UCI University of California, Irvine

GraphQ: Graph Query Processing with Abstraction Refinement

-- Scalable and Programmable Analytics over Very Large Graphs on a Single PC

Kai Wang, Guoqing Xu, University of California, Irvine

Zhendong Su, University of California, Davis

Yu David Liu, SUNY at Binghamton

Big Graph Is Everywhere

Motivation

- The existing graph processing systems all focus on whole graph computation, it's time-consuming
  - Pregelix, [Y. Bu et al., VLDB'15]

    e.g., 32-node cluster, 30 minutes, 70GB web graph
- The whole graph computation seems an overkill for some real-world applications

Find one path between LA and NYC within a certain distance

Find a target group in Southern California with given property (e.g., size)

Find a website with a very high Page Rank value

Analytical Queries

- <u>Observation:</u>

  Many queries can be answered by exploring only a small fraction of the input graph

Questions

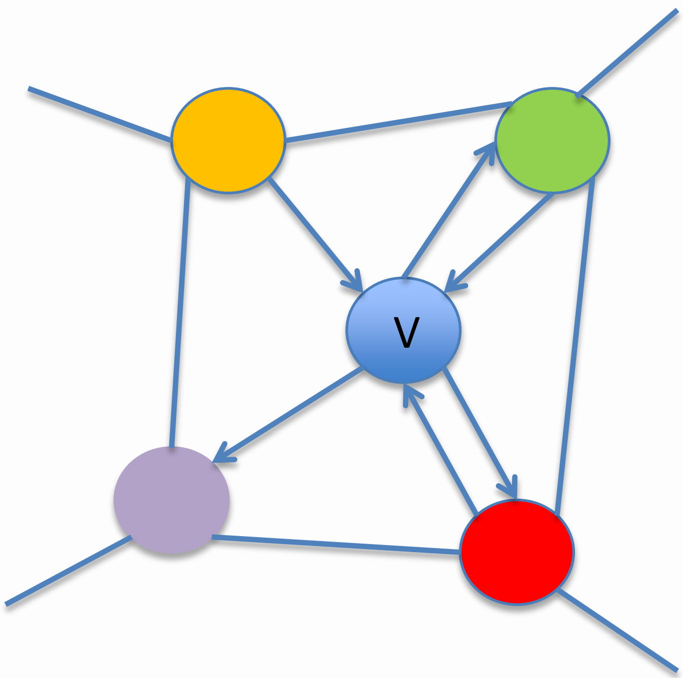
- Can we answer analytical queries **efficiently** by doing computation on **partial graphs**?
- If partial graphs are sufficient, can we process on a **single PC** without resorting to clusters?
  - GraphChi [A. Kyrola et al., OSDI'12]
  - X-Stream [A. Roy et al., SOSP'13]

- GraphQ – Graph Query Processing with Abstraction Refinement over Very Large Graphs on a Single PC

Background

- Graph G = (V,E)
  - each vertex v ∈ V has an associated value
  - each edge e ∈ E has an associated value
  - Vertex and edge values can be modified

Background

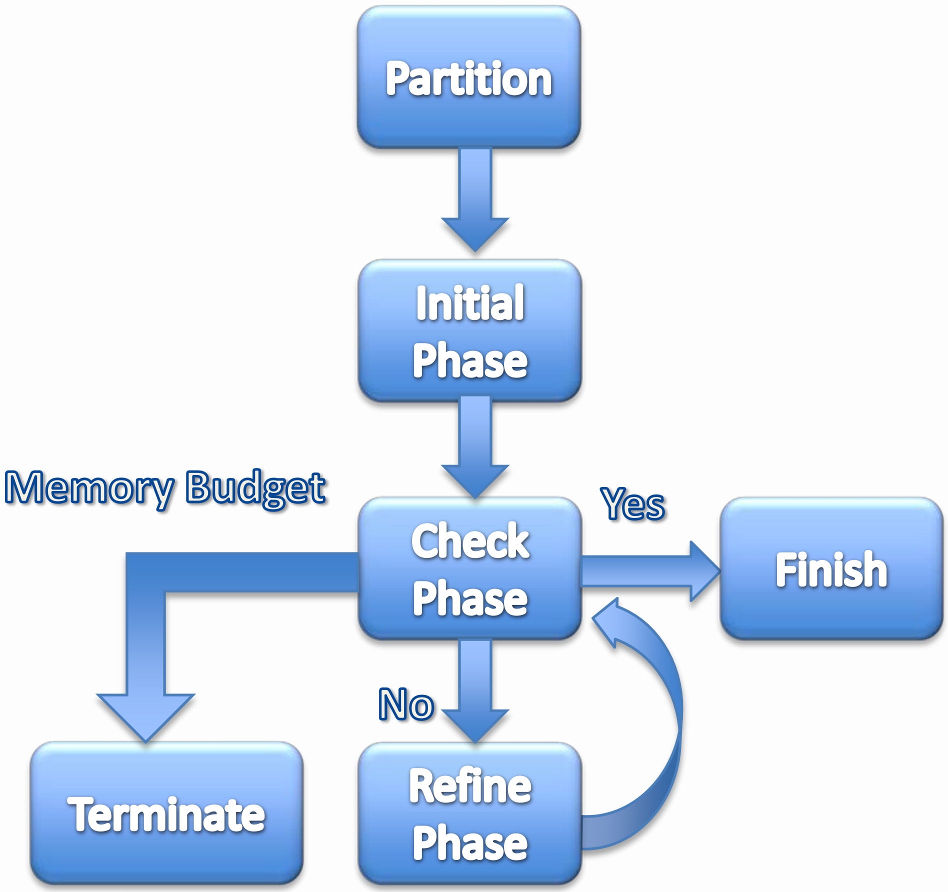


Vertex Centric Programming Model

Vertex V

- Read values from incoming edges
- Update(user-defined function)
- Write values to out-going edges

