744 HW1

Kesong Cao, Mingchen Ma, Xuxiang Sun

September 2022

1 Task 1

In this task, we run the page rank algorithm with partition number by default. The running result is included in the following picture.

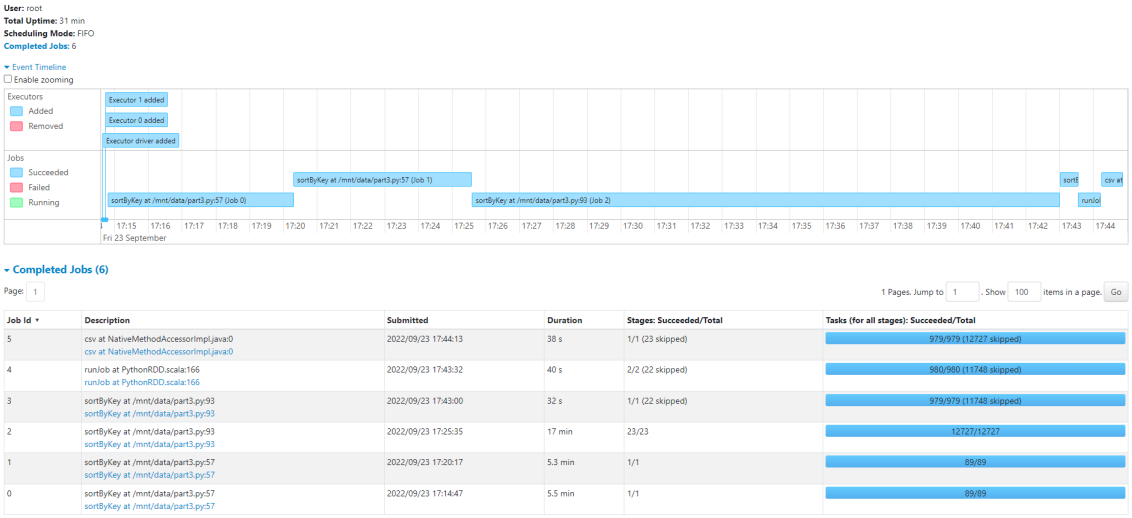

Figure 1: Page rank with partition number by default

We can see that the total running time is 31 minutes.

2 Task 2

In this task, we run the page rank algorithm with different partition numbers. In particular, we run the page rank algorithm with partition number 2, 5, 64,200, 512. We first give a comparsion of the results under different settings in Figure 2.

In this picture, we can see that if we set the partition number to be 64, the running time is 24 minutes. If we set the partition number to be very small, for example, 2 or 5, then the running time will be very long. When the partition number is 5, the running time is 45 minutes and when the partition number is 2, the running time is 1.6 hours. On the other hand, if we set the partition number to be very large, then the running time will not be shorten. When the partition number is 200, the running time is 28 minutes and when the partition number is 512, the running time is 31 minutes.

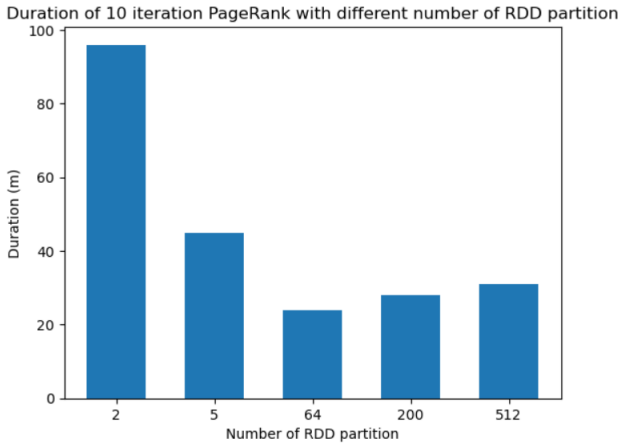

Figure 2: Page rank with different partition numbers

2.1 Discussion for Task 2

In this part, we give a brief discussion for the impact of the partition number for the running time. We have seen from the experiment that too few or too many partitions will not benefit for the running time. When the partition number is too small, it is hard to do computation in parallel and may cause improper resource utilization. This will make the program run longer. On the other hand, if the partition number is too large, then it may take longer time to do scheduling than actual execution time.

3 Task 3

In this task, we compare page algorithms in two different settings. In the first setting we run the page rank algorithm by default. In the second setting we persist an appropriate DataFrame/RDD(s) as in-memory objects. We include the results for the second setting in the following picture.

