Mr. Mirko Varga

Fakultet organizacije i informatike
V a r a ž d i n

UDK: 681.3.06
Prethodno saopćenje

ISPITIVANJE USPJEŠNOSTI ALGORITAMA STRATEGIJE ZAMJENE STRANICA

U radu se razmatraju najpoznatiji algoritmi strategije zamjene stranica. Kroz izbor određenih algoritama, koji su najčešće zastupljeni, željelo se ukazati na neka moguća mjerila uspješnosti te na konkretnim primjerima izračunati i komparirati uspješnost pojedinih algoritama. Nesumnjivo je jedan od najvažnijih zadataka kod izbora odgovarajuće strategije polučiti što veću uspješnost i približavanje optimalnom rješenju kojeg je u praksi gotovo nemoguće ostvariti. Simulacija je pogodna tehnika za uspoređivanje i ispitivanje uspješnosti navedenih algoritama.

Algoritam; strategija zamjene stranica.

1. UVOD

Strategije upravljanja memorijom sa stanovišta operativnog sustava možemo podijeliti u tri skupine:

1. strategije zamjene ("replacement strategies")
2. strategije poziva i/ili punjenja ("fetch strategies")
3. strategije smještaja ("placement strategies").

U ovom radu stavit ćemo naglasak na najčešće primjenjivane strategije odnosno algoritme zamjene stranica.

Navedene algoritme testirat ćemo i mjerit njihovu uspješnost. Kao jedno od mogućih mjerila uspješnosti može poslužiti broj izmjena stranica. Iz svega dosad rečenog vidljivo je da algoritmi, odnosno strategije zamjene stranica, nalaze svoju primjenu kod sustava alokacije memorije u stranicama na zahtjev (" Demand Paging").

Budući da je problematika utvrđivanja uspješnosti pojedinih algoritama i njihove implementacije dosta složena, često puta nije moguće primijeniti optimalni algoritam koji će minimizirati broj izmjena stranica. Uz najpoznatije algoritme kao što su FIFO, BIFO, LRU, LFU i druge koristit ćemo i optimalni algoritam (OPT, Bela 66; MIN, Matt 70) u cilju ispitivanja uspješnosti navedenih bez obzira što optimalni algoritam zahtijeva poznavanje budućeg ponašanja prisutnih procesa u kompjutorskom sustavu.

2. ALGORITMI STRATEGIJE ZAMJENE

Uobičajeno je da se algoritmi strategije zamjene nazivaju određenim akronimima. Tako postoji niz algoritama kao što su: FIFO, BIFO, LRU, LFU, OPT odnosno MIN itd.

U nastavku dajemo kratki prikaz nekih od spomenutih algoritama. Njihovu uspješnost pratit ćemo na temelju slučajno generiranog referentnog stringa tzv: " traga stranica ".

2.1. FIFO algoritam ("First In First Out")

FIFO algoritam jedan je od najjednostavnijih algoritama zamjene stranica koji zamjenjuje stranicu koja je najduže u memoriji ("najstarija stranica") s traženom stranicom.

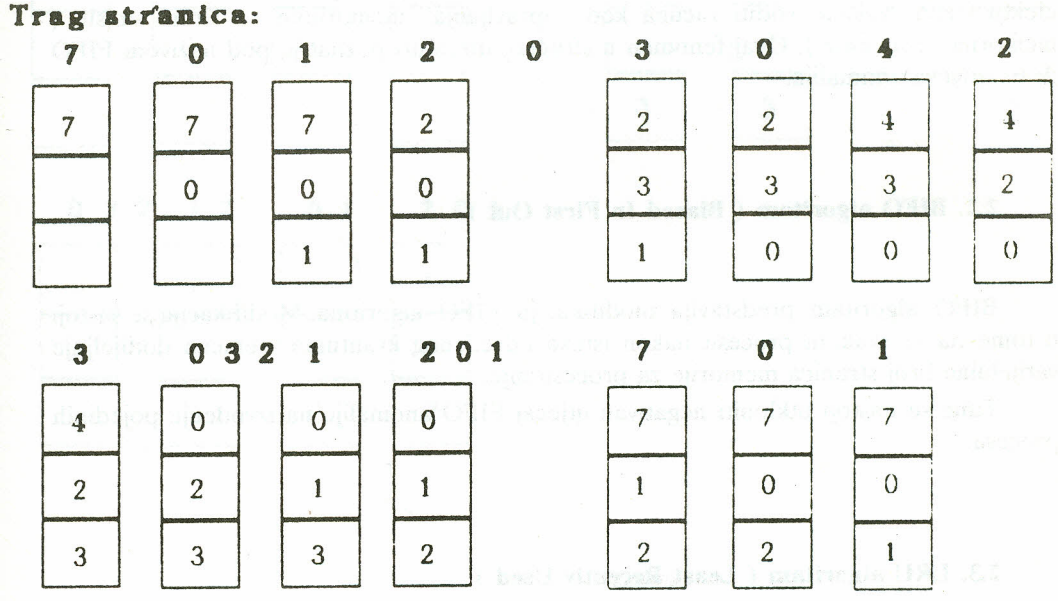
Zbog toga se taj algoritam često naziva "algoritam zamjene najstarije stranice" ("oldest resident ").