Overview



Divide whole graph into partitions

Compute local solutions on subgraphs without inter-partition edges

Check if query answered

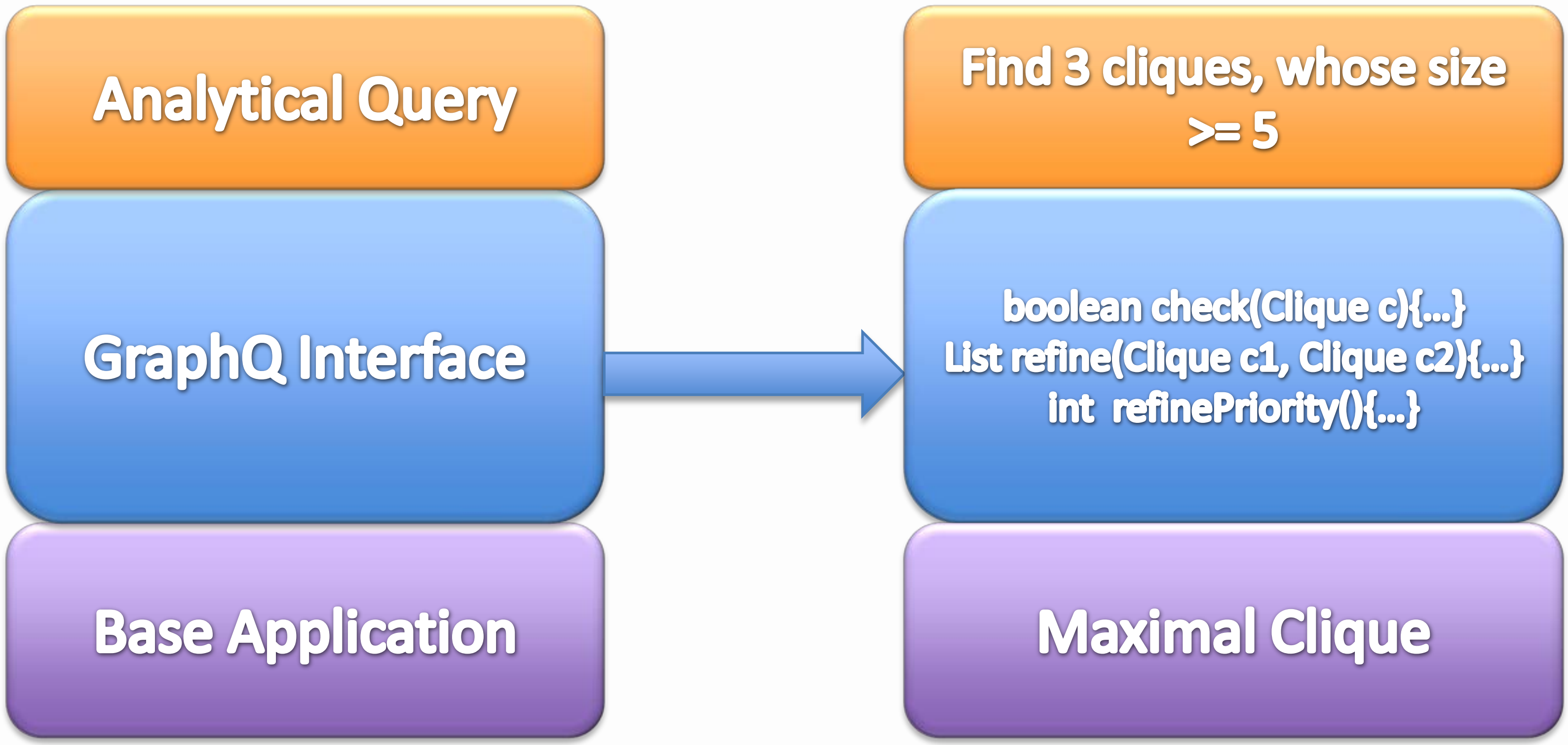
Yes, finish

No, merge partitions
A broader scope of query answering

Reach memory budget, terminate

How To Use GraphQ

Find Δ entities from the graph with a given quantitative property



Goal

Select partitions to merge, hoping that the query can be answered by merging only a very small number of partitions

Abstraction Refinement

[Clarke et al., CAV'00]

- **Abstraction**

  Build abstraction graph to summarize the concrete graph. Abstraction graph serves as a navigation map for checking edge feasibility

- **Refinement**

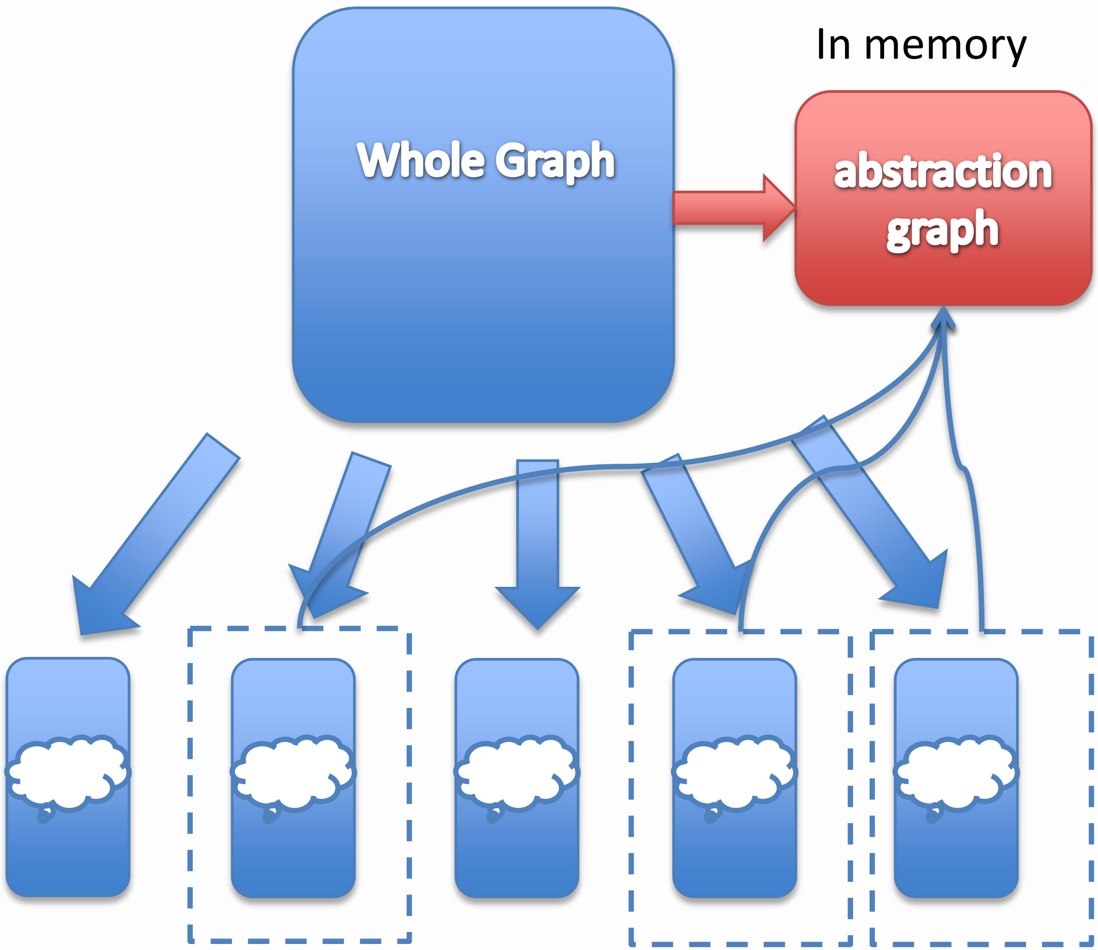
  Merge partitions, recover inter-partition edges to provide a broader scope for query answering

