Virtuelna memorija - mehanizmi OS-a

Operativni sistemi 1

Institut za matematiku i informatiku
Prirodno-matematički fakultet, Kragujevac

doc. dr Miloš Ivanović

Januar 2013. god.

O čemu će biti reči?

Projektovanje sistema za rad sa virtuelnom memorijom

Osnovna pitanja projektovanja

1. Da li se koriste tehnike virtuelne memorije (da li ih hardver podržava)?
2. Upotreba straničenja, segmentacije ili obe tehnike
3. Algoritmi koji su namenjeni raznim aspektima upravljanja memorijom

Politika donošenja
- Po zahtevu
- Predstraničenje

Politika smeštanja

Politika zamene
- Osnovni algoritmi
  - Optimalni
  - Najmanje skoro korišćena (LRU)
  - FIFO
  - Časovnik
- Baferovanje stranica

Upravljanje rezidentnim skupom
- Veličina rezidentnog skupa
  - Fiksna
  - Promenljiva
- Opseg zamene
  - Globalni
  - Lokalni

Politika čišćenja
- Po zahtevu
- Predčišćenje

Upravljanje učitavanjem
- Stepen multiprogramiranja

Politika donošenja i politika smeštanja

Politika donošenja (*fetching*)

1. **Straničenje po zahtevu** - stranica se donosi samo kada neki proces načini referencu prema toj stranici. Neposredno nakon starta procesa generiše se veliki broj grešaka koji se potom smanjuje.
2. **Predstraničenje** - Uglavnom se bazira na optimizaciji pri korišćenju blok uređaja spoljne memorije koji imaju značajno vreme pozicioniranja i rotaciono kašnjenje.

Politika smeštanja

Gde će stranica i/ili segment biti smešten jedino ima smisla razmatrati u sistemu bez straničenja (čista segmentacija) i u multiprocesoru NUMA tipa (*Non-Uniform Memory Access*). Na standardnim sistemima svakoj adresi može da se priđe sa istim vremenom pristupa.

Politika zamene

Osnovno pitanje projektovanja

Koju stranicu iz glavne memorije izabrati za izbacivanje na uređaj sekundarne memorije kada se donese (*fetch*) nova stranica?