Ilustracije radi prikažimo djelotvornost FIFO algoritma na bazi slijedećeg referentnog stringa koji označava " trag stranica" tekućeg procesa u izvođenju:

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

uz promjenu broja stranica odnosno slobodnih blokova ("frames") memorijskog prostora.

Zamjenu stranica primjenom FIFO algoritma uz pretpostavku da koristimo 3 slobodna bloka memorije (3 stranice) prikazuje slika 1.



Sl. 1 Zamjena stranica FIFO algoritmom

Uspješnost FIFO algoritma možemo pratiti utvrđivanjem potrebnog broja izmjena stranica ("number of page faults") odnosno broja neuspješnih referenci u odnosu na ukupan broj traženih referenci. Tako se može izračunati npr: da je u našem slučaju uspješnost FIFO algoritma pod uvjetom da imamo na raspolaganju tri stranice u memoriji svega 25 %.

Naravno, da možemo utjecati na povećanje uspješnosti povećanjem broja raspoloživih stranica u memoriji.

To znači da ako nekom procesu dodijelimo više memorijskog prostora, tj. veći broj stranica, logično je očekivati da će se broj poziva za unošenjem novih stranica smanjivati, a samim time i interno generiran rad računala ("overhead ") prouzročen specijalnom vrstom prekida zbog poziva stranice.

No, kod FIFO algoritma u nekim slučajevima može doći do dijametralno suprotnog efekta koji se očituje u slijedećem: iako procesu dodijeljujemo sve više memorijskog prostora broj izmjena stranica će se povećavati (a ne padati kao što bi se očekivalo) što dovodi do pogoršanja određenih performansi kompjutorskog sustava. O ovom efektu treba svakako voditi računa kod upravljanja memorijom u širem smislu (memorija i procesor). Ovaj fenomen u stručnoj literaturi poznat je pod nazivom FIFO ili Beladyeva anomalija.

2.2. BIFO algoritam (Biased In First Out)

BIFO algoritam predstavlja modifikaciju FIFO algoritma. Modifikacija se sastoji u tome da se svakom procesu nakon isteka određenog kvantuma vremena dodijeljuje varijabilan broj stranica memorije za procesiranje.

Time se nastoji otkloniti negativan utjecaj FIFO anomalije na izvođenje pojedinih procesa.

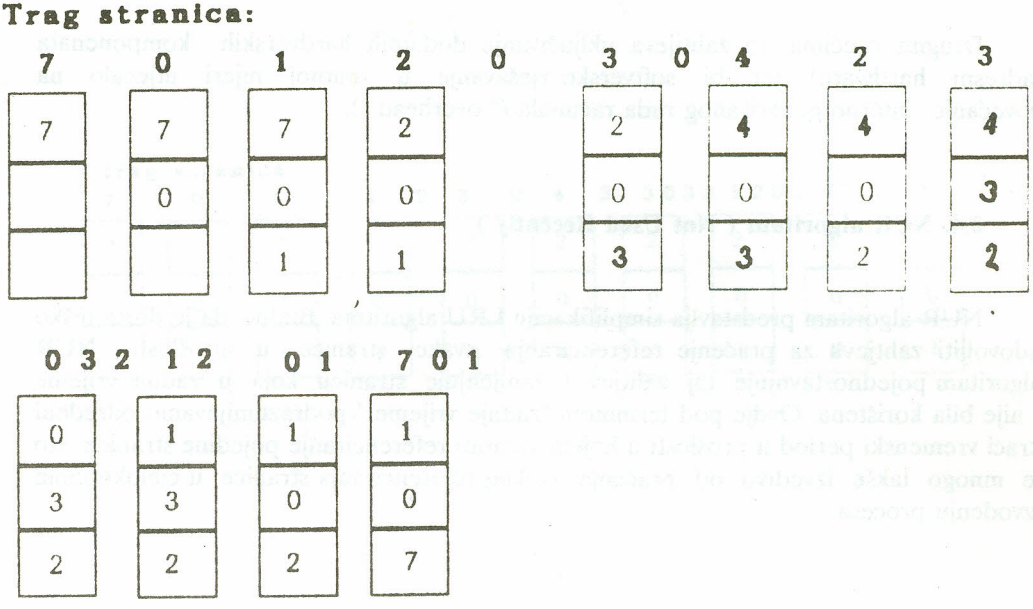
2.3. LRU algoritam (Least Recently Used)

