University of California, Berkeley
Master of Information and Data Science (MIDS)
W205 – Fundamentals of Data Engineering

Week 9 – NoSQL Graph Databases, Part II

Agenda for Today’s Class

- Attendance and Participation
- Announcements
- Schedule and Due Dates
- Work / Life / School Balance
- Asynch High Level Review in a Nutshell
- Project 3
- Breakouts
- Summary

Attendance and Participation

Please record your attendance and participation for today's class:

GitHub => ucb_mids_w205_repo => README.md => Attendance and Participation

Announcements

- Upcoming holidays and/or breaks
- Makeup classes for holidays
- Upcoming events
- Student evaluations
- Etc.

Schedule and Due Dates

Take a quick look at the next couple of weeks' due dates:

GitHub => ucb_mids_w205_repo => README.md => Schedule and Due Dates

Work / Life / School Balance

Open Discussion

Student feedback

- About 5 minutes
- How are things going related to work / life / school balance?
- How is w205 going? Difficulty? Time?
- Impact of any natural and/or man-made disasters
- Etc.

Asynch High Level Review in a Nutshell

Each week we will spend about 15 minutes reviewing the most important high level concepts from the asynch

Graph Centrality Algorithms

- Degree Centrality
  - Measures number of relationships a node has in a graph: incoming, outgoing
  - Compare a node to statistics: average, median, min, max, etc.
- Closeness Centrality
  - Measures average of the shortest path distances between a node and all other nodes
  - High closeness centrality can spread info most efficiently
  - Compare a node to statistics: average, median, min, max, etc.
  - Weak spot: disconnected subgraphs skew calculations

Graph Centrality Algorithms (continued)

- Wasserman and Faust
  - Variation to improve Closeness Centrality
  - Considers reachable nodes and percentage of reachable nodes
- Harmonic Centrality
  - Variation to improve Closeness Centrality
  - Sum inverses of distances instead of distances
  - Unreachable nodes: inverse of zero is infinity
  - Inverse also creates a smoothing effect
  - “Go to” algorithm for Closeness Centrality

Graph Centrality Algorithms (continued)

- Betweenness Centrality
  - Find all pairs shortest paths (weighted)
  - For each node, how many paths pass through the node?
  - High betweenness = control point, bridge, more influence, etc.
    - Pivotal nodes: lies on every path between two other nodes
- Betweenness Centrality of Clusters
  - Group nodes into clusters, each cluster is a node in new graph
  - Can repeat for several layers of hierarchy
  - Scale-free networks

Graph Centrality Algorithms (continued)

- RA-Brandes (Randomized-Approximate Brandes)
  - Betweenness centrality is expensive to calculate
  - Approximates betweenness centrality using random nodes
  - Can throw out random nodes if degree is less than average
- PageRank
  - Larry Page of Google
  - Overall influence of a node in a graph: direct, influence of incoming, incoming of incoming, etc.
  - Knowing a lot of influential people makes you more influential

Graph Centrality Algorithms (continued)

- PageRank Issues
  - Random surfers who are not following links: use a damping factor
  - Rank sinks – no outbound relationships: random teleporting
- Personalized PageRank
  - Perspective from a single node
  - What is important to a single user
  - Target recommendations

Community Detection Algorithms

- Triangle Count
  - Number of triangles that pass through a node
- Clustering Coefficient
  - Probability that neighbors of a node are connected to each other
  - 1 = full clique, every node connected to every other node
- SCC (Strongly Connected Components)
  - Group of nodes where every node is reachable from every other node
  - Direction

Community Detection Algorithms (cont’d)

- Connected Components
  - Direction not considered
- LPA (Label Propagation Algorithm)
  - Fast and good where grouping is less clear
  - Nodes pass labels to neighbors
  - Method to break ties for multiple labels
  - LPA Push - unweighted, less commonly used, serial
  - LPA Pull – weighted, more commonly used, parallel

Community Detection Algorithms (cont’d)

- Louvain Modularity
  - What if analysis
  - How well a node is assigned to a group
  - Creates a hierarchy of group at different scales
  - “Go to” algorithm for Community Detection

Graphs and AI, ML, DL, etc.

- Feature Engineering
    - Features are inputs into AI, ML, DL, etc.
    - Graphy features
    - Graph algorithm features
- Model Evaluation
    - Run AI, ML, DL, etc. and get results of model run
    - Load results into graph database
    - Gather graph stats, run graph algorithms, etc.
    - Helps us decide which model performs best

Project 3

GitHub => ucb_mids_w205_repo => projects => project_3

- Videos going over project 3 are provided, so we won’t spend class time going over it today
- Breakouts today next week will be related to project 3

Breakouts

GitHub => ucb_mids_w205_repo => breakouts

(time permitting, we may not get to all of them)

Summary

Instructor will give a brief (about 2 minute) summary of today’s class.