

©2013 LinkedIn - Get your network map at [inmaps.linkedinlabs.com](http://inmaps.linkedinlabs.com)

# Using Set Cover to Optimize a Large-Scale Low Latency Distributed Graph

Rui Wang  
Staff Software Engineer

# Outline

- LinkedIn's Distributed Graph Infrastructure
- The Scaling Problem
- Set Cover by Example
- Evaluation
- Related Work
- Conclusions and Future Work

# Outline

- **LinkedIn's Distributed Graph Infrastructure**
- The Scaling Problem
- Set Cover by Example
- Evaluation
- Related Work
- Conclusions and Future Work

# LinkedIn Products Backed By Social Graph

## How You're Connected

The screenshot shows a LinkedIn profile page for a company. At the top, the LinkedIn logo is on the left, and the company name "LinkedIn" is in the center. To the right of the name, it says "400,842 followers" and "Following" with a checkmark. A red circle highlights the "400,842 followers" text, with an arrow pointing to it from the right. Below the name is a navigation bar with "Home", "Careers", "Products & Services", and "Insights".

The main banner area features a large image with the LinkedIn logo, a keyboard, a coffee cup, and some papers. Below the banner is a "Recent Updates" section. The first update is from "LinkedIn" and says: "Summertime here at LinkedIn is signaled by the arrival of Food Truck Friday at our Mountain View campus. We like to serve our lunch with a side of vitamin D and community! <http://linkd.in/13L8LOD>". Below the text is a photo of a food truck and people. The update has "Like (93) · Comment (5) · Share · 1 day ago".

On the right side, there is a "How You're Connected" section. It shows four profile pictures of people, each with a "1st" degree label. Below the pictures, it lists: "164 first-degree connections", "3,090 second-degree connections", and "7,355 Employees on LinkedIn". A red circle highlights the "7,355" number, with an arrow pointing to it from the left. Below this section is a "See all" link.

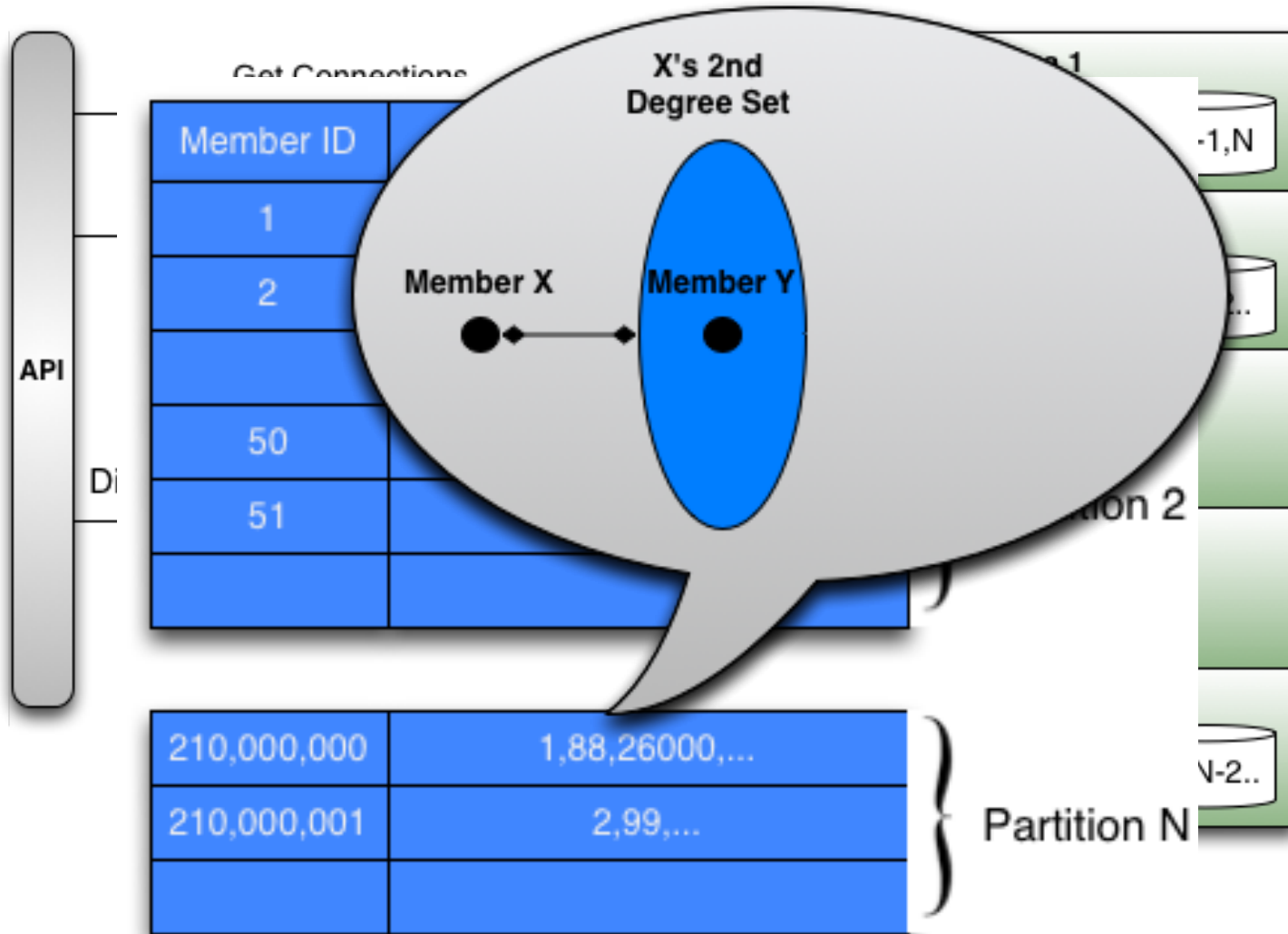
Below the connections section is a "Careers" section with a profile picture and the text "Interested in LinkedIn? Learn about our company and culture. 418 jobs posted". There is a "Learn more" link.

