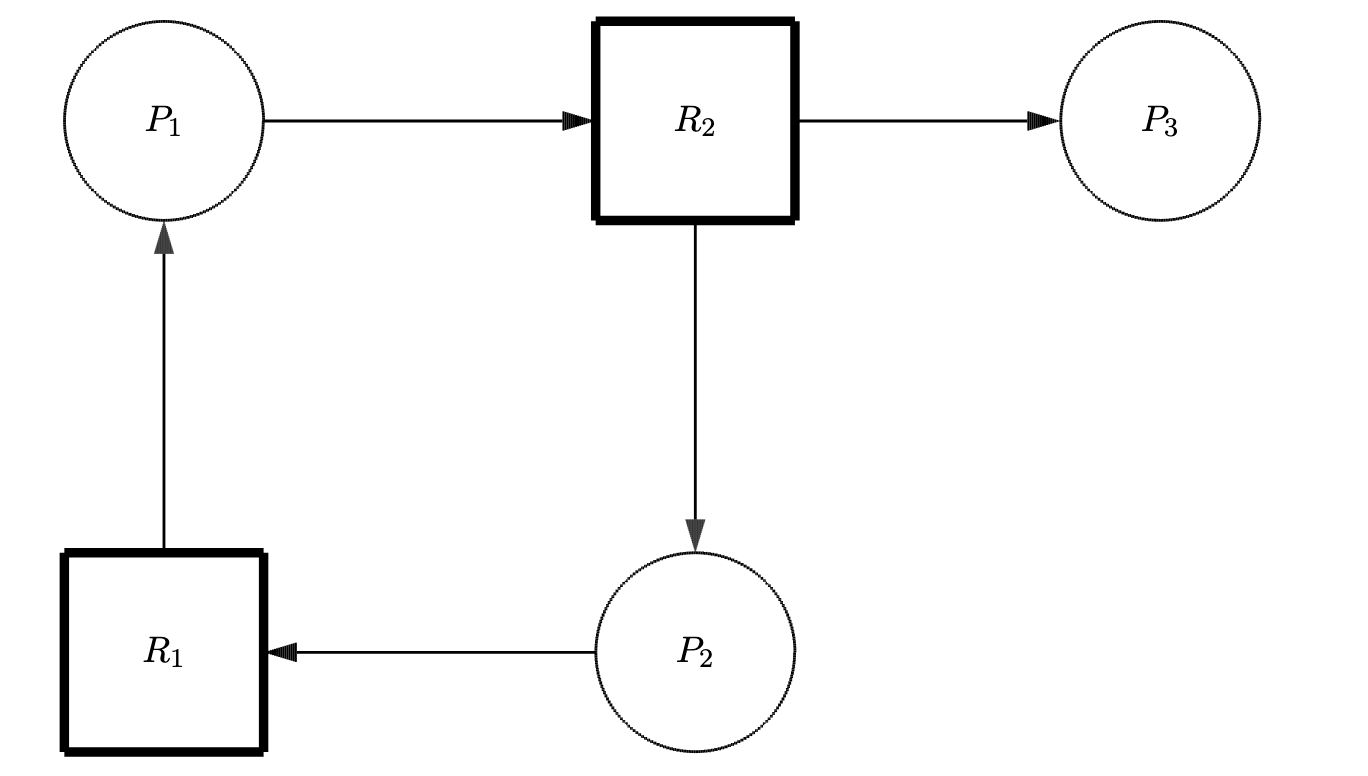
George Suarez

CSE 461

February 20, 2019

Homework 2

1. The following figure shows a resource graph for a system with consumable resources only. A resource is represented by a rectangle with thick lines and labeled as . A process is represented by a circle, labeled .
2. Is the graph a claim-limited graph? Why?
3. Is the graph reducible? Why?