Koncepti upravljanja zamenom

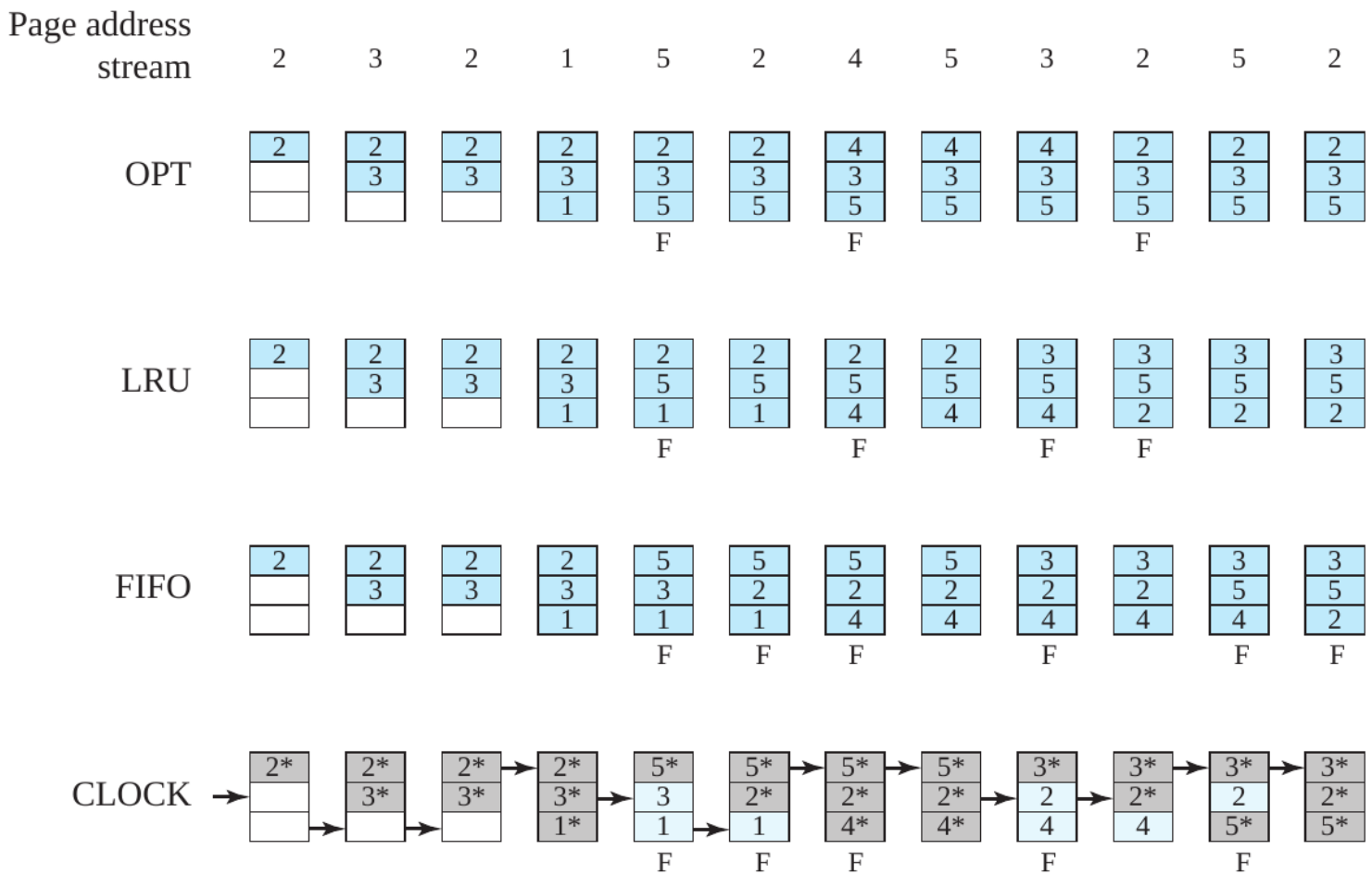
1. Koliko okvira treba dodeliti svakom aktivnom procesu?
2. Da li stranicu za zamenu odabrati iz skupa stranica datog procesa ili iz globalnog skupa?
3. Među stranicama koje dolaze u obzir, koju odabrati za zamenu?

Prva dva koncepta potpadaju pod **upravljanje rezidentnim skupom**, dok se treći bavi **politikom zamene** u užem smislu.
Neke stranice, kao što su stranice koje sadrže kernel OS-a su **zaključane**, tj. ne mogu se zameniti iz RAM-a.

Osnovni algoritmi zamene

1. **Optimalna politika** - bira onu stranicu za koju je vreme do sledećeg referenciranja najduže. Šta je problem?
2. **Least Recently Used (LRU)** - Zamenjuje stranicu koja najduže nije referencirana. Skupa za implementaciju, hardverski ili stek referenci stranice.
3. **FIFO** - Prva unutra, prva napolje. Funkcioniše kao kružni bafer, laka za implementaciju. Pretpostavlja da stranica koja je najranije donesena u RAM može da se zameni, što nije uvek tačno.

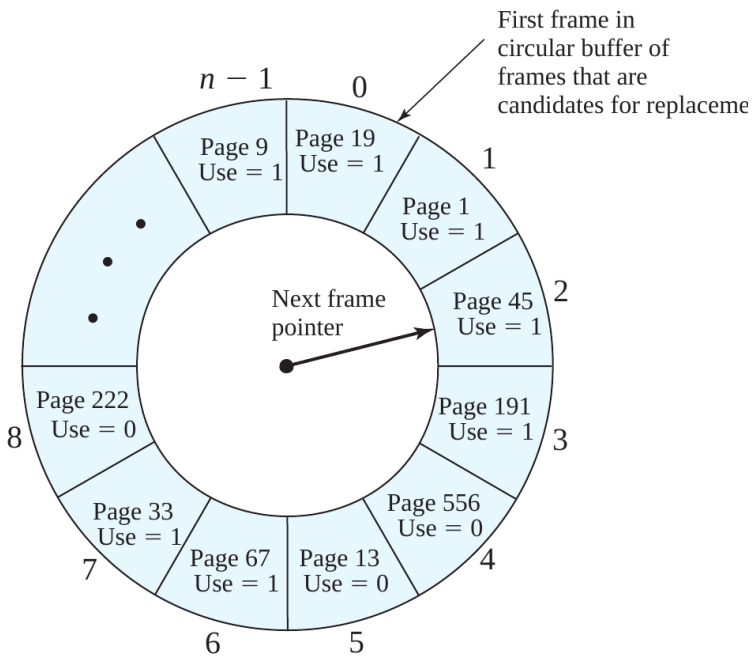
Primer upotrebe osnovnih algoritama zamene

