

---

# Operativni sistemi

## - Alokacija CPU -

Veljko Stanković

---

## — Alokacija CPU (CPU scheduling)

- Predstavlja centralni koncept viseprogramskih OS-a
- Prelaskom sa izvršavanja sa jednog procesa na drugi povećava se efikasnost iskoriscenja sistema.
  - ☑ U glavnoj memoriji se nalazi vise procesa, kada jedna proces mora da saceka sa izvršavanjem, CPU nastavlja sa izvršavanjem drugog procesa.
- Efikasnost algoritma za alokaciju CPU zavisi od uocenih osobina procesa.
  - ☑ Tokom izvršenja procesa se smenjuju ciklusi CPU obrade i I/O poziva
  - ☑ Iako vreme obrade procesa varira od programa do programa i od sistema do sistema, moguće je uociti vezu izmedju broja ciklusa i njihovog trajanja
    - Kraci CPU tj I/O ciklusi se cesce javljaju, i obrnuto, duzi ciklusi se redje javljaju

## — Zadatak alokatora CPU (CPU scheduler) jeste da odabere proces koji ce se izvršavati na CPU.

# Prioritetna alokacija

(Preemptive scheduling)

1010011  
1110100  
1100001  
1010110

- CPU alokacija se može desiti pod sledecim okolnostima
  - Pri prelasku procesa iz stanja running u stanje waiting.
  - Pri prelasku procesa iz stanja running u stanje ready.
  - Pri prelasku procesa iz stanja ready u stanje waiting.
  - Pri završetku izvršenja procesa.
- Nepriemtivna alokacija
  - Alokator čeka sa završenjem procesa da bi ga dodelio nekom drugom procesu
  - Proces se izvršava na CPU sve dok ne završi li pređe u waiting stanje.
- Priemtivna alokacija
  - Alokator može da (“silom”) preuzme CPU od procesa i preda ga nekom drugom procesu
  - Omogućava ravnopravne izvršenje procesa.
  - Zahteva tajmer
  - Neophodni mehanizmi za koordinaciju pristupa procesa podacima kako ne bi više procesa istovremeno menjali/pristupali istim podacima.

# Dispecer (Dispatcher)

1010011  
1110100  
1100001  
1010110

- Dispecer je modul OS-a koji ima zadatak da preda kontrolu nad CPU procesu koji je odabran od strane CPU alokatora.
- Zadaci dispečera su:
  - Promena konteksta CPU
  - Prelazak u korisnički mod
  - Skok na odgovarajucu lokaciju u korisnickom programu
- Neophodno je da dispecer izvrši promenu konteksta za što je kraće vreme.

# Kriterijumi za alokaciju

1010011  
1110100  
1100001  
1010110

- Razliciti algoritmi za alokaciju CPU se razlikuju po svojim osobinama.
- Kriterijumi za ocenu algoritama za alokaciju CPU
  - Iskoriscenje CPU
    - ☑ Cilj je da se CPU iskoristi sto efikasnije. Iskoriscenost moze imati vrednost 0-100%. Obicno se krece u opsegu od 40% do 90%.
  - Propusnost (Throughput)
    - ☑ Predstavlja broj procesa koji se izvrse u jedinici vremena.
  - Vreme izvršenja
    - ☑ Vreme koje protekne od trenutka kada zapocnemo sa izvorsavanjem procesa pa do njegovog zavrsetka.
  - Vreme cekanja
    - ☑ Vreme koje proces provede u ready redu cekanja
  - Vreme odziva
    - ☑ Vreme potrebno da sistem pocne sa davanjem izlaza.

# Kriterijumi za alokaciju

1010011  
1110100  
1100001  
1010110

- Pozeljno je da algoritam za alokaciju CPU ima sto vecu iskoriscenost CPU i propusnost, a sto krace vreme izvršenja, cekanja i odziva.
- U odredjenim slucajevima je potrebno optimizovati minimalne/maksimalne vrednosti ovih parametara na njihove srednje vrednosti.
- Cesto je pozeljno da se minimizuje varijansa ovih parametara.

