BLG312E - Computer Operating Systems - HW3

Enis Teper - 504221511

1 Introduction

In this homework, it is required from us to implement deadlock detection with bankers algorithm. Basically the algorithm checks for the requests of the process and if all requests are allocatable executes that process. Then, the allocations of that process are also loaded back to available resources. If the process requests cannot be satisfied and no more processable process left then the algorithm defines those processes are deadlock. The reason is because of they have already allocated some resources and require more to finish their jobs and until then they keep allocating.

2 Implementation and Pseudocode

Makefile has two main commands, **make output** is used to compile the code and **make clean** to remove executable files. Then results are generated by running the executable file.

```
[teper17@ssh hw3]$ make clean
rm bankers_algorithm
[teper17@ssh hw3]$ make output
gcc bankers_algorithm.c -o bankers_algorithm
[teper17@ssh hw3]$ ./bankers_algorithm

Information for process: P1:
Allocated resources: R1:3 R2:0 R3:1 R4:1 R5:0
Resource request : R1:0 R2:1 R3:7 R4:0 R5:1
Availale resources : R1:0 R2:2 R3:5 R4:1 R5:6

Information for process: P2:
Allocated resources: R1:1 R2:1 R3:0 R4:0 R5:0
Resource request : R1:0 R2:0 R3:1 R4:0 R5:3
Availale resources : R1:0 R2:2 R3:5 R4:1 R5:6

Information for process: P3:
Allocated resources: R1:0 R2:3 R3:0 R4:0 R5:0
Resource request : R1:2 R2:2 R3:0 R4:0 R5:1
Availale resources : R1:0 R2:2 R3:5 R4:1 R5:6

Information for process: P4:
Allocated resources: R1:1 R2:0 R3:0 R4:0 R5:0
Resource request : R1:1 R2:0 R3:1 R4:0 R5:2
Availale resources : R1:0 R2:2 R3:5 R4:1 R5:6

Information for process: P5:
Allocated resources: R1:0 R2:1 R3:4 R4:0 R5:0
Resource request : R1:3 R2:1 R3:0 R4:1 R5:1
Availale resources : R1:0 R2:2 R3:5 R4:1 R5:6

Running order for processes: P2 P4 P3

There is a deadlock: P1 P5 are the cause of deadlock.
[teper17@ssh hw3]$
```

Figure 1: Results

Algorithm 1 Banker's Algorithm

procedure BANKERS_ALGORITHM($n, m, resource\_requests, resource\_allocations, base\_resources$)
    Allocate memory for flags and available_resources arrays
    for $i = 0 \rightarrow n$ **do**
        Initialize flags as false
        Initialize available_resources with as 0
    end for
    for $i = 0 \rightarrow m$ **do**
        Compute available resources as base resources - total current allocation
    end for
    for each process $i = 0 \rightarrow n$ **do**
        Print process information, allocated resources, requested resources, and available resources
    end for
    Initialize variables for tracking execution status and deadlock detection
    while all processes are not done **do**
        if current process is not done **then**
            Check if all requested resources are lower than or equal to available resources
            if so **then**
                Update running order, available resources, flags and reset is_all_done_cnt
            else
                Increment is_all_done_cnt
            end if
        else
            Increment is_all_done_cnt
        end if
        Increase process index and reset to 0 if it reaches total process count
    end while
    for each process $i = 0 \rightarrow n$ **do**
        if the process is not done **then**
            Update deadlocks array
        end if
    end for
    if there is a running order **then**
        Print running order of processes
    end if
    if there are deadlocks **then**
        Print processes causing deadlocks
    end if
    Free all allocated memories
end procedure