Pregel: A System for Large-Scale Graph Processing

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

Submitted by:
Harini Shreedhar
FT5903
Q 1. What is superstep in Pregel graph processing model? In the single source shortest path problem what computation is involved in superstep?

Answer:

Sequence of iteration is called Superstep in a Pregel model. During a superstep, the framework invokes a user defined function for each vertex which runs in parallel. Each superstep comprises of commutation of the vertices and communication followed by a barrier which provides barrier synchronization.

Single Source Shortest Path is used to find shortest path from a source node to all target nodes.

EXAMPLE:

Consider the following example given. Initially, all nodes are active and no nodes are given any value, but are simply started off with A, B, C, D, E. Node A, which is recognized as the start node is given the value 0 and all other nodes are assigned infinity. Communication occurs between each nodes and messages are sent across.
All nodes communicate and messages are sent to each node.
The edge weight between nodes A and B and A and C is 10 and 5 respectively which is communicated and hence the value of B and C becomes 10 and 5 which is also the shortest. Meanwhile node A receives the value infinity from node E which is much much higher than the value it has and hence it retains its value and becomes inactive.

Nodes B and C communicates. Node B sends message to C and D. the value of D is changes to 11 and the value of node C is retained as the 12 is greater than the value it already has. This node goes into inactive state.

Similarly, node C sends message to B, D and E. At node B, the value it possesses is greater than the value it receives which is the SSSP. Hence the node value of B is replaced to 8. The value of node E becomes 7 from infinity. At node D, it receives two value. One from m=node B which is 11 and another from node C
which is 14. The shortest in the two is the value it receives from B which is 11. Hence the value of node D is 11.

At this stage the active nodes communicate and sends messages to its neighboring nodes. The SSSP in each node is retained. And the node which does not change its value goes to inactive mode until triggered.
Here, no node communicates with another node, and hence all the nodes are in inactive state.

**Q 2. What does synchronicity in the Pregel's execution refer to? What benefits can it bring?**

Answer:

Synchronicity refers to the barrier synchronization where the barrier ensures that all the messages between the vertices are passed on from the previous superstep to the next superstep. This completes the communication cycle and only then can the computation phase occur. The benefits of this synchronicity are that it avoids deadlocks and race conditions.

**Q3. How is a Pregel program terminated?**

Answer:
The above figure is the state machine for a vertex. During initialization all the vertices are in active mode. Algorithm termination is based on every vertex voting to halt. In superstep 0, every vertex is in the active state; all active vertices participate in the computation of any given superstep. A vertex deactivates itself by voting to halt (either a minimum or maximum value). This means that the vertex has no further work to do unless triggered externally, and the Pregel framework will not execute that vertex in subsequent supersteps unless it receives a message. If reactivated by a message, a vertex must explicitly deactivate itself again. The algorithm as a whole terminates when all vertices are simultaneously inactive and there are no messages in transit.

The two main conditions for program termination is:

- All vertices are simultaneously inactive
- There are no messages in transit

Q 4. Use figure 2 to illustrate a Pregel programs execution.

Answer:

![Graph](image)

**Figure 2: Maximum Value Example. Dotted lines are messages. Shaded vertices have voted to halt.**

Figure 2 illustrates a strongly connected graph where each vertex contains a value. The largest value is propagated to every vertex. In each superstep any vertex that has learned a larger value from its message sends it to its neighbors. Algorithm terminates when no vertices change in a superstep.

During initialization all vertices are in active mode. The messages are passed between the vertices in a superstep. The largest vertex value is sent across to its neighbours. The vertex compares the value it contains with the one it receives. If the value is more than the value it contains, then the vertex retains
the larger value. In the proceeding supersteps, the vertex again communicates with its neighboring vertices and if it contains a value lesser than the one that is being received, then the value of the vertex changes to the higher value, else it remains the same. When no vertices change its value in a superstep, the algorithm terminates. The below diagrams explains Figure 2 and how communication takes place in each superstep.