BauerMarkus-11721787-H01

The project is exported from IntelliJ and runs with Maven dependencies. For AWS Login to work there is a file in the project root called “awsconfig_template”. Follow the instructions there to make the Login work. At first load the project into IntelliJ as a Maven project and import the dependencies.

1. Starting an EC2 instance

1.1. Development Task

This development task is implemented in class „Task_2_Child“.
It can be executed with these instructions:

- Prerequisites: „input_full.csv“ and „calc_fib.jar“ in project root folder
- execute from IntelliJ with arguments: t2.micro full
  (arg0: InstanceType, arg1: file name ‚input_%.csv‘)

1.2. Experimental Evaluation Task

Finished Computation - TIME MEASUREMENTS(t2.large; Child ID: full; VM ID:i-088c5a71cd2d9142f)

VM Startup: 13624 MS = 13 S

Java Installation: 20812 MS = 20 S

Upload: 547 MS = 0 S

Execution: 216956 MS = 216 S

Download: 243 MS = 0 S

Sum of Times: 252182 MS = 252 S

Total Time Measured (i.e. +waiting time): 267756 MS = 267 S

2. Offload Computing to the cloud

2.1. Development Task

This development task is implemented in class „Task_2_Child“.
It can be executed with these instructions:

- Prerequisites: CSV File manually split into „input_0.csv“, „input_0.csv“ and „calc_fib.jar“ in project root folder
- execute from IntelliJ with arguments: t2.micro t2.large
  (args: InstanceTypes)

The number of instances is variable – though the CSV File needs to be split manually still. (i.e. 3 instances -> 3 csv files)

2.2. Experimental Evaluation Task

Since the complexity of the Fibonacci sequence is 2^n the last Fibonacci numbers in the sequence are gonna take up a lot of execution time. Therefore for this input the last 2 fibonacci numbers are gonna be calculated by a faster t2.large VM and the rest by a smaller t2.micro VM.

- Fibonacci 1-45 was done by Child 0 (t2.micro)
- Fibonacci 46-47 was done by Child 1 (t2.large)

```
---------------------------
Finished Computation - TIME MEASUREMENTS(t2.micro; Child ID: 0; VM ID:i-0b88ff2a3c5b9d614)
VM Startup: 21160 MS = 21 S
Java Installation: 19868 MS = 19 S
Upload: 606 MS = 0 S
Execution: 92645 MS = 92 S
Download: 247 MS = 0 S
---------------------------
Sum of Times: 134526 MS = 134 S
Total Time Measured (i.e. +waiting time): 158434 MS = 158 S
---------------------------
Finished Computation - TIME MEASUREMENTS(t2.large; Child ID: 1; VM ID:i-011be9905849cea9e)
VM Startup: 15405 MS = 15 S
Java Installation: 12853 MS = 12 S
Upload: 562 MS = 0 S
Execution: 133175 MS = 133 S
Download: 246 MS = 0 S
---------------------------
Sum of Times: 162241 MS = 162 S
Total Time Measured (i.e. +waiting time): 186556 MS = 186 S
```

2.3. Observation and Analysis

Local Execution Time 105 S

Locally the performance for this task was the best. This is due to the fact that the computation effort needed is relatively low. So the time for starting, configuring and uploading/downloading the files is too time consuming to get any benefit from it.