LRU algoritam se zasniva na pretpostavci da će ponašanje procesa u budućnosti biti slično ponašanju procesa u prošlosti, tj. da će proces u kraćim vremenskim razmacima zahtijevati korištenje istih stranica. Zbog toga LRU strategija zamjenjuje stranicu procesa koja najduže vremena nije bila korištena.

Ilustracije radi prikačimo djelotvornost LRU algoritma na bazi već ranije korištenog referentnog stringa koji označava " trag stranica" tekućeg procesa u izvođenju:

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

uz pretpostavku da na raspolaganju imamo 3 slobodna bloka memorije ("frames") odnosno 3 stranice.



Sl. 2 Zamjena stranica LRU algoritmom

Uspješnost LRU algoritma možemo pratiti utvrđivanjem potrebnog broja izmjena stranica ("number of page faults") odnosno broja neuspješnih referenci u odnosu na ukupan broj traženih referenci. Tako se može izračunati npr: da je u našem slučaju efikasnost LRU algoritma pod uvjetom da imamo na raspolaganju tri stranice u memoriji 40 % (1 2 / 20) što je mnogo bolje u odnosu na FIFO algoritam (25%).

Ipak treba ukazati i na neke dodatne zahtjeve u primjeni ovog algoritma. LRU algoritam je znatno teže implementirati, jer se prilikom svakog poziva stranice u memoriju mora pronaći stranica koja najduže nije korištena.

To zahtijeva točno praćenje referenciranja svake stranice tokom izvođenja procesa, što se može izvesti na način da se postojeća tabela stranica proširi za jedno polje koje će se ažurirati svakiput kada se pojedina stranica referencira.

Drugim riječima, to zahtijeva uključivanje dodatnih hardverskih komponenata (adresni hardware) jer bi softversko rješavanje u znatnoj mjeri utjecalo na povećanje interno generiranog rada računala (" overhead ").

2.4. NUR algoritam (Not Used Recently)

NUR algoritam predstavlja simplifikaciju LRU algoritma. Budući da je dosta teško udovoljiti zahtjevu za praćenje referenciranja svake stranice u prošlosti, NUR algoritam pojednostavnjuje taj zahtjev i zamjenjuje stranicu koja u "zadnje vrijeme " nije bila korištena. Ovdje pod terminom "zadnje vrijeme " podrazumijevamo određeni kraći vremenski period u prošlosti u kojem se prati referenciranje pojedine stranice što je mnogo lakše izvedivo od praćenja svakog referenciranja stranice u cjelokupnom izvođenju procesa.

2.5. LFU algoritam (Least Frequently Used)

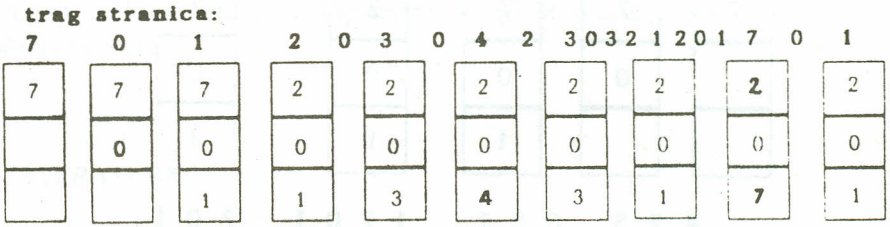
LFU algoritam zamjenjuje stranicu koja je najmanje puta bila upotrebljavana u toku izvođenja procesa.

Implementacija ovog algoritma ima dosta dodirnih točaka s realizacijom LRU strategije. Slično kao i kod LRU algoritma uvodi se brojač + referenciranja svake pojedine stranice čija se vrijednost prilikom svakog korištenja stranice uvećava za 1 (add 1 to counter).

Zamjenjuje se stranica čija je najmanja vrijednost kod pripadajućeg brojača referenciranja stranica.

No, ovdje treba ukazati na jedan problem koji se očituje u slijedećem: novo referencirana (pozvana) stranica imat će inicijalnu vrijednost brojača što može prouzročiti da dođe odmah do izbacivanja upravo te stranice, iako će se ona možda još trebati u toku izvođenja procesa.

