SIGMOD Programming Contest 2014

Efficient Implementation of Social Network Analysis System

Team: H_minor_free

T. Akiba, T. Hayashi, S. Hirahara, T. Ikuta, Y. Iwata, Y. Kawata, N. Ohsaka, K. Oka, Y. Yano (The University of Tokyo, Japan)

R. Okuta (Tohoku University, Japan)

Speaker: Takanori Hayashi

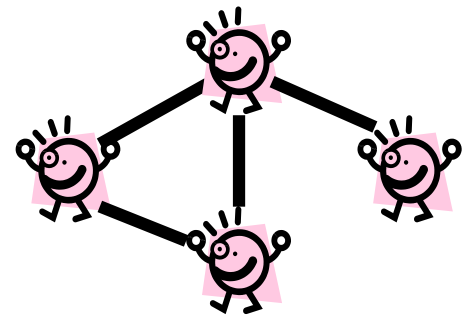
Social networks in this contest.

Synthetic social networks

Generated by
social network benchmark dataset generator
produced by Linked Data Benchmark Council

- Persons
  - Posts (Comments and Replies)
  - Interests
- Friendship between persons

Vertices of the graph



Edges of the graph

$$\#persons \sim 1{,}000{,}000$$

Contest goal

Execute all the queries as quick as possible.

1. Shortest distance over frequent communication paths
2. Interests with Large Communities
3. Socialization Suggestion
4. Most Central People

<u>Evaluation Environment</u>

- Processor 2.67 GHz Intel Xeon E5430
- Configuration 2 processors (8 cores total)
- L2 Cache Size 12MB
- Main Memory 15 GB

Our team

H_minor_free

The University of Tokyo

T. Akiba Y. Iwata
Y. Yano Y. Kawata

Shortest distance query and reachability query
PLL [SIGMOD'13]
PPL [CIKM'13]
PHL [ALENEX'14]
HPLL [WWW'14]

T. Ikuta T. Hayashi S. Hirahara
N. Ohsaka K. Oka

Good at implementing graph algorithms

Tohoku University

R. Okuta

5th place in SIGMOD Programming Contest 2013

Tasks of SIGMOD Programming Contest 2014 are queries on the social networks

Especially, Query1 is the shortest distance query!

We can win?

