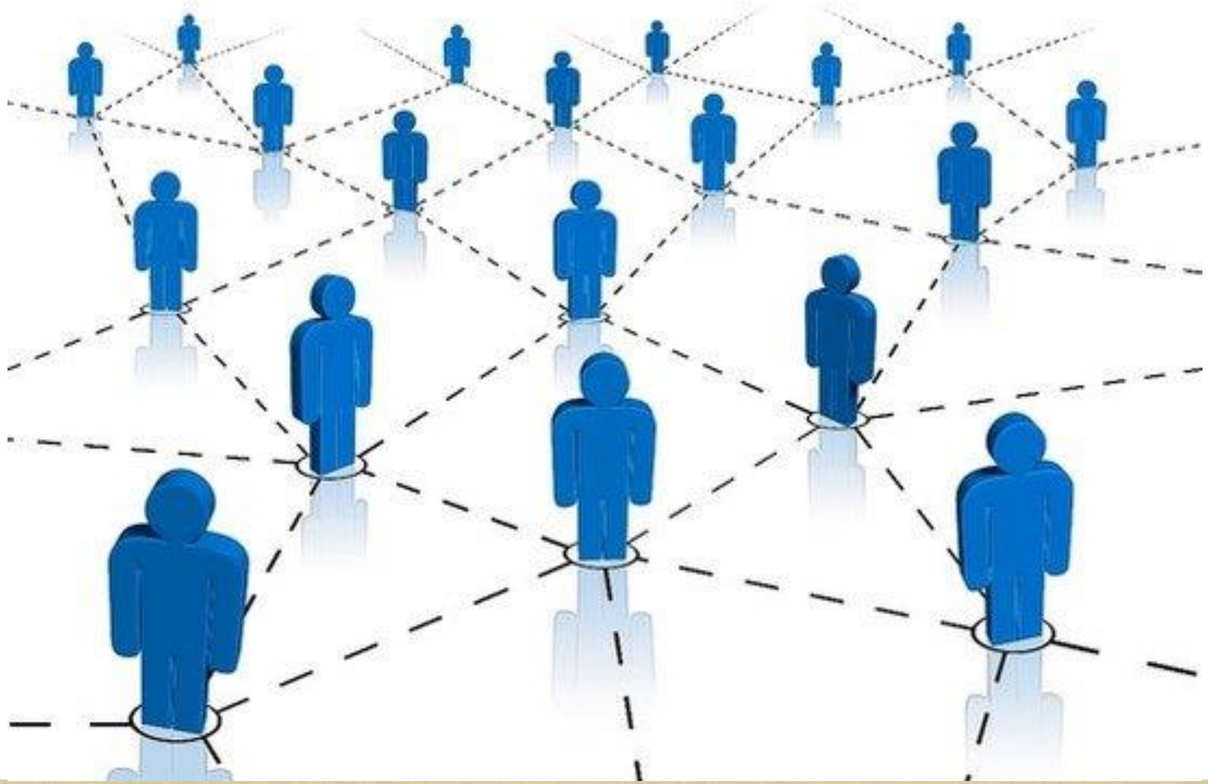


ACM SIGMOD 2014

Programming Contest

Lin Ma, Yuwang Chen, Shian Chen, Yingxia Shao

北京大學

PEKING UNIVERSITY

Introduction

Social network analysis system

Four types of query

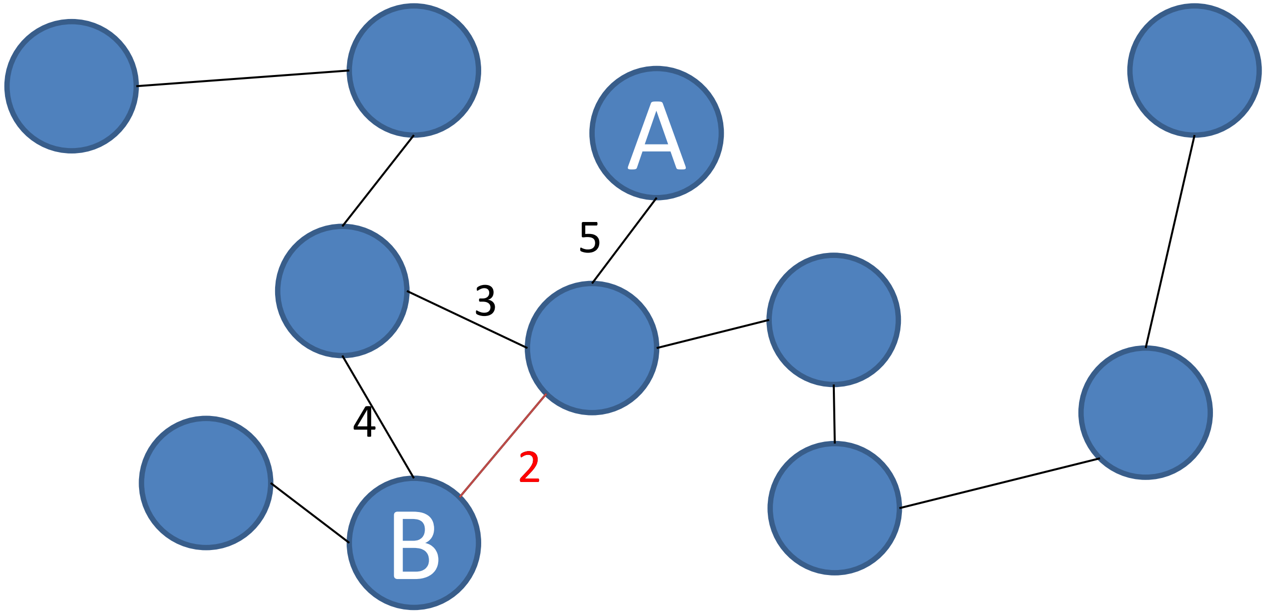
As quickly as possible

What’s for today?

Query Type 1

Definition

- Give *p1, p2, x*
- Minimum hops between *p1, p2*
- At least *x* comments in reply to each other

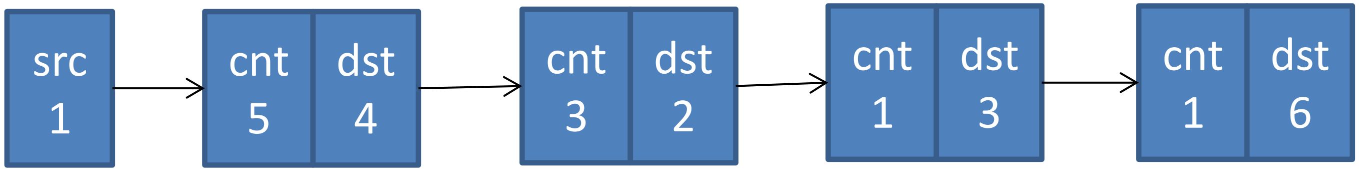


General Idea

- Bidirectional breadth first search

  Search size grows hugely through depth!