Abstraction Function

An *abstraction graph* summarizes a *concrete graph* using *abstraction function*

A sound abstraction:

- All concrete vertices have abstract vertices
- Edge feasibility: If there is no abstract edge, it is guaranteed there is no concrete edge



Partition

Initial Phase

Check Phase

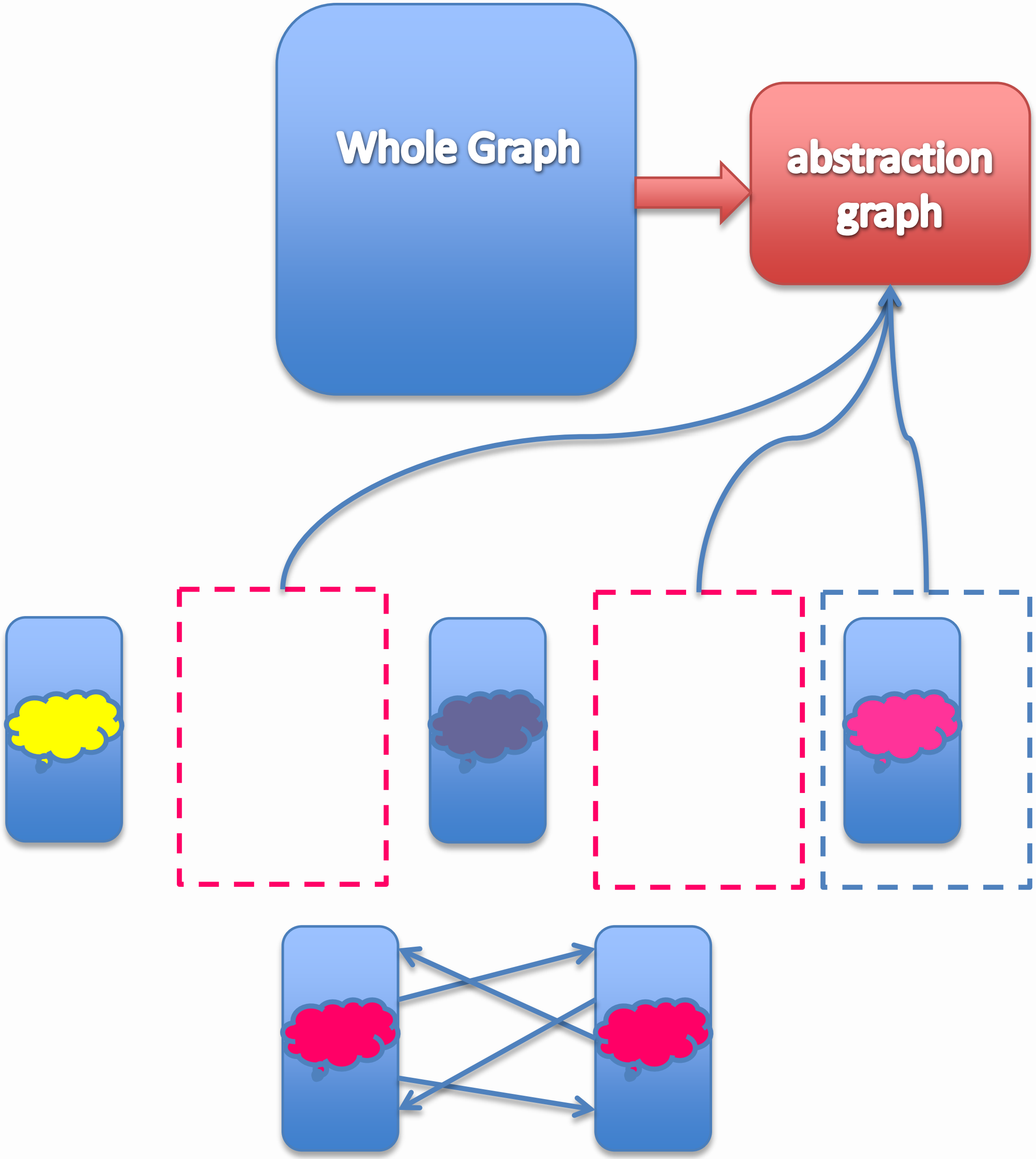
Refine Phase

Local results have priorities

Select results with highest priority

Consultation of abstraction graph

Select partitions to merge



Partition

Initial Phase