Uspješnost LFU algoritma na primjeru korištenog referentnog stringa iznosi 50 %, što je bolje u odnosu na već ranije testirane algoritme (vidi sl. 3).

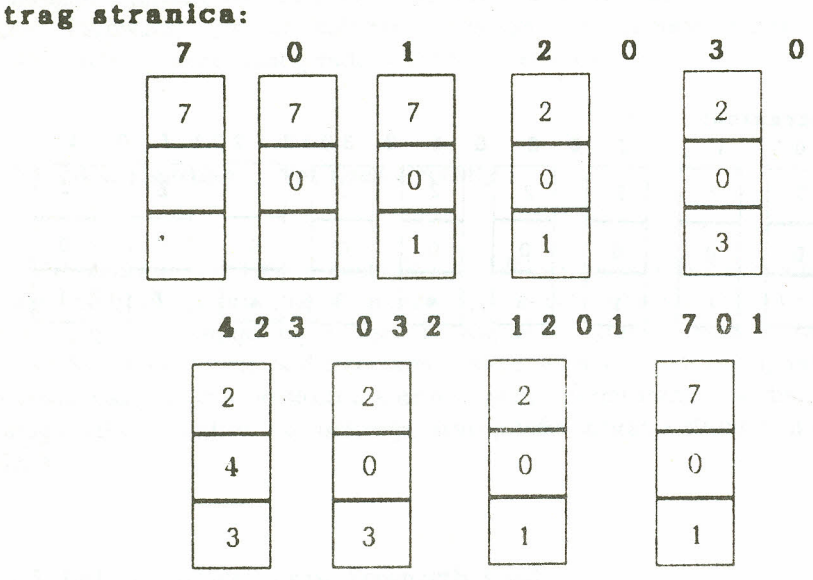


Sl. 3 Zamjena stranica LFU algoritmom

2.6. OPT (OPTimal page replacement)

Optimalni algoritam spominje Belady L. A. 1966, odnosno Mattson R., 1970. godine doduše pod nazivom MIN (od minimalni) što zapravo označava algoritam koji minimizira broj zamjena stranica. Ovaj algoritam ima samo teoretsko značenje te može poslužiti prilikom ispitivanja uspješnosti drugih algoritama.

Prikažimo radi toga optimalnu izmjenu stranica za naš slučaj kako bi mogli usporediti uspješnosti ranije spomenutih algoritama u odnosu na optimalno rješenje (vidi sl. 4).



Sl. 4 Optimalna izmjena stranica

Iz gornjeg prikaza može se zaključiti da je maksimalna uspješnost 55 % (1 1 / 2 0) što ujedno predstavlja i optimum za naš slučaj.

3. ZAKLJUČAK

Strategije zamjene stranica trebaju dati odgovor na ključno pitanje: Koja je stranica najpogodnija za uklanjanje iz memorije ? Naime, nije irelevantno koju stranicu procesa u memoriji zamjenjujemo novom stranicom.

Dokaz za to, jest i postojanje brojnih algoritama zamjene stranica čija je uspješnost različita što je potvrđeno na konkretnom primjeru slučajno generiranog referentnog stringa koji označava "trag stranica".

Pravilnim izborom odgovarajućeg algoritma koji daje smjernice za zamjenu možemo znatno utjecati na povećanje uspješnosti odnosno smanjenje interno generiranog rada računala. Idealno rješenje temelji se na konstruiranju takvog algoritma koji bi zamijenio onu stranicu procesa u memoriji, koju proces najduže vremena neće trebati. Takav algoritam u praksi je vrlo teško implementirati, jer bi to zahtijevalo poznavanje budućeg ponašanja procesa.

LITERATURA:

1. [BEL66] Belady L. A., Study of Replacement Algorithms for Virtual Storage Computer, IBM Systems Journal, vol. 5, No. 5, 1966.
2. [KVA81] Kvaternik R., Uvod u operativne sisteme,Informator, Zagreb, 1981.
3. [MAT70] Mattson R. et al., Evaluation Techniques for Storage Hierarchies, IBM Systems Journal, vol. 9, No. 2, 1970.
4. [P S 8 5] Peterson J. L., Silberschatz, A., Operating System Concepts, Addison-Wesley, 1985.

Primljeno: 1990-06-30

Varga M. Measurements and Efficiency of Page Replacement Algorithms

SUMMARY

The paper describes various page replacement algorithms. FIFO is easy to understand, but suffers from Belady's anomaly. LRU is an approximation of the optimal, but even it may be difficult to implement. Most page replacement algorithms, such as second chance, are approximations of LRU. Optimal (OPT, MIN algorithm) page replacement requires future research.