Cluster Middleware II: System Design

Group No: 4

April 5, 2016

1 Model

In present project, we intend to implement distributed load-balancing cluster middleware. The model is strongly connected, synchronous and reliable. A node (machine) may fail and rejoin. Jobs submitted at a node will be distributed among active nodes and results will be collected. Splitting the job and merging the results will be application dependent. For nodes in cluster, we handle only crash faults.

2 Components and their functionalities

2.1 When job is submitted at a node

- Ask all the neighbouring nodes if they are alive.
- Distribute jobs among active nodes according to load balancing algorithm.
- Send the job at the desired node. Maintain the information about job partitions and executing nodes.
- Keep asking the nodes to whom job was sent, if they are alive.
- If some node is failed, distribute the job assigned to that node to the remaining nodes.
- If some extra node i joins during the process, it will divide its self job (part of the submitted job) into half and send it to i.
- Get the results from all participating nodes and merge them.
- Display the result.

2.2 When node rejoins

- It will send a message to all other nodes that it has joined the system.

2.3 Node on receiving a job from another node

- Puts job in a job queue. Maintain sender's information for the job.
- It will always execute the first job from job queue.
- Upon completion of a job, send result to the sender of that job.

3 Communication messages

- **CheckAlive:** Heartbeat message to check if a node is active. On submission of a job at a node, it sends this message to all other nodes. Also, every node will periodically send this message to other nodes to which a part of a job is given to execute.
- **ReplyAlive:** An active node replies to heartbeat message. Absence of reply indicates failure of node.
- **Job:** Job is sent to an active node according to load balancing algorithm. It may be in the form of executable file and input file.
- **Result:** Result of executed job is sent to the owner node (node at which that job was submitted). It may be in the form of output file.
- **IAmUp:** On startup (or rejoin) a node sends this message to all other machines to indicate that it is active.

4 Implementation and demonstration

Implementation will be in C++ language on Linux operating system. Demonstration will be on command line, multiple processes run on a single machine or several machines can be used.