Pregel

{ Evan Hopkins 11/25/2013

Gregorz Malewicz, Matthew H. Austern, Aart J. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Gregorz Czajkowski

Idea

- Pregel is the solution to graphing extremely large amounts of data. More and more websites, like social networks, contain large amounts of data that needs to be mapped. Current solutions are very resource heavy and Pregel intends to fix that.
- What makes Pregel great is how each 'unit' of a graph operates mostly independent from other units. This allows it to be robust if one part of a graph fails. It allows for these 'units' to be logically stored on separate machines.
- Pregel is a graphing system that…
  - Is scalable from very small to exponentially large
  - Operates in the most efficient manor
  - Is customizable to meet consumer needs
  - Is easy to use and implement
  - Maximizes the use of multithreading on computers
  - Utilizes a network of computers that varies in size

Implementation

- Graph – A network of vertices and edges
- Vertex – An independent node for the graph
  - Contains current state (active or inactive), neighbors (edges), and data
  - Processes it's 'next move' based off incoming messages from other vertices via the user defined Compute() function
- Message – An option outgoing communication with a vertex
  - Contains information that may influence receiving nodes next move
- Superstep – A global 'turn' for all vertices in graph
  - A superstep advances once all vertices have completed their actions
  - Each vertex completes the following sequence of actions in a superstep:
    - Start -> receive messages -> Compute() -> send messages -> end
- Machines – Interconnected computers
  - Master – responsible for managing and load balancing workers
  - Worker – a machine that holds a portion of the whole graph
- Combiner – groups messages that must travel across machines into a single message
- Aggregators – Generate data available to all vertices
  - E.G: The total number of vertices

Analysis

- I agree that the world is in need of a graphing system capable of handling very large amounts of data. There should be a graphing system that can handle mapping something as large as a social network.
- The system will only operate properly when the vertices of the graph do the majority of communication with their neighbors. If there is lots of communication between nodes very far away, a better system can be implemented.
- The easy to use API opens many doors for graphing systems to be developed on top of Pregel. For instance, a graphing system could be built to show the usage of computers across a large college campus (think Penn State). This system could then be ported to other colleges.

Advantages

- Advantages
  - Handles extremely large amounts of data as well as medium amounts of data
  - Easy to adapt to more data by adding more ‘workers’ (scalability)
  - Uses multithreading to maximize efficiency
  - Logically groups vertices to minimize cross worker messages.
  - Very customizable in terms of…
    - Format of exported data
    - Functionality of vertices
- Disadvantages
  - Overkill for small graphing needs
  - Only fast when the majority of messages stay on the local ‘worker’
  - While it handles failures, they still significantly slow the system. In very large systems this could constantly slow the system as a whole down.
  - If systems are very data heavy and very large at the same time, saving a backup to persistent storage could take a significant amount of time.

Real World Use

- Page Ranking
  - Arranges a set of pages into a hierarchy. For this example, page rank is determined by the amount of pages linking to a given page. Each superstep, each vertex sends it current page rank to its neighbors. Using a combination of a vertex's current page rank, neighbor's page ranks, and amount of neighbors
- Shortest Path Calculation
  - Determines the shortest path to a destinations. The user inputs a destination and starting vertex. As moves are made, a current vertex value tracks where it currently is in the graph. Each vertex determines its own shortest path to the destination. The current vertex will see which of its neighbors has the current shortest path and move to it. This process will continue until the destination is reached.
- Social Group Mapping
  - Generates map of a group of people where mutual friends is the determining value. A base person will be the central vertex. The closest vertices to the central vertex have the most mutual friends with the base person. As you go further from the central vertex you get people who are more removed from the base persons central friend group.