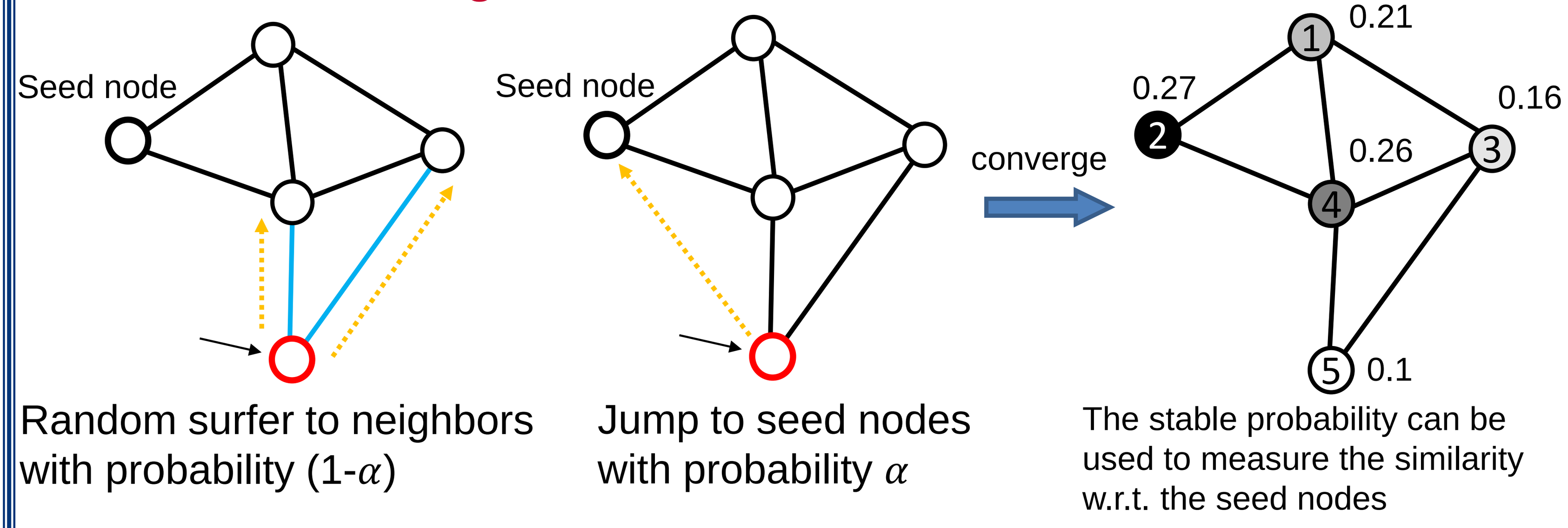
Distributed Algorithms on Exact Personalized PageRank

Tao Guo, Xin Cao , Gao Cong, Jiaheng Lu, Xuemin Lin

NANYANG TECHNOLOGICAL UNIVERSITY · UNSW AUSTRALIA · UNIVERSITY OF HELSINKI

Problem Statement:

❑Personalized PageRank Model



❑Problem

- **Input**: Given a graph G, a set of seed nodes P, and teleport probability α.
- **Output**: Find Personalized PageRank Vector(PPV) $\boldsymbol{r}_P$ which is computed as

$$\boldsymbol{r}_P = (1-\alpha)A^T\boldsymbol{r}_P + \alpha\,\boldsymbol{u}_P,$$

where

- A^T is the normalized adjacency matrix,
- $\boldsymbol{u}_P$ is the user preference vector.

❑ Challenge:

- **Exactness**. Most existing methods focus on approximate PPV computation, exact PPV is hard to compute.
- **Parallel**. It is hard to design scalable distributed algorithm to compute PPV that works in iteration.
- **Costs**. It requires high time, space and network costs for distributed graph computation.

Background:

❑ From PPV to random tours

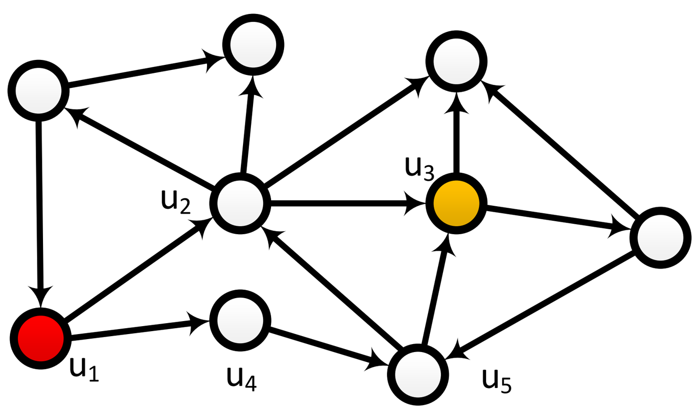
- PPV scores can be computed by random tours

❖ Example:
there are 3 random tours from u_1 to u_3:

t_1: $u_1 \to u_2 \to u_3$

t_2: $u_1 \to u_4 \to u_5 \to u_3$

t_3: $u_1 \to u_4 \to u_5 \to u_2 \to u_3$



- The PPV score can be computed by adding up the weight of all possible random tours.