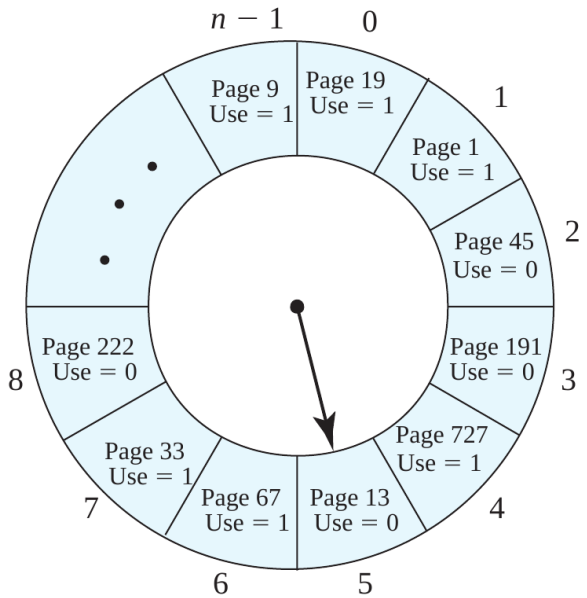


Politika časovnika - osnovni algoritam

Algoritam časovnika

Svakom okviru se pridruži **bit upotrebe**. Pri svakom referenciranju se postavlja na 1. Algoritam zamene ide redom po kružnom baferu u potrazi za prvim okvirom koji ima bit upotrebe 0. Usput anulira bitove upotrebe okvira preko kojih prolazi.





Politika časovnika - kompleksniji algoritam

Umesto jednog bita, svakom okviru se dodeljuje 2 bita, **bit upotrebe** u i **bit promene** m. Na osnovu ova dva bita, stranice se dele u 4 kategorije:

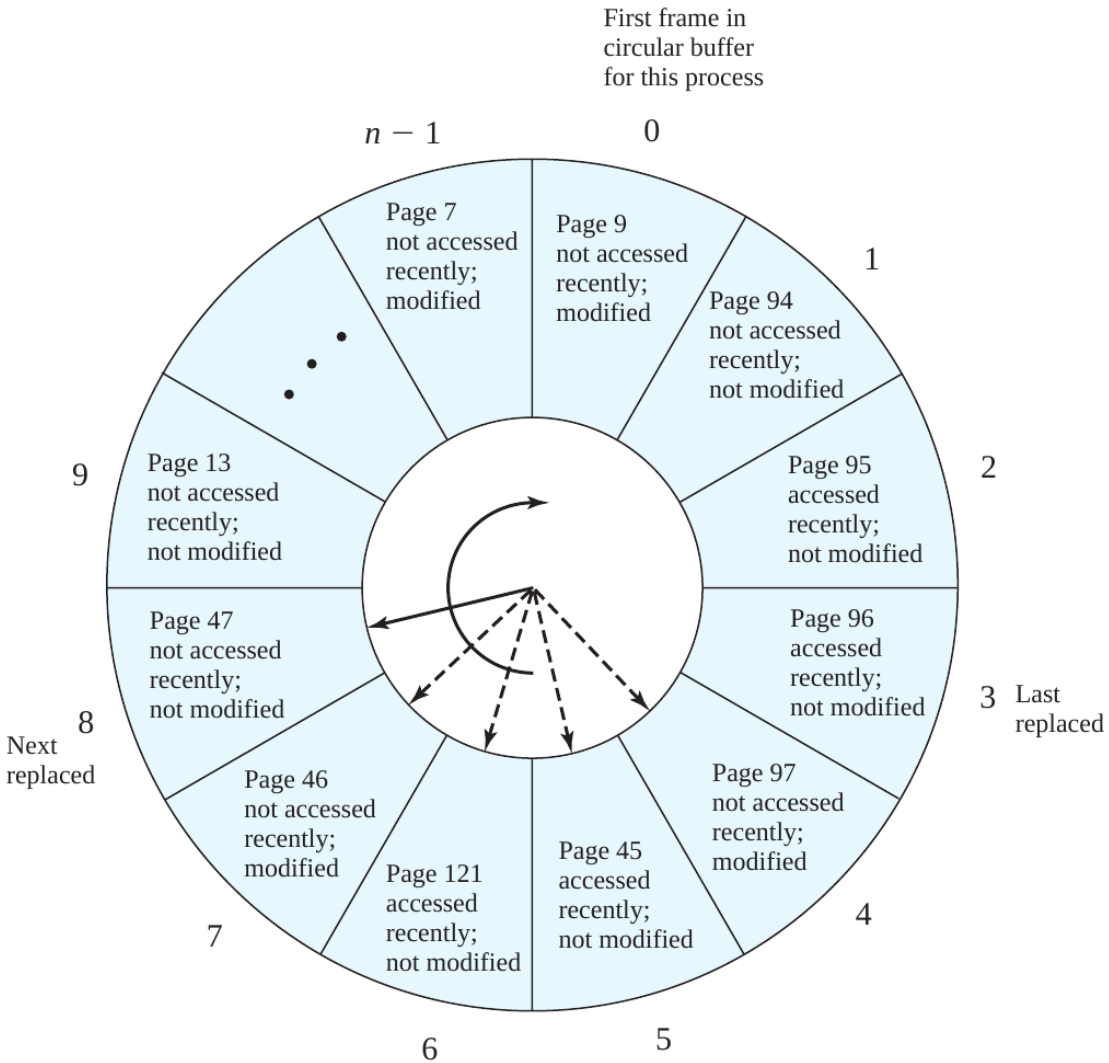
1. Nije joj se skoro pristupalo, nije menjana (u=0,m=0)
2. Skoro joj se pristupalo, nije menjana (u=1,m=0)
3. Nije joj se skoro pristupalo, menjana (u=0,m=1)
4. Skoro joj se pristupalo, menjana (u=1,m=1)

Algoritam zamene

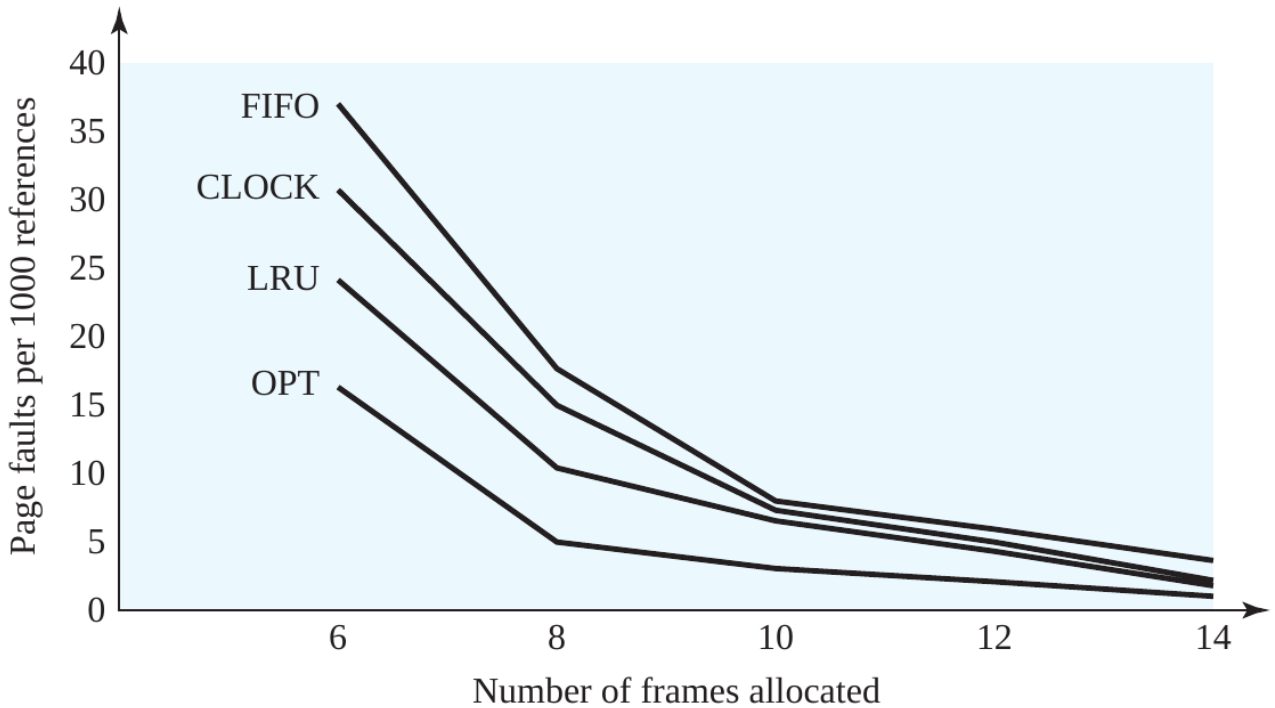
1. Skenira bafer počevši od trenutne pozicije. U toku skeniranja ne menja bit u. Bira prvi okvir sa (u=0,m=0).
2. Ako ne uspe postupak 1, skenira tražeći (u=0,m=1). U toku skeniranja anulira u na svakom okviru na koji naiđe.
3. Ako ne uspe postupak 2, vraća pokazivač na poč. položaj i ponavlja postupak 1 i po potrebi postupak 2.