# Algoritmi za alokaciju CPU

## - First Come First Served -

1010011  
1110100  
1100001  
1010110

- Jedan od najjednostavnijih algoritama za alokaciju CPU
- Proces koji prvi zahteva CPU, preuzima kontrolu nad CPU-om
- Lako se implementira primenom FIFO reda cekanja
- Vreme cekanja je obicno veoma dugo
- FCFS algoritam je nepriemltivan
  - ↳ Nepogodan za primenu u multitasking sistemima

# Algoritmi za alokaciju CPU

## - Shortest Job First -

1010011  
1110100  
1100001  
1010110

- CPU se dodeljuje procesima na osnovu trajanja njihovog sledeceg CPU ciklusa
- Najkraci procesi imaju se prvi izvorsavaju na CPU.
  - Ako dva i vise procesa imaju isto trajanje CPU ciklusa, nad njima se primenjuje FCFS opsluzivanje
- SJF algoritam je optimalan u pogledu minimalnog srednjeg vremena cekanja.
- Problem predstavlja estimacija vremena izvorsavanja procesa.
  - U batch sistemima korisnik zadaje vremenska ogranicenja
  - Ne moze se implementirati na nivou kratkorocnog alokatora jer ne postoji nacin da tacno odredimo vreme izvorsenja procesa u kratkom intervalu
  - Mozemo da procenimo vreme izvorsenja narednog CPU ciklusa procesa na osnovu trajanja prethodnih CPU ciklusa.



# Algoritmi za alokaciju CPU

## - Shortest Job First -

1010011  
1110100  
1100001  
1010110

- Srednje vreme CPU ciklusa se najcesce procenjuje primenom sledece jednacine:

$$\tau_{n+1} = \alpha t_n + (1 - \alpha)\tau_n$$

gde je  $\tau_n$  srednje vrem trajanja CPU ciklusa do trenutka  $n$ ,  $t_n$  je trajanje CPU ciklusa u  $n$ -tom trenutku i parametar  $0 \leq \alpha \leq 1$

- SJF moze biti priemtivan ili nepriemptivan.

➡ Priemtivni SJF algoritam se ponekad naziva najkrace-preostalo-vreme-prvi (shortest remaining time first)

# Algoritmi za alokaciju CPU

## - Prioritetna alokacija -

1010011  
1110100  
1100001  
1010110

- Svakom procesu se dodeljuje prioritet i onda se opsluzuje proces sa najvisim prioritetom
  - Procesi sa istim prioritetom se raspoređuju po FCFS algoritmu
- Prioritet procesa se može definisati interno ili eksterno.
- Prioritet procesa se definiše na osnovu neke velicine koja može da se izmeri
  - Interno: vremenska ograničenja, zahtev za memorijom, broj otvorenih fajlova, odnos procesnog trajanja I/O ciklusa prema prosečnom trajanju CPU ciklusa.
  - Eksterno: Vaznost procesa, cena upotrebe računara, politike, korisnika...
- Problem se javlja jer neki procesi sa niskim prioritetom mogu da čekaju na izvršenje neograničeno dugo (“izgladnjuju se”).
  - Primenjuje se tehnika gde prioritet procesa postepeno raste sa vremenom tokom kojeg čeka na izvršenje (“starenje”)

# Algoritmi za alokaciju CPU

## - Round-Robin -

1010011  
1110100  
1100001  
1010110

- Razvijen narocito za multitasking sisteme
- Slican FCFS algoritmu.
- Svakom procesu se ciklicno dodeljuje CPU neki tokom nekog fiksnog vremenskog intervala (vremenski kvant, 10 – 100 ms).
- Srednje vreme cekanja je obicno veoma dugo.
- Performanse RR algoritma zavise od velicine vremenskog kvanta.
  - Ako je to vreme veoma dugo, RR se svodi na FCFS
  - Ako je vremenski kvant veoma kratak, stvara se iuzija istovremenog izvršenja procesa.
  - Prilikom cestih promena konteksta CPU dolazi do izrazaja vreme potrebno za promenu konteksta.
  - Vreme izvšenja se poboljsava ako vecina procesa završi izvršavanja tokom jednog vremenskog kvanta.