Our strategy

T. Hayashi
Y. Iwata

Query1
Shortest Distance over frequent communication paths

Y. Yano
K. Oka

Query2
Interests with large communities

T. Ikuta
S. Hirahara

Query3
Socialization Suggestion

Y. Kawata
N. Ohsaka

Query4
Most central people

Analyze the networks: T. Akiba

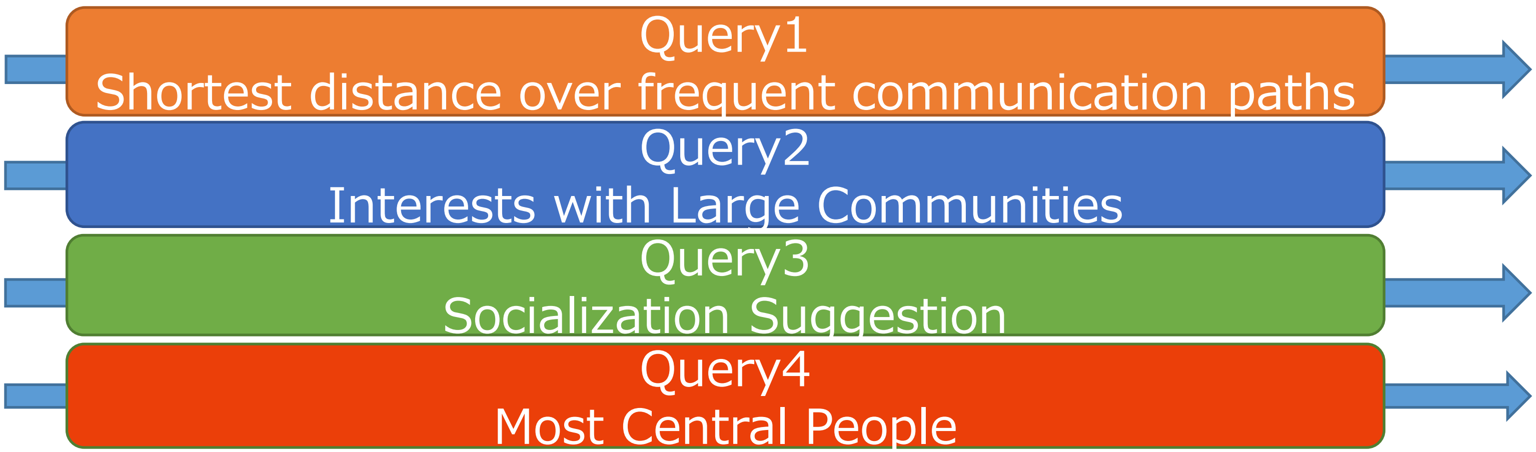
Optimize programs R. Okuta

↓

At March 18th, We became 1st place

Overview of our system

- Implement one program for each query type.
- Run programs simultaneously.



$$TotalElapsedTime \\ = \max(\text{total elapsed time of query type } i \mid 1 \le i \le 4)$$

The most difficult query

At March 27th,(about 20 days before the deadline)

In the middle size network (#persons = 100,000),

- Query1 (1,000 queries) → 9 sec (2 threads)
- Query2 (10 queries) → 2 sec (1 thread)
- Query3 (9 queries) → 6 sec (1 thread)
- Query4 (8 queries) → 18 sec (8 threads)

Query4 is the most difficult query

→ Query4 is the main topic of this talk

Query4: Most Central People (Top-k Closeness Centrality)

Query4(k, t):

Answer k persons with the highest centrality values in the induced subgraph depending on a tag name t

The centrality value $C(p)$ is defined as follows

$$C(p) := \frac{(R(p) - 1)^2}{(n - 1) \times S(p)}$$

$R(p) :=$ #(reachable persons from p)

$S(p) := \sum_{q:reachable\ from\ p} d(p, q)$

$C(p) \propto {}^{1}/_{S(p)}$ in a connected graph.

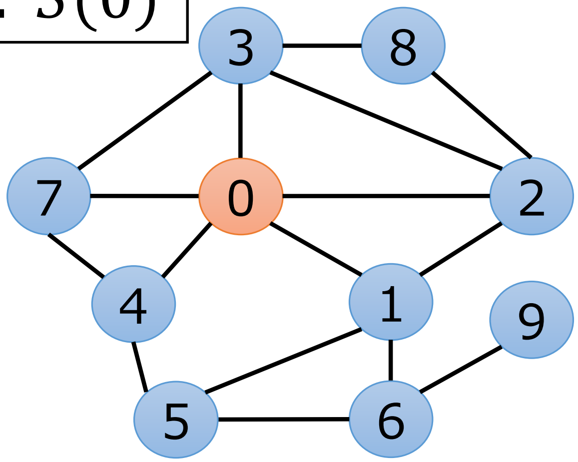
Query4: Most Central People (Top-k Closeness Centrality)

Query4'(k, t):

Answer k persons with the smallest sum of shortest distances in the induced subgraph depending on a tag name t.