We can see from the picture that the running time is shorter then the case when we run the algorithm by default. We try to explain this phenomenon briefly. When we compute the page rank, sometimes we actually need to use some RDD multiple times. Assume we don't persist RDD in memory, then we may need to repeat the RDD evaluation several times and it may take a longer time. This implies if we persist appropriate RDD in memory, then we will save some time.

4 Task 4

In this task, we will kill a worker when application reaches 25% and 75% of its lifetime. In general, we summarize the results in the following picture.

When we kill a worker when the application reaches 25% its lifetime, we use roughly 39 minutes to complete the total task. When we kill a worker when the application reaches 75% its lifetime, we use roughly 48 minutes to complete the total task.

We first include the running results for the first case in the following picture.

We can see that when we kill a worker, the program has not started to compute the rank of each web page by iteration. Thus, when a worker fails we only need to perform job 1 again. Furthermore, since there is only one worker left, it will take us more time to compute the rank of each web page. Finally, it takes about 38 minutes to finish the task.

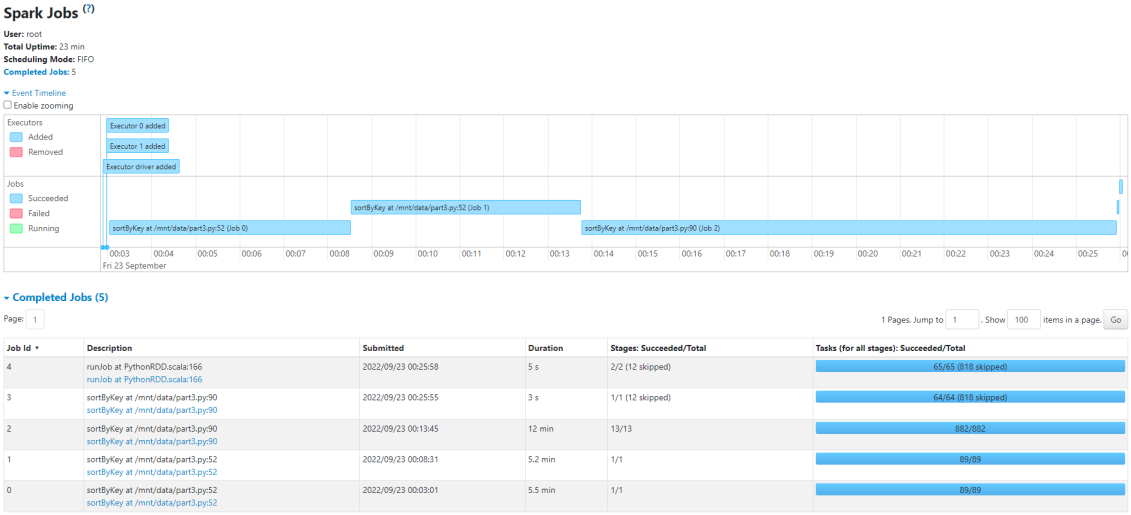

Figure 3: Persist RDD in memory

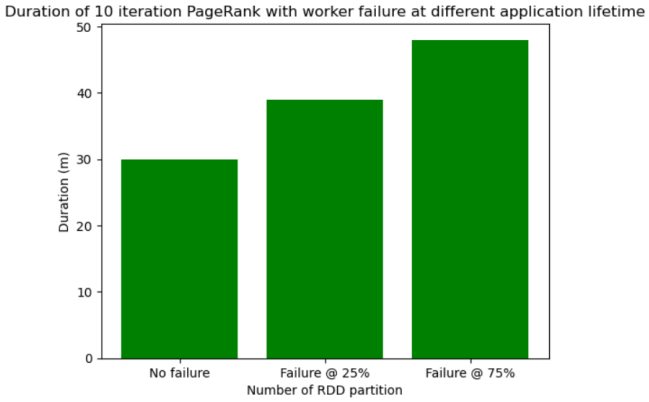

Figure 4: Page rank with failure times

Next, we include the running results for the second case in the following picture.

When we kill a worker, the program has performed many iterations to compute the rank of each web page and has finished job 1 and job 2. This implies that the program don't need to perform job 1 and job 2 again, but instead need to perform computation that has been performed by the killed worker. Furthermore, the computation should be done by a single node and thus need more time.

5 Contribution

In this assignment, Kesong and Xuxiang set up the enviroment. Kesong wrote down the python code for task 2 and task 3. Kesong, Mingchen and Xuxiang perform experiments for task 1-4. Mingchen organized the running report and wrote this report. Xuxiang organized the code for the assignment.