Politika časovnika - kompleksniji algoritam

Primer



Performase osnovnih algoritama zamene
Simulacija



$0.25x10^6$ referenci u Fortran programu. Veličina stranice je 256 reči. Najpovoljnije je ponašanje sistema na kolenu krive. Zašto?

Baferovanje stranica

- Iako su LRU i politika časovnika značajno bolje od FIFO, **pate od dodatnih režijskih troškova**
- Ukoliko se zamenjene stranice ne upisuju odmah na disk, nego baferuju, to može delom da **pokrije nedostatak običnog FIFO algoritma**
- Na VAX VMS-u je algoritam običan FIFO, ali uz bafer koji radi kao keš stranica
- Još jedna dodatna prednost ovakvog sistema je što se stranice upisuju u grupama (povoljnost za blok uređaje sekundarne memorije jer se smanjuje broj U/I operacija)

Upravljanje rezidentnim skupom
Veličina rezidentnog skupa

Činjenice

- Što manje stranica bude dodeljeno pojedinačnom procesu, to više prcesa može da stane u RAM.
- Što manji broj stranica po procesu - veća učestalost grešaka stranica uprkos principu lokalnosti
- Ako se premaši određeni broj stranica za dati proces - to **neće imati neki značajan efekat na performanse** (opet zbog principa lokalnosti)

Veličina skupa i opseg zamene

Politike veličine rezidentnog skupa

- **Fiksno dodeljivanje** - Procesu se u toku učitavanja dodljuje neki fiksni broj okvira koji ostaje konstantan tokom celog njegovog životnog veka
- **Promenljivo dodeljivanje** - Dozvljava da se broj dodeljenih okvira menja u toku životnog veka procesa. Npr. ako proces generiše veliki broj PF (*Page Fault*) grešaka, dodeli mu se više stanica.

Opseg zamene

- **Politika lokalne zamene** - bira stranicu za zamenu samo iz rezidentnog skupa procesa.
- **Politika globalne zamene** - bira stranicu za zamenu od svih rezidentnih nezaključanih

Upravljanje rezidentnim skupom
Moguće kombinacije

Fiksno dodeljivanje, lokalni opseg

- Potrebno je unapred odlučiti o broju stranica koji se dodeljuje procesu
- Koriste se standardni algoritmi zamene
- **Nedostaci**: Malo dodeljenih okvira–mnogo PF-ova, mnogo dodeljenih okvira–premalo programa u RAM-u

Promenljivo dodeljivanje, globalni opseg

- Najlakše se implementira, stranica za zamenu se bira nekim od standardnih algoritma
- **Nedostatak**: Pošto bilo koji proces može da izgubi okvir u RAM-u, taj izbor ne mora biti optimalan
- Baferovanje stranica može smanjiti loše efekte jer izbor stranice postaje manje značajan