At the bottom right, there is a "Products And Services" section.

# LinkedIn's Distributed Graph APIs

- Graph APIs
  - Get Connections
    - “Who does member X know?”
  - Get Shared Connections
    - “Who do I know in common with member Y”?
  - Get Graph Distances
    - “What’s the graph distances for each member returned from a search query?”
    - “Is member Z within my second degree network so that I can view her profile?”
- Over 50% queries is to get graph distances

# Distributed Graph Architecture Diagram



# LinkedIn's Distributed Graph Infrastructure

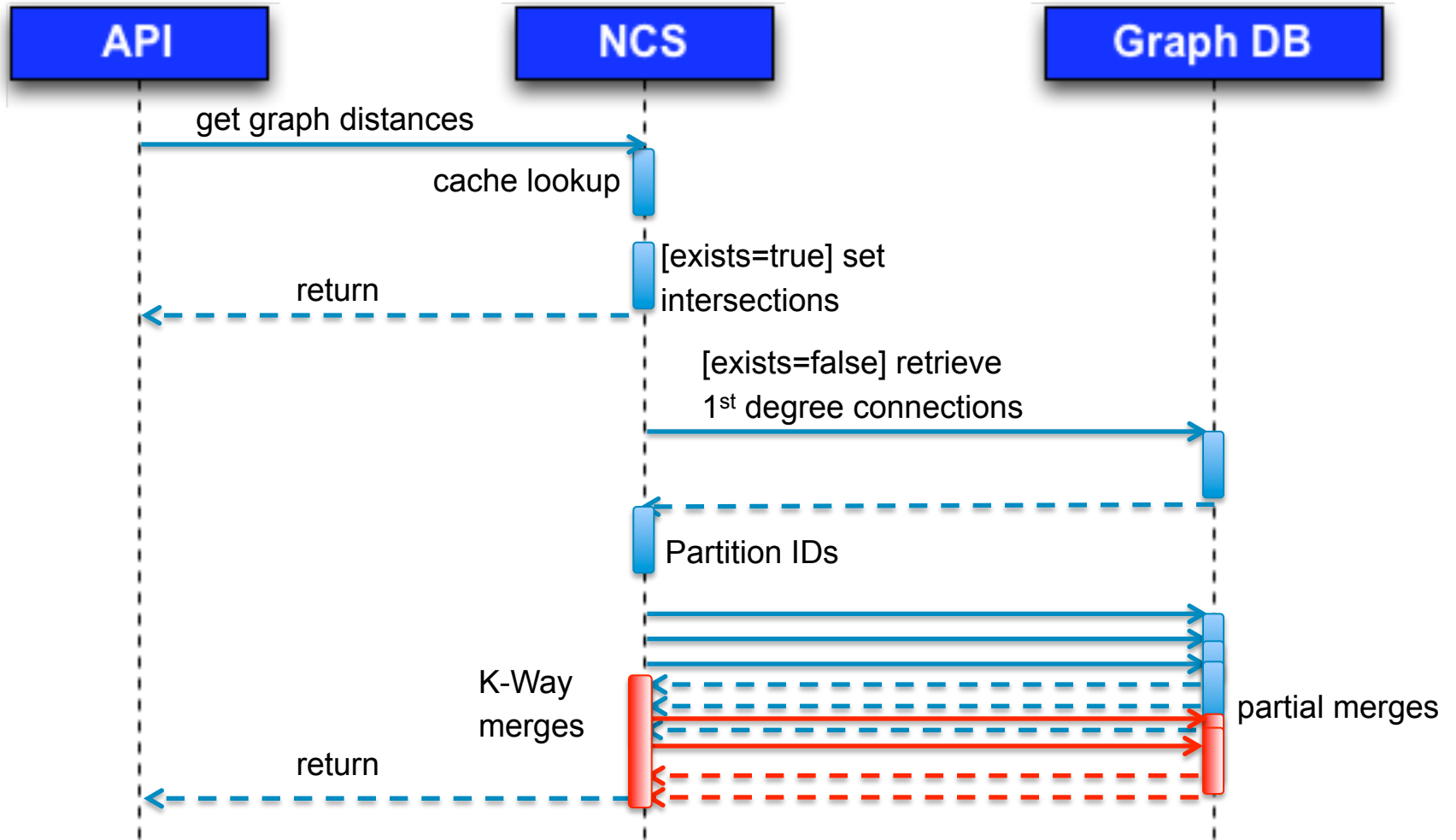
