Hamburg,
Graph processing on Hadoop
Pelle Jakovits
Outline

- Last Talk
- Hamburg (Hadoop)
- Hama (Hadoop)
- Bulk Synchronous Parallel
- Pregel (Google)
- From here
Last Talk

• I talked about GraphGrep
  – Searching for sub graph matches in a database of graphs
  – Is not suited for very large graphs

• Decided to look into parallel graph algorithms and packages for cloud computing.
  – Hamburg
Hamburg

- Graph computing framework for Hadoop
- Moved under another project called Hama in the end of September 2009
- Based on Bulk Synchronous Parallel model
- Still work in the process
Hama

- Distributed scientific package on Hadoop
- Based on Map/Reduce and Bulk Synchronous Parallel models
- Consists of two packages:
  - Matrix package
  - Graph package
- Hama has been incubating since 19 May, 2008
Bulk Synchronous Parallel

• Computing model for parallel programming
• Iterations consist of 3 Supersteps:
  – Concurrent local computation
  – Communication between nodes
  – Barrier synchronisation
Bulk Synchronous Parallel

Processors

Local Computation

Communication

Barrier Synchronisation
Why BSP?

- Communication is done in mass:
  - Easier to manage
  - Easier to quantify
- Using barrier synchronisation
  - leaves no possibility of deadlocks
  - easier to achieve fault tolerance
- Compared to Map/Reduce it is more suited for problems that require graph traversing.
Hamburg (cont.)

- Graph data is divided between nodes
- Tries to store neighbouring vertices near each other by preprocessing the graph using Map/Reduce
- Computation iterations consist of two main steps:
  - Local computation one each node
  - Bulk synchronization between nodes
Hamburg
Hamburg example

On distributed storage
Hamburg example

- Find distances from all vertices to root vertex 1

![Diagram showing the Hamburg example with distances from vertices to root vertex.](image-url)
Pregel (Google)

- Scalable infrastructure for graph computations
- Uses Bulk Synchronous Parallel model
- Consists of sequences of Supersteps, which are performed on all active vertices concurrently.
- Google claims Pregel scales up to billions of vertices and edges.
- But very little concrete information published.
Pregel Superstep

- User defined function `Compute()` invoked concurrently on each vertex of the graph.

- At each iteration step `S`:
  - Read messages that were sent to it in step `S-1`
  - Do work on vertex `V`
  - If required, send messages to connected vertices, which will be received at step `S+1`
  - Wait for barrier synchronisation
  - Go to step `S+1`
From here

- Why use BSP? Is Map/Reduce model really not enough for graph problems?
- Currently preparing to start a thesis on topic: “Reducing scientific computing problems to MapReduce”
Thank you for listening

• Any questions?
References

- "Pregel: a system for large-scale graph processing", August 2009 http://portal.acm.org/citation.cfm?id=1582716.1582723