$$S(p) \coloneqq \sum_{q: reachable\ from\ p} d(p, q)$$

Example: $S(0)$



$\{v \mid d(0, v) = 1\} = \{1, 2, 3, 4, 7\}$

$\{v \mid d(0, v) = 2\} = \{5, 6, 8\}$

$\{v \mid d(0, v) = 3\} = \{9\}$

$$S(0) = 0 + 1 \times 5 + 2 \times 3 + 3 = 14$$

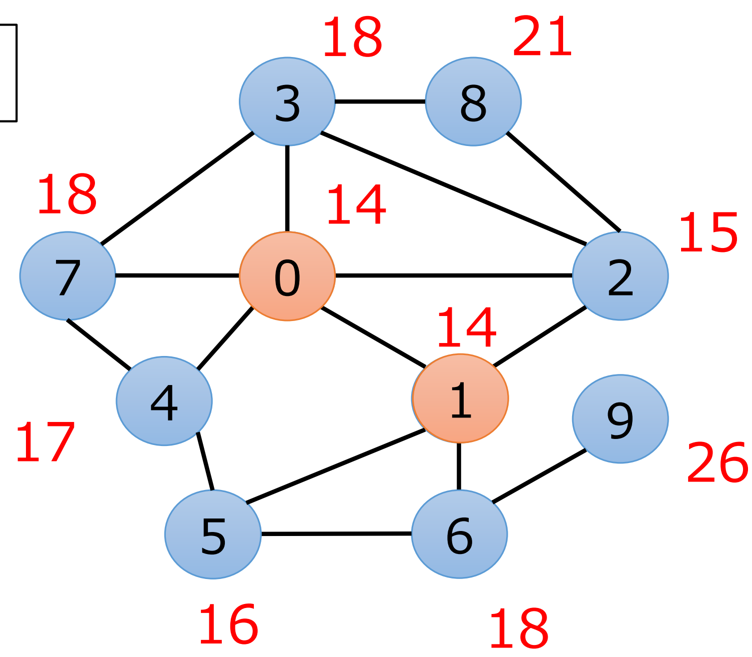
Query4: Most Central People (Top-k Closeness Centrality)

Query4′(k, t):

Answer k persons with the smallest sum of shortest distances in the induced subgraph depending on t a tag name

$$S(p) := \sum_{q:reachable\ from\ p} d(p, q)$$

Example: Query4′(2, t)



$$Query4'(2, t) = \{0, 1\}$$

Naïve approach

$n \coloneqq \#persons$
$m \coloneqq \#friendships$

Conduct BFSs from all persons.

1. $S(p)$ can be computed in $O(m)$ time.
2. Pick up k persons with highest centrality values.

Too inefficient

- $O(nm)$ time. ($n > 50{,}000$ in large networks)

Our algorithm in Query4

Our algorithm should be

Exact → Answer correct values on any instance

Simple → Easy to parallelize
Easy to optimize

Note: $O(km)$ time algorithm is not known.

Our algorithm in Query4

Prune the BFS by using current top-k values.

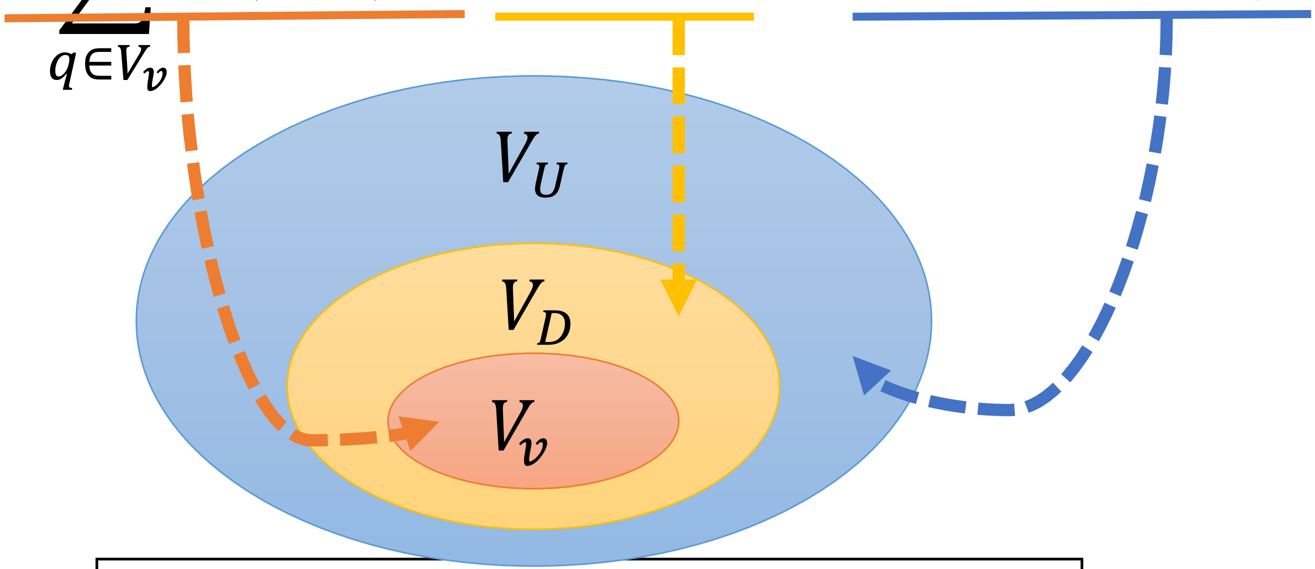
- Conduct BFSs in the decreasing order of degree.
- Estimate lowerbound of $S(p)$ during the BFSs

  If the lowerbound of $S(p)$ is large, the BFS is pruned.