- Graph Database (Graph DB)
  - Partitioned and replicated graph database
  - Distributed adjacency list
- Distributed Network Cache (NCS)
  - LRU cache stores second degree network for active members
  - Graph traversals are converted to set intersections
- Graph APIs
  - Get Connections
  - Get Shared Connections
  - Get Graph Distances

# Outline

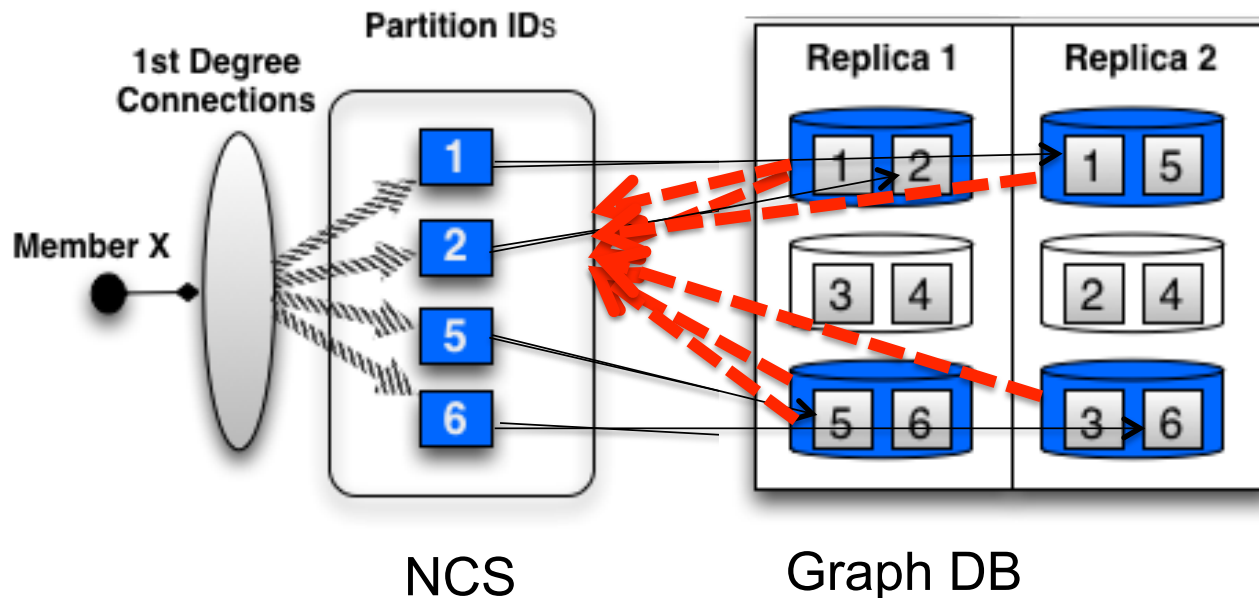
- LinkedIn's Distributed Graph Infrastructure
- **The Scaling Problem**
- Set Cover by Example
- Evaluation
- Related Work
- Conclusions and Future Work



