Date: 2022-11-24

Group MHA

Murtadha Alobaidi , mhaao@kth.se

Hasan Alzubeidi , haz@kth.se

Ahmed Alkhalaf, ahmedalk@kth.se

1) What is the meaning of the term busy waiting? Can busy waiting be avoided altogether? Explain your answer.

**Answer:**It means that the process will check if a condition is valid multiple times. For instance, it can check if the key “enter” is clicked. The process will check that multiple times. Or just the process goes into a loop without doing anything just to make delay when a second process waits for some data to be resolved before resolving new data. For instance, if we have p1 and p2, which require c1 to happen before c2.

Graphical user interface, text, application

Description automatically generated

Lecture 5 Semphore Usage (2/2) slid 33

Semaphores can be implemented without Busy waiting by invoking block and Wakeup. These two functions will be the controller of the waiting queue and the ready queue of processes. There the block will be invoked when the process is pleased in the waiting queue, and the wakeup will be useful to remove the process from the waiting queue and place it in the ready queue.

2) Show that, if the wait() and signal() semaphore operations are not executed atomically, then mutual exclusion may be violate.

**Answer:**The idea behind the wait and signal is to share the same value and increment och decrement this value. The value is the number of available resources, and when the wait is invoked, this value should be incremented, and the opposite for the signal. So, if two operations are not executed automatically and are executed on a semaphore when its value is 1. It may both operations decrement the semaphore and so during that violating mutual exclusion.

3) Consider the following snapshot of a system, and answer the following questions using the banker’s algorithm:

Table

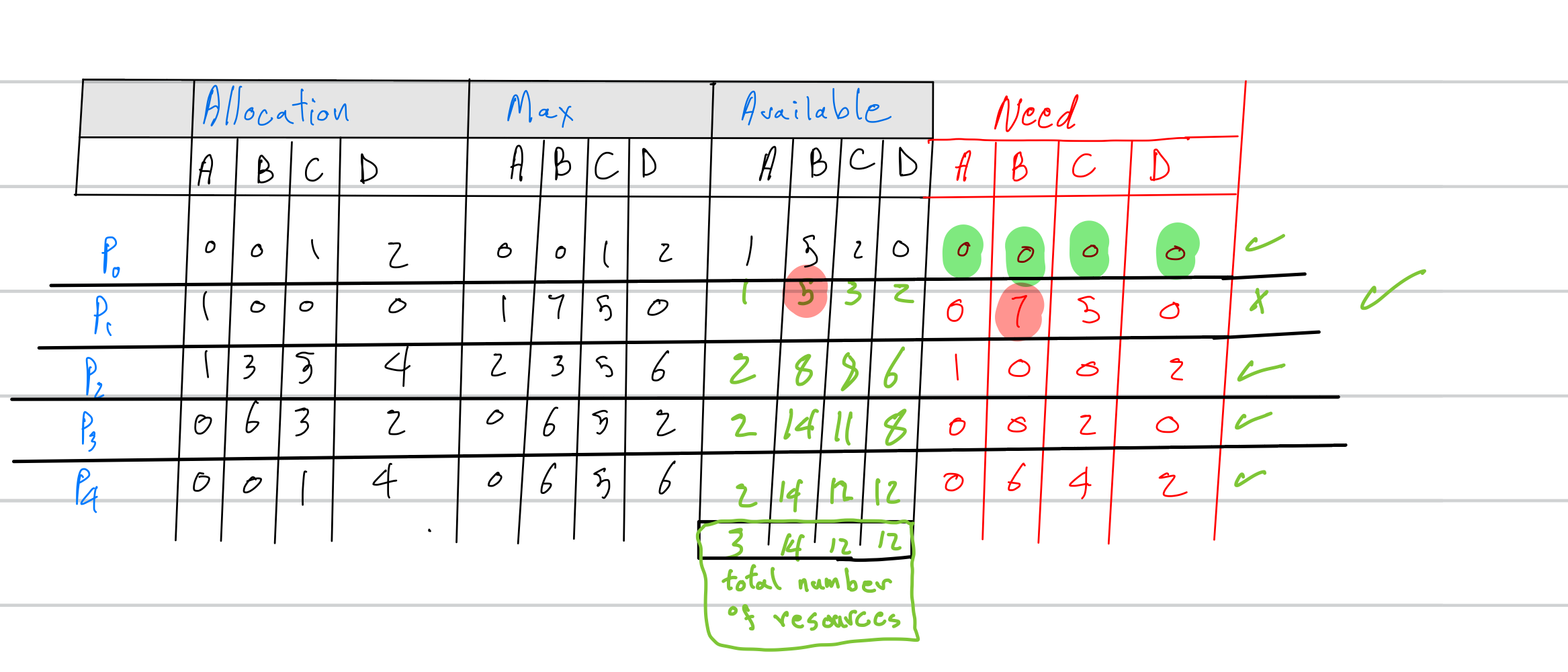
Description automatically generated with medium confidence

1. What is the content of the matrix Need?
2. Is the system in a safe state?
3. If a request from thread T1 arrives for (0, 4, 2, 0), can the request be granted immediately?

**Answer:   
a.** The content of the matrix **Need** .

P0 (0,0,0,0)   
P1 (0,7,5,0)  
P2 (1,0,0,2)  
P3 (0,0,2,0)  
P4 (0,6,4,2)

**b.** The system in a **safe state**:



We have the **total available resources: 3 14 12 12**The **safe sequence we have** : **p0, p2, p3,p4 and p1**

**The system is in the safe state,** we can see after executing all the processes the recourse available is equal to the **total available resources.**

**The answer is YES** the system in a safe state.

**c.** The request can be granted immediately.   
The value of Available: (1,1, 0,0)

The processes ordering: p0, p2, p3,p1 and p4

4) Consider the following resource-allocation policy. Requests for and releases of resources are allowed at any time. If a request for resources cannot be satised because the resources are not available, then we check any threads that are blocked waiting for resources. If a blocked thread has the desired resources, then these resources are taken away from it and are given to the requesting thread. The vector of resources for which the blocked thread is waiting is increased to include the resources that were taken away.

For example, a system has three resource types, and the vector Available is initialized to (4, 2, 2). If thread T0 asks for (2, 2, 1), it gets them. If T1 asks for (1, 0, 1), it gets them. Then, if T0 asks for (0, 0, 1), it is blocked (resource not available). If T2 now asks for (2, 0, 0), it gets the available one (1, 0, 0), as well as one that was allocated to T0 (since T0 is blocked). T0 Allocation vector goes down to (1, 2, 1), and its Need vector goes up to (1, 0, 1).

1. Can deadlock occur? If you answer “yes”, give an example. If you answer “no”, specify which necessary condition cannot occur.
2. Can deadlock occur? If you answer "yes", give an example. If you answer "no", specify which necessary condition cannot occur.

**Answer:**

1. No. The existence of the preemption makes the deadlock not happen.
2. There will be an indefinite deadlock when there are a series of requests which makes not all resources required.

**SOURCES:**

(ABRAHAM SILBERSCHATZ, 2018)