Operativni sistemi
- Blokada izvrsenja programa -

Veljko Stanković

Uvod

- U viseprogramskom okruzenju, vise programa se nadmecu za pristup konacnom broju resursa racunarskog sistema.
- Desava se da proces ceka na resurs koji je zauzet od strane nekog drugog procesa koji takodje ceka da se oslobodi neki resurs.
- Racunarski sistem se sastoji iz skupa resursa koji su grupisani prema svojim osobinama i slicnosti.
  - Memorijski prostor, CPU ciklusi, I/O uredjaji, fajlovi...

Uvod

- Pod normalnm uslovima, proces moze da pristupa resursima samo po sledecem redosledu:
  - Zahtev
    - ☑ Proces zahteva pristup odredjenom resursu. Proces mora da ceka da se trazeni resurs oslobodi.
  - Upotreba/Obada
    - ☑ Proce koristi odredjeni resurs
  - Oslobadjanje resursa
    - ☑ Nako zavrsene obrade proces oslobadja resurs sistema.
- Za svaki resurs se formiraju redovi cekanja procesa koji su zahtevali pristup.

Neophodni uslovi za blokiranje procesa

- Do blokiranja procesa dolazi ako su sledeca cetiri uslova istovremeno zadovoljena
  - Medjusobna iskljucivost
    - ☑ Samo jedan proces moze da pristupi odredjenom resursu.
    - ☑ Ako neki drugi proes zahteva pristup istom resursu, on se smesta u red cekanja dok se taj resurs ne oslobodi.
  - Pristupi i cekaj
    - ☑ Proces treba da ima pristup najmanje jednom resursu i ceka na druge resurse koji su dodeljeni drugim procesima.
  - Piemtivnost
    - ☑ Resursu se nemoze pristupiti dok ga proces kome je dodeljen ne oslobodi.
  - Cirkularno cekanje
    - ☑ Posmatramo skup procesa pri cemu proces P(i) ceka na resurs koji je dodeljen procesu P(i+1)...

Graf dodele resursa

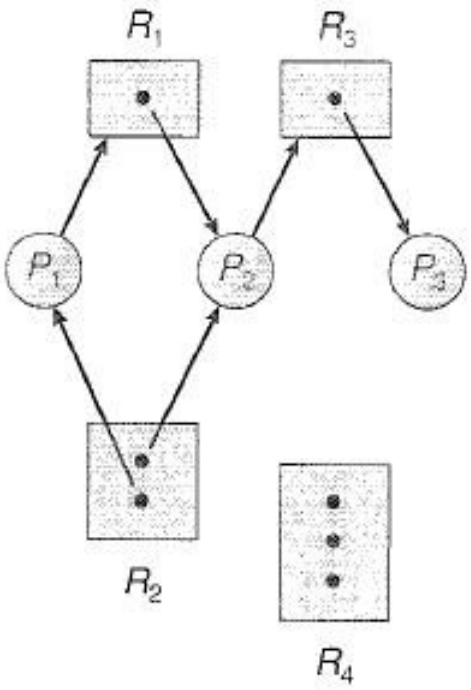
- Proces blokiranja se moze opisati uz pomoc usmerenog grada koji zovemo graf dodele resursa sistema.
  - Cvorovi grafa
    - ☑ Procesi
    - ☑ Resursi
  - Grane grafa
    - ☑ Veze izmedju procesa i resursa
- Usmerena veza od procesa P_i ka resursu R_j se oznacava sa $P_i \rightarrow R_j$ i pokazuje da je proces P_i izdao zahtev da pristupi resursu R_j i ceka da postane dostupan.
- Usmerena veza od resursa R_j ka procesu P_i se oznacava sa $R_j \rightarrow P_i$ i pokazuje da je resurs R_j automatno dodeljen procesu P_i.