# Graph Distance Queries and Second Degree Creation



# The Scaling Problem Illustrated



- Increased Query Distribution
- Merging and De-duping shift to single NCS node

# Outline

- LinkedIn's Distributed Graph Infrastructure
- The Scaling Problem
- **Set Cover and Example**
- Evaluation
- Related Work
- Conclusions and Future Work

# Set Cover Problem

- Definition

- Given a set of elements  $\mathcal{U} = \{1, 2, \dots, m\}$  (called the universe) and a family  $\mathcal{S}$  of  $n$  sets whose union equals the universe, the set cover problem is to identify the smallest subset of  $\mathcal{S}$  the union of which contains all elements in the universe.

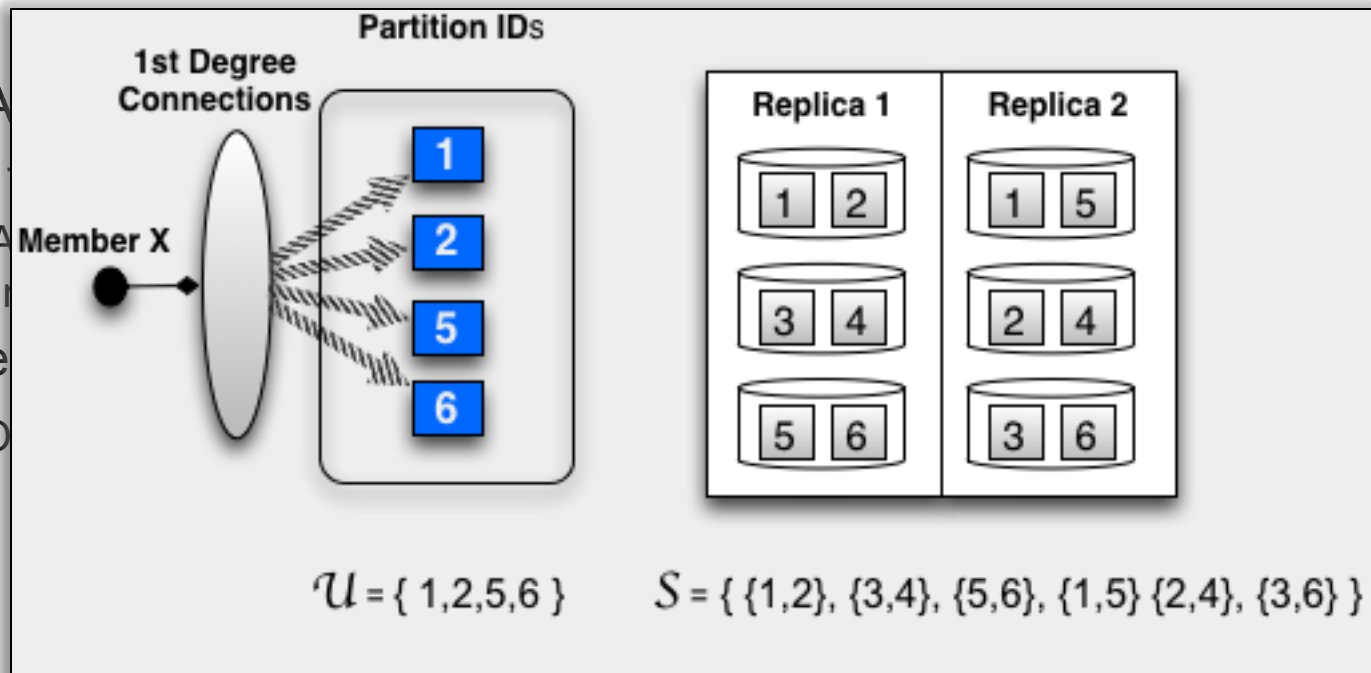
- Greedy Algorithm

- Rule

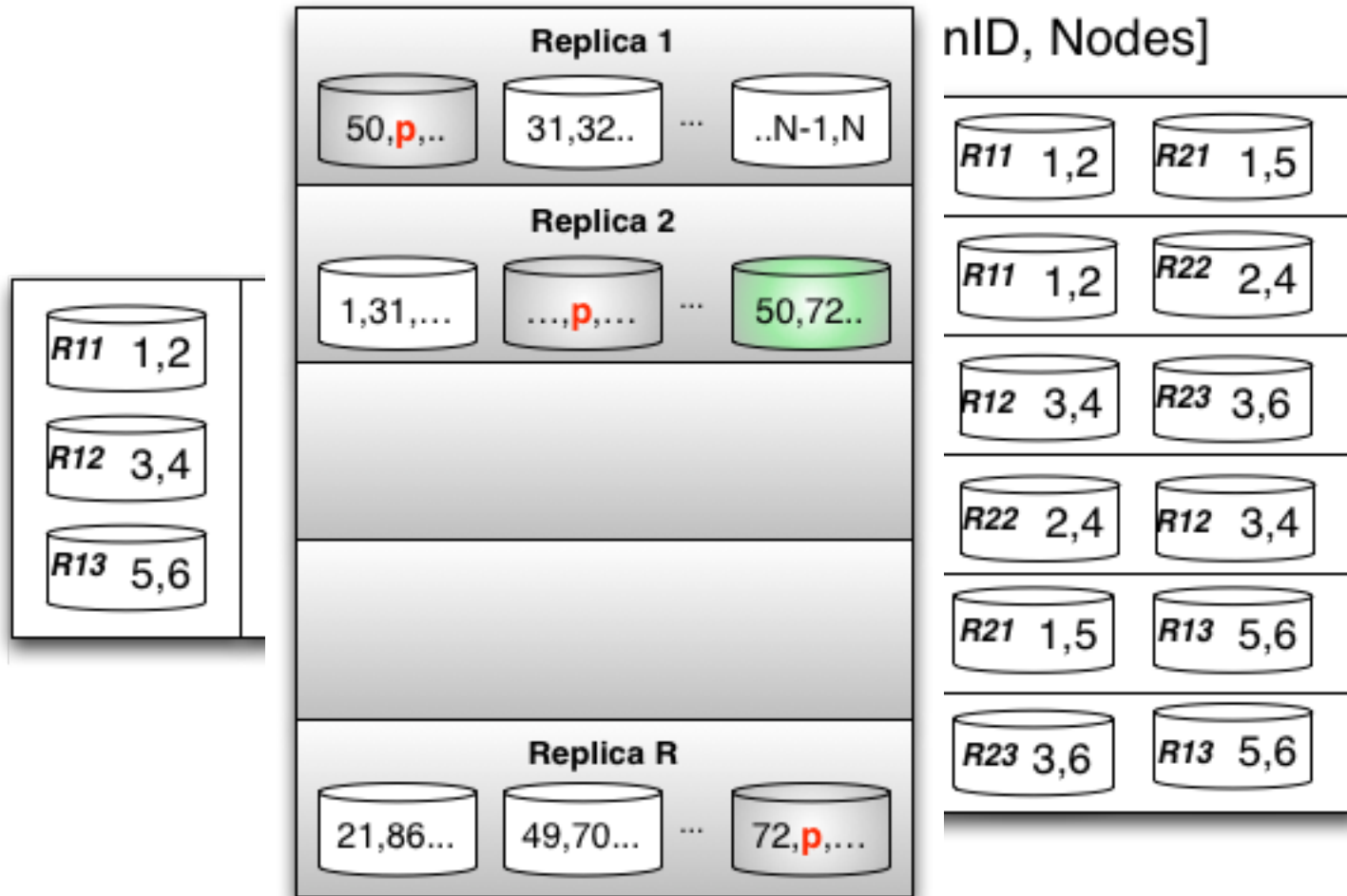
  - At each step, choose the set that covers the most uncovered elements.

- Upper bound

  - On a set of  $n$  elements, the greedy algorithm will find a set cover of size at most  $H(n)$ .