Promenljivo dodeljivanje, lokalni opseg

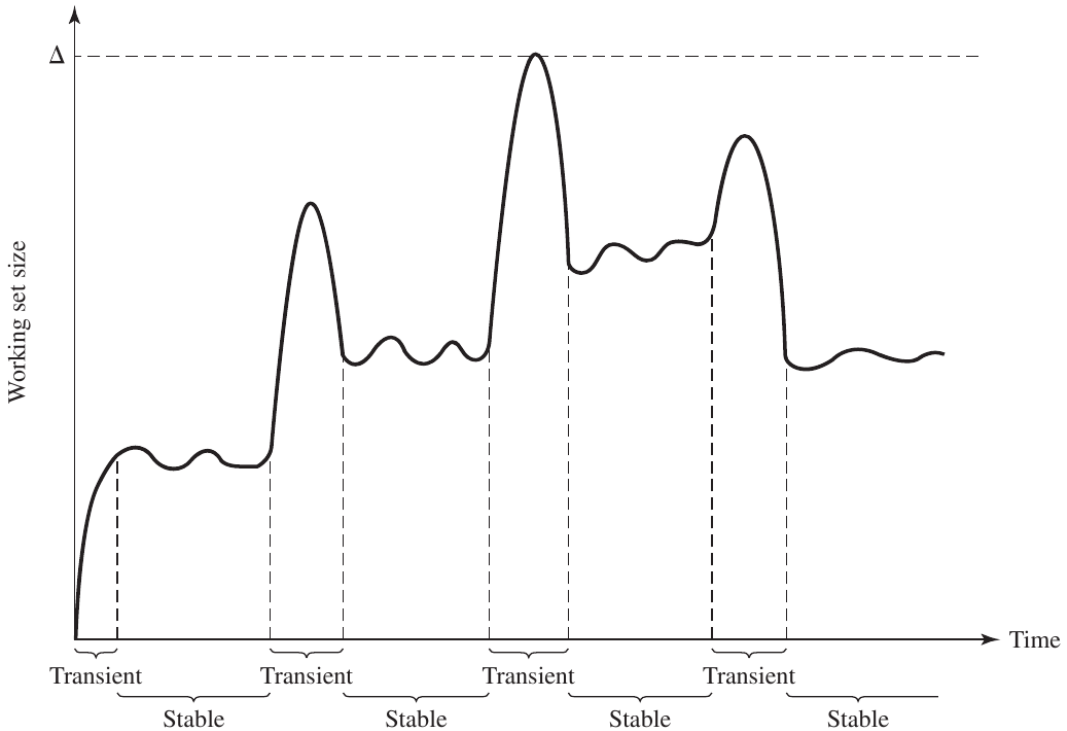
- Kada se učita novi proces, dodeli mu se neki broj okvira
- Kada dođe do PF, za zamenu se bira neka od stranica datog procesa
- Povremeno se procenjuje situacija i broj dodeljenih okvira se povećava ili smanjuje
- Optimizacijom se bavi tzv. **strategija radnog skupa**

Strategija radnog skupa
Promenljivo dodeljivanje, lokalni opseg

- $W(t, \Delta)$ - skup stranica koje su referencirane u poslednjih Δ jedinica virtuelnog vremena, za teki proces u trenutku t
- $W = (t, \Delta + 1) \supseteq W(t, \Delta)$ - veličina radnog skupa je neopadajuća funkcija veličine okvira Δ
- $1 \leq |W(t, \Delta)| \leq min(\Delta, N)$ - radni skup je i funkcija vremena

Sequence of Page References	Window Size, Δ			
W	2	3	4	5
24	24	24	24	24
15	24 15	24 15	24 15	24 15
18	15 18	24 15 18	24 15 18	24 15 18
23	18 23	15 18 23	24 15 18 23	24 15 18 23
24	23 24	18 23 24	•	•
17	24 17	23 24 17	18 23 24 17	15 18 23 24 17
18	17 18	24 17 18	•	18 23 24 17
24	18 24	•	24 17 18	•
18	•	18 24	•	24 17 18
17	18 17	24 18 17	•	•
17	17	18 17	•	•
15	17 15	17 15	18 17 15	24 18 17 15
24	15 24	17 15 24	17 15 24	•
17	24 17	•	•	17 15 24
24	•	24 17	•	•
18	24 18	17 24 18	17 24 18	15 17 24 18

Dijagram veličine radnog skupa



1. Nadgledati radni skup svakog procesa
2. Periodično uklanjati iz rezidentnog skupa stranice koje nisu u radnom skupu
3. Primarni cilj je da rezidentni skup obuhvati radni skup

PFF - *Page Fault Frequency* algoritam

Aproksimacija strategije nadgledanja r.s.