# Algoritmi za alokaciju CPU

## - Multilevel queue scheduling -

1010011  
1110100  
1100001  
1010110

- Hijerarhijski red cekanja se koristi kada je moguće grupisati procese.
- Procesi iz različitih grupa se dodeljuju različitim ready redovima cekanja na osnovu nekog kriterijuma (velicine memorije, prioriteta, tipa procesa).
- U svakom redu cekanja se primenjuje posebni algoritam za alokaciju.
- Potrebno je takođe definisati i prioritet redova cekanja i algoritam za alokaciju CPU između redova.
  - ✚ Primer organizacije redova cekanja:
    - ☑ Sistemski procesi
    - ☑ Interaktivni procesi
    - ☑ Interaktivni editorski procesi
    - ☑ Batch procesi

# Algoritmi za alokaciju CPU

## - Multilevel feedback – queue scheduling -

1010011  
1110100  
1100001  
1010110

- Kada se koristi hijerarhijska alokacija, procesi su za stalno svrstani u određene redove cekanja.
  - Ovakav pristup je prilično nefleksibilan.
- Hijerarhijska alokacija sa povratnom informacijom
  - Proces se mogu prebacivati iz jednog u drugi red cekanja.
  - Proces se grupisu prema karakteristikama njihovih CPU ciklusa.
    - ☑ Ako proces koristi isuvise vremena na izvršavanju na CPU, prebacuje se u red cekanja sa nizim prioritetom.
    - ☑ Proces koji je duze vremena proveo u redu cekanja sa niskim prioritetom se moze prebaciti u red cekanja sa visim prioritetom.

# Algoritmi za alokaciju CPU

## - Multilevel feedback – queue scheduling -

1010011  
1110100  
1100001  
1010110

### — Primer: Hijerarhijska alokacija sa 3 reda cekanja

- Redovi cekanja su klasifikovani prema trajanju CPU ciklusa.
- Oznaceni su rednim brojevima 0-2 i njima odgovara trajanje CPU ciklusa od 8, 16 i 24 ms.
- Procesi u redu 1 se izvršavaju tek kada je red 0 prazan, a u redu 2 tek kada su redovi 1 i 0 prazni.
- Ukoliko se prilikom izvršavanja procesa u redu 1 (2) javi proces u redu 0 (0 ili 1) proces prekida sa izvršavanjem i izvršava se proces sa visim prioritetom.
- Unutar odredjenog reda cekanja procesi se obradjuju po algoritmu FCFS

# Algoritmi za alokaciju CPU

## - Multilevel feedback – queue scheduling -

1010011  
1110100  
1100001  
1010110

- Hijerarhijski protokol sa povratnom informacijom je određen:
  - Brojem redova cekanja
  - Algoritmom za alokaciju u okviru posebnih redova
  - Kriterijumom za prebacivanje procesa u red sa visim prioritetom
  - Kriterijumom za prebacivanje procesa u red sa nizim prioritetom
  - Kriterijumom za odredjivanje reda cekanja u koji treba smestiti proces koji zahteva opsluzivanje
- Ovo je najfleksibilniji ali i najslozeniji alogritam za alokaciju CPU.

