Pregel: A System for Large-Scale Graph Processing

M Ravi Kumar   Dr. Ch Sobhan Babu

Mar 2023

Table of Contents

Motivation

Many practical computing problems concern large graphs (web graph, social networks, transportation network).

Example :

- Shortest Path
- Clustering
- Minimum Cut
- Connected Components

Related Work

- Creating a custom distributed framework for every new algorithm.
- Existing distributed framework MapReduce :
  Sub-optimal performance and have usability issues.
- Single-computer graph algorithm libraries like NetworkX,BGL:
  It is not scalable on large data.
- Existing parallel graph systems like parallel BGL:
  These are do not handle fault tolerance and other issues.

Need for a scalable distributed solution

Pregel

- Google come up with solution, distributed graph parallel computation frame work *Pregel*.
- Vertex centric computation (Think like a vertex).
- Inspired by *Valiant's Bulk Synchronous Parallel* model.
- Scalable and Fault-tolerant platform.
- API with flexibility to express arbitrary algorithm.

Computation Model

- Bulk Synchronous Parallel
  - Series of synchronous iterations (supersteps).
  - Vertex asynchronously executes some user-defined function in parallel in each superstep.
- Message-passing Model
  - Vertex reads messages sent in previous superstep.
  - Vertex sends messages, to be read by other vertices in the next superstep.
  - Vertex updates states of itself and its outgoing edges.

Bulk Synchronous Parallel

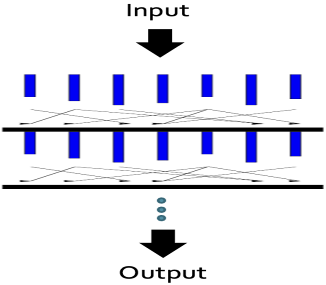

Figure: Bulk Synchronous Parallel Model

Sequence of iterations/superstep.

Bulk Synchronous Parallel

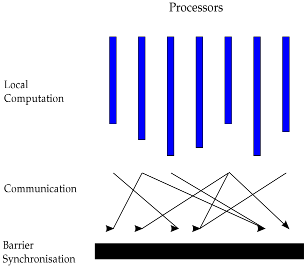

Figure: Bulk Synchronous Parallel Model (Single Superstep)

- Local Computation: every participating processor/thread may perform local computations.
- Communication: The processes exchange data with other process.
- Barrier synchronization : The process wait until all other process complete above two steps.

Vertex State Machine

Execution stops when all vertices have voted to halt and no vertices have messages.

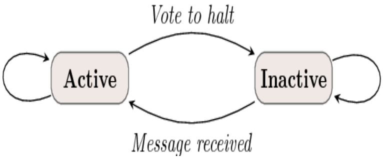

Figure: Vertex State Machine

Computation Model

- Input
  - Directed graph
- Pregel Computation
  - Partition the vertices and allocate them to CPU's/Threads.
  - Superstep 0: All vertices active, initialize vertex value, send message to the out going neighbours.
  - Superstep 1..N-1
    - Active vertex receive message from previous step.
    - Compute user defended function and update its value.
    - Sends messages to outgoing vertices.
    - Votes to halt if it has no further work to do.
    - Program terminated if all vertices are inactive.
- Output
  - Set of vertex updated values.

Maximum Vertex Value Example

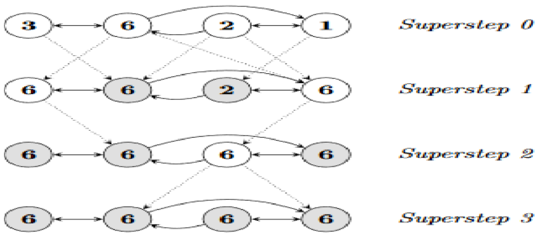

Figure: Maximum Vertex Value Example

Dotted lines are messages. Shaded vertices have voted to halt.

Applications

- PageRank
- Shortest Path
- Bipartite Matching
- Semi-clustering

PageRank

PageRank is link analysis alogithm to identify importance of a document based on the number of references to it and the importance of the source documents themselves.

A = A is given page
$T_1 \dots T_n$ = Pages that point to page A (citations)
d = Damping factor
C(T) = No of outgoing links of page T
N = Total no of pages.
PR(A) = PageRank of A

$$PR(A) = (1 - d)/N + d\left(\frac{PR(T_1)}{C(T_1)} + \dots + \frac{PR(T_n)}{C(T_n)}\right)$$

PageRank

```
class PageRankVertex
    : public Vertex<double, void, double> {
 public:
  virtual void Compute(MessageIterator* msgs) {
    if (superstep() >= 1) {
      double sum = 0;
      for (; !msgs->Done(); msgs->Next())
        sum += msgs->Value();
      *MutableValue() =
          0.15 / NumVertices() + 0.85 * sum;
    }

    if (superstep() < 30) {
      const int64 n = GetOutEdgeIterator().size();
      SendMessageToAllNeighbors(GetValue() / n);
    } else {
      VoteToHalt();
    }
  }
};
```

System Architecture

Pregel system uses the master/worker model

- Master
  - Partition the graph and assign input to workers.
  - Keep track of which worker holds which portion.
  - Recovers faults of workers.
- Worker
  - Load its portion of graph into memory.
  - Receive messages from neighboring vertices, process the task.
  - Update states of vertices, edges.

Fault Tolerance

- Checkpointing
  - The master periodically instructs the workers to save the state of their partitions to persistent storage system.
    e.g., Vertex values, edge values, incoming messages.
- Failure detection
  - Using regular "ping" messages
- Recovery
  - The master reassigns graph partitions to the currently available workers.
  - The workers all reload their partition state from most recent available checkpoint.

References

[1] G. Malewicz, M.H. Austern, A.J. Bik, J.C. Dehnert, I. Horn, N. Leiser, G. Czajkowski, Pregel: a system for large-scale graph processing, in SIGMOD (2010).

[2] Richard Miller, A Library for Bulk-Synchronous Parallel Programming. in Proc. British Computer Society Parallel Processing Specialist Group Workshop on General Purpose Parallel Computing, 1993.

[3] Luiz Barroso, Jeffrey Dean, and Urs Hoelzle, Web search for a planet: The Google Cluster Architecture. IEEE Micro 23(2), 2003, 22–28.

[4] Andrew Lumsdaine, Douglas Gregor, Bruce Hendrickson, and Jonathan W. Berry, Challenges in Parallel Graph Processing. Parallel Processing Letters 17, 2007, 5-20.

Thank you!