- Sort all edges in adjacency list based on comment counts



src 1 → cnt 5 | dst 4 → cnt 3 | dst 2 → cnt 1 | dst 3 → cnt 1 | dst 6

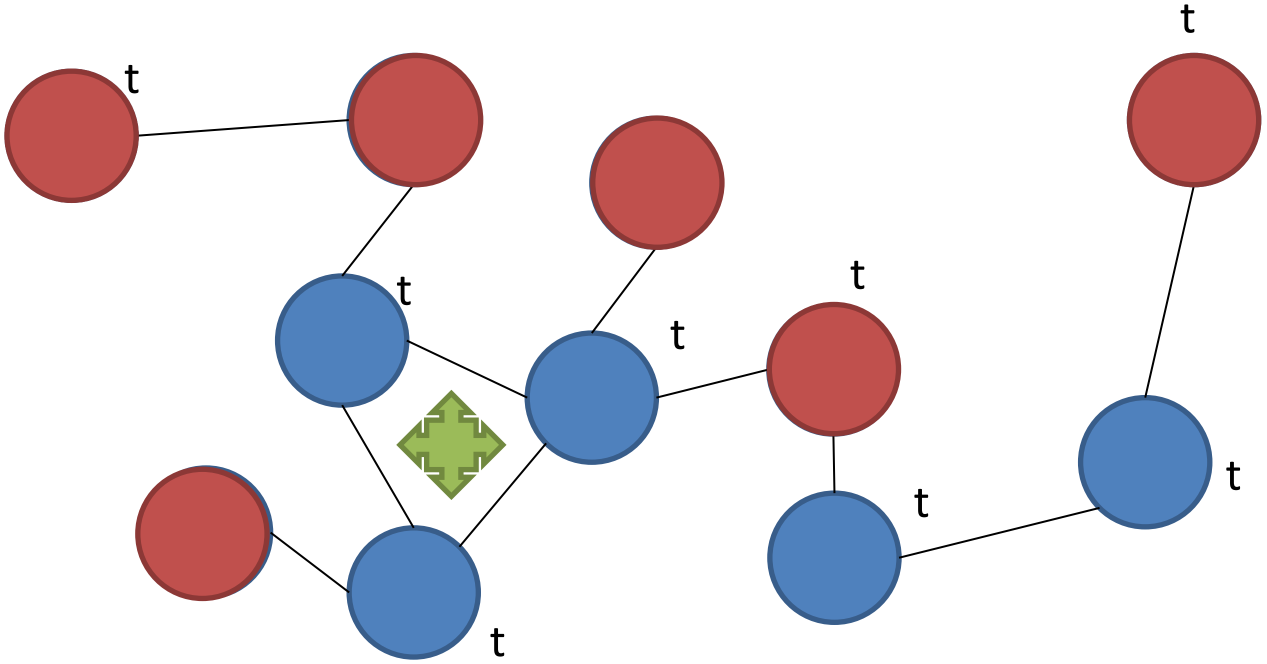
Cost most

Disk I/O 80%~90%

Query Type 2

Definition

- Give *t, d*
- Largest size of connected component
- Have the interest tag *t*
- Born on or later than *d*



Single Tag’s Connected Component

- Disjoint Set and Find-Union Algorithm

Reuse of information?

- Offline Computation. Sort the queries by *d*
- Incrementally add persons to Disjoint Set
- $O(\alpha(n) \cdot (n + m) \cdot T)$

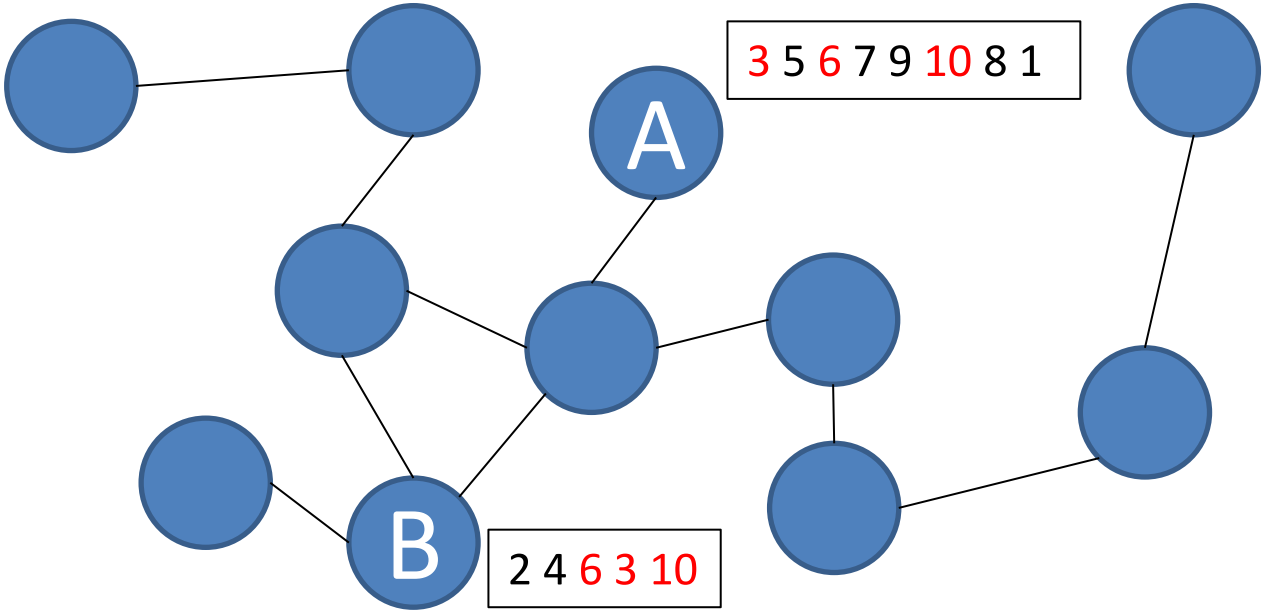
Tag number is relevantly *small*!

