COMP 512 P3 Distributed Computing Platform

1 Description of the Architecture

Our implementation includes the following znodes:

Znode	type	Usage
/dist42/master	ephemeral	the master node in zookeeper
/dist42/workers	persistent	contains all the workers in the zookeeper
/dist42/workers/workerid	EPHEMERAL	an individual worker znode
/dist42/tasks	persistent	listed all the tasks that client proposes
/dist42/tasks/taskid	persistent-sequential	the particular id of one task
/dist42/tasks/taskid/result	persistent	the result of the corresponding task
/dist42/tasks/taskid/assigned	ephemeral	indicate whether the tasks is assigned to a worker

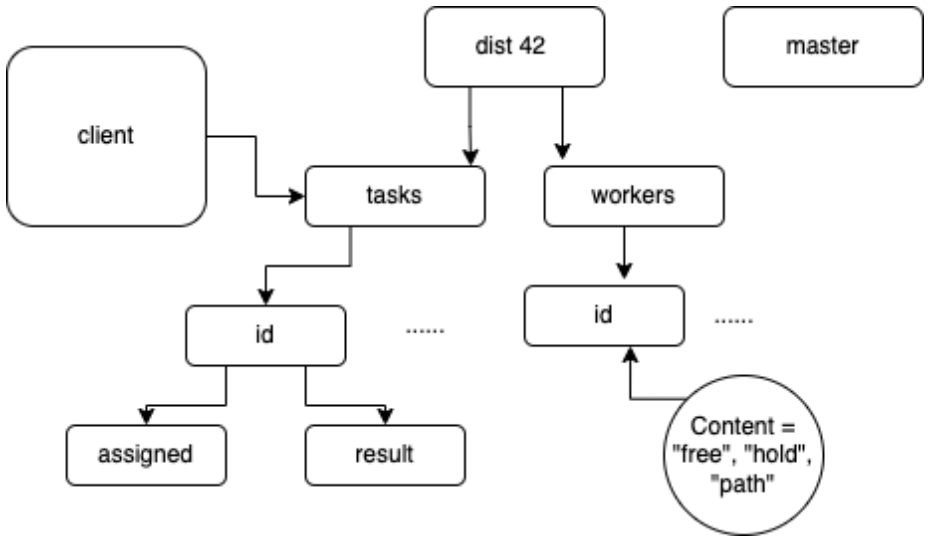

Figure 1: The tree diagram of our implementation

2 Work Flow

2.1 Master

Initially when a server is launched, it will try to create the "/dist42/master" znode in the process and becomes the master if succeeded.

This master will install asynchronous watcher to the "/dist42/workers" directory to detect if there is a new worker added.

Similarly, it will also install watcher to the "/dist42/tasks" directory to check for new tasks added.

When a new worker is added, the master process will try to find an unassigned task through the callback function: Loops through the children of "/dist42/tasks" directory, if there exist a task that does not have a "/assigned" child node, The unassigned task will be assigned to the free worker by writing unassigned taskpath to the content of free worker's node. It also reinstall the watcher to the "/dist42/workers" znode.

Similarly, When a new task is added, the master will also try to find a free worker by checking which the content of which worker node is "free", if successful, master write the unassigned taskpath to the free worker's directory. It also reinstall the watcher to the "/dist42/tasks" znode.

When any assignment between unassigned task and free worker is successful, the master install a watcher under the "/dist42/tasks/taskid/result" directory. If the watcher spot a creation of the znode "/dist42/tasks/taskid/result", the master would know there is a worker finished computation and becomes free. Then the master do the same thing as it spot a new free worker is added

2.2 Workers

When a second or later server is launched, it will first try to create the master znode, since the master already exist, it will raise and catch an nee exception then try to create a worker znode at "/dist42/workers/"+¡workerid¿. At the same time, they will register a workerwatcher on themselves. If the master write any taskpath in the contents of the "workers/workerid", it will be spot by the workerwatcher.

The contents inside the workers are the following: (in Bytes)

"free": this is the initial state of a worker, indicates that it is not calculating a task and free to be assigned

"/dist42/tasks/"+¡taskid¿:

this is the path of an unperformed task, once the watch of the worker spot that the content is not "free", they will get the Data inside the task node through this taskpath. It will then process the computation in another thread to avoid extra waiting. Once the task is finished, the worker node will again set to "free", a result under the "/dist42/tasks/id/result" directory is also created and the workerwatcher is re-installed by the worker on itself.

The workers creation, writing actions on their result directory will be triggered by this resultwatcher. When the node is created by the worker, this means

that a worker has finished the current task and is thus free. In this case, it will be treated the same as if the worker is created: Master will loop through the tasks directory, find an unassigned task (if there is any) and write the task path to the content of a free worker's znode.