Graf dodele resursa

$P = \{P_1, P_2, P_3\}$

$R = \{R_1, R_2, R_3, R_4\}$

$E = \{P_1 \rightarrow R_1, P_2 \rightarrow R_3, R_1 \rightarrow P_2, R_2 \rightarrow P_2, R_2 \rightarrow P_1, R_3 \rightarrow P_3\}$

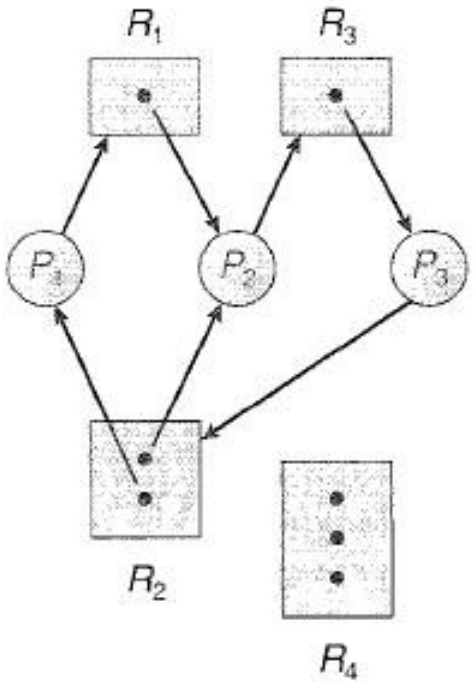


Graf dodele resursa

- Moze se pokazati da do blokade nece doci ako u grafu alokacije resursa ne postoje zatvorne petlje.
  - U slucaju da postoji tacno jedna instanca svakog od resursa unutar petlje, tada je postojanje same potlje dovoljan i potreban uslov da sistem bude blokiran.
    - ☑ Na slici postoje dve petlje:

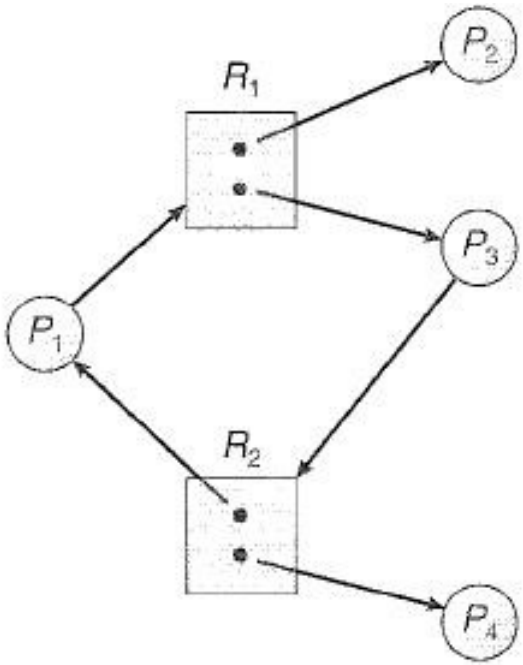
$$P_1 \rightarrow R_1 \rightarrow P_2 \rightarrow R_3 \rightarrow P_3 \rightarrow R_2 \rightarrow P_1$$
$$P_2 \rightarrow R_3 \rightarrow P_3 \rightarrow R_2 \rightarrow P_2$$

- ☑ Proces P1 ceka na proces P2 da oslobodi R1
- ☑ Proces P2 ceka da proces P3 oslobodi R3
- ☑ Proces P3 ceka da ili proces P1 ili proces P2 oslobode R2



Graf dodele resursa

— Na sledecem grafu nece doci do blokade jer iako postoji petlja, proces P4 ce pre ili kasnije oslobidti resurs R2 i proces P3 ce nastaviti sa izvrsavanjem.



Razresavanje konflikata pri dodeli resursa

- Konflikt oko dodele resursa se moze resiti na jedna od sledeca tri nacina:
  - Mozemo definisati protokol kojim cemo obezbediti da predupredimo pojavu blokada i da nikada sistem ne bude blokiran.
    - ☑ Mehanizmi za prevenciju blokada
      - OS treba da obezbedi da barem jedan od potrebnih uslova za pojavu blokade ne bude ispunjen.
    - ☑ Mehanizmi za izbegavanje blokada
      - Procesi saopstavajui OS-u unapred koji resursi i kojim redosledom ce im trebati. OS odlucuje kada procesi i na koje zahteve trebaju da cekaju.
  - Mozemo omoguciti da sistem detektuje blokadu sistema i definisati mehanizme za oporavak sistema.
  - Mozemo ignorisati problem i ponasati se kao da do blokada sistema nikada ne dolazi.
    - ☑ Najcesce korisceno resenje kod modernih OS-a

