**AKIN: A Streaming Graph Partitioning Algorithm for Distributed Graph Storage Systems**

Wei Zhang, Yong Chen, Dong Dai
Department of Computer Science, Texas Tech University

---

**Abstract**

- Many graph applications face the challenge of managing excessive and ever-growing graph data.
- A graph partitioning algorithm is needed to distribute graph data onto multiple machines as the graph data comes in.
- Existing graph partitioning algorithms either fail to work on streaming workloads, or leave edge-cut ratio to be further improved.
- Our novel graph partitioning algorithm - AKIN exploit the similarity measure on the degree of vertices in a streaming setting.
- Our evaluation shows that AKIN algorithm is able to further reduce edge-cut ratio while maintaining partition balance.

**Application Scenarios**

- Large Scale Social Network
- Gene Sequencing
- Protein-Protein Interaction Simulation
- Road Construction Simulation
- Data Mining on World Wide Web
- Other Large Scale Instant Graph Analysis

**Methods**

**AKIN Data Structure**

- AKIN data structure is a $t^2$-scale 2D table.
- Top $t$ source vertices who have the largest number of neighbors will be maintained in this in data structure.
- Top $t$ destination vertices who have the largest number of neighbors will be maintained for each source vertex.

**On Edge Stream:**

- We evaluate each edge in the edge stream to see if data migration is needed for increasing data locality.
- Every data migration has to be complied with partition size constraint.

**On Vertex Stream**

**Algorithm 1** Determine partition $P_i$ for arriving vertex $v$

1. $i \leftarrow h(v)$
2. If $i \in \{0, \ldots, k-1\}$ then
3. If not exists($s, P_i$) then
4. assign $v$ to $P_i$
5. end if
6. end if

**Algorithm 2** Determine vertex migration for arriving edge $(u,v)$

1. $i \leftarrow h(v)$
2. If $(K_u \in P_i)$ or $(K_v \in P_i)$ then
3. $max_{score} \leftarrow 0$
4. end if
5. for all $p$ such that $0 \leq p < k$ do
6. $x \leftarrow h(p)$
7. if $p > max_{score}$ and $x < _F(s, P_i)$ then
8. $max_{score} \leftarrow x$
9. $t \leftarrow p$
10. end if
11. end for
12. if $t \neq i$ then
13. migrate $u$ and $v$ to partitions $P_t$
14. $K_u \leftarrow _F(s, P_t)$
15. $K_v \leftarrow _F(s, P_t)$
16. end if
17. maintain reference key $K_u$ at partition $P_t$
18. maintain reference key $K_v$ at partition $P_t$
19. end if
20. end if

**Pseudocode**

**References:**


**Acknowledgement:**

This material is based upon work supported by the National Science Foundation under grant CCF-1409946 and CNS1338078.