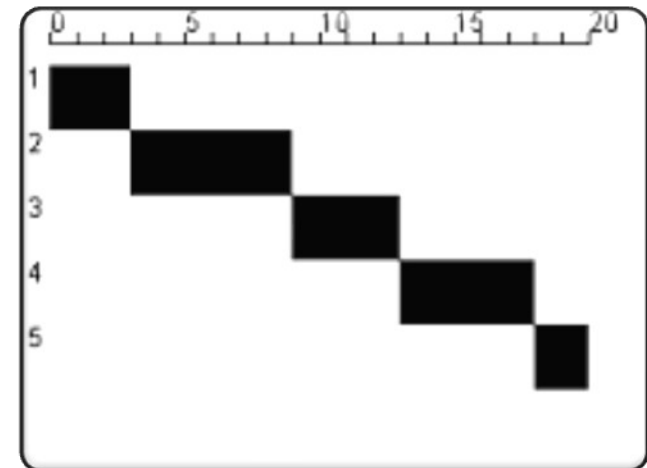
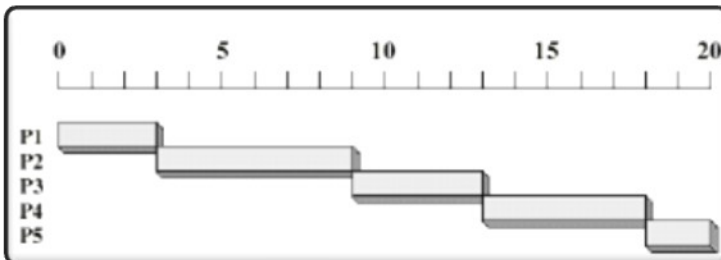
## — FCFS

Proces	Vreme pristizanja	Vreme obrade
P1	0	3
P2	2	6
P3	4	4
P4	6	5
P5	8	2



## — FCFS

Proces	Vreme pristizanja	Vreme obrade
P1	0	3
P2	2	6
P3	4	4
P4	6	5
P5	8	2



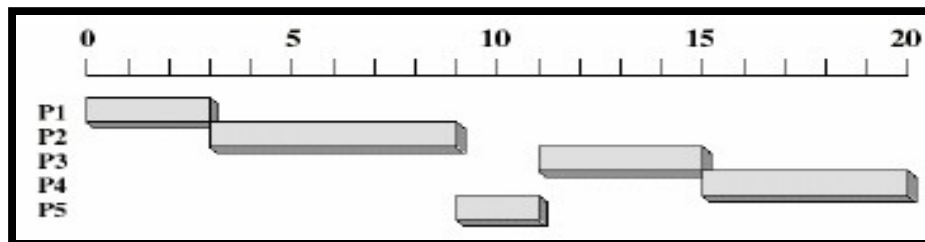
## — FCFS

Proces	Vreme obrade
P1	30
P2	20
P3	10

P1	
P2	
P3	

## — Shortest Job First (SJF)

Proces	Vreme pristizanja	Vreme obrade
P1	0	3
P2	2	6
P3	4	4
P4	6	5
P5	8	2



## — Shortest Job First (SJF)

Proces	Vreme obrade
P1	30
P2	20
P3	10

P1	
P2	
P3	

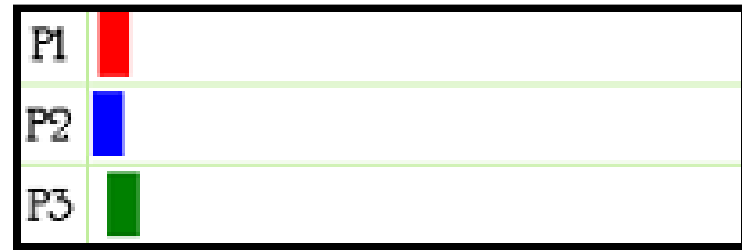
## — Shortest Remaining Time First (SRTF)