Check Phase

Refine Phase

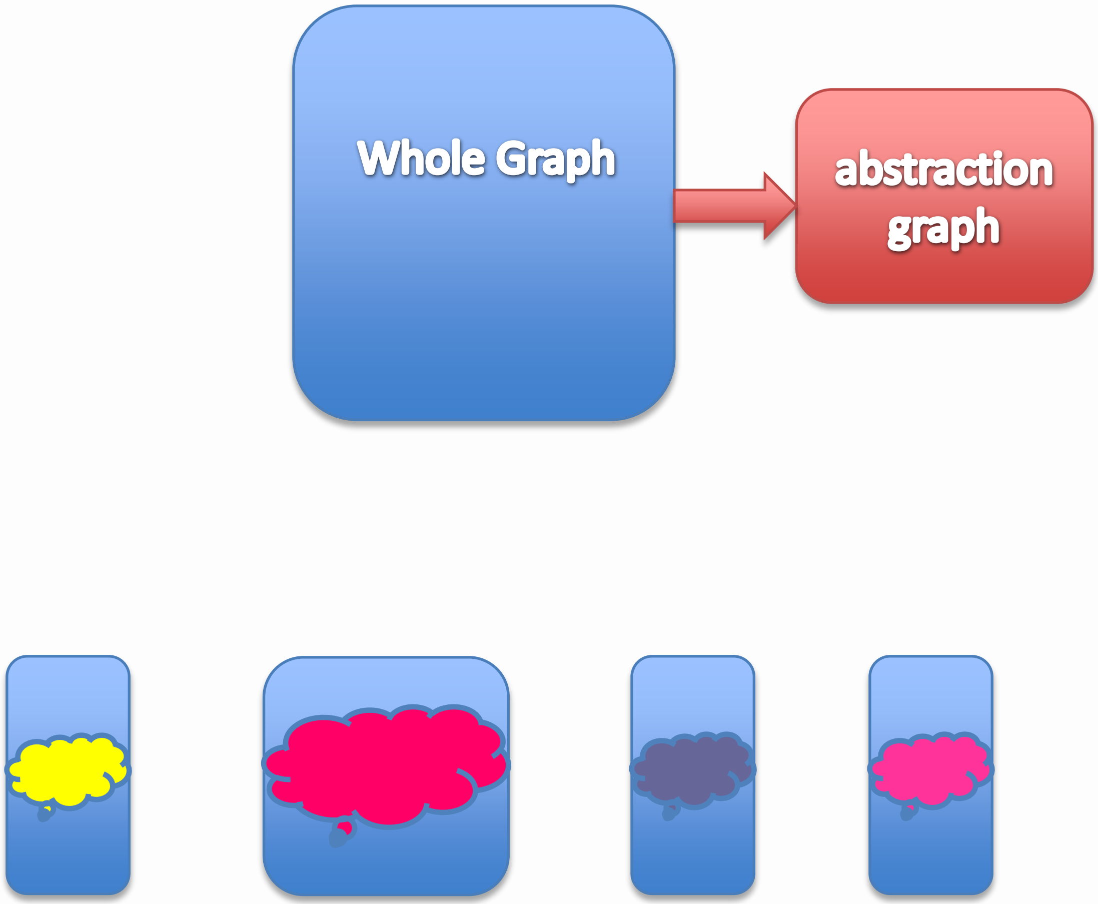
Local results have priorities

Select results with highest priority

Consultation of abstraction graph

Select partitions to merge

Recover inter-partition edges



Partition

Initial Phase

Check Phase

Refine Phase

Local results have priorities

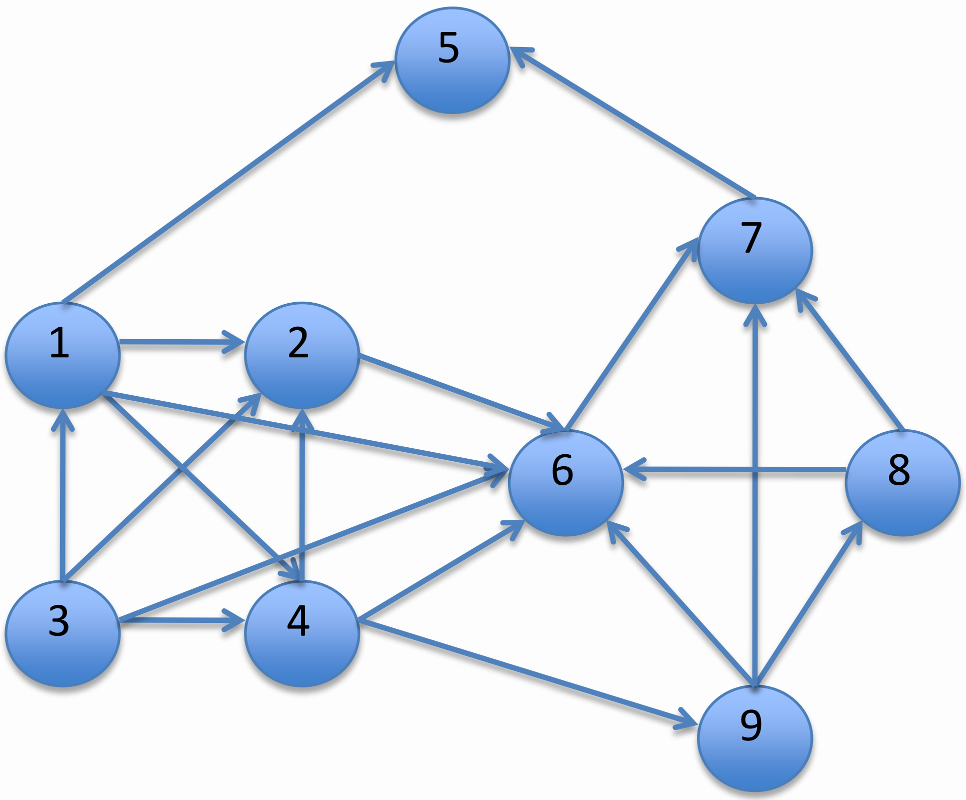
Select results with highest priority

Consultation of abstraction graph

Select partitions to merge

Recover inter-partition edges

Example



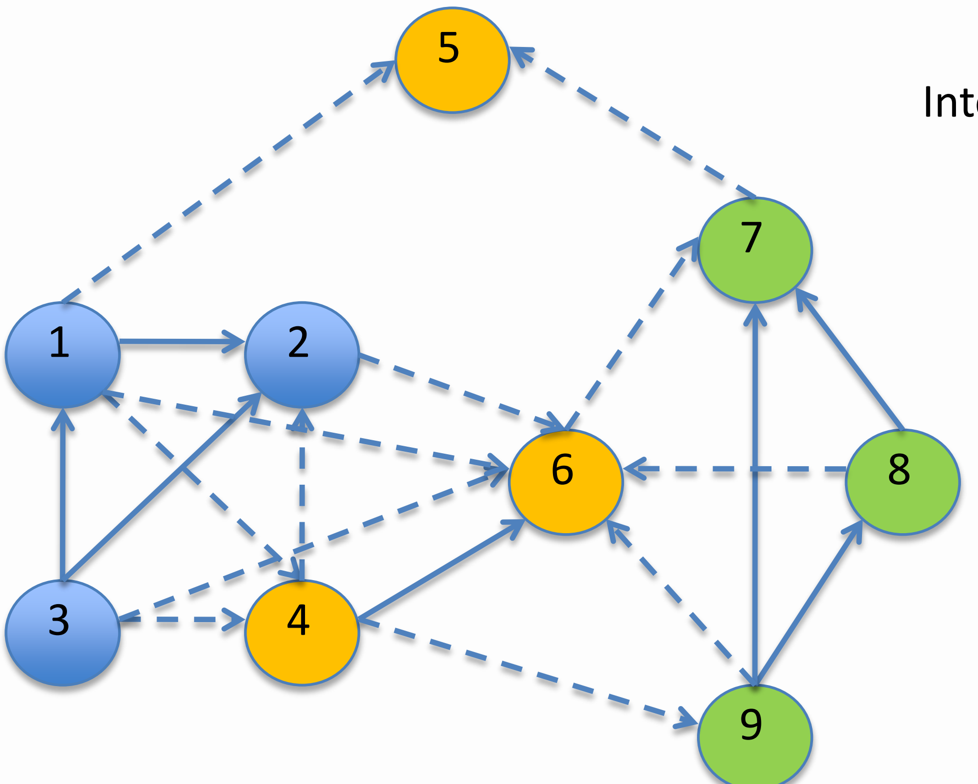
