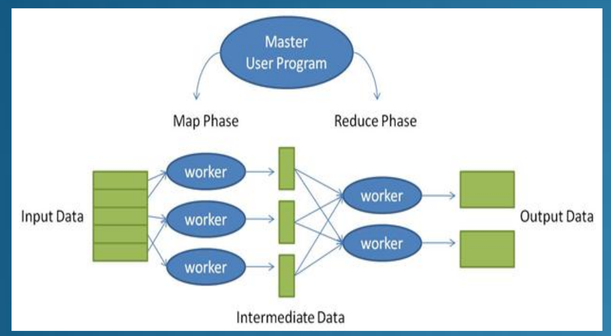
Analyzing Patterns in Large-Scale Graphs Using MapReduce in Hadoop

Joshua Schultz[1] (Undergraduate), Jonathan Vieyra[2] (Undergraduate), Enyue Lu[1] (Faculty Mentor)
[1]Dept. of Math. & Computer Science, Salisbury University, Salisbury, USA
[2]Dept. of Computer Science, California State Polytechnic University, Pomona, USA

Motivation

Analyzing patterns in large-scale graphs, such as social networks (e.g. Facebook, Linkedin, Twitter) has many applications including community identification, blog analysis, intrusion and spamming detections. Currently, it is impossible to process information in large–scale graphs with millions even billions of edges with a single computer. In this project, we take advantage of MapReduce, a programming model for processing large datasets, to detect important graph patterns using open source Hadoop on Amazon EC2. The aim of this paper is to show how MapReduce cloud computing with the application of graph pattern detection scales on real world data.

MapReduce Programming Model



Contributions

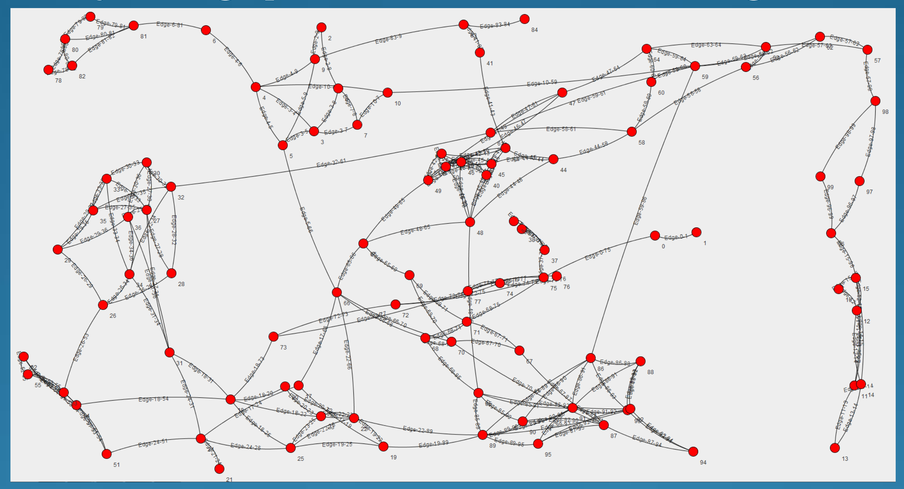
- Implement MapReduce graph algorithms to enumerate important patterns including
  - Triangles: three-vertex complete graphs
  - Rectangles: four-vertex cycles
  - K-trusses: every edge is in K-2 triangles
  - Components: there is a path between any pair of vertices
  - Barycentric clusters: highly connected subgraphs
- Analyze the performance of MapReduce graph algorithms
- Create a visualization algorithm to visualize the detected graph patterns

Acknowledgment

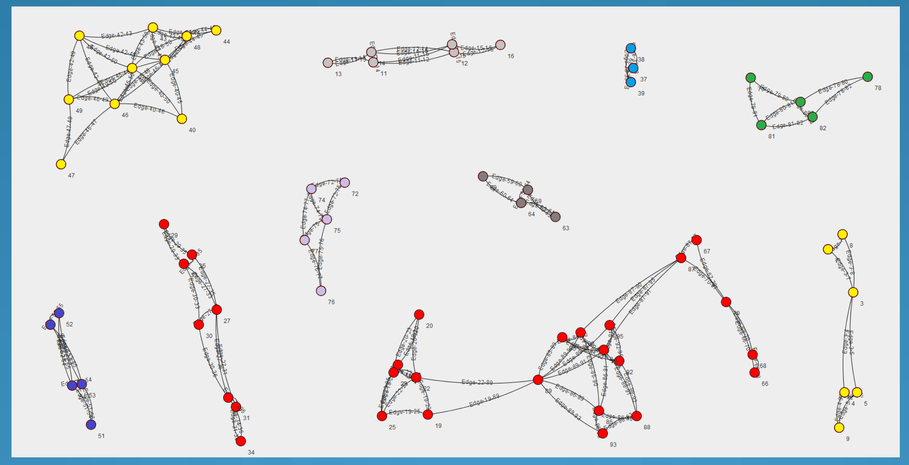
This research is funded by National Science Foundation CCF-1156509 under the Research Experience for Undergraduates Program. We would like to thank Professor Randal Burns at Johns Hopkins University for his valuable inputs on our work.

