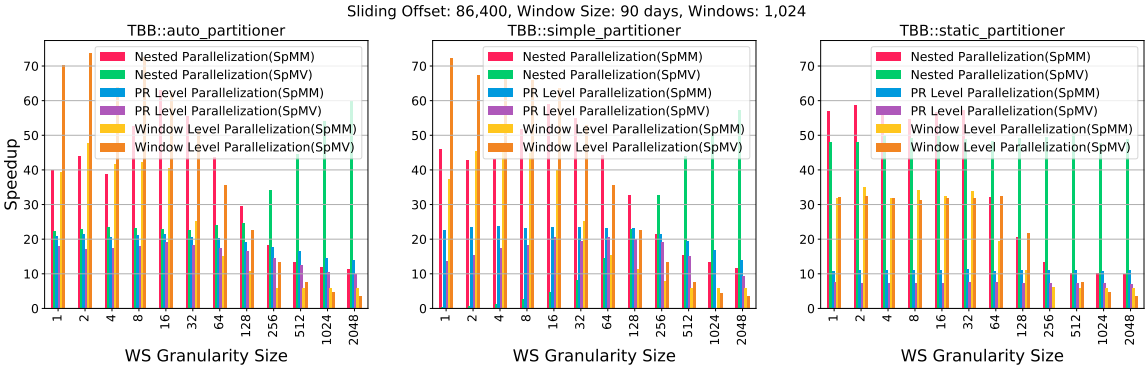
The opportunity

- A multi-window graph represents multiple graphs for different times.
- When two graphs in the same multi-window are traversed, the same memory location in the multi-window graph are traversed.
- When the Pagerank of two window-graphs are computed, once can synchronize the access to the Pagerank vectors for more memory regularity.

Interleaved execution

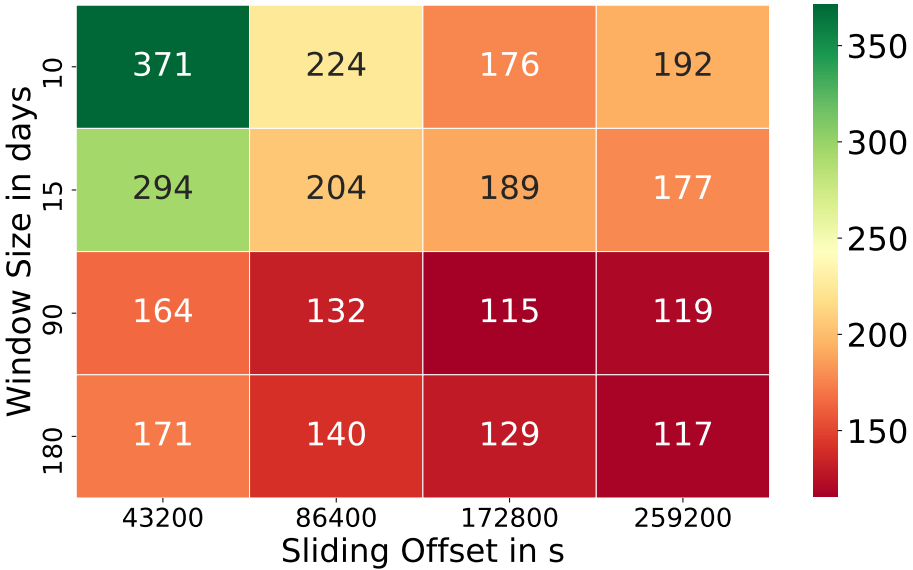
- Compute Pagerank on k graphs at a time inside a multi window.
- Interleave the Pagerank vectors in a Pagerank matrix to regularize memory accesses.
- Compute Pagerank on non consecutive graphs.
  - $g_0, g_{10}, g_{20}, g_{30}$, first
  - $g_1, g_{11}, g_{21}, g_{31}$, then benefit from partial initialization

SpMV vs SpMM for different partitioners (wiki-talk)



Outline

Best performance gain by postmortem compared to Streaming



Conclusion

Postmortem is an important HPC case for temporal network analysis.

- Partial initialization of postmortem Pagerank analysis can bring significance improvement.
- Postmortem analysis enables multi-level parallelization for Pagerank.
- Our work show the significance of the Work-stealing model for the postmortem analysis.
- We show the comparison of SpMV and SpMM Pagerank for the postmortem graph analysis.
- Postmortem analysis can be conducted from 50 to 800 times faster than a streaming analysis.

Thank you!

Questions?

Image references

- https://totalfootballanalysis.com/data-analysis/using-google-page-rank-algorithm-build-up-crucial
- https://www.cs.cmu.edu/~christos/PUBLICATIONS/siam04.pdf
- https://arxiv.org/abs/2003.13212