Load Balancing for Traffic in Networks

Feiyu Shi, Dóra Erdös, Evimaria Terzi

Motivation

- Traffic control — find two sets of intersections that have the same traffic.
- Information control — find two groups of nodes that have the same knowledge.

Problem

- **Definition:** Let $G(V, E)$ be a Directed Acyclic Graph (DAG). Given a set of nodes $V_0 \subseteq V$, where $|V_0| = k$, partition it into two groups V_1 and V_2, such that the difference of total number of shortest paths covered by each group is minimal.
- Node centrality $I(v)$: the number of shortest paths covered by node v.
- Group centrality $I(V)$: the number of shortest paths covered by group V.
- Conditioinal centrality $I_A(v)$: the number of shortest paths covered by node v but not by group A.

Framework

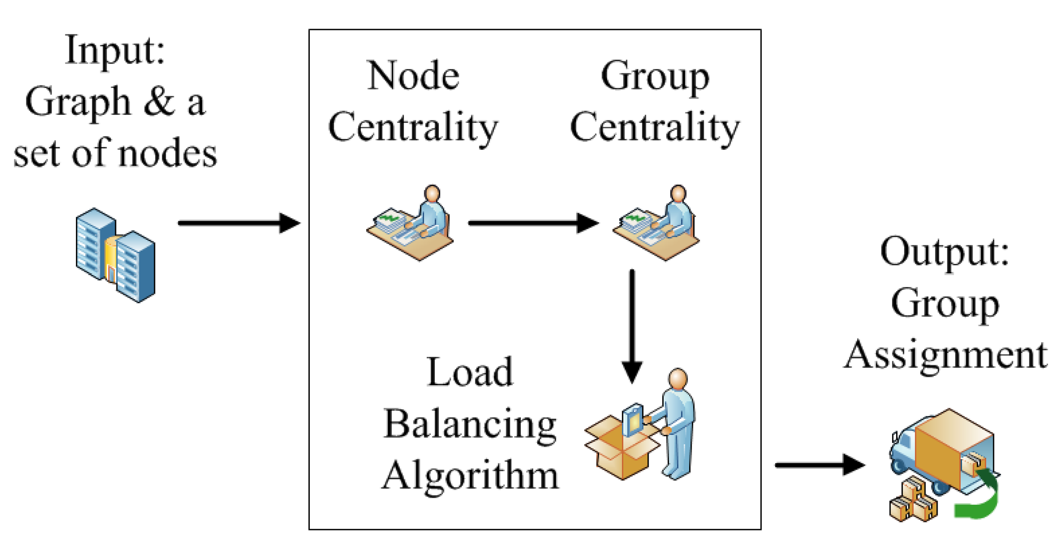

Figure 1: Overview of solving load balancing problem.

Proposed Algorithms

Basic Idea

- We have two baskets V_1 and V_2. V_1 contains all the nodes in V_0, and V_2 is empty initially. Then, we start picking up nodes from V_1 and put it in V_2, until the difference of the group centrality of V_1 and V_2 is minimal.

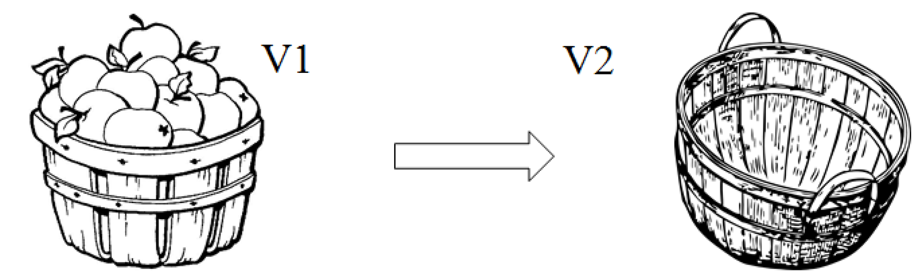

Figure 2: Illustration of proposed greedy algorithms.

- How to get the new difference after moving node v without computing group centrality again?

$$I(V_1') - I(V_2') = (I(V_1) - I(V_2)) - (I_{V_1-\{v\}}(v) + I_{V_2}(v))$$

Greedy Algorithm and Greedy Search Algorithm

- Greedy heuristic: find the node v whose absolute value of the new difference is minimal.
- Greedy algorithm picks up only one node that is the best. It can only give one solution. Time complexity: $\mathcal{O}(k^2|E|\Delta)$.
- Greedy search algorithm tries all possible best nodes at each step, so it's possible to give more than one solutions. Time complexity: best case $\mathcal{O}(k^2|E|\Delta)$, worst case $\mathcal{O}(k!|E|\Delta)$.