Problemi sa strategijom nadgledanja radnog skupa

1. Prošlost ne predviđa uvek budućnost
2. Pravo merenje radnog skupa za svaki proces je nepraktično
3. Optimalna vrednost za Δ je nepoznata, a i menja se tokom vremena

Strategija PFF

- Merenje učestanosti PF svakog procesa
- Ako je PFF ispod definisanog minimalnogpraga, rezidentni skup može da se smanji
- Ako je PFF iznad definisanog maksimalnog praga, rezidentni skup treba povećati
- Ne radi najbolje u toku prelazinih perioda! Zašto?

Politika čišćenja

- Suprotna je politici donošenja (*fetch*)
- Bavi se odlučivanjem kada stranica treba da se upiše napolje, u sekundarnu memoriju

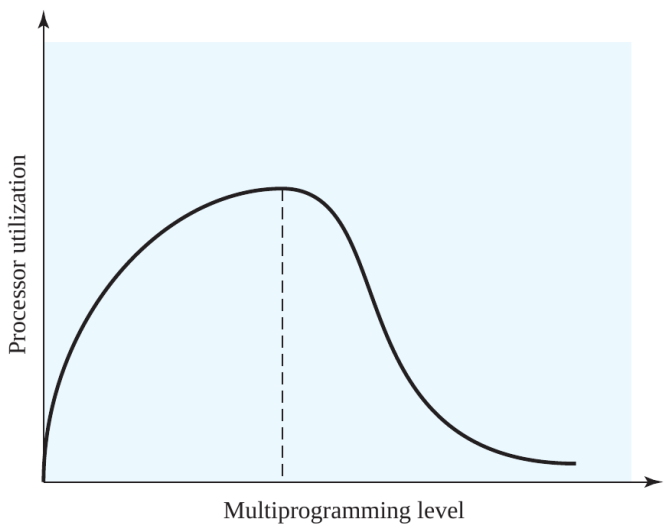
Vrste politike čišćenja

- **Čišćenje po zahtevu** - Stranica se upisuje samo kada biva izabrana za zamenu.
- **Predčišćenje** - Upisuje promenjene stranice pre nego što se one deklarišu za izbacivanje. **Prednost** je što se omogućava upis u paketima, što smanjuje broj U/I operacija. **Mana** je u tome što se modifikovane stranice ispisuju napolje iako će već uskoro ponovo biti promenjene.

Dobro rešenje se ogleda u **baferovanju stranica** na zameni napolje. Održavaju dve liste: **lista promenjenih** i **lista nepromenjenih**. Samo stranice sa liste promenjenih se povremeno upisuju na disk.

Upravljanje učitavanjem - nivo multiprogramiranja

- **Pitanje:** Koliko procesa sme da se istovremeno nalazi u memoriji, a da ne dođe do pada performansi?
- **Više procesa - veće iskorišćenje CPU. Više procesa - manji rezidentni skupovi procesa, češći PF.**
- **Metod 1: Kriterijum** $L = S$: Podešava se nivo multiprogramiranja tako da je srednje vreme između PF-ova jednako vremenu potrebnom da se donese stranica sa diska.
- **Metod 2: Nadgledanje brzine kretanja pokazivača kod algoritma časovnika** Nadgleda se brzina i duzina skeniranja.



Suspenzija procesa

Ako stepen multiprogramiranja treba smanjiti, jedan ili više procesa mora biti suspendovano (razmenjeno napolje). Kako izabrati odgovarajuče?

- **Proces najnižeg prioriteta**
- **Proces koji pravi česte PF** - verovatno ima mali rezidentni skup pa će performanse najmanje trpeti ako se on suspenduje.
- **Poslednji aktiviran proces**
- **Proces sa najmanjim rezidentim skupom**
- **Najveći proces** - dobija se dosta slobodnih okvira