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

Query Type 3

Definition

- Give *k, h*
- Top-*k* similar pairs of persons (within h hops) based on the number of common interest tags
- Some location limitation (index)



What we do

- Just search each person’s h-hop neighbor

Common interest comparison

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

Pruning

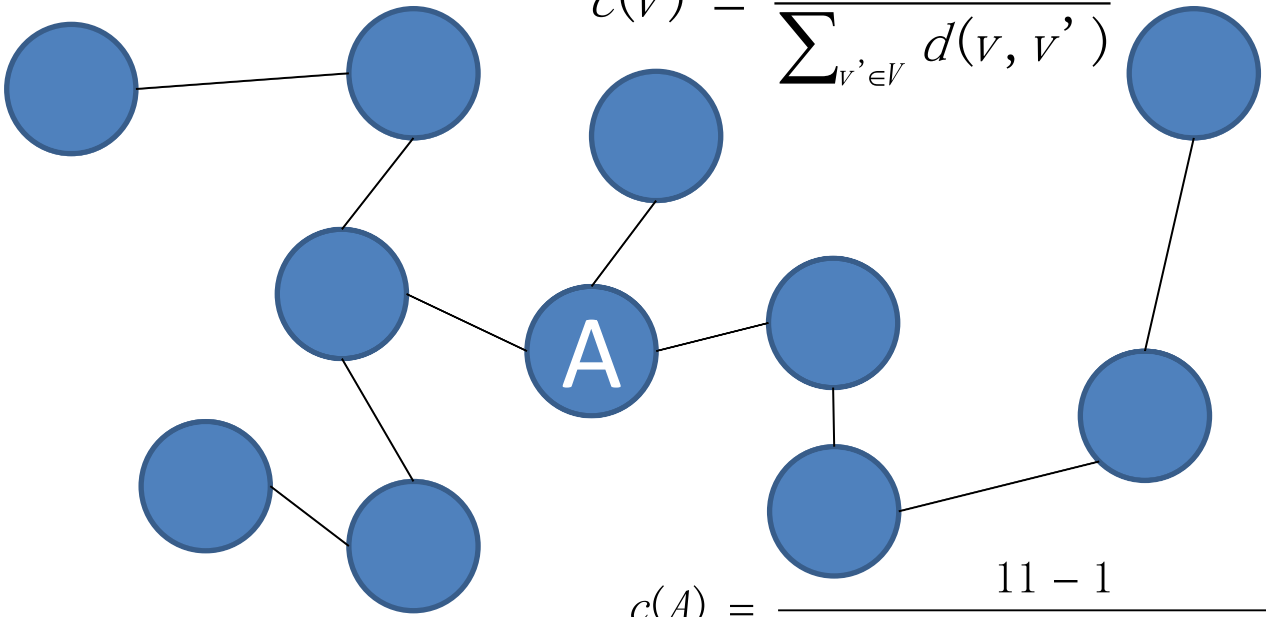
- Eliminate the persons with less tags than top-k common interest tag number

Query Type 4

Definition

- Give *k*, *t*
- Find the *k* persons who have the highest closeness centrality values. Persons must have a given tag *t*.

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



$$c(A) = \frac{11 - 1}{1 \bullet 3 + 2 \bullet 3 + 3 \bullet 3 + 4 \bullet 1} = 0.45$$

