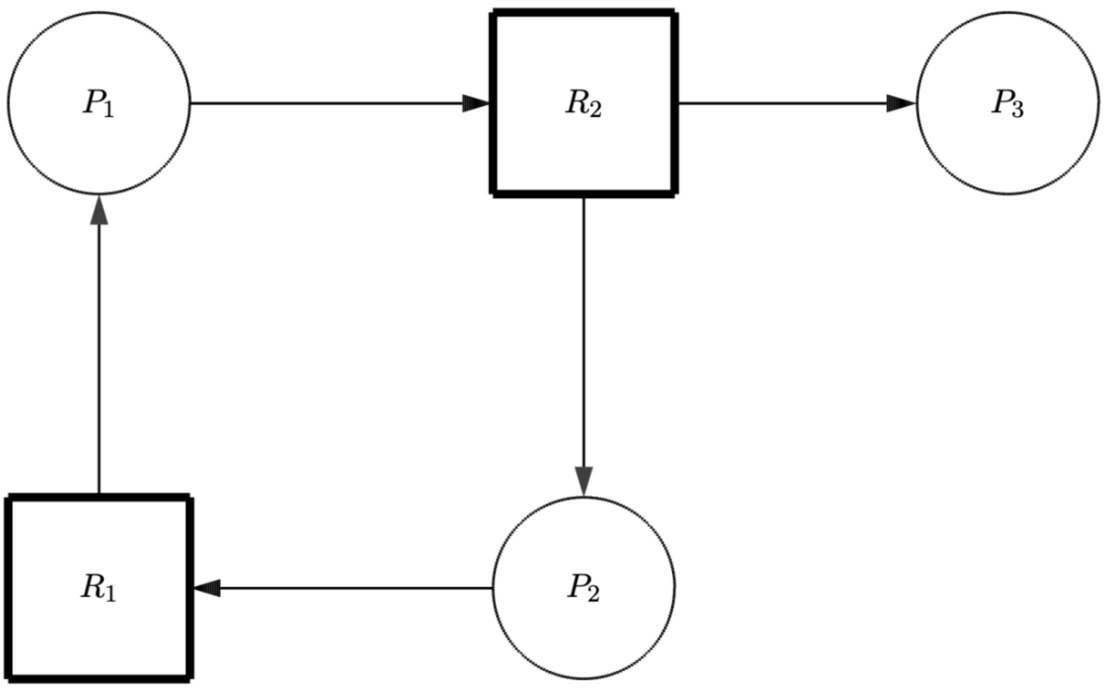
George Suarez
CSE 461
February 20, 2019

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 R_i. A process is represented by a circle, labeled P_i.

   (a) Is the graph a claim-limited graph? Why?
   (b) Is the graph reducible? Why?



Solution:

(a) 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 (P_i, R_j) which P_i is a consumer of R_j.
(b) The graph is reducible since P_3 is a producer of P_2 and it is not blocked which means it can produce a unit for R_2 that reduces the edge (P_1, R_2) and all other edges will be reduced as a result.

2. Assume a system has P processes and R identical units of a reusable resource. If each process can claim at most N units of the resource, determine whether each of the following is true or false and prove your claim:

   (a) If the system is deadlock free, then $R \geq P(N-1)+1$.
   (b) If $R \geq P(N-1)+1$ then the system is deadlock free.

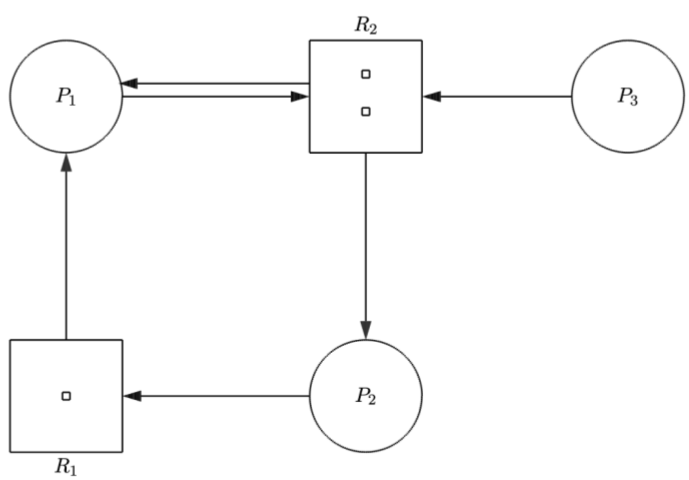
   Solution:

   (a) **False** because if the system is already deadlock free, then the process does not need an additional resource.