How to estimate good lowerbound?

Straightforward lowerbound

$$L_1(p) \coloneqq \sum_{q \in V_v} d(p,q) + D \times |V_D| + (D+1) \times |V_U|$$



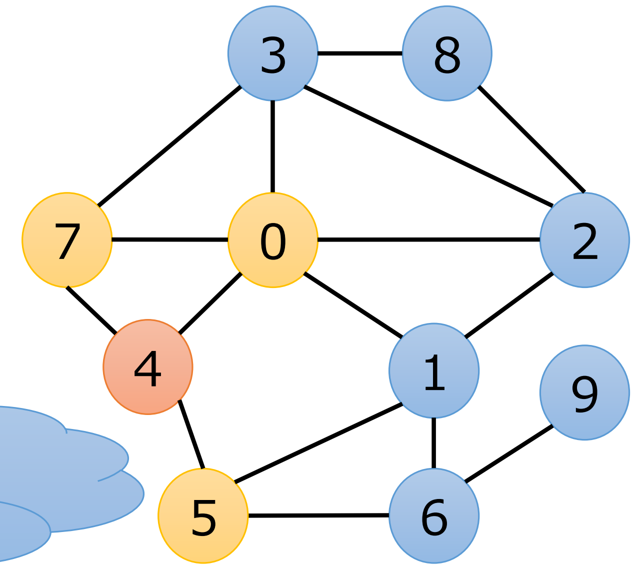
$V_v \coloneqq$ #(already visited persons)
$V_D \coloneqq$ #(persons in the queue)
$V_U \coloneqq$ #(unreached persons)
$D \coloneqq$ distance to vertices in V_D

Straightforward lowerbound Example

Example: BFS from a person 4

Assume $k = 2$ and $S(0) = S(1) = 14$

$L_1(4) = 0 + 1 \times 3 + 2 \times 6 = 15 > 14$



$V_v = \{4\}$

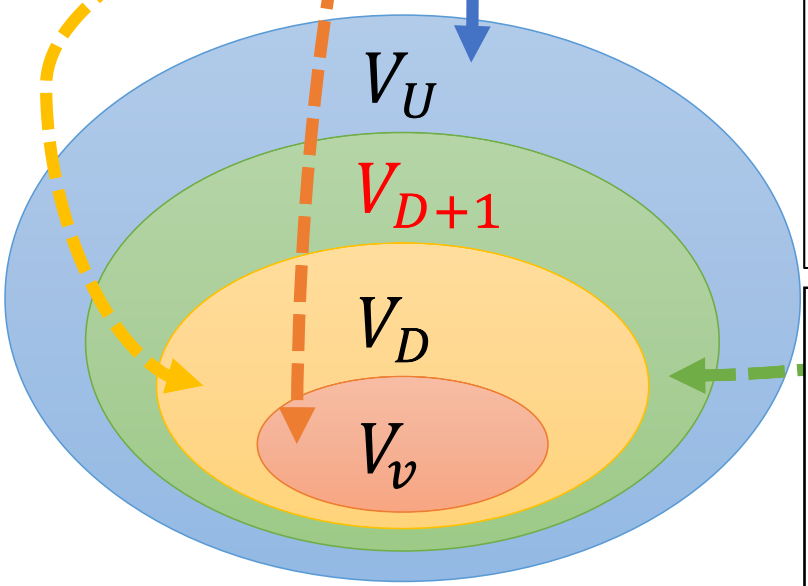
$V_D = \{0, 5, 7\}$

$V_U = \{1, 2, 3, 6, 8, 9\}$

BFS from a person 4 is pruned!