Proces	Vreme pristizanja	Vreme obrade
P1	0	3
P2	2	6
P3	4	4
P4	6	5
P5	8	2



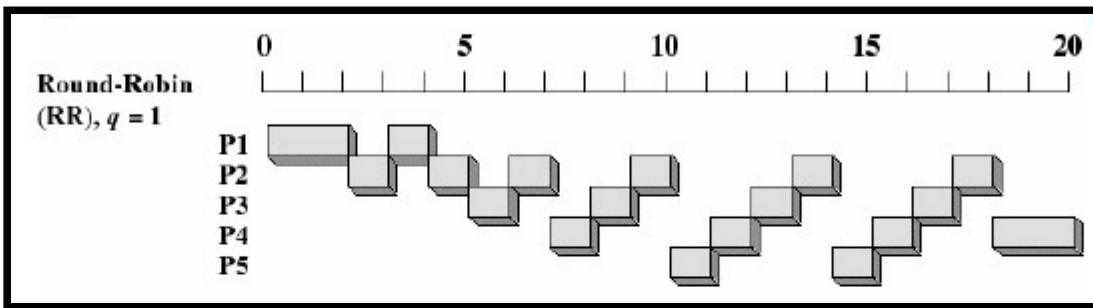
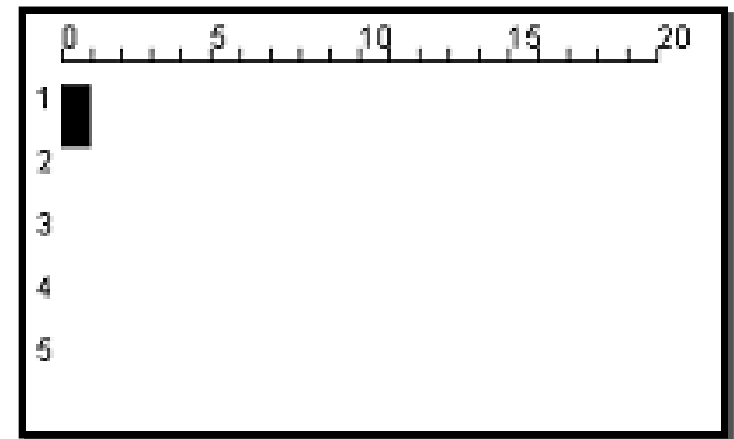
## — Shortest Remaining Time First (SRTF)

Proces	Vreme obrade
P1	30
P2	20
P3	10






## — Round Robin (RR)

Proces	Vreme pristizanja	Vreme obrade
P1	0	3
P2	2	6
P3	4	4
P4	6	5
P5	8	2



## — Round Robin (RR)

Proces	Vreme obrade
P1	30
P2	20
P3	10

P1	
P2	
P3	



## — Zadatak

Dat je sledeći skup procesa čija su vremena izvršavanja na procesoru i prioriteti dati u tabeli: Procesi su u sistem naišli u poretku P1, P2, P3, P4, P5, svi približno u trenutku  $t=0$ .

(a) Nacrtati *Gantt*-ove karte dodele procesora ukoliko se raspoređivanje vrši na osnovu sledećih algoritama: FCFS, SJF bez istiskivanja, PS bez istiskivanja i RR sa kvantomom  $Q=1$ .

(b) Odrediti vreme potrebno za kompletiranje procesa za svaki proces (za sve gore pomenute algoritme).

(c) Odrediti vreme čekanja za svaki proces i srednje vreme čekanja (za sve gore pomenute algoritme). Za koji algoritam je srednje vreme čekanja najmanje?

Proces	Vreme izvršavanja	Prioritet
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

(a) *Gantt*-ove karte dodele procesora:

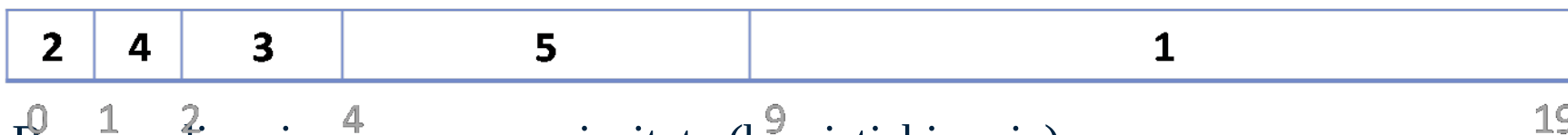
FCFS:



RR(Q=1):



SJF (bez istiskivanja):



Raspoređivanje na osnovu prioriteta (bez istiskivanja):