Baseline Algorithms

- Brute-force algorithm: enumerates all possible assignments and then gives all the best. $\mathcal{O}(k2^k|E|\Delta)$.
- Full search algorithm: similar to greedy search algorithm, but without using the heuristic. $\mathcal{O}(kk!|E|\Delta)$.

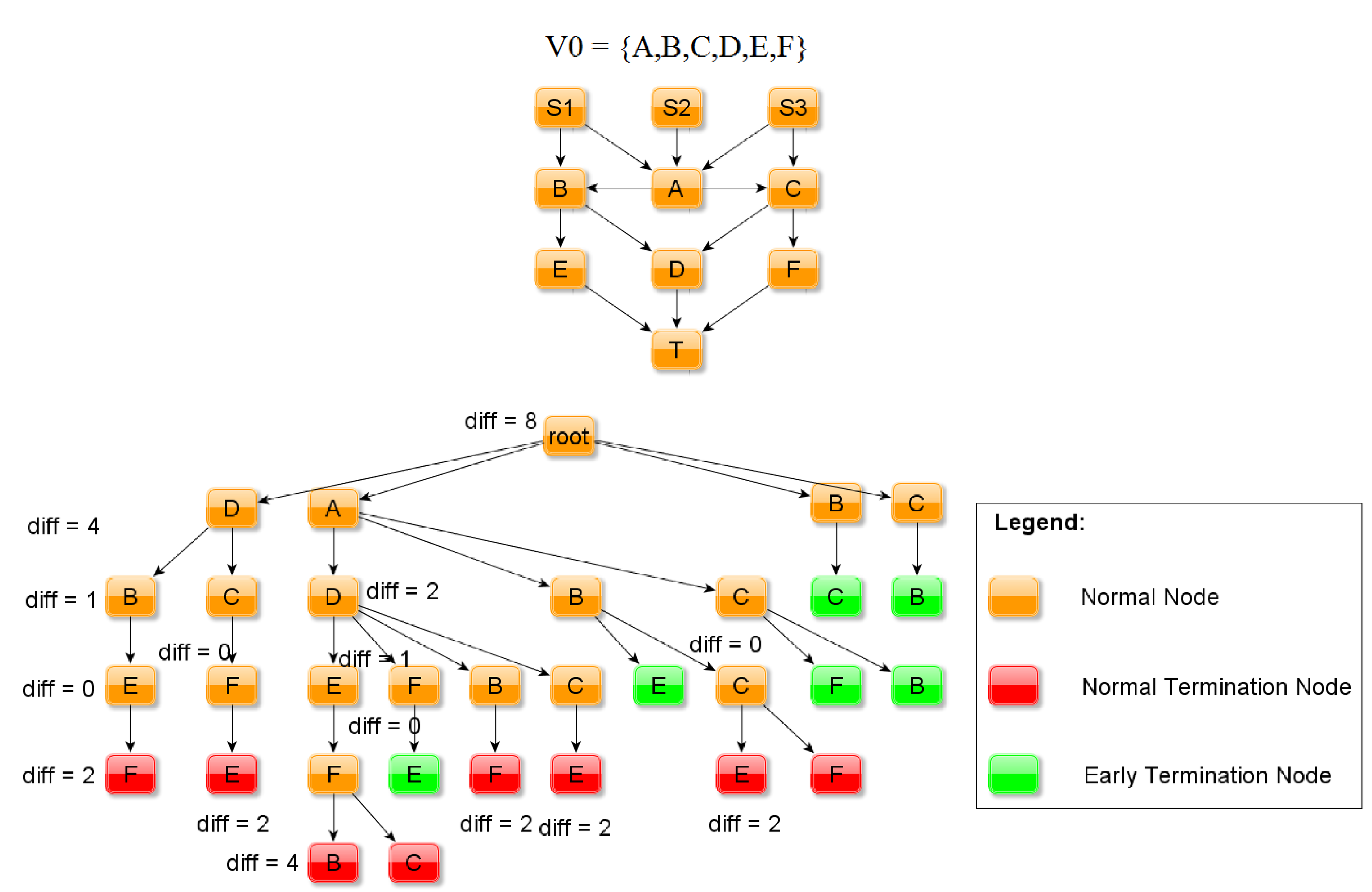

Figure 3: Toy example for greedy search.