# Modified Set Cover Algorithm Example Cont.

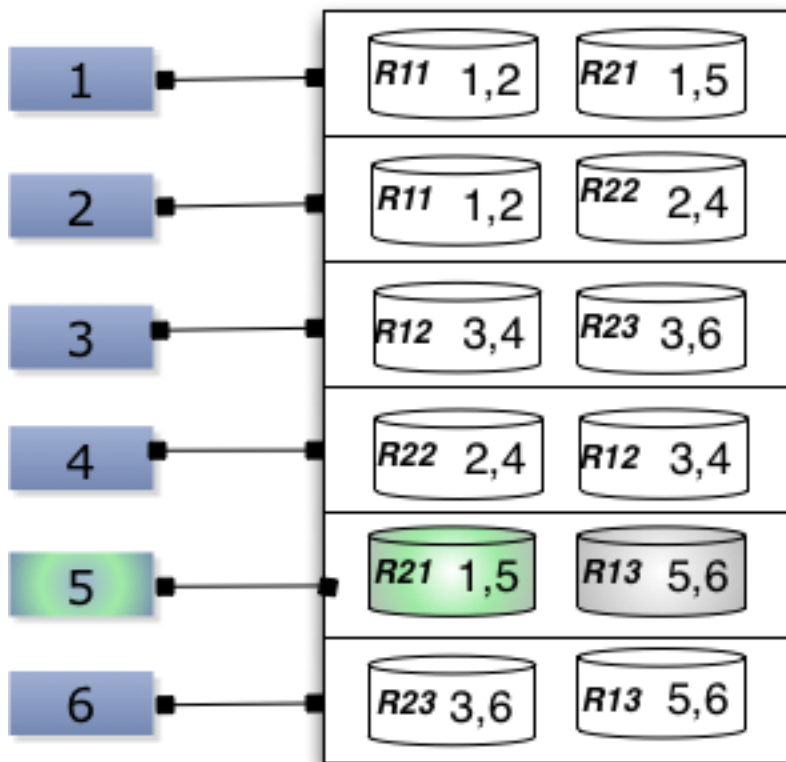


- Build a map

des for the input graph

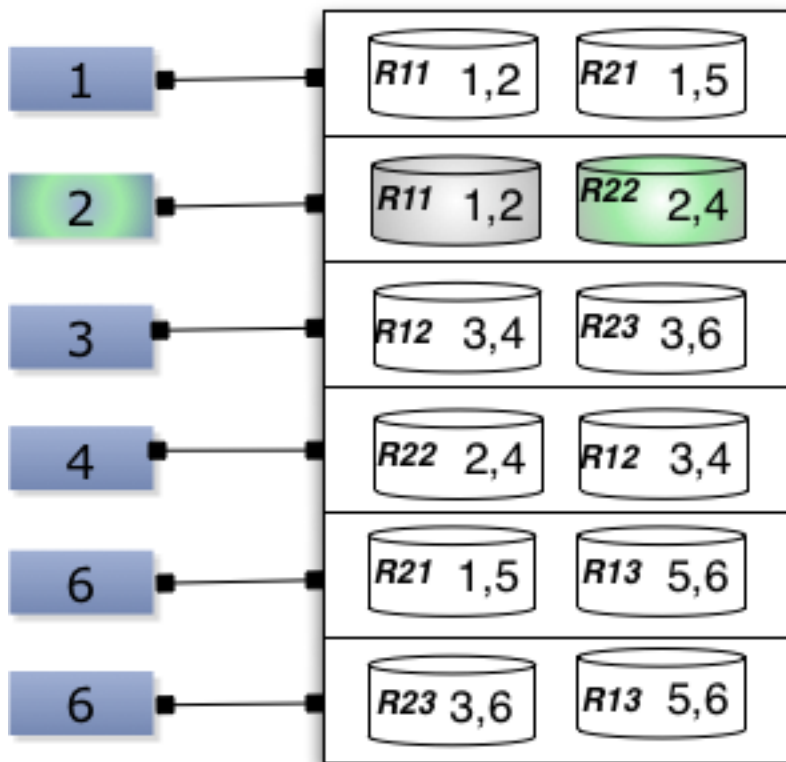
Graph DB

# Modified Set Cover Algorithm Example Cont.



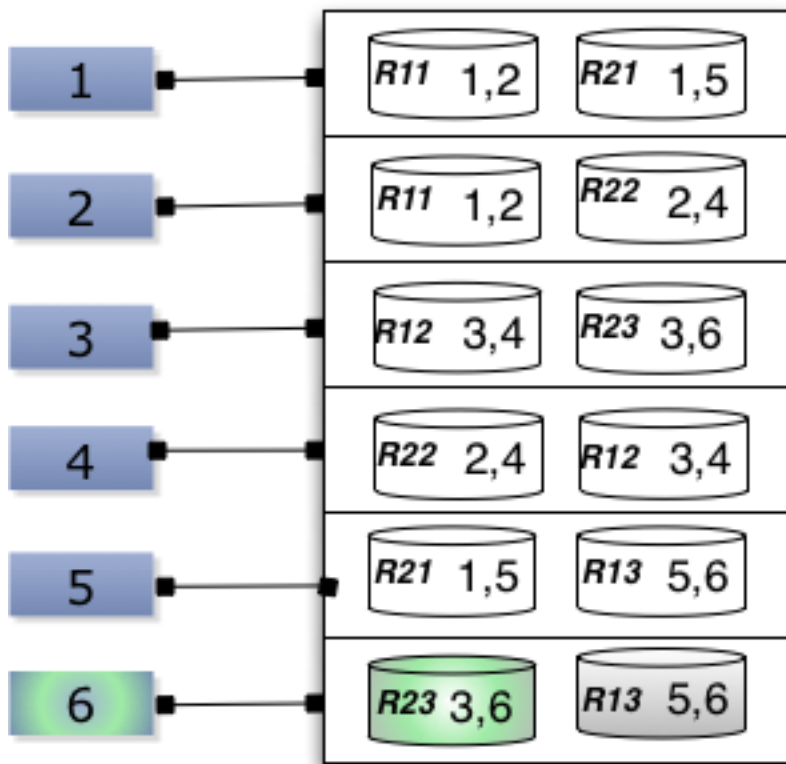
- Randomly pick an element from  $\mathcal{U} = \{1, 2, 5, 6\}$ 
  - $e = 5$
- Retrieve from map
  - nodes =  $\{R21, R13\}$
- Intersect
  - $R21 = \{1, 5\}$  with  $\mathcal{U}$ ,
  - $R13 = \{5, 6\}$  with  $\mathcal{U}$
- Select R21
  - $\mathcal{U} = \{2, 6\}$ ,  $C = \{R21\}$

# Modified Set Cover Algorithm Example Cont.



- Randomly pick an element from  $\mathcal{U} = \{ 2, 6 \}$ 
  - $e = 2$
- Retrieve from map
  - nodes =  $\{ R11, R22 \}$
- Intersect
  - $R11 = \{ 1, 2 \}$  with  $\mathcal{U}$ ,
  - $R22 = \{ 2, 4 \}$  with  $\mathcal{U}$
- Select R22
  - $\mathcal{U} = \{ 6 \}$ ,  $C = \{ R21, R22 \}$

# Modified Set Cover Algorithm Example Cont.



- Pick the final element from  $\mathcal{U} = \{ 6 \}$ 
  - $e = 6$
- Retrieve from map
  - nodes = { R23, R13 }
- Intersect
  - $R23 = \{ 3,6 \}$  with  $\mathcal{U}$ ,
  - $R13 = \{ 5,6 \}$  with  $\mathcal{U}$
- Select R23
  - $\mathcal{U} = \{ \}$ ,  $C = \{ R21, R22, R23 \}$



# Modified Set Cover Algorithm Example Solution

- Solution Compared to Optimal Result for  $\mathcal{U}$ , where  $\mathcal{U} = \{1,2,5,6\}$