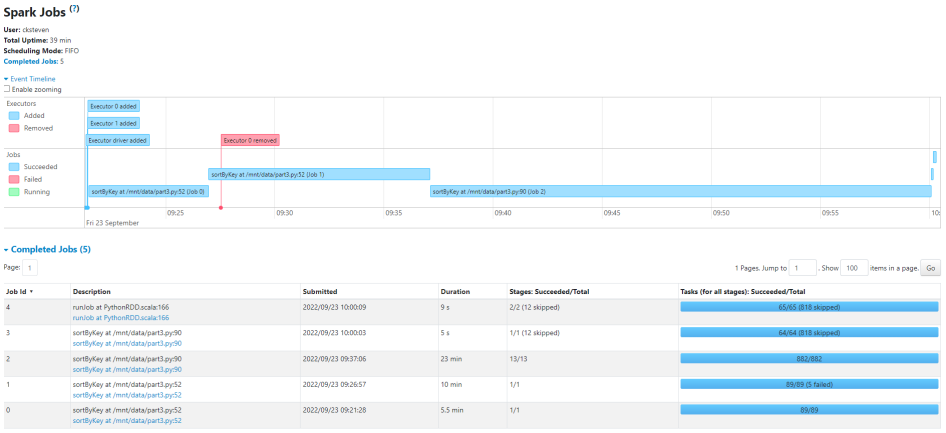

Figure 5: Kill a worker at 25% time

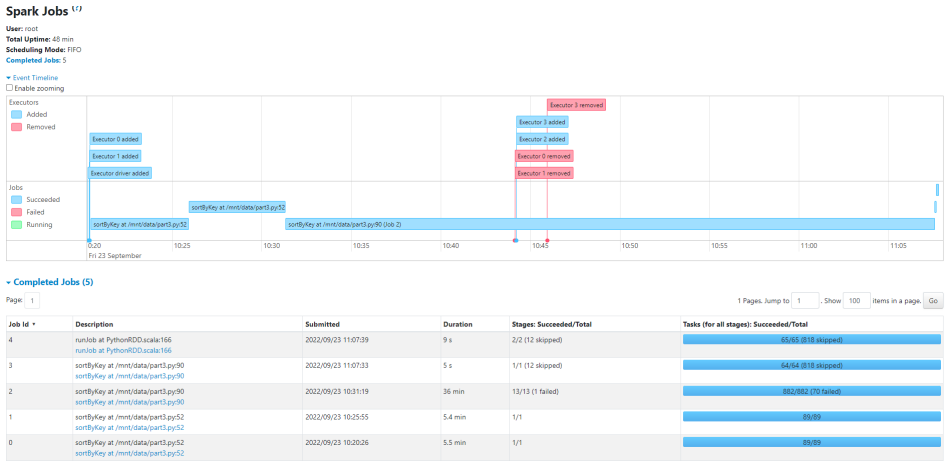

Figure 6: Kill a worker at 75% time

A Pictures for Task 2

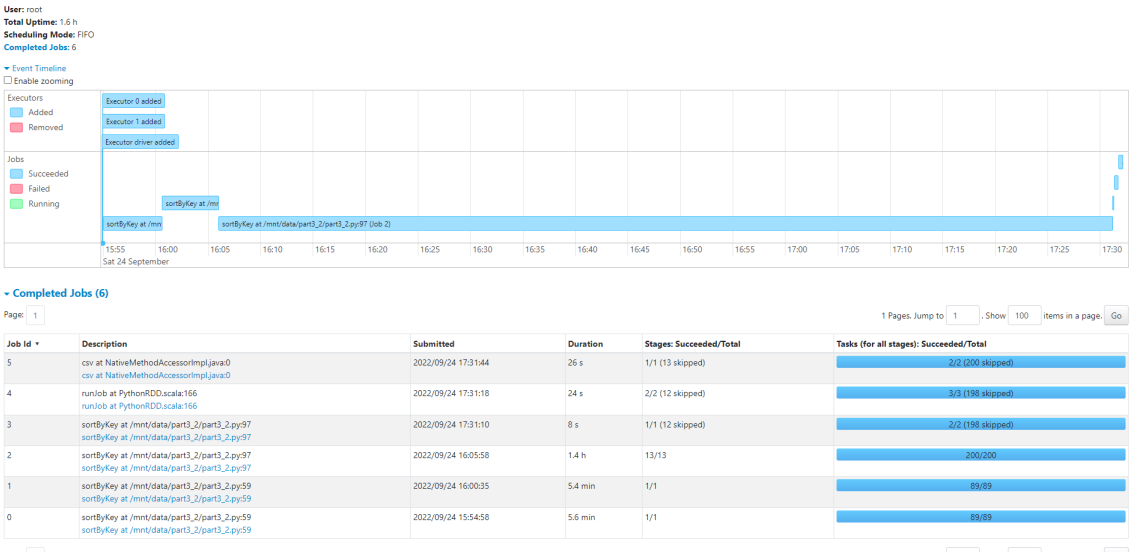