Razresavanje konflikata pri dodeli resursa
- Prevencija blokada -

— Medjusobna iskljucivost

- Za sve resurse koji se nemogu deliti mora da vazi princip medjusobne iskljucivosti
- Resursi kao read-only fajlovi se mogu deliti izmedju vise procesa i oni ne moraju da cekaju kada zahtevaju pristup takvim resursima.

— Pristupi i cekaj

- Da se ne bi desilo da proces pristupi nekom resursu a potom udje u stanje cekanja, mroamo da obezbedimo da kad aproces zahteva neki resurs, on nema pristup nijednom drugom resursu.
  - ☑ Moze se postici tako sto bi svaki proces zahtevao sve resurse i ne bi pocinjao sa izvrsavanjem pre nogo sto mu se dodele.
  - ☑ Drugi pristup bi bio da se dozvoli procesu da zahteva resurse samo kada nema nijedan, tj. proces mora da oslobodi sve resurse koje do tada koristio pre nego mu se dozvoli da pristupi novim.

Razresavanje konflikata pri dodeli resursa
- Prevencija blokada -

— Priemtivnost

- Jedna od nacina da se obezbedi ovaj uslov je da ukoliko proces zahteva pristup nekom resursu koji trenutno nije slobodan, tada se pristup tog procesa bilo kojim drugim resursima moze suspendovati, tj. ovi resursi postaju implicitno slobodni i dodaju se listi resursa na koje ovaj proces ceka da se oslobode.
  - ☑ Pre dodele resursa procesu proveravamo da li su slobodni ili pripadaju nekom drugom procesu koji trenutno ceka sa izvrsavanjem.
  - ☑ Ako je to slucaj, proces koji je ispostavio zahtev za pristup resursima ih preuzima od procesa koji ceka

— Cirkularno cekanje

- Jedan nacin da se izbegne cirkularno cekanje jeste da se definise redosled resursa i da proces pristupa resursima po tom redosledu.

Razresavanje konflikata pri dodeli resursa
- Izbegavanje konfikata/blokada -

- Algoritmi za prevenciju blokada funkcionisu tako sto kontolisu nacin na koji procesi zahtevaju resurse.
  - Potencijalno rezultiraju manjom efikasnoscu iskoriscenja sistema.
- Algoritmi za izbegavanje blokada funkcionisu tako sto zahtevaju vise informacija o tome kako ce procesi pristupiti odredjenim resursima i idrektno uticu na redosled izvrsavanja procesa i pristup resursima.
  - U prethodnom slucaju sistem reaguje neposredno pre desavanja konflikata.

Razresavanje konflikata pri dodeli resursa
- Izbegavanje konfikata/blokada -

— Sigurno stanje

- Za neko stanje sistema se kaze da je sigurno ako sistem moze da dodeli resurse svakom procesu odredjenim redosledom i da ne dodje do blokade sistema.
- Stanje sistema je sigurno ako postoji neki redosled izvrsavanja procesa kada ce svakom procesu biti dostupni potrebni resursi.