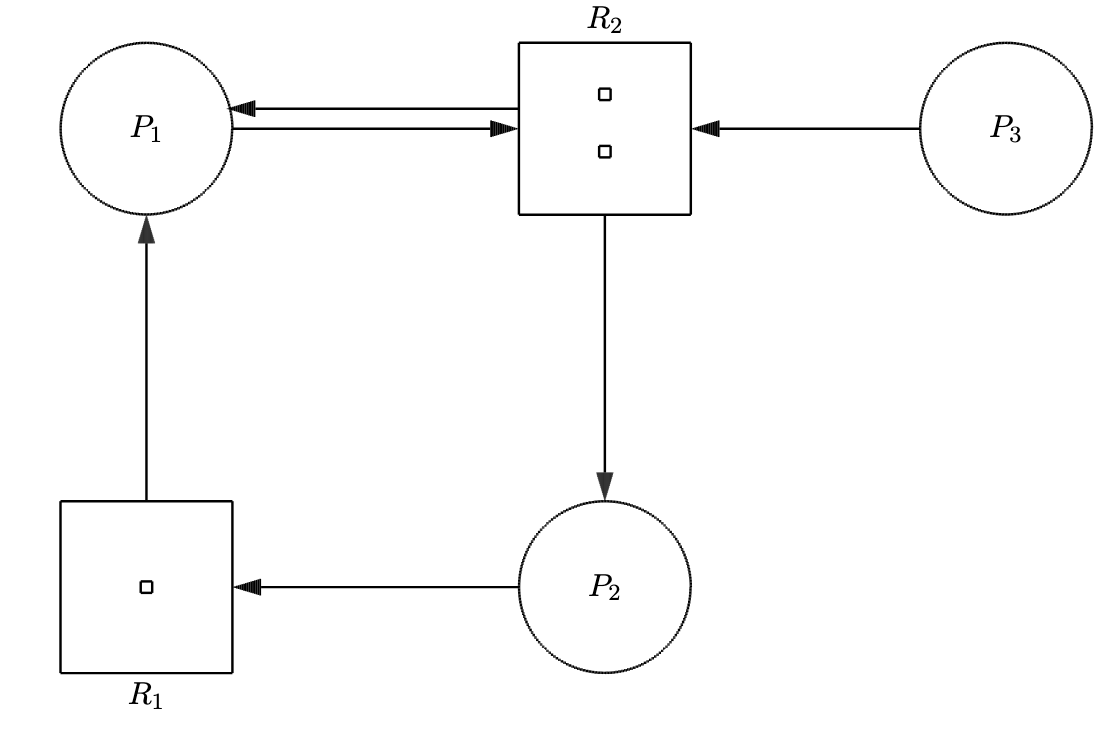


Solution:

1. The graph is a claim-limited graph because it represents a consumable resource system since each resource has 0 available units and each node has a request edge ( which is a consumer of .
2. The graph is reducible since is a producer of and it is not blocked which means it can produce a unit for that reduces the edge and all other edges will be reduced as a result.
3. Assume a system hasprocesses and identical units of a reusable resource. If each process can claim at most units of the resource, determine whether each of the following is true or false and prove your claim:
4. If the system is deadlock free, then .
5. If then the system is deadlock free.

Solution:

1. **False** because if the system is already deadlock free, then the process does not need an additional resource.
2. **True** because if each process claims the same amount of identical units, then we are still left with 1 more available resource to use. Thus, the system is deadlock free if .
3. The following figure shows a resource graph for a system with reusable resources only. A resource is represented by a rectangle, in which a small square indicates a unit of the resource.
4. Is the graph expedient? Why?
5. Is there any knot in the graph? Why?
6. Is there any deadlock in the system? Why?



Solution:

1. The graph is expedient because processes have outstanding requests that are blocked.
2. The graph does have a knot because the subgraph G = are able to reach other, but process cannot be reached by any of the nodes the subgraph G.
3. The system is in a deadlock state because we have a knot in the graph which is a sufficient condition for a deadlock.
4. In this problem you are to compare reading a file using a single-threaded file server and a multithreaded server. It takes 15 milliseconds to get a request for work, dispatch it, and do the rest of the necessary processing, assuming that the data needed are in a cache in main memory. If a disk operation is needed, as is the case one-third of the time, an additional 75 milliseconds is required, during which time the thread sleeps. How many requests/sec can the server handle if it is single threaded? If it is multithreaded?

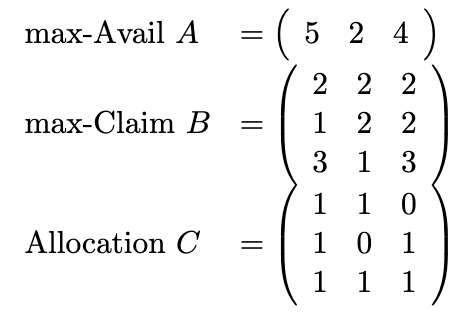
Solution:

For the single-threaded case, the total time it takes when reading a file is:

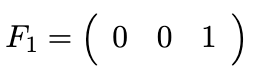
Which the single-threaded file server can handle:

For the multi-threaded case, the time we wait for the disk operation overlaps, so it only takes 15 milliseconds. Therefore, the multi-threaded file server takes:

1. Consider the state of a system with processes , defined by the following matrices:



1. Find the available matrix and the need matrix in this state.
2. Suppose now process makes a request with



If the request were granted, what would be and in the resulted state?

1. To ensure the system be safe, should the request be granted? Why? Give your reasons in detail.

Solution:

1. Available Matrix Max-Avail Matrix Allocation Matrix

Need Matrix = Max-Claim Matrix Allocation Matrix

1. Using the safe-state check algorithm:

: *finished*

Since the processes and are not finished because and is not . This means that the system is not safe, and the request is blocked. Thus, the system needs to be reset by doing the following operations:

Which now we get:

And now process is now unfinished.