BLG312E Homework 3 Report

Ahmet Secaettin Alıcı - 150190097,

31.05.2023

1 Introduction

We have fixed amount of resources and fixed number of processes. In the beginning processes declares how much and what kind of resources they will need. We have 3 table, one shows all resources, one shows already allocated resources and last one shows requested/needed resources. Via Banker's Algorithm resources will be allocated optimally, but after allocation if there is a still unfinished processes, that means these processes are the cause of deadlock. Let's see implementation of this program.

2 The Program

First we have auxiliary function named read-files(), it takes pointers to files and arrays. It reads and fills the arrays. We have 3 main arrays: req for requests, alloc for allocations, res for all resources.

Second, we have work array for the algorithm, finish array to follow which processes is finished or not.

Third we have results array to report finished processes, deadlocks array to report criminal processes that cause deadlock. We initialize these two arrays' slots as -1 so that we can understand unused slots. We have results-num and deadlocks-num int variables to follow how many of their slots are used.

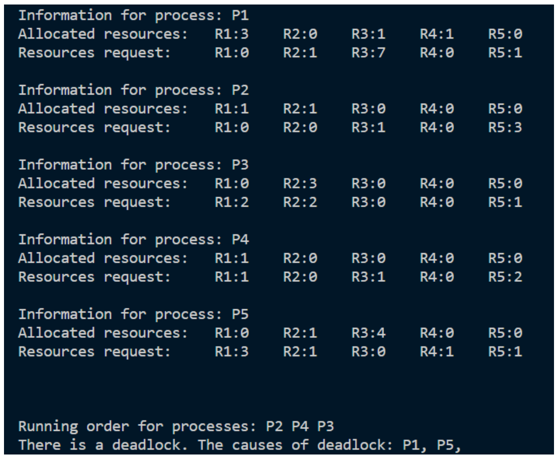
After reading files and filling arrays, we output the explanatory information about processes and resources to the terminal.

We fill work array like this: work=resources-allocations or (work=available).

flag1 variable is initialized as 1. After we enter loop it become 0 and the only way for it to become 1 again is that after looping and checking all processes and after that at least one process should be finished and added to the results array. If we find a unfinished process, flag2 checks for requests less than or equal work, flag2 means it is true. If flag2==1 that means that process can proceed and finish. After finish, it release its allocated resources and we add this resources to work array, flag1 becomes 1, and we add this process to results array.

Eventually flag1 becomes 0 but important part is why did it become 0? Like we have said, flag1 becomes 0 if we don't add any processes to results array after one scanning of all processes. There could be two reason for that, there may not be any unfinished process left and all processes are already added to results array or some unfinished processes demand more resources than we already have as available. We check for unfinished processes and add them to array named deadlocks, if there is any. If there is deadlock we set flag1 to 0, thus flag1 now can tell us if there is deadlock or not.

Lastly, we output the results array, and if there is deadlock we output deadlocks array too accordingly.



```
Information for process: P1
Allocated resources:   R1:3    R2:0    R3:1    R4:1    R5:0
Resources request:     R1:0    R2:1    R3:7    R4:0    R5:1

Information for process: P2
Allocated resources:   R1:1    R2:1    R3:0    R4:0    R5:0
Resources request:     R1:0    R2:0    R3:1    R4:0    R5:3

Information for process: P3
Allocated resources:   R1:0    R2:3    R3:0    R4:0    R5:0
Resources request:     R1:2    R2:2    R3:0    R4:0    R5:1

Information for process: P4
Allocated resources:   R1:1    R2:0    R3:0    R4:0    R5:0
Resources request:     R1:1    R2:0    R3:1    R4:0    R5:2

Information for process: P5
Allocated resources:   R1:0    R2:1    R3:4    R4:0    R5:0
Resources request:     R1:3    R2:1    R3:0    R4:1    R5:1



Running order for processes: P2 P4 P3
There is a deadlock. The causes of deadlock: P1, P5,
```

Figure 1: Outputs of The Program