CHALLENGE

Really time consuming

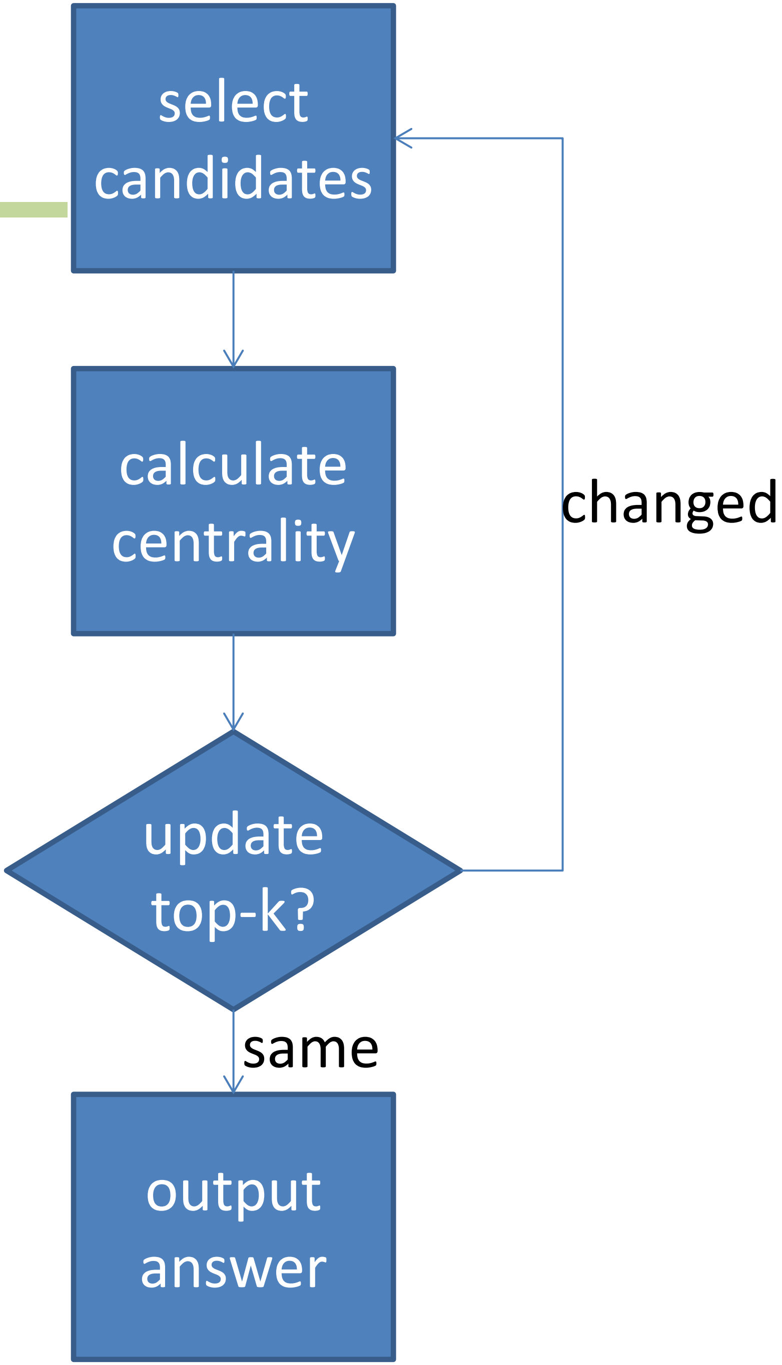
Can’t avoid APSP

Approximate?

Accurate

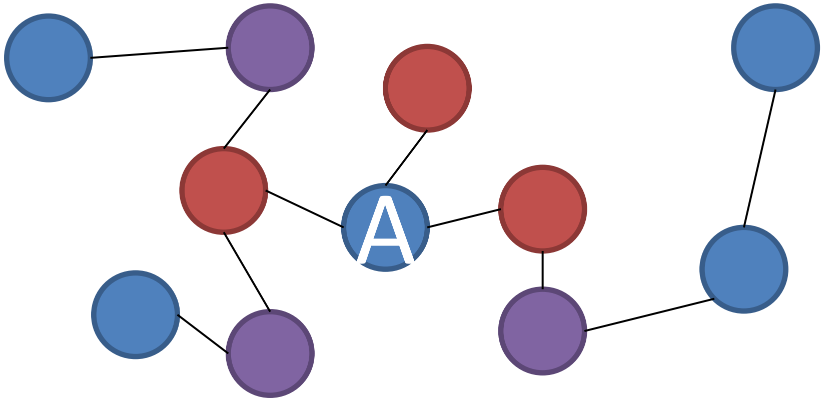
Approximate

Idea


select candidates
calculate centrality
update top-k?
changed
same
output answer

Approximate values

- Tried: Degree Centrality, Random Sampling, FM-Sketch
- Finally: 2-hop neighbors’ number!



Exact values

- Incremental method: Efficient Top-k Closeness Centrality Search, ICDE 2014

Parallelism

Type Level (process)

Limited resource!!

- IO consuming: Query Type 1
- CPU consuming: Query Type 3, 4
- Memory consuming: Query Type1, 4

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

Query Level (thread)

- Share some data structure
- Except Query Type 2

Thanks!
Questions?