Operating Systems - Review Questions 2

Deadline: Oct. 28, 2022

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

   Answer. Busy waiting means that a process is waiting for a condition to be satised in a tight loop without relinquishing the processor. One strategy to avoid busy waiting temporarily puts the waiting process to sleep and awakens it when the appropriate program state is reached, but this solution incurs the overhead.

---

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

   Answer. A `wait()` operation atomically decrements the value associated with a semaphore. If two `wait()` operations are executed on a semaphore when its value is 1 and the operations are not performed atomically, then both operations might decrement the semaphore value, thereby violating mutual exclusion.

---

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

	Allocation	*Max*	*Available*
	A B C D	*A B C D*	*A B C D*
T_0	0 0 1 2	0 0 1 2	1 5 2 0
T_1	1 0 0 0	1 7 5 0	
T_2	1 3 5 4	2 3 5 6	
T_3	0 6 3 2	0 6 5 2	
T_4	0 0 1 4	0 6 5 6	

   a. What is the content of the matrix `Need`?

   b. Is the system in a safe state?

   c. If a request from thread T_1 arrives for (0, 4, 2, 0), can the request be granted immediately?

   Answer.

   a. The values of `Need` for processes P_0 through P_4, respectively, are (0, 0, 0, 0), (0, 7, 5, 0), (1, 0, 0, 2), (0, 0, 2, 0), and (0, 6, 4, 2).

   b. The system is in a safe state. With `Available` equal to (1, 5, 2, 0), either process P_0 or P_3 could run. Once process P_3 runs, it releases its resources, which allows all other existing processes to run.

   c. The request can be granted immediately. The value of `Available is` then (1, 1, 0, 0).

---

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 T_0 asks for (2, 2, 1), it gets them. If T_1 asks for (1, 0, 1), it gets them. Then, if T_0 asks for (0, 0, 1), it is blocked (resource not available). If T_2 now asks for (2, 0, 0), it gets the available one (1, 0, 0), as well as one that was allocated to T_0 (since T_0 is blocked). T_0's `Allocation` vector goes down to (1, 2, 1), and its `Need` vector goes up to (1, 0, 1).

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

   b. Can indefinite blocking occur? Explain your answer.

   Answer.

   a. Deadlock cannot occur, because preemption exists.

   b. Yes. A process may never acquire all the resources it needs if they are continuously preempted by a series of requests such as those of process C.