— Vreme potrebno za zavrsetak procesa

	FCFS	RR	SJF	PS
P1	10	19	19	16
P2	11	2	1	1
P3	13	7	4	18
P4	14	4	2	19
P5	19	14	9	6

## — Vreme cekanja

	FCFS	RR	SJF	PS
P1	0	9	9	6
P2	10	1	0	0
P3	11	5	2	16
P4	13	3	1	18
P5	14	9	4	1
Srednje vreme	9,6	5,4	3,2	8,2

## — Zadatak

Dat je sledeći skup procesa čija su vremena nailaska u sistem i izvršavanja na procesoru dati u tabeli (izraženi u ms):

Nacrtati *Gantt*-ovu kartu i odrediti srednje vreme za kompletiranje procesa i srednje vreme čekanja ukoliko se raspoređivanje procesa vrši po:

- (a) FCFS algoritmu,
- (b) SJF algoritmu bez istiskivanja,
- (c) SJF algoritmu sa vremenom čekanja na procese  $t_{idle}=1$ ,
- (d) SRTF algoritmu sa istiskivanjem.

Proces	Vreme izvršavanja	Vreme nailaska
P1	8	0
P2	4	0,4
P3	1	1

## — FCFS



	Vreme kompletiranja	Vreme čekanja
P1	8	0
P2	$12 - 0,4 = 11,6$	$8 - 0,4 = 7,6$
P3	$13 - 1 = 12$	$12 - 1 = 11$
Srednje vreme	10,53	6,2

— SJF



	Vreme kompletiranja	Vreme čekanja
P1	8	0
P2	$13 - 0,4 = 12,6$	$9 - 0,4 = 7,6$
P3	$9 - 1 = 8$	$8 - 1 = 7$
Srednje vreme	9,53	5,2

# Vezbe

1010011  
1110100  
1100001  
1010110

— SJF,  $T_{idle} = 1$



	Vreme kompletiranja	Vreme čekanja
P1	14	6
P2	$6 - 0,4 = 5,6$	$2 - 0,4 = 1,6$
P3	$2 - 1 = 1$	0
Srednje vreme	6,86	2,53



## — SRTF

<b>1</b>	<b>2</b>	<b>3</b>	<b>2</b>	<b>1</b>		
0	0,4	1	2	5,4	6	13

	Vreme kompletiranja	Vreme čekanja
<b>P1</b>	13	$5,4 - 0,4 = 5$
<b>P2</b>	$5,4 - 0,4 = 5$	$2 - 1 = 1$
<b>P3</b>	$2 - 1 = 1$	0
<b>Srednje vreme</b>	6,33	2

## — Zadatak

Četiri procesa su u trenutku  $t=0$  ušli u red čekanja na procesor u sledećem redosledu: P1, P2, P3, P4. Vremena izvršavanja na procesoru za ova četiri procesa su 6, 3, 1 i 7 vremenskih jedinica, respektivno. Ukoliko se raspoređivanje vrši RR algoritmom sa kvantomom  $Q=2$ :

nacrtati Gantt-ovu kartu i odrediti srednje vreme izvršavanja na procesoru i srednje vreme čekanja,

odrediti koliko puta je obavljena zamena konteksta i koliko je ukupno vremena potrebno da sva četiri procesa završe aktivnosti (kašnjenje dispečera je  $d_l=0,01$  vremenskih jedinica).

$$t=17+9*0.01=17.09$$

<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>1</b>	<b>2</b>	<b>4</b>	<b>1</b>	<b>4</b>	<b>4</b>
0	2	4	5	7	9	10	12	14	16

	Vreme kompletiranja	Vreme čekanja
<b>P1</b>	14	5+3=8
<b>P2</b>	10	2+5=7
<b>P3</b>	5	4
<b>P4</b>	17	5+3+2=10
<b>Srednje vreme</b>	11,5	7,25

# Domaci zadatak

1010011  
1110100  
1100001  
1010110

---