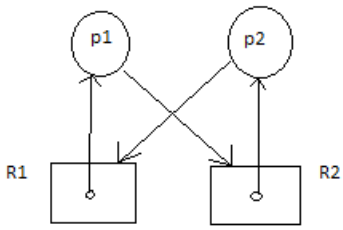
5. Deadlocks



Deadlock betingelser

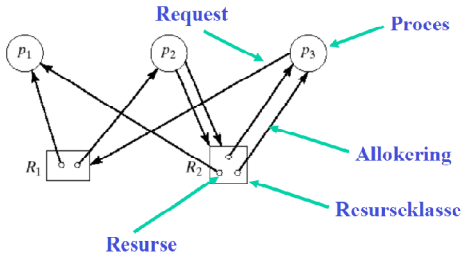
Når to processer står og venter på hinanden, og ikke kan komme ud af tilstanden, uanset hvad der gøres.

a. Mutual exclusion: en resurse kan kun anvendes af en proces
b. Hold and wait: en proces der allerede har en resurse kan bede om en mere
c. No preemption: ikke muligt at tvinge en proces til at frigive en resurse
d. Circular wait: der kan dannes en cycle af processer der venter på hinandens frigivelse af resurser

Deadlock detection

Resursegrafer. Grafreduktion:

1. Vælg proces p der ikke er blokeret
2. Eliminer p og alle tilhørende request og allokeringskanter.
3. Gentag 1 og 2 indtil der ikke er flere ikke-blokerede processer
4. Processer der ikke kan elimineres er i deadlock! Hvis grafen ikke kan elimeres helt er den i deadlock.



- Request kan tilføjes fra proces p når:
  - ingen request findes fra p i forvejen
  - antal ønskede resurser ikke overstiger total
- Acquire kan udføres når:
  - alle requests kan tilfredsstilles samtidig
- Release kan udføres når:
  - ingen request findes fra p

Løsning: Terminer proces (frigør dens resurser) af gangen og tjek for deadlock.

Resurse-preemption: fjerne "problem"-resursen

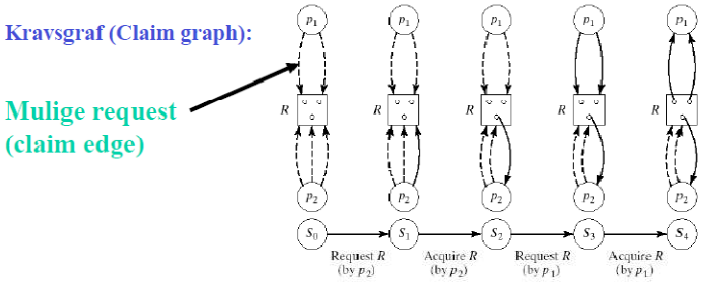
Direkte preemption: Dealloker resursen midlertidigt, giv til anden, og request igen.

Proces rollback: Resursen fjernes, og vi går tilbage til punktet, hvor den endnu ikke var allokeret.

Bryder "No preemption".

Deadlock avoidance

Tjek på runtime. Bruger claim grafer. Forsinke erhvervelsen af resurser, der kan give problemer. Banker's Algorithm (udnytter OS vælger requests der opfyldes):



1. Antag at requests for proces P opfyldes i S og modificér kravsgrafen til at reflektere dette
2. Betragt alle kravs-kanter som request-kanter og reducer kravsgrafen
3. Hvis alle processer kan elimineres, er det sikkert at opfylde den givne request

Deadlock prevention

1. Statisk tilgang. Lav regler der skal overholdes så deadlocks ikke kan opstå.
2. Eliminer en af følgende betingelser for deadlock, så kan deadlocks ikke opstå
   a. Mutual exclusion: read-only, virtualisering
   b. Hold and wait: alle resurser skal allokeres med det samme / alle resurser skal releases før der kan laves nye requests / hvis en requested resurse ikke er tilgængelig, releases alle nuværende resurser
   c. Circular wait: ordning af requests