$$r_{u_1}(u_3) = P(t_1) + P(t_2) + P(t_3)$$

The weight of a tour t

$$P(t) = \alpha(1-\alpha)^{L(t)} \prod_{i=1}^{L(t)} \frac{1}{|Out(w_i)|}$$

❑ Random Tour Decomposition

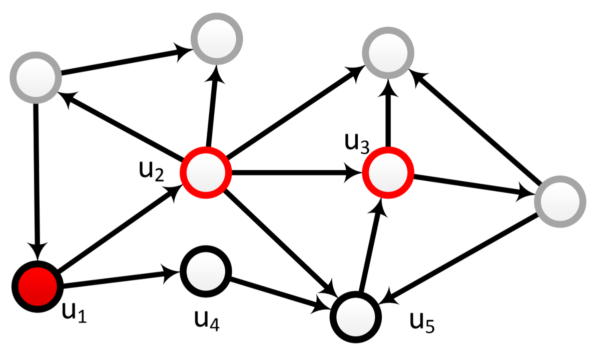
- If we select some nodes to be **hub nodes**

1. The random tours can be decomposed by these hub nodes.
2. Result in two types of tours
   - **Partial vector**: tours passing through no hub nodes
     $pt_{u_1} = P(t_1) + P(t_2) = P(u_1 \to u_4) + P(u_1 \to u_4 \to u_5)$
   - **Skeleton vector**: tours stop at a hub node
     $sk_{u_1} = P(t_3) + P(t_4) + P(t_5) + P(t_6)$
     $= P(u_1 \to u_2) + P(u_1 \to u_2 \to u_3) + P(u_1 \to u_2 \to u_5 \to u_3)$
     $+ P(u_1 \to u_4 \to u_5 \to u_3)$

All possible tours can be constructed by partial vectors and skeleton vectors.

- Example:
Consider a tour $u_1 \to u_2 \to u_3 \to u_6$

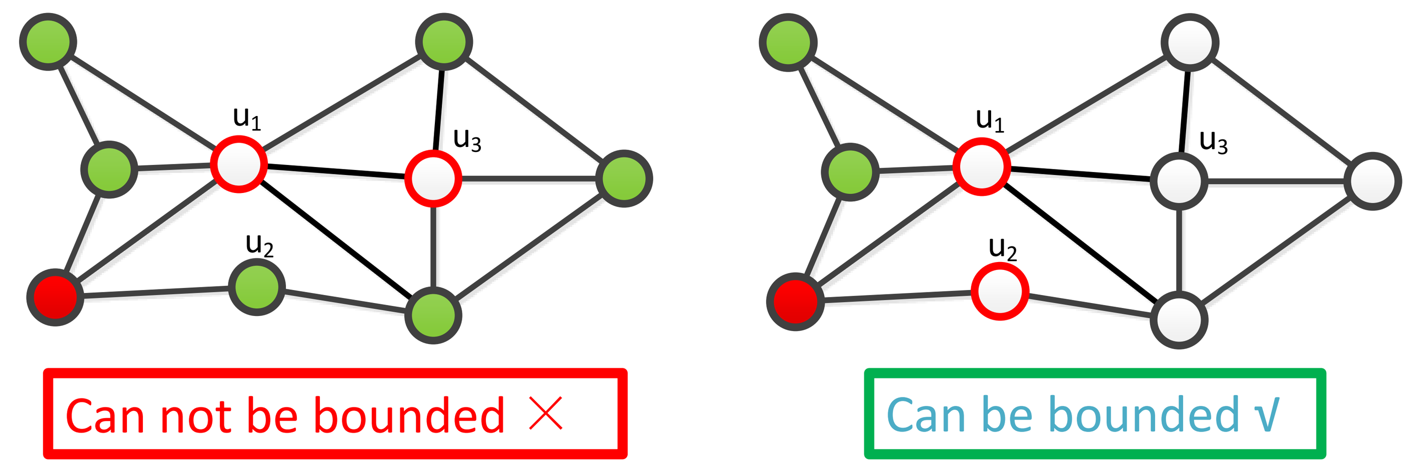
In skeleton vector of u_1 | In partial vector of u_3