Tighter lowerbound

$$L_2(p) := \sum_{q \in V_v} d(p,q) + D \times |V_D| + (D+1) \times |V_{D+1}|$$
$$+ (D+2) \times (|V_U| - |V_{D+1}|)$$



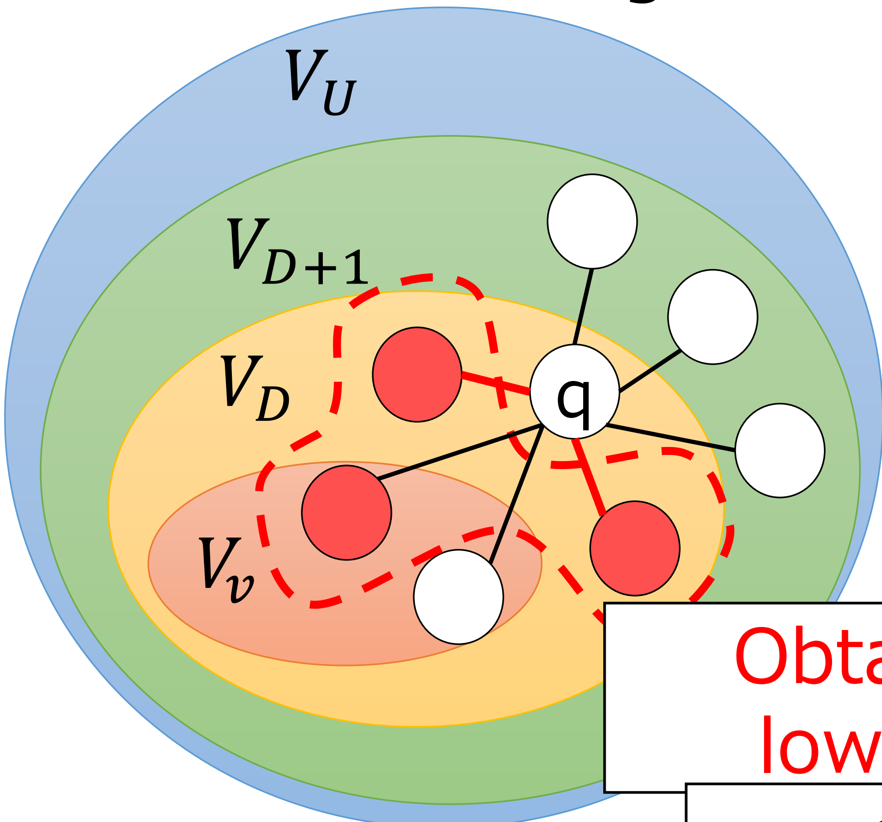
Estimate the number of persons whose distance from p is $D+1$.

$$|V_{D+1}| := \sum_{q \in V_D} (\deg(q) - 1)$$

V_v := #(already visited persons)
V_D := #(persons in the queue)
V_U := #(unreached persons)
D := distance to vertices in V_D
V_{D+1} := #(persons whose distance from p is $D+1$)

Update lowerbound

After scanning the edges from $q \in V_D$,



are newly reached vertices or the parent vertex in shortest path tree.