   (b) **True** because if each process claims the same amount of R identical units, then we are still left with 1 more available resource to use. Thus, the system is deadlock free if $R \geq P(N-1)+1$.

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.

   (a) Is the graph expedient? Why?
   (b) Is there any knot in the graph? Why?
   (c) Is there any deadlock in the system? Why?



   Solution:

   (a) The graph is expedient because processes P_1, P_2, P_3 have outstanding requests that are blocked.

(b) The graph does have a knot because the subgraph G = $\{P_1, R_2, P_2, R_1\}$ are able to reach other, but process P_3 cannot be reached by any of the nodes the subgraph G.
(c) 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:

$$\frac{2}{3} \times\ 15\, msecs + \frac{1}{3} \times 90\, msecs$$

$$= 10\, msecs + 30\, msecs$$

$$= 40\, msecs$$

   Which the single-threaded file server can handle:

$$\frac{1\, request}{40\, msecs} \times \frac{1000\, msecs}{1\, sec} = 25 \frac{requests}{sec}$$

   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:

$$\frac{1\, request}{15\, msecs} \times \frac{1000\, msecs}{1\, sec} = 66.67 \frac{requests}{sec}$$

5. Consider the state of a system with processes P_1, P_2, P_3, defined by the following matrices:

$$\begin{aligned}
\text{max-Avail } A &= \begin{pmatrix} 5 & 2 & 4 \end{pmatrix} \\
\text{max-Claim } B &= \begin{pmatrix} 2 & 2 & 2 \\ 1 & 2 & 2 \\ 3 & 1 & 3 \end{pmatrix} \\
\text{Allocation } C &= \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}
\end{aligned}$$

(a) Find the available matrix D and the need matrix E in this state.

(b) Suppose now process P_1 makes a request with

$$F_1 = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}$$

If the request were granted, what would be $D, C,$ and E in the resulted state?

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

Solution:

(a) Available Matrix D = Max-Avail Matrix A – Allocation Matrix C

$$D = (5 \quad 2 \quad 4) - (3 \quad 2 \quad 2)$$

$$D = (2 \quad 0 \quad 2)$$

Need Matrix E = Max-Claim Matrix B – Allocation Matrix C

$$E = \begin{matrix} 1 & 1 & 2 \\ 0 & 2 & 1 \\ 2 & 0 & 2 \end{matrix}$$

(b) $D = D - F_1 = (2 \quad 0 \quad 2) - (0 \quad 0 \quad 1) = (2 \quad 0 \quad 1)$

$C_i = C_i + F_i$

$$C = \begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 2 \\ 1 & 1 & 2 \end{matrix}$$

$E_i = E_i - F_i$

$$E = \begin{matrix} 1 & 1 & 1 \\ 0 & 2 & 0 \\ 2 & 0 & 1 \end{matrix}$$

(c) Using the safe-state check algorithm:

$P_3 : D = (2 \quad 0 \quad 1) + (1 \quad 1 \quad 2) = (3 \quad 1 \quad 3)$ *finished*

Since the processes P_1 and P_2 are not finished because E_1 and E_2 is not $\leq D$. 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:

$$D = D + F_1$$
$$C_i = C_i - F_i$$
$$E_i = E_i + F_i$$

Which now we get:

$$D = (2 \quad 0 \quad 2)$$

$$C = \begin{matrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{matrix}$$

$$E = \begin{matrix} 1 & 1 & 2 \\ 0 & 2 & 1 \\ 2 & 0 & 2 \end{matrix}$$

And now process P_3 is now unfinished.