A directed graph

Divide concrete graph into three partitions:

A: {1,2,3}
B: {4,5,6}
C: {7,8,9}

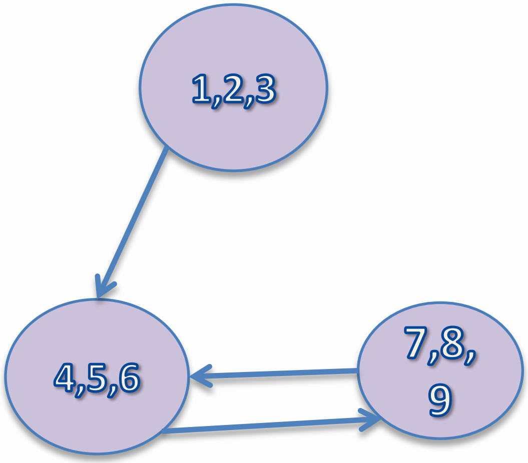
Example

Is there a clique, whose size is no less than 5?



A directed graph

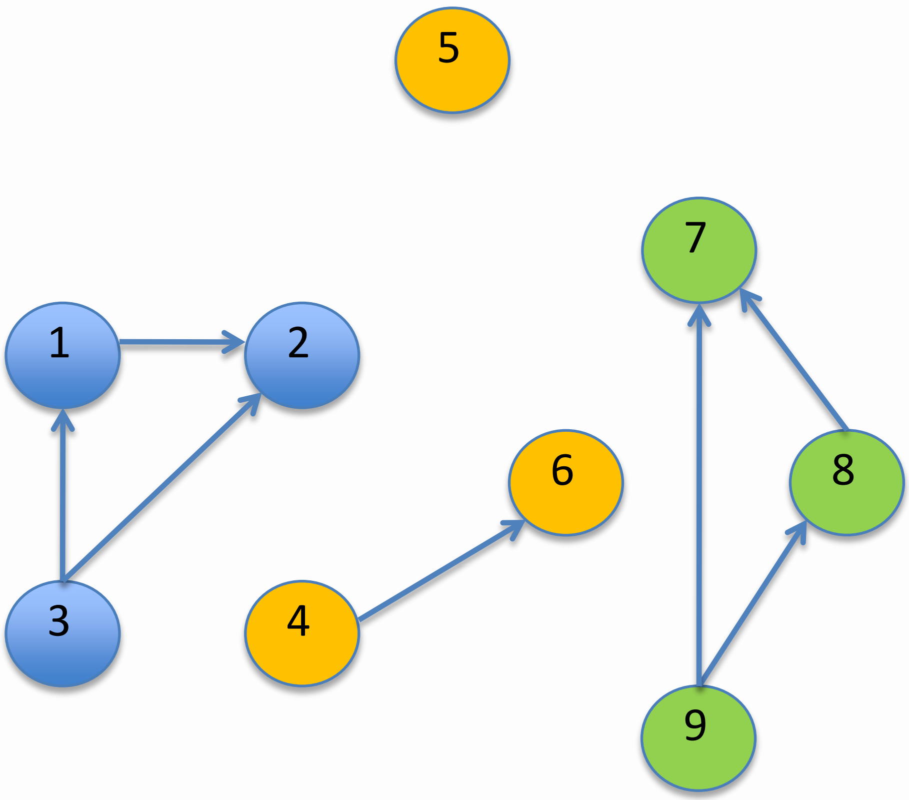
Interval domain [P. Cousot et al., POPL'77]



The abstraction graph

Initial Phase

Is there a clique, whose size is no less than 5?



A directed graph

Four local cliques
{1,2,3}
{4,6}
{5}
{7,8,9}

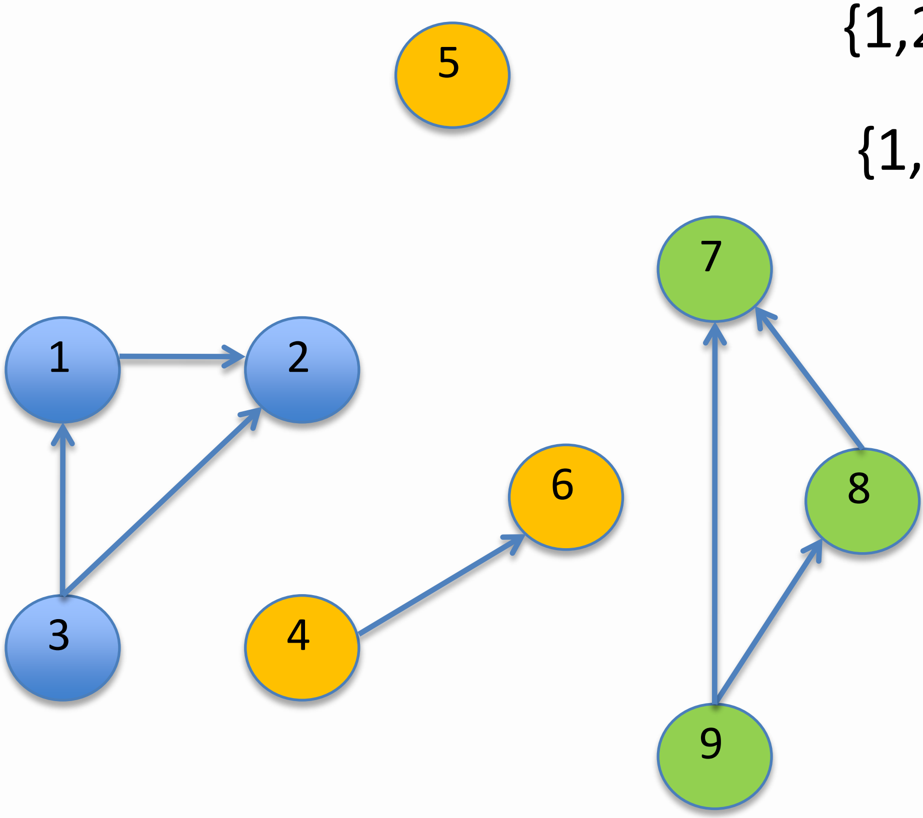
Clique size >= 5?

