

IN5020 Distributed Systems - Assignment 1

Team members: Caroline Santos Alvær and Andreas Askim Vatne

Implementation and Design

The implementation of the client and server was mostly provided by the assignment text. However, there were certain design choices that were made by us.

Cache

The cache for both the server and the client was created using a LinkedHashMap. This gave us the necessary functionality to be able to update the cache accordingly and release the least recently used entries as necessary.

At the end of the program execution the client creates the server_cache.txt output file by calling upon the remote method getServerCaches in the proxy server. We decided this was the best solution seeing as the proxy server could easily bind to the stub of each server and access their individual caches, and return this as a list to the client.

Client

On the client side, we decided to place every remote method invocation in a runnable task and place every task within its own thread. The thread was then added to an array list of threads. To avoid that the main thread executes the calculation of the average times and creation of output files before the pool of threads were completed, we used the .join method on all threads created to ensure their completion before the main thread was allowed to resume.

Server

In the server and client class we implemented a simulate latency method to simulate the communication latency that would exist in a distributed network.

ProxyServer

The proxy was the part of the assignment where we were most free to do as we please regarding the design and implementation. In the proxy class we created an inner class called ServerNode with attributes to keep track of the workload of the different servers, as well as how many tasks the proxy had assigned to each server. The proxy server then started each server and binded each server to a name in the local registry.

To keep track of the workload of the servers without disturbing their execution of the client queries we implemented a thread pool with 5 threads that are reused. When a task to check the server workload is created, it is submitted to the executor object and one of the available threads handles the task asynchronously.

