Hacettepe University

Computer Engineering

BBM 203 – HOMEWORK 2
REPORT

Name: Zekeriya Onur Yakışkan
Student id: 21527539

Problem

Implementing a basic client-server simulation. In this simulation, there are two types of machine, clients and a server. Each client and server can receive processes and interrupts. Clients and server can hold a defined number of process and and interrupts which can be different for each machine. Client can send its process or interrupt to the server. In that case, server would add a process to itself regardless of what it receives(process or interrupt). Server cant send anything(obviously) bot it can operate. In operate and send operations, interrupts has always a priority. There is also operations for adding a process to a client processes and adding an interrupt to machines interrupts. All operations and size of the processes and interrupts are taken from two txt files(one for operations, one for initializing process and interrupt size).

About Operations

Add Process

In txt file format will be given as ’A’ ’Client Number’ ’Process Character Name’. This will add a new process, if there is enough room for a process in that specific machine, to the given client. If there is no room this process it will write ‘1’ to machines history.

Add Interrupt

In txt file format will be given as ’’I’ ’Client/Server Number’ ’Interrupt Character Name’. This will add a new interrupt, if there is enough room for a interrupt in that specific machine, to the given machine. If there is no room this interrupt it will write ‘2’ to machines history.

Send Command

In txt file format will be given as ’S’ ’Client Number’ ’G’. G can be ignored in this command. If there is an interrupt, the last added interrupt will be sended to the servers processes and then deleted from clients.. If there is a process, first process will be sended to the servers processes and then deleted from clients. It will also write the sended interrupt or process name to clients history. If there is no room in servers processes, it will write ‘1’ to servers history. If there is nothing to send, it will write ‘3’ to clients history.

Operate Command

In txt file format will be given as ’O’ ’G’ ’G’. Two G’s can be ignored in this command. It will operate the last added interrupt and delete it if any. If there is no interrupt, it will operate the first process in process, then delete it. It will also write the operated interrupt or process name to servers history. If there is nothing to send, it

will write ‘3’ to clients history.

Main Data Structures

Machines

There are two things that each machine have, interrupts and processes. I hold each machines interrupts in a stack structure and processes in a circular queue structure. There is also a machine structure that has two attributes, a queue for processes and a stack for interrupts. There is also a stack for keeping clients history.

List of Machines

Since I have multiple machines, I hold them in a list which starts from first machine and ends with the server.

Input

There will be two inputs in txt format. Its names will be taken as arguments.

First Input

Its name will be given as first argument. Its first line is a number which represent how many machines there will be in the simulation. After the first line, each line has two numbers which is seperated with a whitespace character. Digits represents process and interrupts size respectively of first clients and it goes on like this to the last client. Last line is the servers processes and clients in the same format.

Second Input

Its name will be given as first argument. First line contains a number which represents number of commands given. After that, each lines represents a command.

Output

Output is a txt file and its name will be given as the third argument. Its each line will be the history of the clients(first line first client, second line second client...). Last line will be servers history.

Solution

First I create a list of machines with respect to first input. While creating machines, I initial the history size of each machine to command line number which is given in the first line of the second input. Then I do the tasks that has explained before with respect to second input file. Finally, I read from each machines history stack and write it to the output file which name has given as third argument.

Algorithm

```
typedef{
	stack interrupts
	stack history
	queue processes
}machine

main(){

	numCommandLine = getint(input2)
	numMachineLine = getint(input1)

	machine machines[numMachineLine]
	for(i = 0; i < numMachineLine; i++)
		one = getint(input1)
		two = getint(input1)
		createMachine(machines[i],one,two)
		machines[i].histrory.size = numCommandLine

	for(i = 0; i < numCommandLine; i++)
		one = getchar(input2)
		two = getint(input2)
		three = getchar(input2)
		if(one == 'A')
			addProcess(machines + two - 1, three)
		else if(one == 'I')
			addInterrupt(machines + two - 1, three)
		else if(one == 'S')
			sendToServer(machines + two  -1, machines + numMachineLine -
1)
		else if(one == 'O')
			operate(machines + numMachineLine - 1)

	for(machine in machines)
		write(output, machine.history.list)

	return 0
}
```