ACM SIGMOD 2014 Programming Contest

Lin Ma, Yuwang Chen, Shian Chen, Yingxia Shao

♣ Peking University ♣

1. Task Statement

The task for this year's contest is to construct a social network analysis system. And the goal is to execute a set of queries as quickly as possible.

There are four types of queries.

2. Query Type 1

Definition: Find the minimum number of hops between two persons p1 and p2. The graph is induced by persons who know each other and have made more than x comments in reply to each other.

Solution:

1) Sort all edges in adjacency list based on comment counts

2) Bidirectional breadth first search

Search size grows hugely through depth!

3. Query Type 2

Definition: Find the interest tag *t* with the largest size of connected component in the graph. Persons in that component must have the interest tag *t* and were born on or later than a given date *d*.

Property: Tag number is relevantly *small*!

People Num	1k	10k	100k	1000k
Tag Num	1458	4567	12144	15676

Come to tag number in real queries, even smaller!

Solution:

1) Offline Computation. Sort the queries by date *d*.
2) For each tag showed up, keep a *Disjoint Set*.
3) In reverse order of birthday, incrementally add persons to each tag's *Disjoint Set* using *Find-Union Algorithm*.
4) Sort the answers and output them based on input order.

4. Query Type 3

Definition: Find the top-*k* similar pairs of persons based on the number of common interest tags. The pair of persons must have location relationship, and be no more than given hops away from each other.

Solution:

1) Linear comparison of common interest tags.

1		3		5	6	7	8	9	10
	2	3	4		6				10

2) Build index for location and pruning skills.

5. Query Type 4

Definition: Find the *k* persons who have the highest closeness centrality values. Persons must have a given tag *t*.

$$c(v) = \frac{|V| - 1}{\sum_{v' \in V} d(v, v')}$$

The BIG one!!

Challenge:

1) Really time consuming to calculate exact centrality values.
2) Can't avoid ALL-Pairs Shortest Path.
3) Approximate methods are not good enough for top-k queries.

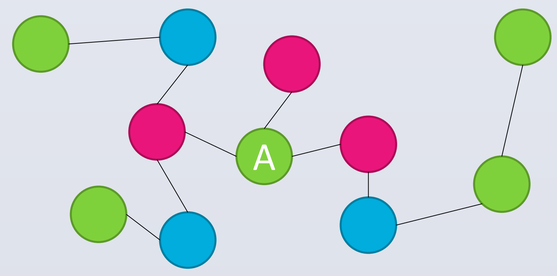
Outline:

1) Use heuristic methods to calculate approximate centrality values. Select possible candidates.
2) Calculate exact closeness centrality values for candidates.
3) Ensure correctness.

For Approximate Values:

Tried: Degree Centrality, Random Sampling [1], FM-Sketch [2]

We finally use 2-hop neighbors' number!



For Exact Values:

We use an incremental method mentioned on ICDE 2014 [3].

For Correctness:

6. Parallelism

Type Level: No enough resource. Only enable 2 types to parallel!

IO consuming: Query Type 1

CPU consuming: Query Type 3, Query Type 4

Memory consuming: Query Type1, Query Type 4

(Query Type 1, 3) -> (Query Type 2, 4)

Query Level: Inter query parallelism inside each type.

Reference

[1] K. Okamoto, W. Chen, and X. Li. Ranking of Closeness Centrality for Large-Scale Social Networks. Faw, 2008

[2] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. J. Comput. System Sci.,1985.

[3] P. Olsen, A. Labouseur, J. Hwang. Efficient Top-k Closeness Centrality Search. ICDE, 2014