coreservlets.com – Hadoop Course

Job Execution on YARN

In this exercise, you will run a MapReduce job that was designed to randomly fail tasks. This will give you a chance to observe how YARN and MapReduce framework deal with failures. You will also have a chance to become further familiar with Management UI and how it relates to the framework components.

Approx. Time: 45 minutes

Perform

Run mapRed.jobOnYARN.JobWithFailures MapReduce job and observe it's behavior.

```
$ yarn jar $PLAY_AREA/Exercises.jar \
      mapRed.jobOnYARN.JobWithFailures \
      /training/data/hamlet.txt \
      /training/playArea/JobsOnYarn
```

With the given input this job will generate 3 splits = 3 map tasks. Each mapper will run for 5 minutes with a 25% chance of failure every 25 seconds. Eventually tasks will start failing and the framework will perform recovery process. Enough failures may case your job to fail but no run will be quite the same. I guess that's part of the fun.

Answer the following questions by re-running mapRed.jobOnYARN.JobWithFailures and observing its behavior.

Answer

1. Identify which page represents YARN Resource Manager UI
2. Identify which page represents Application's information
   - Observe this page while a job is running and after completion. How does it differ? Why does it differ?
3. Identify which page represents MapReduce Application Master Page
   - What happens to this page after job's completion?
4. Identify which page represents Job's information which includes both tasks and ApplicationMaster (in this case MRAppMaster)
   - Observe this page while a job is running and after completion. Why does it differ?
5. On the job's page identify areas that represent tasks and areas that represent attempts. How do tasks and attempts relate?
6. There will be a number of tasks failing. What happens when a map tasks fails? Identify recovery mechanism and how MapReduce framework is handling recovery?
7. Eventually the job will fail. Why did it fail? You may see that some tasks failed and some were killed, why the difference in status?
   - If the job doesn't fail right away, re-run it as chance is on your side that it will fail eventually.

Solution

1. YARN Resource manager can be found on port 8088: http://localhost:8088/cluster. This management UI lists information about recent applications. A full history is maintained by the MapReduce History Server that runs on port 19888: http://127.0.1.1:19888/jobhistory
2. From Resource Manager home locate your application, by default it's the last on the list and may not be on the first page. You can sort by dates or go to the last page. Click on the link and it will take you to an Application's Management UI.

   While application is running a link under 'Tracking URL' will show 'Application Master'. After the job completes this link will say 'History'. This happens because while an application is running Application Master will host management and metrics information about the job and upon completion metrics and management information is shipped over to the history server. Resource Manager proxies to the proper places depending on the current state of an application.
3. From Application information page click on 'Application Master' labeled with 'Tracking URL'. If the link instead displays 'History' then the application completed and Application Master doesn't exist anymore. You will be directed to the history server for the job's information.
4. While the job is running you can get to Job's information page by clicking on the job-id link from 'Application Master' page. After the job completes the Application information page will take you directly to the Job's page on the history server. The page will have a minor variation between MRAppMaster vs. History Server.
5. On the bottom the panel is broken down into task and attempts. Each task consists of attempt. If an attempt fails the task will be re-tried with another attempt. By default framework will create up to four attempts for a task. For example, you can click on 'Map' link in the task panel and it will show you current Map tasks. You can then click on a task to see its attempts. Since this job was designed to fail task you are very likely to see several attempts for a task.
6. The failure is propagated back to MRAppMaster and task's attempt is labeled as failed. If attempt's JVM is still running it will be shut down. By default another attempt will be created as long as there are less than 4 failures. If a task has 4 failed attempts then that task will be labeled as failed.
7. The job failed because a task had 4 failed attempts. A failure in the task will cause job failure and any currently running task will be killed. This is why you may see some tasks labeled as failed and some as killed. In case of this task even the killed job are likely to have failed attempts. See if you can find those attempts.