	Maximum Needs	Current Needs
P_0	10	5
P_1	4	2
P_2	9	2

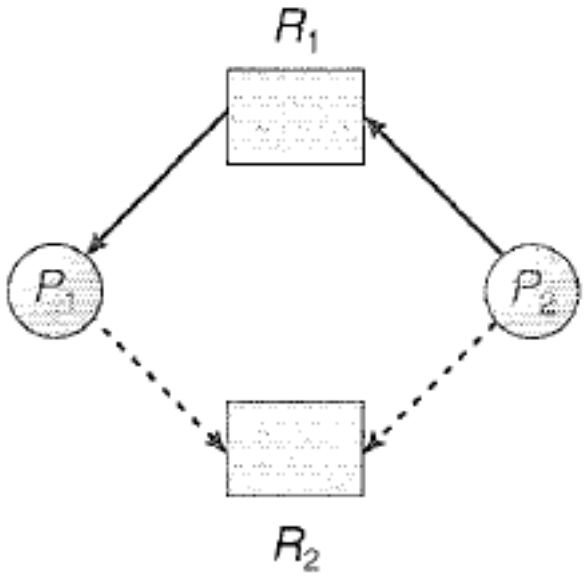
$< P_1, P_0, P_2 >$

Razresavanje konflikata pri dodeli resursa
- Izbegavanje konfikata/blokada -

— Modifikovani graf dodele resursa

- Pored dve vrste veza kojima se pokazuje koji su resursi dodeljeni procesima, na koje resurse procesi cekaju uvodi se jos jedna vrsta veza: veza kojom procesi polazu pravo pristupa resursu, tj. da ce tokom izvrsavanja u nekom trenutku zahtevati pristup datom resursu.
  - ☑ Obelezava se isprekidanom strelicom od procesa ka resursu.

- Kada proces zahteva odredjeni resurs, veza prava pristupa se pretvara u vezu zahteva pristupa.
- Proces pristupa odredjenom resursu, veza postaje veza pristupa procesa resursu.
- Nakon zavrsetka obrade, veza pristupa se ponovo menja u vezu prava pristupa.



Razresavanje konflikata pri dodeli resursa
- Bankarov algoritam -

- Modifikovani graf dodele resursa nije pogodan za primenu kada imamo vise instanci jednog resursa.
- Bankarov algoritam je primenljiv u takvim situacijama ali je i manje efikasan od prethodnog algoritma.
  - Ime dobio jer se moze koristiti na primeru banke koja mora tako da rasporedi svoj novac da uvek bude u stanju da zadovolji potrebe svojih korisnika.
  - Kada novi proces pocne sa izvrsavanjem, on mora da deklarise koliko mu je instanci svakog resursa potrebno za izvrsavanje.
  - OS mora da utvrdi da li ce sistem nakon dodele resursa ostati u sigurnom stanju (safe state), i tek onda dodeljuje resurse procesu.
  - Ako OS utvrdi da u slucaju dodele resursa prelazi u potencijalno nesigurno stanje, tada proces mora da ceka da neki drugi proces oslobodi dovoljno resursa.

Razresavanje konflikata pri dodeli resursa
- Bankarov algoritam -

1010011
1110100
1100001
1010110

- Uvode se nove strukture podataka koje su potrebne da bi se predstavilo stanje sistema.
  - *n* je broj procesa koji se izvrsavaju na sistemu, a *m* je broj resursa sistema
- Strukture podataka
  - Available (Dostupan)
    - ☑ Vektor duzine m, koji pokazuje koliko je instanci svakog resursa dostupno.
  - Max
    - ☑ Matrica dimenzija n × m koja pokazuje za svaki proces maksimalan broj instanci pojedinih resursa koje su mu potrebne za izvrsavanje.
  - Allocation (Dodela)
    - ☑ Matrica dimenzija n × m koja pokazuje za svaki proces koliko mu je instanci pojedinih resursa dodeljeno.
  - Need (Potrebe)
    - ☑ Matrica dimenzija n × m koja pokazuje preostale potrebe procesa za resursima.

Razresavanje konflikata pri dodeli resursa
- Bankarov algoritam -

— Algoritam kojim se utvrdjuje da li se sistem nalazi u sigurnom stanju

1. Let *Work* and *Finish* be vectors of length m and n, respectively. Initialize $Work = Available$ and $Finish[i] = false$ for $i = 0, 1, ..., n - 1$.
2. Find an i such that both
   a. $Finish[i] == false$
   b. $Need_i \leq Work$

   If no such i exists, go to step 4.
3. $Work = Work + Allocation_i$
   $Finish[i] = true$
   Go to step 2.
4. If $Finish[i] == true$ for all i, then the system is in a safe state.

