

1 Avditorne 30. 5. 2019

1. Na računalniku tečejo štiri procesi (P1-P4). Ti med izvajanjem potrebujejo različne vire (R1-R5). števnost virov v računalniškem sistemu je:

R1: 1, R2: 1, R3: 1, R4: 1, R5: 2.

Potrebe so podane v naslednji matriki (v vrsticah so procesi, v stolpcih pa viri):

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 \end{bmatrix}$$

- (a) Trenutno ima P1 v lasti R1, R2 in R5, P4 ima v lasti R5.
(b) Trenutno ima P1 v lasti R1 in R2, P4 ima v lasti 2 x R5.
(c) Trenutno ima P1 v lasti R2 in R5, P2 ima v lasti R1, P4 ima v lasti R4 in R5.

Zapišite sled delovanja algoritma za zaznavo smrtnega objema. Ali je prišlo v sistemu do smrtnega objema?

2. Za spodnji tok naslovov strani, ki so potrebne za izvajanje nekega programa, kjer ima proces v glavnem pomnilniku na voljo **tri** okvirje, zapišite postopek delovanja zamenjevalnega algoritma Princip ure (angl. *Clock Policy*). Pred začetkom izvajanja programa je glavni pomnilnik prazen.

2 1 3 8 4 3 6 5 1 2 3 4 3 2 4

3. Za spodnjih pet procesov skicirajte delovanje razporejanja s povratkom (angl. feedback), kjer velja: $n = 1, i = 0..n, q = 2^i$.

Proces	Čas dospelja	Čas izvajanja
A	0	4
B	3	3
C	4	2
D	5	3
E	7	3

4. Na sistemu zaganjamo dve aplikaciji, vsaka ima dve niti. Prekinitve za namen štetja izvajanja niti in aplikacije se pojavljajo 60-krat v sekundi. Preračun prioritete se zgodi enkrat na sekundo. Osnovna prioriteta ob zagonu je 60. Prva aplikacija naj bo izbrana v 2/3 primerov. Sledite (skicirajte) izvajanju algoritma pravičnega razporejanja (FSS) od časa 0 sekund do časa 7 sekund, kjer poračunajte še vrednosti prioritete!

2 Avditorne 31. 5. 2019

- Po postopku Sistem prijateljev (angl. *Buddy System*) narišite razdeljevanje pomnilnika velikosti 512 KB za primer naslednjih zahtev: A – 60 KB, B – 256 KB, C – 128 KB, sprosti B, D – 50 KB, sprosti C, E – 30 KB, sprosti A, sprosti E, sprosti D. V vsaki vrstici skice, ki predstavlja delovanje posameznega koraka algoritma, pazite na pravilno velikost celic.
- Kratkoročno razporejanje - Najprej tisti z največjim odzivnim razmerjem (HRRN) Skicirajte delovanje algoritma “najprej tisti z največjim odzivnim razmerjem” (HRRN) za spodnje podatke.

proces	čas prispetja	čas izvajanja
P1	0	4
P2	3	3
P3	4	2
P4	5	3
P5	7	3

- Pri kartkoročnem razporejanju smo omenili tudi algoritma Najprej najstarejši (First Come First Served) in Kriterij konstantne časovne rezine (Round Robin). Za spodnje podatke (levo) skicirajte delovanje obeh algoritmov v spodaj pripravljena diagrama! Pri drugem algoritmu upoštevajte, da je časovna rezina procesorskega časa enaka $q=2$ časovnima enotama na diagramu.

proces	čas prispetja	čas izvajanja
P1	0	5
P2	1	3
P3	2	1
P4	4	3

Tabela 1: procesi

- Pri razporejanju v realnem času smo spoznali algoritem *monotono razporejanje* (angl. *RMS*). V sistemu imamo opravka s štirimi periodičnimi procesi. Prvi ima čas izvajanja 20 ms in periodo 100 ms. Drugi ima čas izvajanja 25 ms in periodo 100 ms. Tretji ima čas izvajanja 50 ms in periodo 200 ms. Četrta pa ima čas izvajanja 10 ms in periodo 200 ms. Ali sistem ulovi vse roke? (Argumentirajte z izračunom. Pomoč, da ne rabimo kalkulatorjev: $\sqrt[4]{2} \approx 1'19$.)
- (a) Za spodnji tok naslovov strani, ki so potrebne za izvajanje nekega programa, kjer ima proces v glavnem pomnilniku na voljo tri okvirje, zapišite postopek delovanja zamenjevalnega algoritma Najdlje neuporabljen – LRU (angl. *Least Recently Used*). Pred začetkom izvajanja programa je glavni pomnilnik prazen.
2 1 3 8 4 3 6 5 1 2 3 4 3 2 4
- (b) Kakšna je učinkovitost algoritma (razmerje zadetkov) za ta tok zahtev?