Screenshots of client

```
o Carolines-MBP:INS020 carolsal$ java ProxyServer
Server 1 is running...
Server 2 is running...
Server 3 is running...
Server 4 is running...
Server 5 is running...
Proxy is running...
[]

getNumberOfCountries 32 63390 152912 Zone:2: 30 --turnaroundtime: 49795 executiontime: 121 waitingtime: 49674
getNumberOfCountries 16 17136 71980 Zone:5: 106 --turnaroundtime: 49217 executiontime: 84 waitingtime: 49132
getNumberOfCountries 32 63390 152912 Zone:2: 30 --turnaroundtime: 49874 executiontime: 85 waitingtime: 49789
getNumberOfCountries 16 17136 71980 Zone:5: 106 --turnaroundtime: 49295 executiontime: 85 waitingtime: 49210
getNumberOfCountries 32 63390 152912 Zone:2: 30 --turnaroundtime: 49953 executiontime: 85 waitingtime: 49866
getNumberOfCountries 16 17136 71980 Zone:5: 106 --turnaroundtime: 49374 executiontime: 85 waitingtime: 49289
getNumberOfCountries 32 63390 152912 Zone:2: 30 --turnaroundtime: 50033 executiontime: 85 waitingtime: 49947
getNumberOfCountries 26 36349 85074 Zone:5: 48 --turnaroundtime: 49430 executiontime: 123 waitingtime: 49307
getNumberOfCountries 18 69984 119037 Zone:2: 29 --turnaroundtime: 50146 executiontime: 118 waitingtime: 50026
getNumberOfCountries 26 36349 85074 Zone:5: 48 --turnaroundtime: 49505 executiontime: 80 waitingtime: 49424
getNumberOfCountries 1 75277 110899 Zone:2: 149 --turnaroundtime: 50250 executiontime: 125 waitingtime: 50125
getNumberOfCountries 41 99830 129236 Zone:5: 6 --turnaroundtime: 49622 executiontime: 123 waitingtime: 49497
getNumberOfCountries 1 75277 110899 Zone:2: 149 --turnaroundtime: 50329 executiontime: 85 waitingtime: 50244
getNumberOfCountries 23 53247 109979 Zone:5: 37 --turnaroundtime: 49719 executiontime: 117 waitingtime: 49602
getNumberOfCountries 49 40125 66331 Zone:2: 18 --turnaroundtime: 50325 executiontime: 124 waitingtime: 50201
getNumberOfCountries 23 53247 109979 Zone:5: 37 --turnaroundtime: 49792 executiontime: 80 waitingtime: 49711
getNumberOfCountries 49 40125 66331 Zone:2: 18 --turnaroundtime: 50404 executiontime: 85 waitingtime: 50319
getNumberOfCountries 23 53247 109979 Zone:5: 37 --turnaroundtime: 49871 executiontime: 85 waitingtime: 49786
getNumberOfCountries 49 40125 66331 Zone:2: 18 --turnaroundtime: 50482 executiontime: 84 waitingtime: 50398
getNumberOfCountries 23 53247 109979 Zone:5: 37 --turnaroundtime: 49948 executiontime: 81 waitingtime: 49866
getNumberOfCountries 2 42487 106841 Zone:2: 149 --turnaroundtime: 50577 executiontime: 125 waitingtime: 50451
getNumberOfCountries 37 59132 141719 Zone:5: 29 --turnaroundtime: 50048 executiontime: 119 waitingtime: 49927
getNumberOfCountries 40 59748 120315 Zone:2: 23 --turnaroundtime: 50627 executiontime: 119 waitingtime: 50508
getNumberOfCountries 1 35536 38625 Zone:2: 112 --turnaroundtime: 50685 executiontime: 122 waitingtime: 50563
getNumberOfCountries 1 35536 38625 Zone:2: 112 --turnaroundtime: 50761 executiontime: 81 waitingtime: 50679
getNumberOfCountries 1 35536 38625 Zone:2: 112 --turnaroundtime: 50835 executiontime: 82 waitingtime: 50753
getNumberOfCountries 1 35536 38625 Zone:2: 112 --turnaroundtime: 50911 executiontime: 81 waitingtime: 50830
getNumberOfCountries 31 41697 128218 Zone:2: 42 --turnaroundtime: 50990 executiontime: 120 waitingtime: 50869
getNumberOfCountries 31 41697 128218 Zone:2: 42 --turnaroundtime: 51055 executiontime: 81 waitingtime: 50972
getNumberOfCountries 31 41697 128218 Zone:2: 42 --turnaroundtime: 51134 executiontime: 80 waitingtime: 51044
getNumberOfCountries 50 79312 155371 Zone:2: 14 --turnaroundtime: 51207 executiontime: 122 waitingtime: 51084
getNumberOfCountries 40 45661 102827 Zone:2: 29 --turnaroundtime: 51318 executiontime: 119 waitingtime: 51199
getNumberOfCountries 40 45661 102827 Zone:2: 29 --turnaroundtime: 51397 executiontime: 85 waitingtime: 51312
getNumberOfCountries 40 45661 102827 Zone:2: 29 --turnaroundtime: 51476 executiontime: 85 waitingtime: 51391
getNumberOfCountries 40 45661 102827 Zone:2: 29 --turnaroundtime: 51551 executiontime: 81 waitingtime: 51469
getNumberOfCountries 5 74568 91737 Zone:2: 45 --turnaroundtime: 51607 executiontime: 124 waitingtime: 51483
getNumberOfCountries 5 74568 91737 Zone:2: 45 --turnaroundtime: 51685 executiontime: 84 waitingtime: 51600
getNumberOfCountries 35 94789 129232 Zone:2: 8 --turnaroundtime: 51724 executiontime: 120 waitingtime: 51603
getNumberOfCountries 26 108363 151844 Zone:2: 10 --turnaroundtime: 51797 executiontime: 122 waitingtime: 51675
getNumberOfCountries 45 109430 135643 Zone:2: 3 --turnaroundtime: 51848 executiontime: 121 waitingtime: 51727
getNumberOfCountries 42 48717 74733 Zone:2: 19 --turnaroundtime: 51959 executiontime: 121 waitingtime: 51838
getNumberOfCountries 42 48717 74733 Zone:2: 19 --turnaroundtime: 52037 executiontime: 81 waitingtime: 51956
getNumberOfCountries 42 48717 74733 Zone:2: 19 --turnaroundtime: 52116 executiontime: 85 waitingtime: 52031
```

User Guide

The distributed application can be compiled and run in the following two ways:

1 - Manually compile and run the application

Compile: `javac *.java`

Run: `java ProxyServer exercise_1_input.txt`

2 - Create a deployable jar file and run the application

Create a directory called bin

Compile: `javac -d bin *.java`

Jar-file creation: `jar cf assignment1.jar -C bin .`

Extract Manifest file: `jar xvf assignment1.jar META-INF/MANIFEST.MF`

Add main class path to manifest file: Main-Class: ProxyServer

Save changes to manifest file: `jar uvfm assignment1.jar META-INF/MANIFEST.MF`

Run: `java -jar assignment1.jar exercise_1_input.txt`

Division of workload

Caroline and Andreas did 50/50 of the workload, co-wrote the whole code.