Figure 7: Page rank with partition number 2

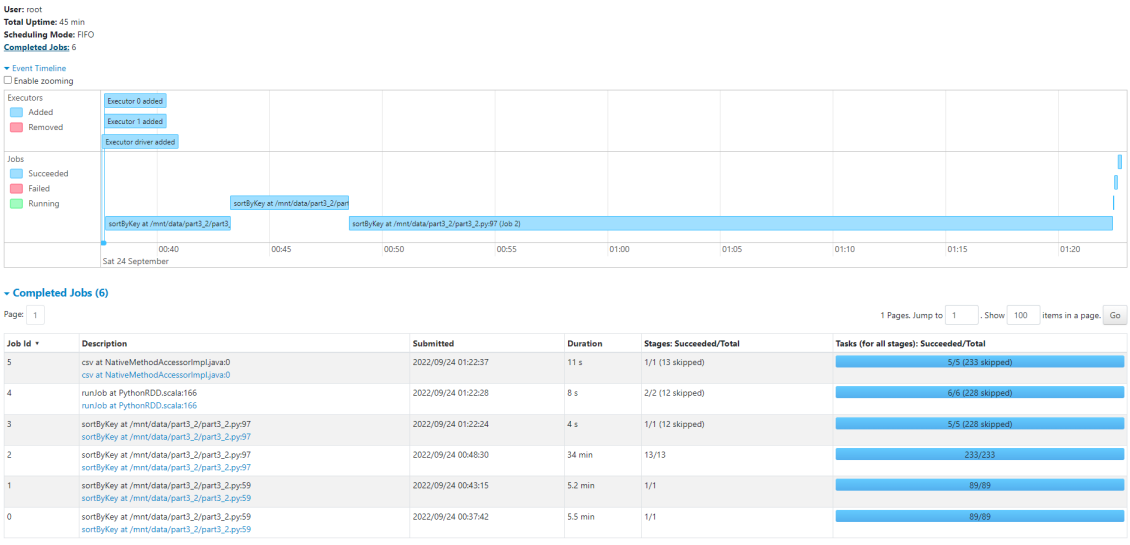

Figure 8: Page rank with partition number 5

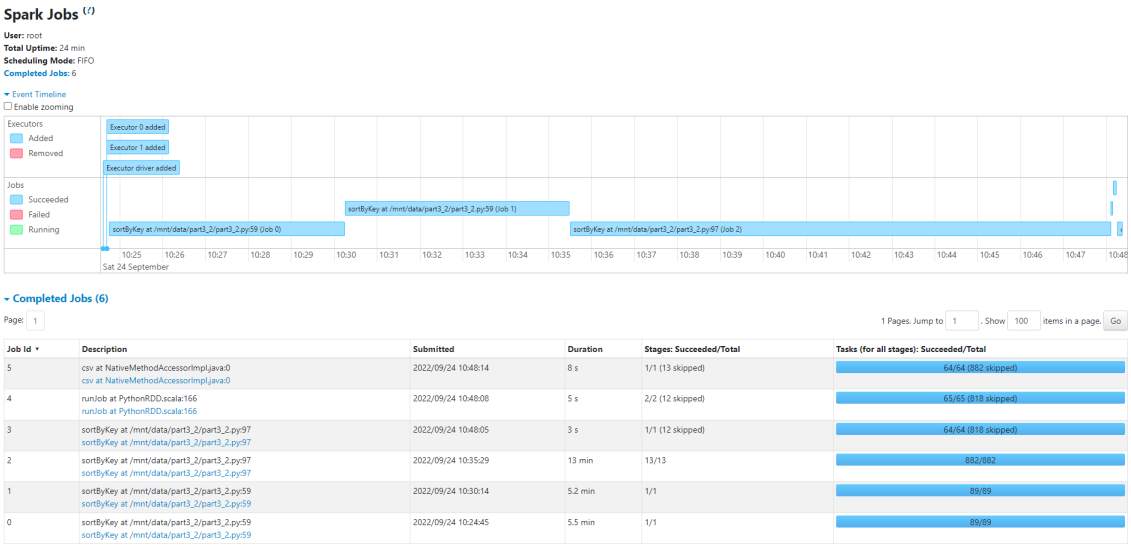

