Resource Utilization Maximiser for Distributed Systems

Name: Murtaza Karimi
Student ID: 45606153

Introduction

Our task in this project is to design and implement a new client in java which can dispatch/ schedule jobs to servers in a distributed system. Normally, scheduling contains many constraints such as execution time, turnaround time, resource utilization and the cost of the executions. Our job is to design and implement at least one new scheduling algorithm that can optimise one of the above constraints. We should aim our new scheduling algorithm to either minimise the average turnaround time, minimise the total server cost or maximise the average resource utilization. Whilst we're designing this new scheduling algorithm, we must keep in mind that the optimisation of one of the constraints can lead to sacrificing the optimisation of the other constraints. Hence we must be careful when designing the new algorithm by making compromises at the right places. We are also required to discuss our algorithm results in comparison with the three baseline algorithms.

Problem Definition

Our problem in this project is that when we dispatch and/or schedule jobs to servers in a distributed system, they normally contain many constraints such as execution time, turnaround time, resource utilization and cost. Whilst designing our new algorithm, we made sure to be careful because sometimes when you optimise the constraints, another constraint might need to be sacrificed. In our newly designed algorithm, we made it so that we maximise the average resource utilization but at the cost of increasing the turnaround time.

Algorithm Description

We use all to largest method by finding the largest server and then scheduling the job to that one.

Below are example scheduling scenarios.

```
# Welcome  Murtaza!
# The system information can be read from 'ds-system.xml'
SENT OK
RCVD REDY
SENT JOBN 32 0 728 1 700 600
RCVD GETS All
SENT DATA 3 124
RCVD OK
SENT tiny 0 inactive -1 1 4000 32000 0 0
small 0 inactive -1 2 8000 64000 0 0
medium 0 inactive -1 4 16000 128000 0 0
RCVD OK
SENT .
RCVD SCHD 0 medium 0
t:         32 job     0 (waiting) on # 0 of server medium (booting) SCHEDULED
SENT OK
RCVD REDY
SENT JOBN 54 1 1144 1 400 800
RCVD SCHD 1 medium 0
t:         54 job     1 (waiting) on # 0 of server medium (booting) SCHEDULED
SENT OK
RCVD REDY
SENT JOBN 55 2 260 2 900 1600
RCVD SCHD 2 medium 0
t:         55 job     2 (waiting) on # 0 of server medium (booting) SCHEDULED
SENT OK
RCVD REDY
t:         92 job     0 on # 0 of server medium RUNNING
t:         92 job     1 on # 0 of server medium RUNNING
t:         92 job     2 on # 0 of server medium RUNNING
SENT JOBN 108 3 151 2 500 3300
RCVD SCHD 3 medium 0
t:        108 job     3 (waiting) on # 0 of server medium (active) SCHEDULED
SENT OK
RCVD REDY
t:        269 job     2 on # 0 of server medium COMPLETED
t:        269 job     3 on # 0 of server medium RUNNING
SENT JCPL 269 2 medium 0
RCVD REDY
SENT JOBN 287 4 3936 4 1600 4600
RCVD SCHD 4 medium 0
t:        287 job     4 (waiting) on # 0 of server medium (active) SCHEDULED
SENT OK
RCVD REDY
t:        406 job     3 on # 0 of server medium COMPLETED
SENT JCPL 406 3 medium 0
RCVD REDY
t:       1922 job     0 on # 0 of server medium COMPLETED
SENT JCPL 1922 0 medium 0
RCVD REDY
t:       2285 job     1 on # 0 of server medium COMPLETED
t:       2285 job     4 on # 0 of server medium RUNNING
SENT JCPL 2285 1 medium 0
RCVD REDY
t:       4826 job     4 on # 0 of server medium COMPLETED
```

```
t:         55 job     2 (waiting) on # 0 of server medium (booting) SCHEDULED
SENT OK
RCVD REDY
t:         92 job     0 on # 0 of server medium RUNNING
t:         92 job     1 on # 0 of server medium RUNNING
t:         92 job     2 on # 0 of server medium RUNNING
SENT JOBN 108 3 151 2 500 3300
RCVD SCHD 3 medium 0
t:        108 job     3 (waiting) on # 0 of server medium (active) SCHEDULED
SENT OK
RCVD REDY
t:        269 job     2 on # 0 of server medium COMPLETED
t:        269 job     3 on # 0 of server medium RUNNING
SENT JCPL 269 2 medium 0
RCVD REDY
SENT JOBN 287 4 3936 4 1600 4600
RCVD SCHD 4 medium 0
t:        287 job     4 (waiting) on # 0 of server medium (active) SCHEDULED
SENT OK
RCVD REDY
t:        406 job     3 on # 0 of server medium COMPLETED
SENT JCPL 406 3 medium 0
RCVD REDY
t:       1922 job     0 on # 0 of server medium COMPLETED
SENT JCPL 1922 0 medium 0
RCVD REDY
t:       2285 job     1 on # 0 of server medium COMPLETED
t:       2285 job     4 on # 0 of server medium RUNNING
SENT JCPL 2285 1 medium 0
RCVD REDY
t:       4826 job     4 on # 0 of server medium COMPLETED
SENT JCPL 4826 4 medium 0
RCVD REDY
SENT NONE
RCVD QUIT
SENT QUIT
# -------------------------------------------------------------------------------------
# 0 tiny servers used with a utilisation of 0.00 at the cost of $0.00
# 0 small servers used with a utilisation of 0.00 at the cost of $0.00
# 1 medium servers used with a utilisation of 100.00 at the cost of $1.05
# ================================== [ Summary ] ==================================
# actual simulation end time: 4826, #jobs: 5 (failed 0 times)
# total #servers used: 1, avg util: 100.00% (ef. usage: 100.00%), total cost: $1.05
# avg waiting time: 458, avg exec time: 1375, avg turnaround time: 1833
```

Implementation Details/ Data Structures

The data structures that we used in this project are strings and integer arrays. We also used Sockets, DataInputStreams and PrintStreams.

Evaluation

There are a range of benefits as to why we designed our algorithm this way. This is because we tried to maximize resource utilization when scheduling jobs in a distributed system. This improved one of the constraints but at the cost of effecting another. The cons of our newly designed algorithm did however maximise the resource utilization but it also increased the turnaround time.

Conclusion

In conclusion, for this project we were asked to create and implement a new client in java which can dispatch/ schedule jobs to servers in distributed systems. Now, when scheduling jobs to servers, there are many issues and restrictions which we face such as the turnaround time, resource utilization, execution time and cost. So my job was to design a new scheduling algorithm which would overcome at least one or more of these constraints. I designed my algorithm so that it can maximise the resource utilization but at the cost of increased turnaround time. All in all, you can improve some of the constraints but at the cost of sacrificing some other constraints.

Reference:

Github link

- https://github.com/Murtaza280/Distributed-System-Assignment-2
- https://en.wikipedia.org/wiki/C_dynamic_memory_allocation
- https://www.geeksforgeeks.org/first-fit-allocation-in-operating-systems/
- https://www.programiz.com/c-programming/c-dynamic-memory-allocation