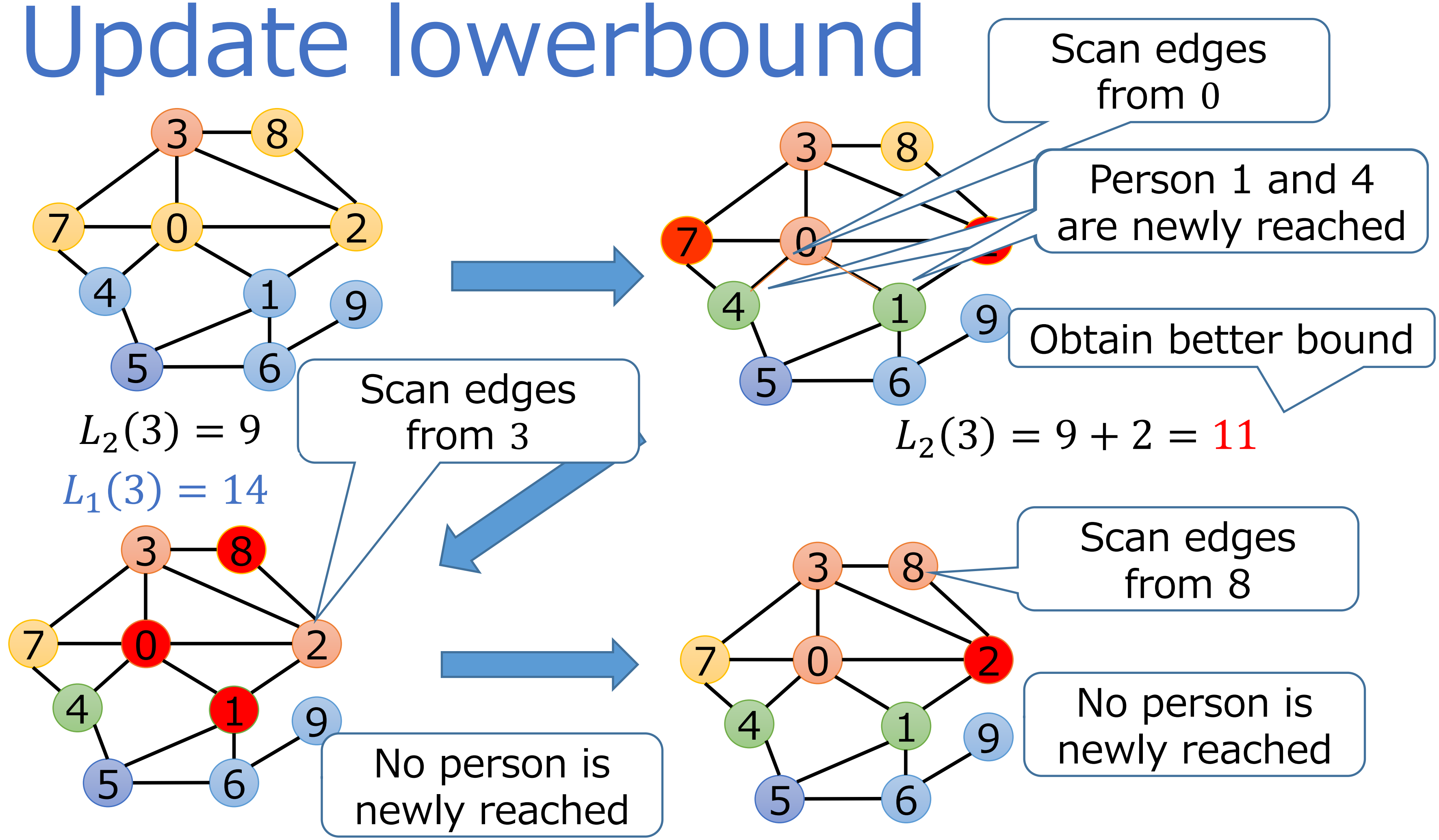
$|V_{D+1}|$ is overestimated by #

$$L_2(p) \leftarrow L_2(p) + \#$$

Obtain tighter lowerbound!

$$= \deg(q) - 1 - \#(\textit{newly visited vertices})$$

Update lowerbound



$L_2(3) = 11 + 3 = 14$

$L_2(3) = 14 + 1 = 15 > L_1(3) = 14$

BFS from a person 3 is pruned before scanning the edges from person 7!

Speedup by pruning

In 5,000 persons network induced by a tag t from 10,000 persons network,

Naïve: 425 ms

Pruning(L_1): 35 ms

Pruning(L_2): 28 ms

15x faster!

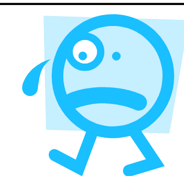
Results

In the middle size network $(\#people = 100{,}000)$,

Match 27th	April 15th
Query1: 9 sec.	Query1: 6 sec.
Query2: 2 sec.	Query2: 2 sec.
Query3: 6 sec.	Query3: 2 sec.
Query4: 18 sec.	Query4: 6 sec.

Structure of the system is still naïve…

Each program is written in the different way.



Summary

Query1 — Shortest Distance over frequent communication paths (T. Hayashi, Y. Iwata)

Query2 — Interests with large communities (Y. Yano, K. Oka)

Query3 — Socialization Suggestion (T. Ikuta, S. Hirahara)

Query4 — Most central people (Y. Kawata, N. Ohsaka)

Analyze the networks (T. Akiba)

Optimize programs (R. Okuta)

At March 18th, 1^{st} place

At April 15th, 3^{th} place

- BFS with pruning
- Lowerbound estimation
- Other optimization