NO!

Refine Phase

{1,2,3} + {7,8,9} ? **NO!**

{1,2,3} + {4, 6} ? **MAYBE!**

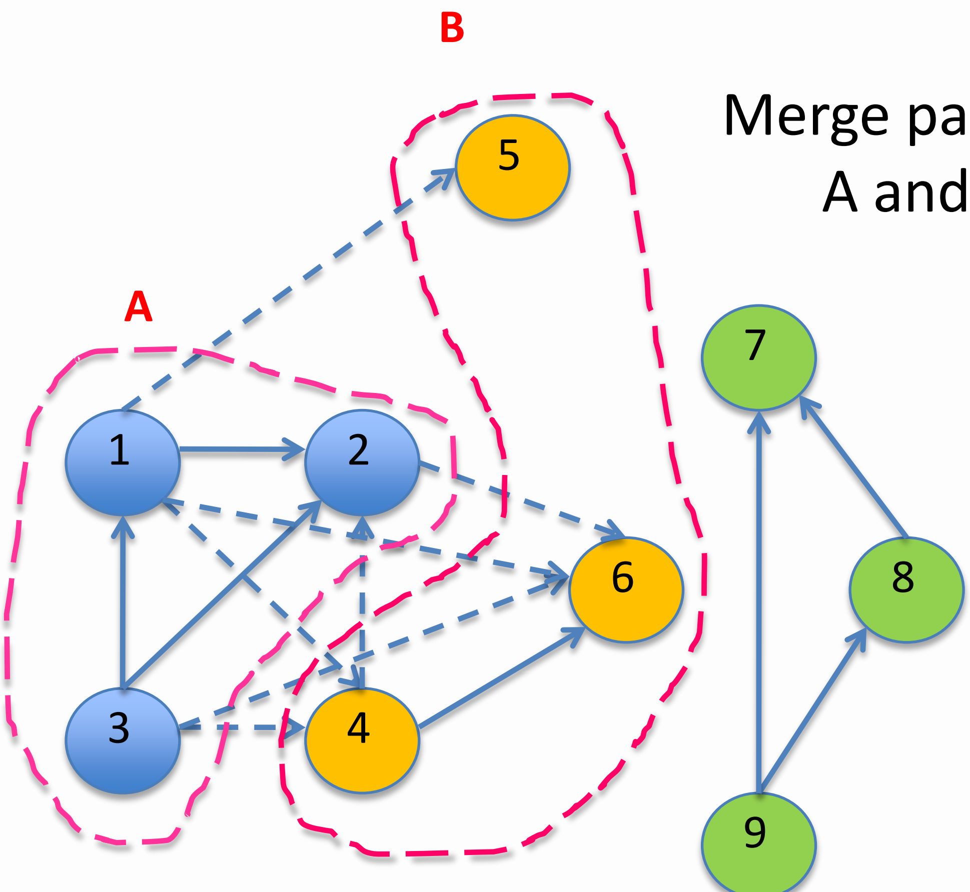


A directed graph

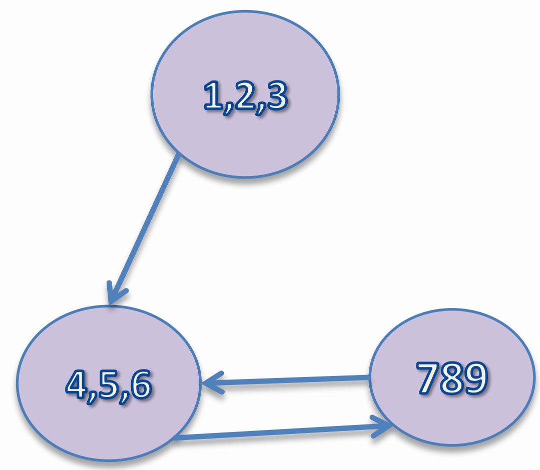
The abstraction graph

Refine Phase

Merge partitions
A and B



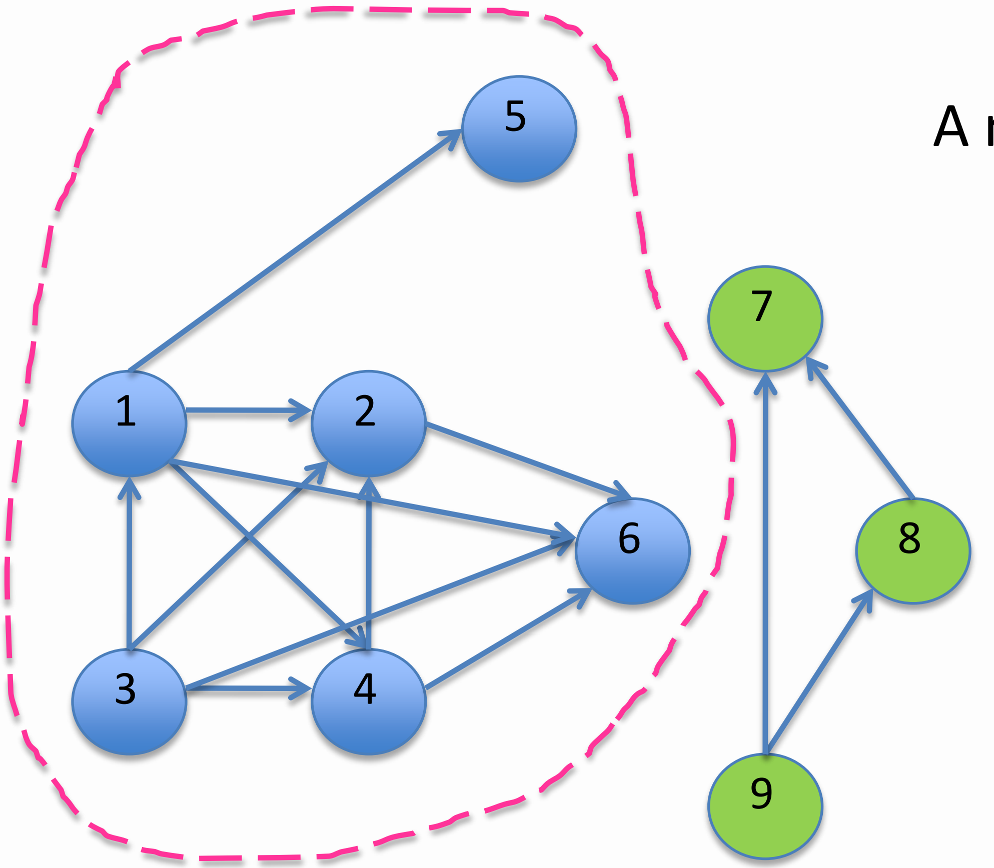
A directed graph



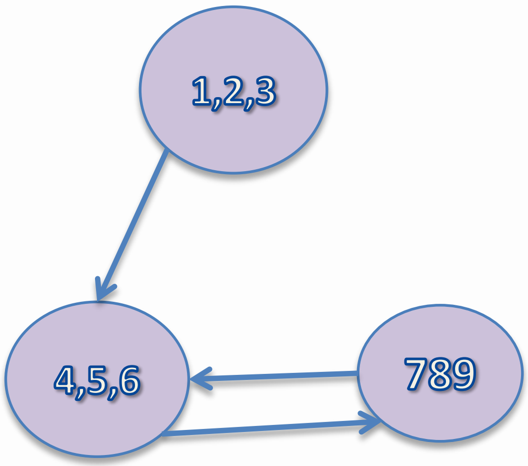
The abstraction graph

Refine Phase

A new clique {1,2,3,4,6}



A directed graph



The abstraction graph

Answerability

- Works for a class of queries with monotone update functions
- Doesn't answer all kinds of queries
  - Belief propagation, probability in a vertex may fluctuate

Implementation

- Based on GraphChi[A. Kyrola et al., OSDI'12], a single-machine graph processing system