3 Avditorne 3. 6. 2019

- Kakšna je učinkovitost algoritma N -koračno skeniranje (angl. *N-step-scan*) pri $N=3$, če ima disk 2000 sledi, je na začetku glava diska na sledi 150 in se na začetku brani premika proti manjšim sledem, znotraj posameznega koraka algoritma pa ohranja na začetku smer prejšnjega koraka? Razporejevalnik je dobil zahteve po sledih v naslednjem vrstnem redu: 90, 150, 130, 166, 88, 75, 166, 44, 65, 200, 88, 175.
 - Kakšna pa je učinkovitost v primeru algoritma Skeniranje (angl. *Scan*) za ta tok zahtev?
 - Kateri algoritem je za ta tok zahtev boljši?
- Imamo glavni pomnilnik, ki ima dva okvirja (torej prostora za 2 strani). Pri izvajanju programa se zaporedno pojavijo zahteve za naslednje strani v navideznem pomnilniku: 3, 37, 10, 4, 4, 3, 4, 3, 37, 4.
 - Simulirajte zamenjave strani po principu FIFO: za vsak dostop narišite sliko glavnega pomnilnika.
 - Učinkovitost pristopa primerjajte z optimalno strategijo (na istem primeru), za oba primera izračunajte razmerje zadetkov. Kateri je učinkovitejši?
 - Opišite, kako sploh deluje optimalna strategija. Kaj je predpogoj, da lahko uporabimo optimalno strategijo?
- Izračunajte učinek multiprogramiranja v primerjavi z uniprogramiranjem za naslednji primer:

	zahteva 1	zahteva 2	zahteva 3
procesorske zahteve	80%	5%	10%
trajanje	5 min.	10 min.	10 min.
pomnilniške zahteve	100 M	50 M	25 M
potrebuje disk?	ne	ne	da
potrebuje terminal?	ne	da	ne
potrebuje tiskalnik?	ne	ne	da

Na voljo imamo 200 M pomnilnika. Izračunati morate: uporabo procesorja v %, uporabo pomnilnika v %, uporabo diska v %, uporabo tiskalnika v %, čas celotnega izvajanja v minutah ter zmogljivost v zahtevah/uro, in sicer tako za primer uniprogramiranja, kot za primer multiprogramiranja. Pomagate si lahko z utilizacijskim histogramom za vsak vir.

- Kateri dodatni strojni značilnosti smo potrebovali, da smo lahko implementirali podporo multiprogramiranju?
- Tisti, ki ste pridno spremljali predavanja (ali se učili) ste spoznali dva pristopa za izogibanje smrtnemu objemu. Eden od teh je zavrnitev zagona novega procesa. Na preprostem računalniku v Južnokorejskem avtomobilu iz leta 1999 poganjamo tri procese P1 - P3, ki potrebujejo vire R1 - R6. Potrebe so podane v matriki C, trenutne alokacije (lastništva virov po procesih) pa v matriki A. Pri obeh matrikah so procesi podani po vrsticah, viri pa po stolpcih. Števnost virov v tem avtomobilskem računalniku je podana z vektorjem R.

$$C = \begin{bmatrix} 1 & 3 & 2 & 1 & 1 & 0 \\ 3 & 2 & 1 & 1 & 1 & 1 \\ 3 & 0 & 2 & 4 & 2 & 0 \end{bmatrix}$$
$$A = \begin{bmatrix} 1 & 2 & 1 & 1 & 1 & 0 \\ 3 & 2 & 0 & 0 & 0 & 0 \\ 3 & 0 & 1 & 4 & 2 & 0 \end{bmatrix}$$

$$R = [8 \ 6 \ 3 \ 5 \ 3 \ 1]$$

- (a) Ali lahko poženemo tudi proces $P4$, $C(P4) = [110010]$, brez nevarnosti smrtnega objema? Odgovor argumentirajte!
- (b) Kakšna je pomankljivost pristopa zavrnitve zagona novega procesa?
- (c) Kateri drugi pristop ste tisti, ki ste se učili, še spoznali?
5. Zloglasni zajec Rajko-Zajko je letos le obiskoval predavanja, kjer je spoznal tudi algoritem realno časovnega razporejanja po principu prvega roka z neizsiljenim časom čakanja (angl. Earliest deadline with unforced idle times). Ker je med predavanji klepetal, nima pojma in ji boš moral/-a skicirati časovnico izvajanja 20 ms procesov brez preklapljanja iz katerega so jasno razvidne dolžine in vrstni red izvajanj.

Proces	Čas prispetja [ms]	Rok pričetka izvajanja [ms]
A	5	30
B	20	120
C	35	35
D	50	85
E	70	110