Razresavanje konflikata pri dodeli resursa
- Bankarov algoritam -

— Algoritam za dodelu resursa

➥ Utvrdjuje da li je bezbedno da se nekom procesu dodele resursi

1. If $Request_i \leq Need_i$, go to step 2. Otherwise, raise an error condition, since the process has exceeded its maximum claim.
2. If $Request_i \leq Available$, go to step 3. Otherwise, P_i must wait, since the resources are not available.
3. Have the system pretend to have allocated the requested resources to process P_i by modifying the state as follows:

$$Available = Available - Request_i;$$
$$Allocation_i = Allocation_i + Request_i;$$
$$Need_i = Need_i - Request_i;$$

If the resulting resource-allocation state is safe, the transaction is completed, and process P_i is allocated its resources. However, if the new state is unsafe, then P_i must wait for $Request_i$, and the old resource-allocation state is restored.

Oporavak od blokade sistema

— Algoritam za detekciju blokade sistema

- Modifikacija Bankarovog algoritma
- Graf dodele resursa postaje graf čekanja (wait-for)

1. Let *Work* and *Finish* be vectors of length m and n, respectively. Initialize $Work = Available$. For $i = 0, 1, ..., n\text{-}1$, if $Allocation_i \neq 0$, then $Finish[i] = false$; otherwise, $Finish[i] = true$.
2. Find an index i such that both
   a. $Finish[i] == false$
   b. $Request_i \leq Work$

   If no such i exists, go to step 4.
3. $Work = Work + Allocation_i$
   $Finish[i] = true$
   Go to step 2.
4. If $Finish[i] == false$, for some i, $0 \leq i < n$, then the system is in a deadlocked state. Moreover, if $Finish[i] == false$, then process P_i is deadlocked.

Oporavak od blokade sistema

- Kada OS detektuje da je doslo do blokade sistema
  - Moze obavestiti korisnika da on razresi problem
  - Moze pristupiti automatskoj deblokadi sistema
    - ☑ Prekidanjem izvrsavanja jednog ili vise procesa koji su blokirani
    - ☑ Suspendovanjem pristupa jednom ili vise resursa

Oporavak od blokade sistema

- Prekid izvrsavanja blokiranih procesa
  - Prekida se izvrsavanje svih procesa koji su blokirani
    - ☑ Sigurno razresava sve konflikte pri izvrsavanju
    - ☑ Veoma neefikasan jer je moguce da su se pojedini procesi izvrsavali veoma dugo.
  - Prekida se izvrsavanja jednog po jednog procesa sve dok se sistem ne deblokira
    - ☑ Zahteva dosta dodatne obrade jer se nakon prekida izvrsavanja svakog procesa mora proveriti da li je sistem jos uvek blokiran

Oporavak od blokade sistema

- Koji proces prekinuti sa izvrsavanjem?
  - Koji je prioritet procesa?
  - Koliko dugo se proces izvrsavao i koliko mu je jos potrebno da bi zavrsio sa izvrsavanjem?
  - Koliko i koje vrste resursa je do sada koristio proces?
  - Koliko resursa je potrebno procesu da bi zavrsio sa izvrsavanjem?
  - Koliko procesa cemo morati da prekinemo sa izvrsavanjem?

Oporavak od blokade sistema

- Suspenzija pristupu resursu
  - Koji proces i koje resurse prekinuti sa izvrsavanjem?
    - ☑ Moramo utvrditi redosled kojim cemo suspendovati pristup resursima kako bi minimizovali gubitke.
  - Kako se vratiti korak unazad sa izvrsavanjem procesa?
    - ☑ Sta uraditi sa procesom kome smo oduzeli pravo pristupa resursu cije je izvrsavanje suspendovano?
  - Izgladnjivanje procesa
    - ☑ Kako utvrditi da resursi nece stalno biti oduzimani istom procesu koji za posledicu nikada nece zavrsiti sa svojim izvrsavanjem.
    - ☑ Proces se moze vracati unazad pri izvrsenju samo odredjeni broj puta.

Domaci zadatak

1010011
1110100
1100001
1010110