  - Modify shard construction in preprocessing
  - Abstraction Graph Construction
  - Modify parallel sliding window

More details in the paper

Evaluation

- Test setup
  - 10GB RAM
  - 256GB SSD
  - Intel Core i5, 3.2GHz
- Input graphs:
  - twitter-2010: 42M vertices, 1.5B edges
  - uk-2005: 39M vertices, 0.75B edges

Evaluation

- Queries

  Page Rank, Max Clique, Community Detection, SSSP, Triangle Counting

- Methodology

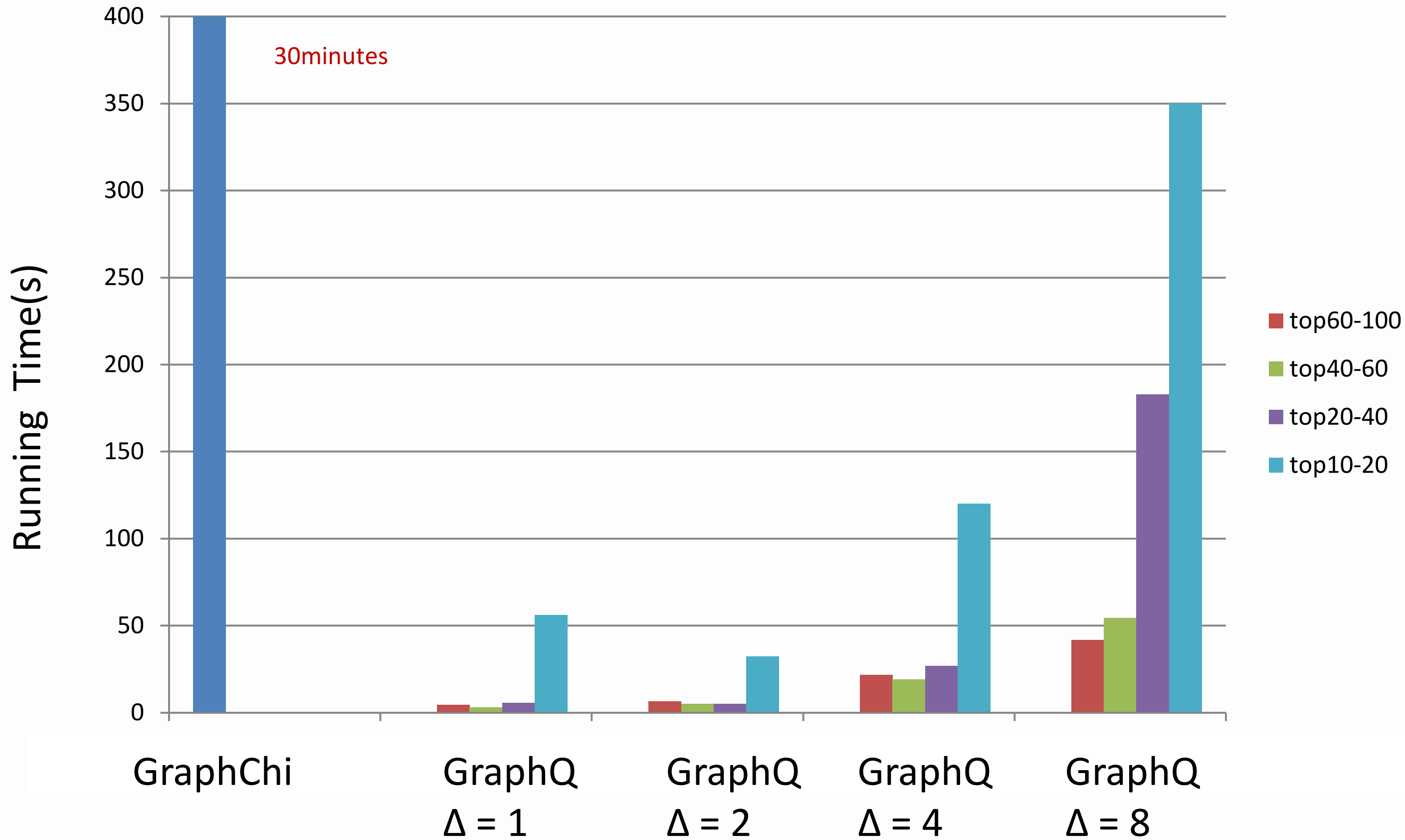
  Three sets of experiments

  - Queries with various goals: find *Δ* entities with a given *quantitative* property
  - Comparison between GraphQ and modified GraphChi
  - Vary abstraction granularity