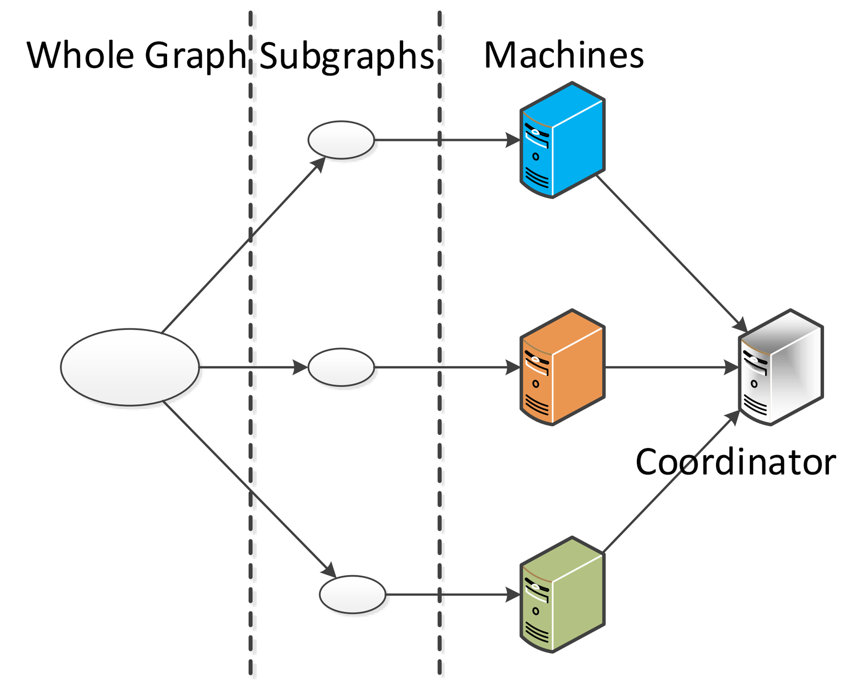
Approaches:

❑Graph Partition Based Algorithm

If we choose the hub nodes that can separate the graph, the size of partial vector can be bounded inside a subgraph.

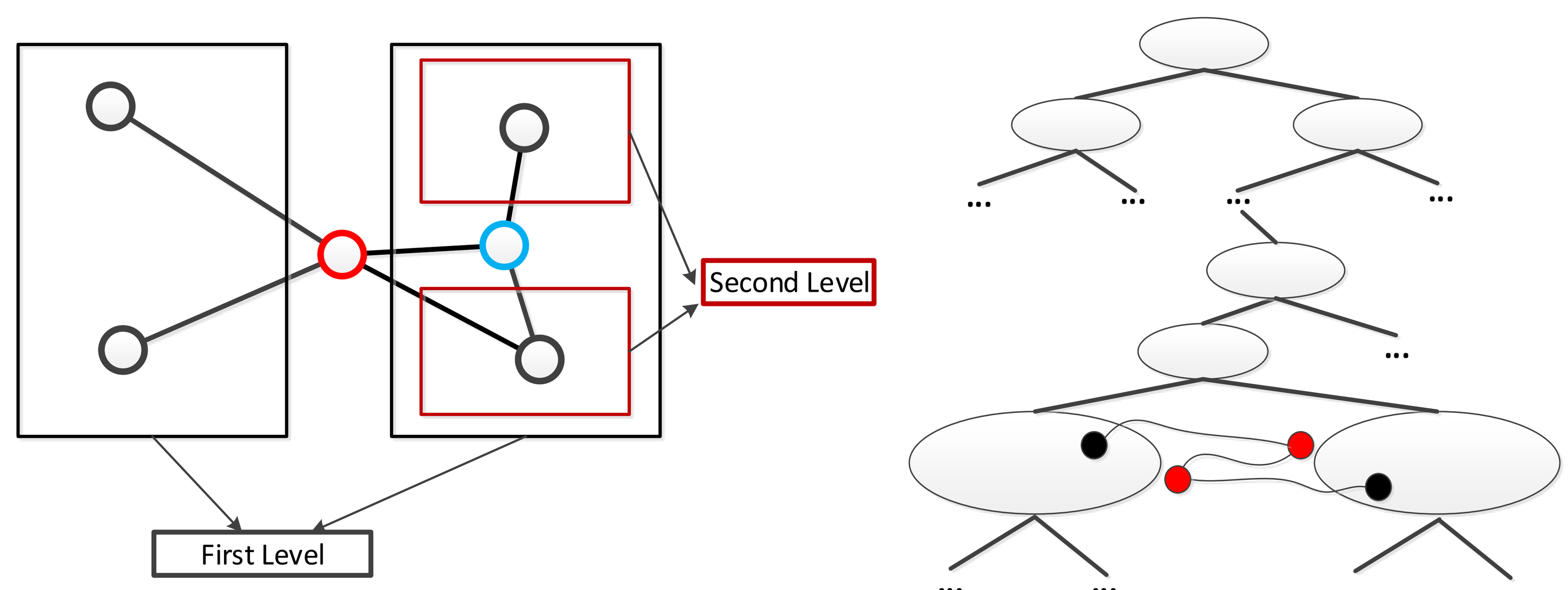


We partition the graph to disjoint components and distribute each subgraph on each machine to compute PPV.



❑Hierarchical Graph Partition Based Algorithm

The partial vector computation in a subgraph is to compute a "local" PPV. We can further partition the subgraph recursively.



Experimental Results:

❑ Baselines:

- Approximate : FastPPV [Fanwei Zhu, PVLDB 2013]
- Exact : Power Iteration
- Graph Processing Systems: Pregel+ [Da Yan, VLDB 2014], Blogel [*Da Yan*, VLDB 2014]