Experimental Results

Validation Results

Test	1	2	3	4	5	6	7	8	9	10
B #	16	16	16	8	4	2	1	1	1	1
G #	1	1	1	1	1	1	1	1	1	1
GS #	16	16	16	7	3	1	1	1	1	1
R %	100	100	100	87.5	75	50	100	100	100	100

Test	11	12	13	14	15	16	17	18	19	20
B #	1	1	2	4	4	8	8	8	8	8
G #	1	1	1	1	1	1	0	1	1	0
GS #	1	1	1	3	3	7	0	7	7	0
R %	100	100	50	75	75	87.5	0	87.5	87.5	0

Note: B # is total number of results from baseline methods. G #: the number of greedy method results. GS #: the number of greedy search results . R: retrieval rate.

Figure 4: Table shows the number of correct results retrieved.

Time Complexity

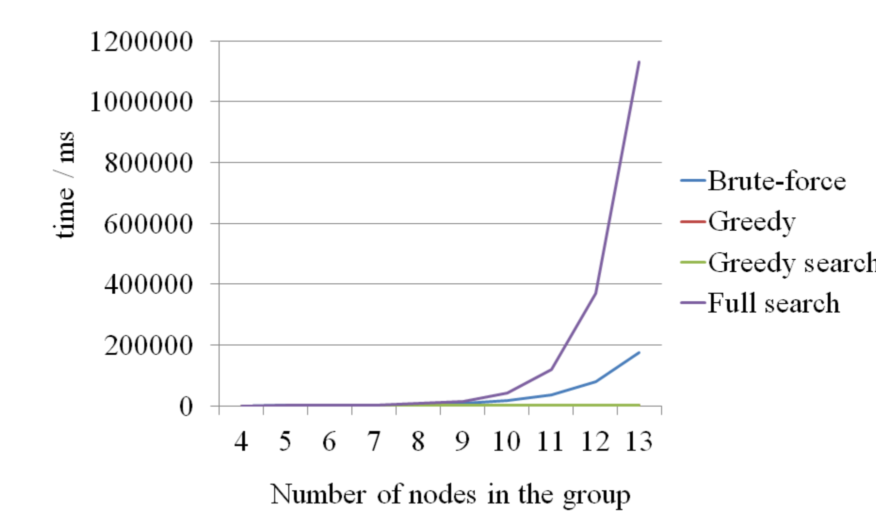

Figure 5: Running time comparison of the four methods on a DAG.

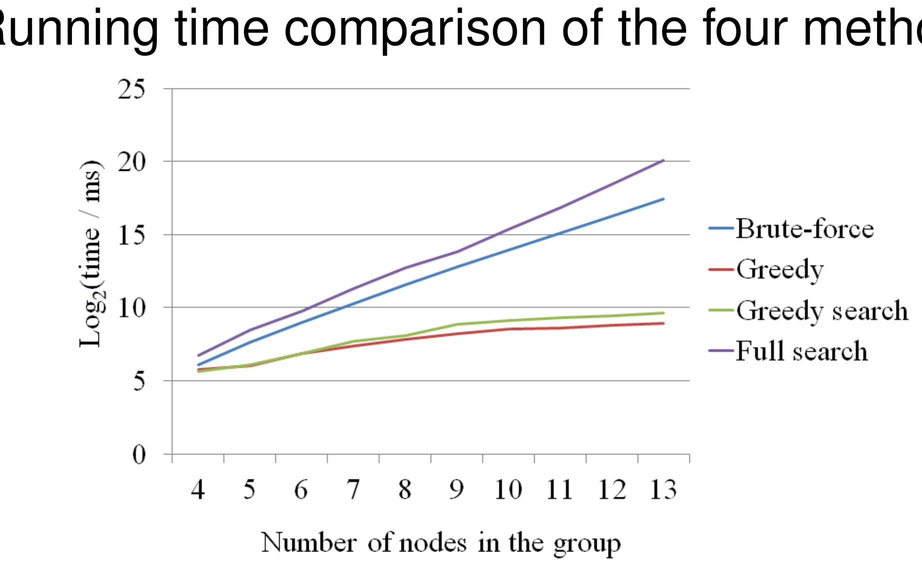

Figure 6: Logarithm time comparison of the four methods.

Reference

1. Ishakian, Vatche, Dóra Erdös, Evimaria Terzi, and Azer Bestavros. "A Framework for the Evaluation and Management of Network Centrality." In *SDM*, pp. 427-438. 2012.