

2/13/2019

Homework 2 - CSE 461 - Dr Tong, Yu Housat Farior

SID # 26

#1 (10 pts) - The following figure shows a resource graph for a system with consumable resources only. A resource is represented by a rectangle w/ 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?

→ Each resource has no available unit.

P_1 is a consumer of R_1 , P_2 and P_3 are consumers of R_2 . \therefore making this graph claim-limited.

(b) Is the graph reducible? why?

→ This graph is reducible.

b/c P_1 is a producer of R_2 and b/c it is

unblocked, P_1 can produce 2 units, that P_2 and P_3 can consume. Since P_1 needs only 1 unit of R_1 , this can be produced by P_2 .

\therefore All process requests can be granted, and making it reducible.

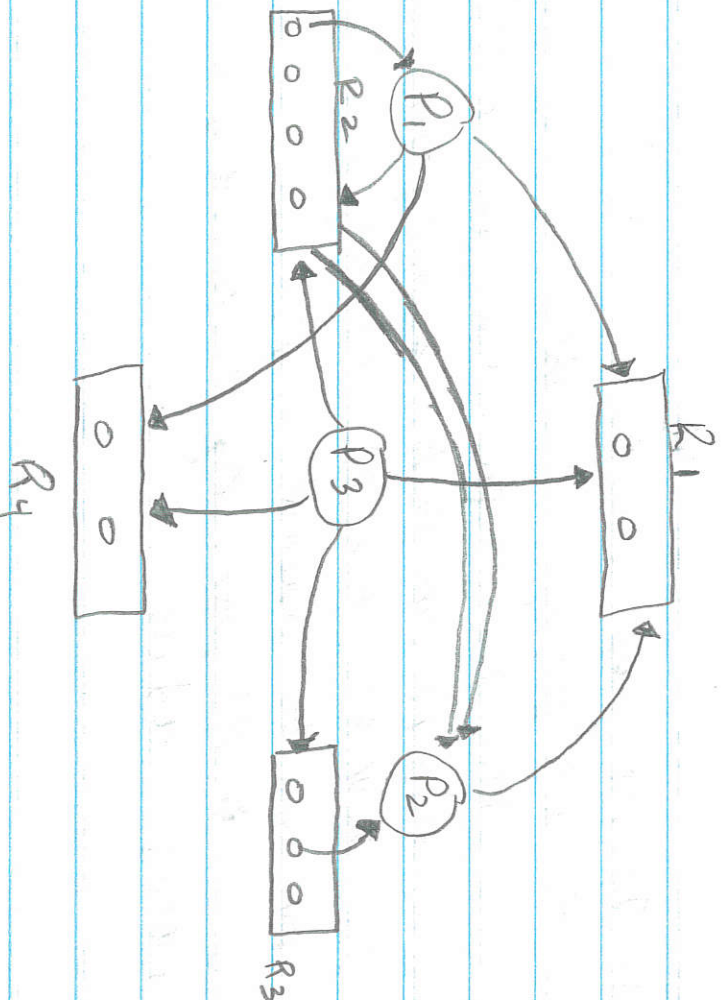
10 pts

Q2 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 a deadlock free then $R \geq P(N-1) + 1$.

→ Since each process can hold $N-1$ units,

we can assume:



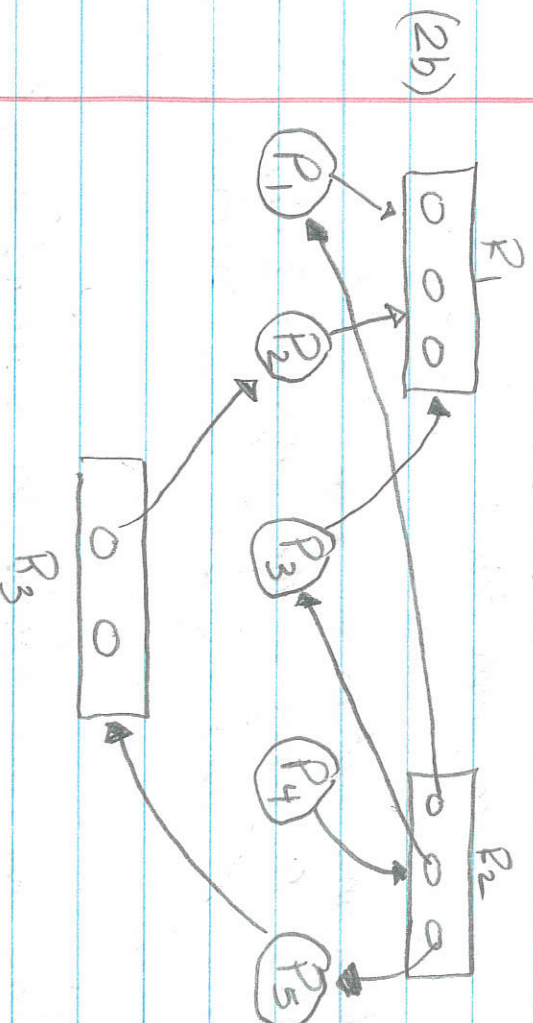
2a) Assume that there are 3 processes P_1, P_2, P_3 and $N=4$