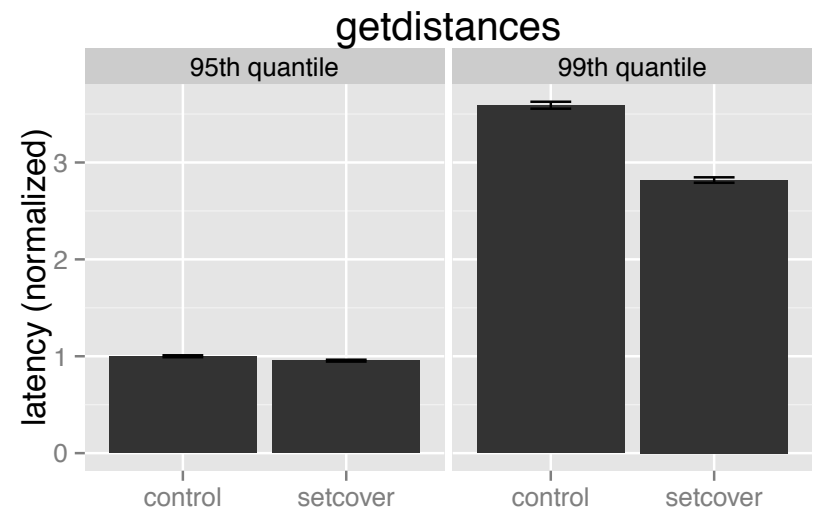
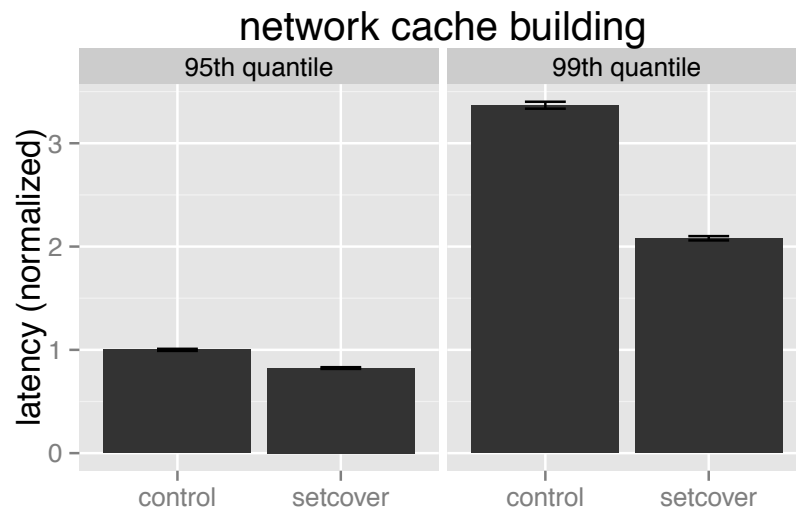
# Outline

- LinkedIn's Distributed Graph Infrastructure
- The Scaling Problem
- Set Cover by Example
- **Evaluation**
- Related Work
- Conclusions and Future Work

# Evaluation

## ■ Production Results

- Second degree cache creation drops 38% on 99<sup>th</sup> percentile
- Graph distance calculation drops 25% on 99<sup>th</sup> percentile
- Outbound traffic drops 40%; Inbound traffic drops 10%



# Outline

- LinkedIn's Distributed Graph Infrastructure
- The Scaling Problem
- Set Cover by Example
- Evaluation
- **Related Work**
- Conclusions and Future Work

# Related Work

- Scaling through Replications
  - Collocating neighbors [Pujol2010]
  - Replication based on read/write frequencies [Mondal2012]
  - Replication based on locality [Carrasco2011]
- Multi-Core Implementations
  - Parallel graph exploration [Hong2011]
- Offline Graph Systems
  - Google's Pregel [Malewicz2010]
  - Distributed GraphLab [Low2012]

# Outline

- LinkedIn's Distributed Graph Infrastructure
- The Scaling Problem
- Set Cover by Example
- Evaluation
- Related Work
- **Conclusions and Future Work**

# Conclusions and Future Work

- Future Work

- Incremental cache updates
- Replication on GraphDB nodes through LRU caching
- New graph traversal algorithms

- Conclusions

- Key challenges tackled
  - Work distribution balancing
  - Communication Bandwidth
- Set cover optimized latency for distributed query by
  - Identifying a much smaller set of GraphDB nodes serving queries
  - Shifting workload to GraphDB to utilize parallel powers
  - Alleviating the K-way merging costs on NCS by reducing K
- Available at

*<https://github.com/linkedin-sna/norbert/blob/master/network/src/main/scala/com/linkedin/norbert/network/partitioned/loadbalancer/DefaultPartitionedLoadBalancerFactory.scala>*