Figure 9: Page rank with partition number 64

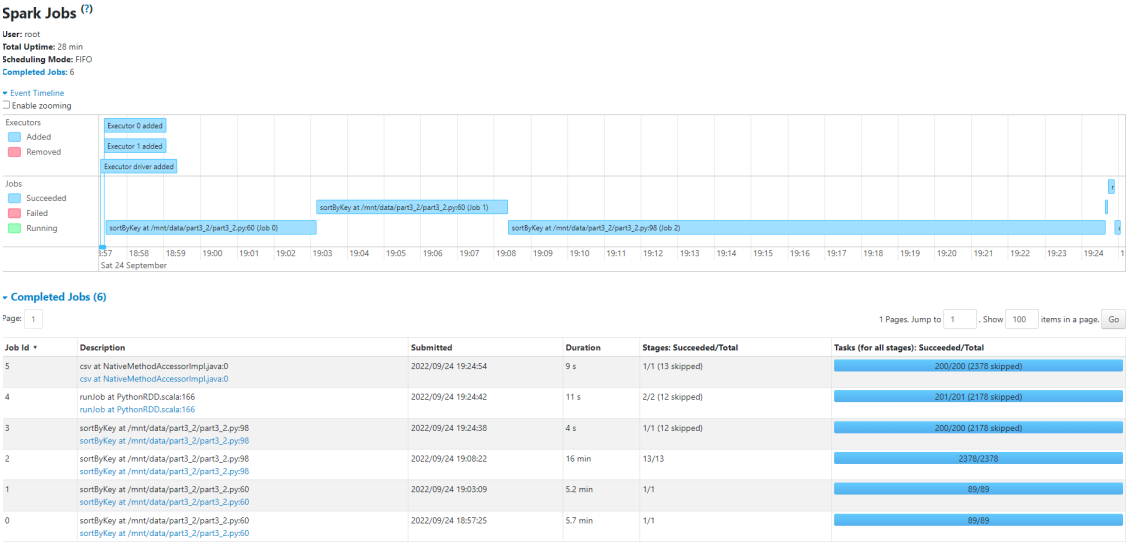

Figure 10: Page rank with partition number 200

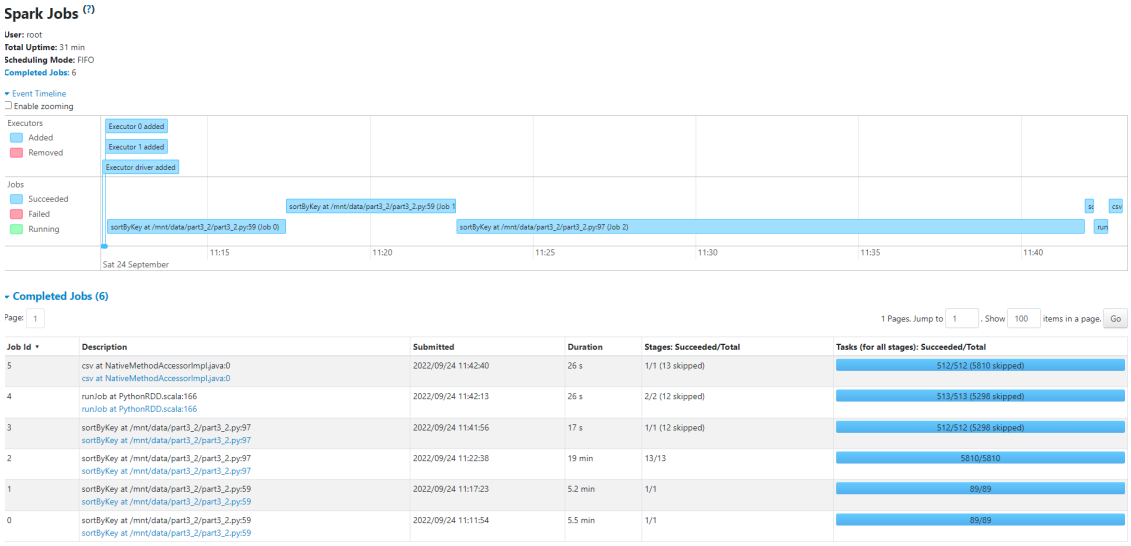

Figure 11: Page rank with partition number 512