Based on the figure, # of resources are 22

$$\hookrightarrow P(N-1) = 3(4-1) + 1 = 10$$

which makes the graph a deadlock when

$$R \geq P(N-1) + 1$$



• R_1 is allocated to P_1, P_2, P_3

$$R = 8$$

$$P = 5 \quad 8 \geq 5(2-1) + 1$$

$$N = 2$$

- If R_1 is allocated to P_1, P_2, P_3 . P_1 can finish working & release to R_2 . R_2 can be allocated to P_4 . P_5 can finish its work as 2 resources are allocated. P_5 finishes, and releases R_1 & R_2 . P_1 and P_3 are assigned. When it finishes the work, P_4 can then continue.

#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?

→ Yes, All processes have requests that are blocked, which makes the graph expedient

(B) Is there any knot in the graph? why?

→ Yes, All nodes in the subgraph $\{P_1, P_2, P_3, P_4\}$ are reached from every other node. This makes it a knot.

(C) Is there any deadlock in the system? why?

→ Yes, Since there is a knot present, $\{P_1, P_2, P_3, P_4\}$ this is enough for a deadlock.

(4) In this problem you are to compare reading a file using a single-threaded server and a multi-threaded server. T1 takes 15 mseconds 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 msec is required, during which time the thread sleeps. How many requests/sec can the server handle if it is single threaded? If it is multi-threaded?

→ In a single thread server, 15 mseconds to get a request for work. Does the rest of the "necessary" processing if the data is in the cache in main memory.

→ Takes an additional $\frac{1}{3}$ of the time, an additional 75 mseconds, (90 mseconds if disk operation is needed)

∴ $\frac{1}{3}(90) + \frac{2}{3}(15) = 40$ mseconds, which is the total time required for reading a file using a single thread. which can perform 25 requests/sec.

↳ In multi threaded server, waiting for disk is overlapped.
∴ it takes 15 mseconds for reading a file. Hence, it can perform 1000/15 requests per second
= 66 (2/3) requests/second

(5) Consider the state of the system with processes P_1, P_2 and P_3 , defined by the following matrices:

$$\text{max-Avail } A = \begin{pmatrix} 5 & 2 & 4 \end{pmatrix}$$

$$\text{max-Claim } B = \begin{pmatrix} 2 & 2 & 2 \\ 1 & 2 & 2 \\ 3 & 1 & 3 \end{pmatrix}$$

$$\text{Allocation } C = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

(A) Find the available Matrix D and the need Matrix E in this state.

$$\text{Available Matrix } D = A - \sum_{i=1}^n C_i$$

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

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

$$E = B - C$$

$$\begin{pmatrix} 2 & 2 & 2 \\ 1 & 2 & 2 \\ 3 & 1 & 3 \end{pmatrix} - \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 2 \\ 0 & 2 & 1 \\ 2 & 0 & 2 \end{pmatrix}$$

(B) Suppose now P_1 makes a request with $F_1 = (001)$

$$\text{Available } D = D - F_1$$

$$= (202) - (001)$$

$$D = (201)$$

$$\text{Allocation } C = C_1 + F_1$$

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

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

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

$$E = E_1 + F_1$$

$$= \begin{pmatrix} 1 & 1 & 2 \\ 0 & 2 & 1 \\ 2 & 0 & 2 \end{pmatrix} - \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 2 & 1 \\ 2 & 0 & 2 \end{pmatrix}$$

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

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

→ The request is granted when the next state is "safe" state.

- Use safe-state check algorithm

$P_1 (111) \leq (201) \rightarrow \text{false}$

$P_2 (021) \leq (201) \rightarrow \text{false}$

$P_3 (202) \leq (201) \rightarrow \text{False}$

∴ the system is not in safe state as available matrix doesn't have enough resources given by the matrix.