Graph Visualization for Detected Patterns

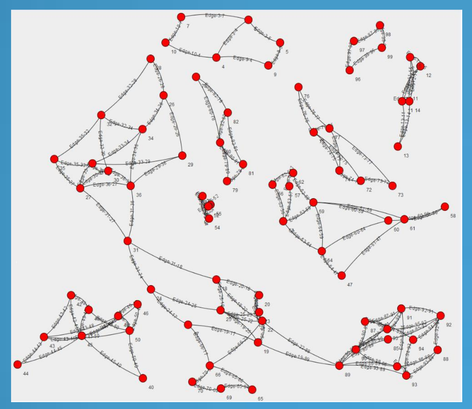
A synthetic graph G with 100 vertices and 398 edges



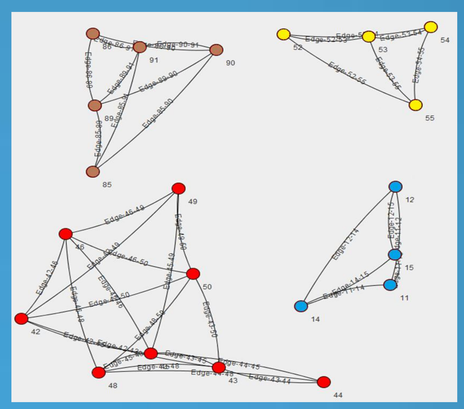
Enumerating Triangles on graph G



Enumerating Rectangles on G



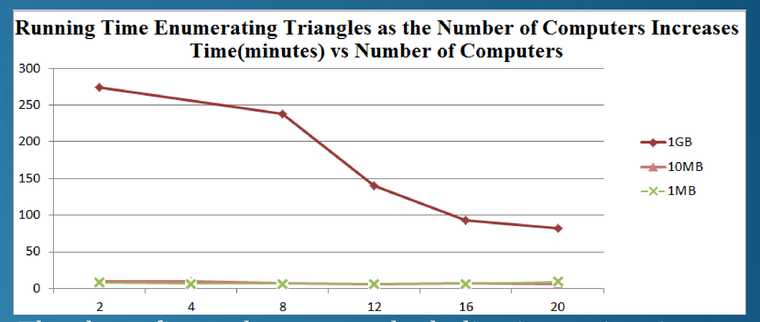
Enumerating 4-truss on G



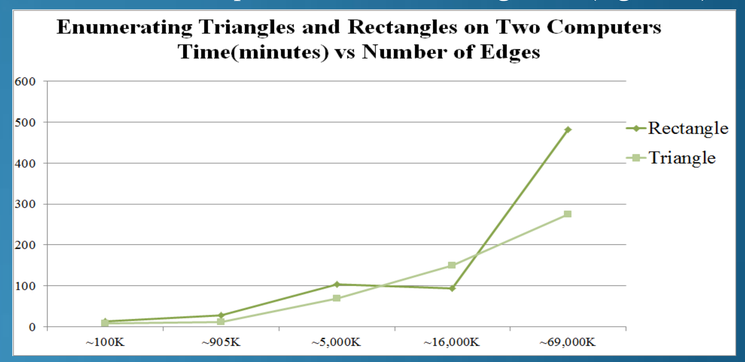
Experimental Setting

Data processed on a cluster was ran on Amazons Elastic MapReduce "small" computers. Each computer was outfitted with 1.7 GB of memory 160 GB of storage and the equivalent of a 1.7GHz Xeon processor. We used different datasets ranging from 1 MB to 1GB including wiki-Vote (7,115 Vertices, 103,689 Edges, 1MB), soc-Slashdot0811 (77,360 Vertices, 905,468 Edges, 10MB), and soc-LiveJournal1 (4,847,571 Vertices, 68,993,773 Edges, 1GB) from Snap Stanford.

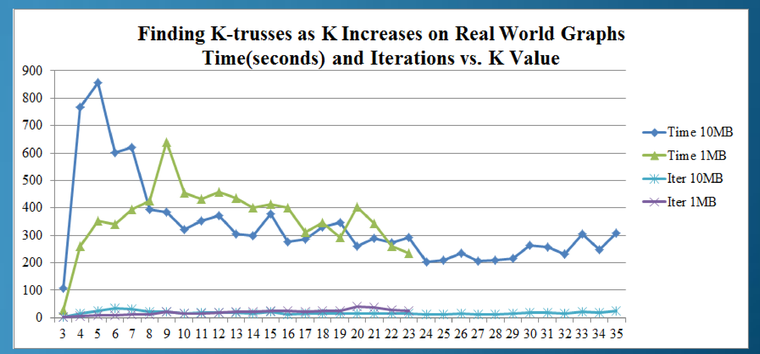
Experimental Results



The above figure shows a steady decline in running time as the number of computers increases for large data (e.g. 1GB).



As shown in the figure above, triangle and rectangle enumerating algorithms scale well when datasets get larger.



The above figure indicates that the scalability of the truss algorithm depends on the number of MapReduce iterations.