CS33211 - Operating Systems - Assignment2 - Documentation

Instructor: Dr. Qiang Guan
Author: Maxim Mamotlivi
Date: 4/25/2024

Assignment Description

Topic: Bankers Algorithm
Considering a system with five processes P0 through P4 and three resources of type A, B, C. Resource type A has 10 instances, B has 5 instances and type C has 7 instances. Suppose at time t0 following snapshot of the system has been taken:

Process	Allocation	Max	Available
	A B C	A B C	A B C
P0	0 1 0	7 5 3	3 3 2
P1	2 0 0	3 2 2	
P2	3 0 2	9 0 2	
P3	2 1 1	2 2 2	
P4	0 0 2	4 3 3	

Implement the Banker’s algorithm to answer the following question: Is the system in a safe state? If Yes, then what is the safe sequence?

Implementation

This implementation of Dijkstra’s “Banker’s Algorithm” utilizes a text file storing data from the allocation, max, and available tables to populate their respective 2D and 1D arrays. Arrays for “need” and “sequence” are also created according to the execution of Dijkstra’s “Banker’s Algorithm.”
Allocation and state is then evaluated through a series of rounds representing each process row and recheck operation via nested for-loops where the 2D “need” array is compared against the “avail” array.
In cases where there are enough available resources, the corresponding “alloc” array is added to the “avail” array, and the process in question is marked by advancement under its corresponding index in the “seq” array. In cases where there are not enough resources, addition is prohibited; the process is flagged via negative numbers reflecting the current round.
Upon the completion of round 1, the algorithm proceeds similarly, except it only considers those processes marked negative in the previous round. If resources are found to be insufficient again, the doom flag is raised. Unless a deficient process is later satisfied, the doom flag remains and the system is marked as unsafe. Otherwise, the sequential order of advancement is reported for each process.

Compilation & Execution Commands

As specified by the assignment, the following commands are used to compile and execute “banker.cpp”:

```
$ g++ -o banker banker.cpp
$ ./banker
```

Libraries Used

The following C++ libraries were used for standard operations and for reading from the .txt file:

```
#include <stdlib.h>   //standard lib
#include <iostream>   //in-out stream
#include <fstream>    //file stream
```

File List

Project files:

```
banker.cpp
tables.txt
```

Classes

N/A, everything runs in “int main(){}”

Dependencies

Aforementioned libraries; <fstream> is particularly important.

Notable Functionality

It should be noted that multiple sections used for debugging have been commented out, uncommented or pasted in other sections of code to obtain information regarding the state of each array.

The majority of calculations occur in round-based availability assessments which allows the algorithm to work on datasets of varying size or complexity. Additionally, using the “seq” array to report round-deficiency *and* advancement sequence saves memory.

See code after the following line:
“//calculate availability states, recheck availability”

Additional Resources

The list below includes documentation links for notable libraries as well as certain functions associated with shared memory and semaphores:

Library Documentation
https://en.cppreference.com/w/cpp/io/basic_fstream
https://en.cppreference.com/w/cpp/header/iostream

Method Documentation
https://www.javatpoint.com/bankers-algorithm-in-cpp