How To Make Queries

- Run whole graph computation on GraphChi and get all results
- Select top100 values
- Divide into intervals, each interval has a lower bound and an upper bound value
- Generate 20 queries for each interval

1. Queries With Various Goals

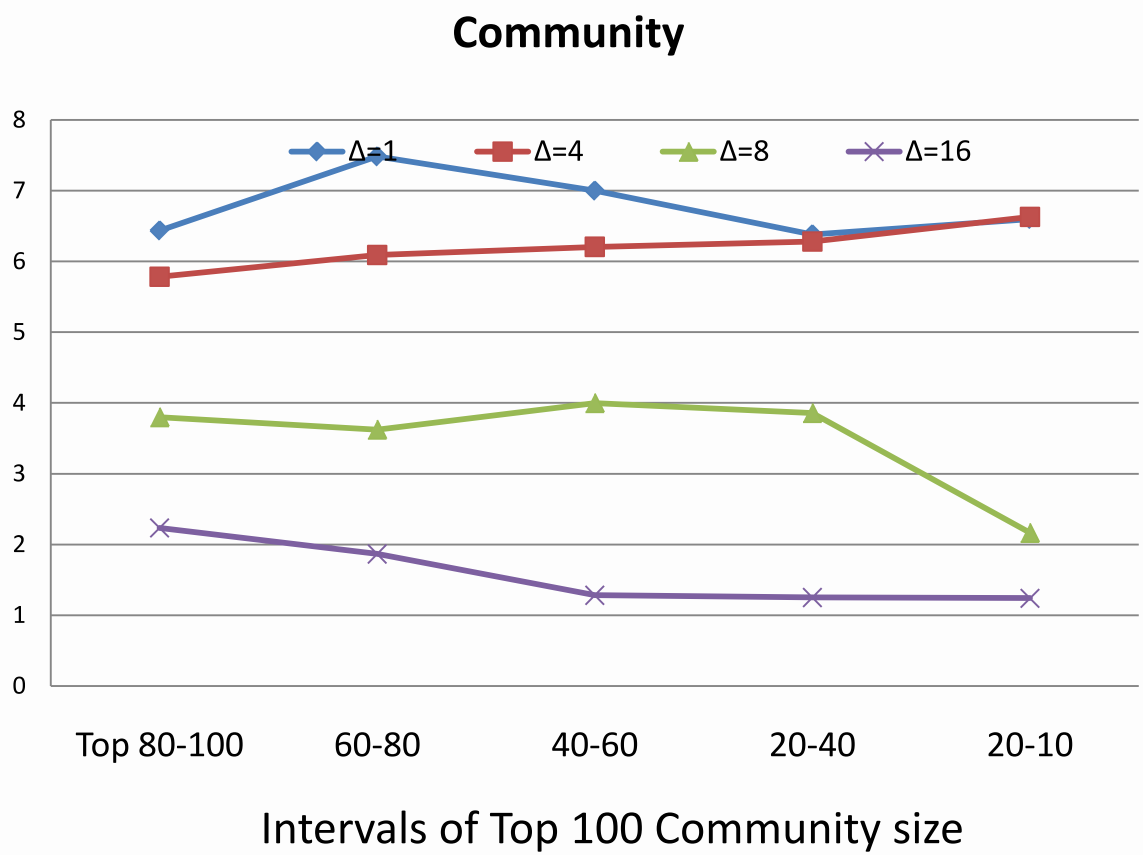


Page Rank queries over uk-2005

2. Compared to Modified GraphChi

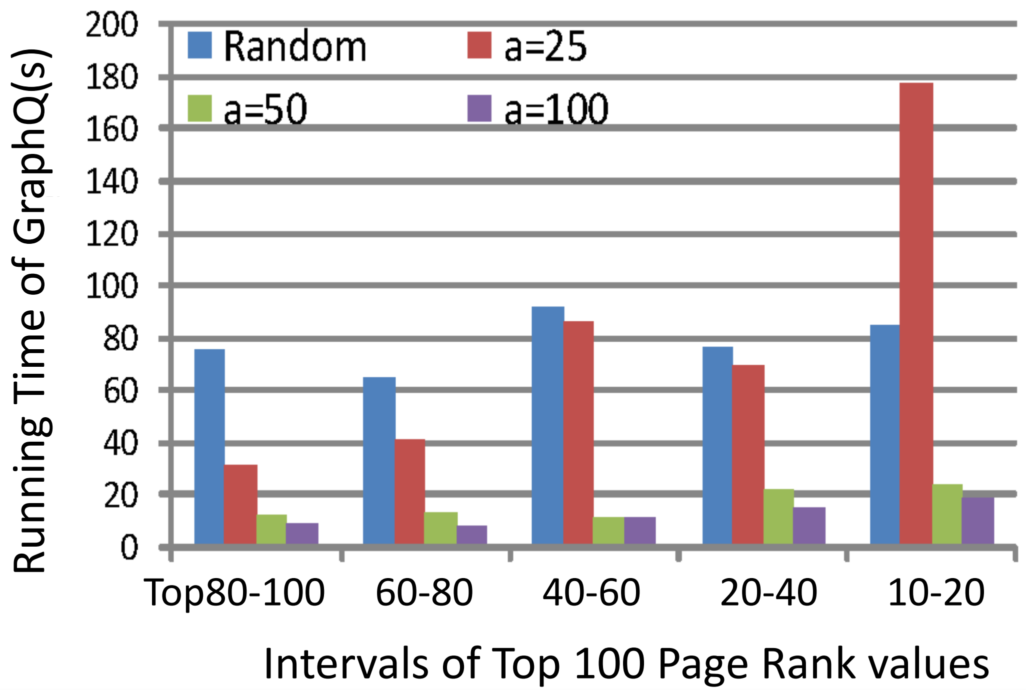
Ratios:

modified GraphChi and GraphQ



Community Detection over twitter-2010
Max:7.5X, Min:1.3X, GeoMean:3.8X

3. Vary Abstraction Granularity



Page Rank queries over twitter-2010

Conclusions

- GraphQ, a graph query answering system based on abstraction refinement
- Efficiently answer analytical queries over partial graphs
- Open up new possibilities to scale up Big Graph processing with small amounts of resources

Thanks!

Q&A