

Raspoređivanje procesa (CPU Scheduling)

- **Osnovni koncepti**
- **Kriterijumi za raspoređivanje**
- **Algoritmi za raspoređivanje**
- **Raspoređivanje u višeprocorskoj okolini**
- **Raspoređivanje u realnom vremenu**
- **Ispitivanje algoritama**

Osnovni koncepti

■ **Maksimalno iskorišćenje CPU-a**

- ☞ postiže se
- ☞ **multiprogramiranjem**

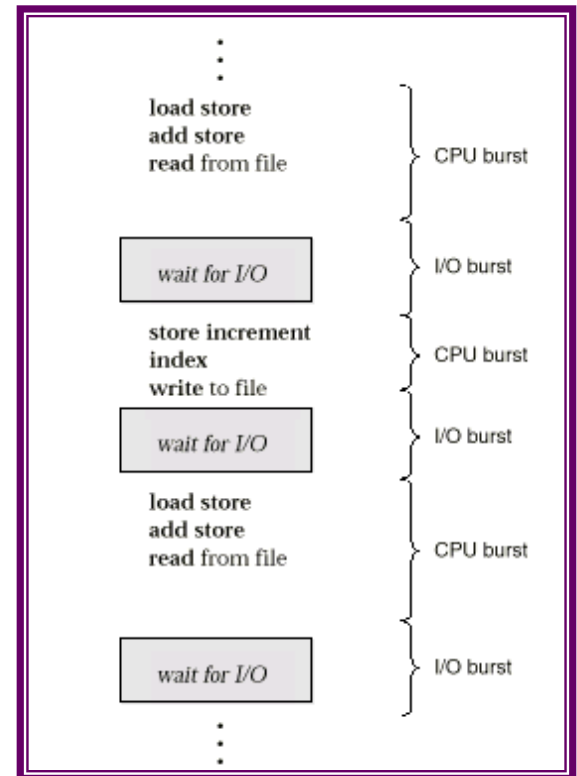
■ **CPU-I/O Burst Cycle:**

■ Izvršavanje procesa se sastoji od

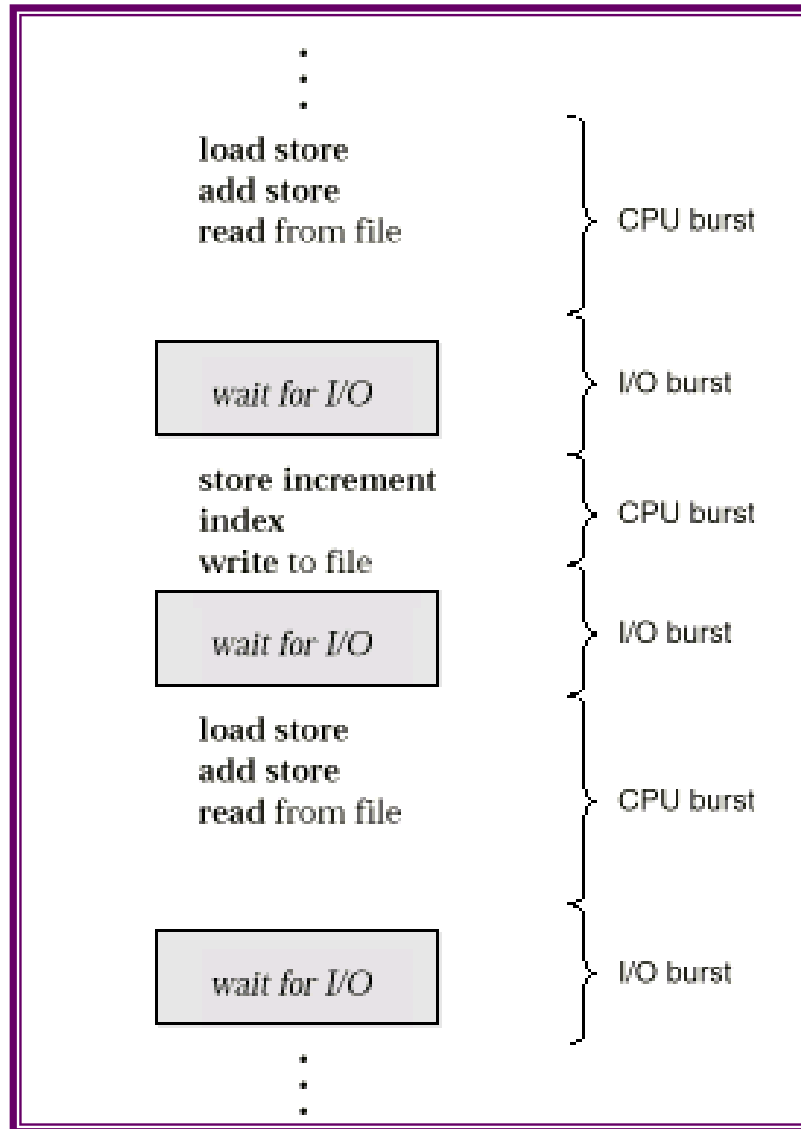
- ☞ **ciklusa CPU izvršavanja**
- ☞ **i**
- ☞ **ciklusa čekanja na I/O operacije**

■ **CPU burst distribucija**

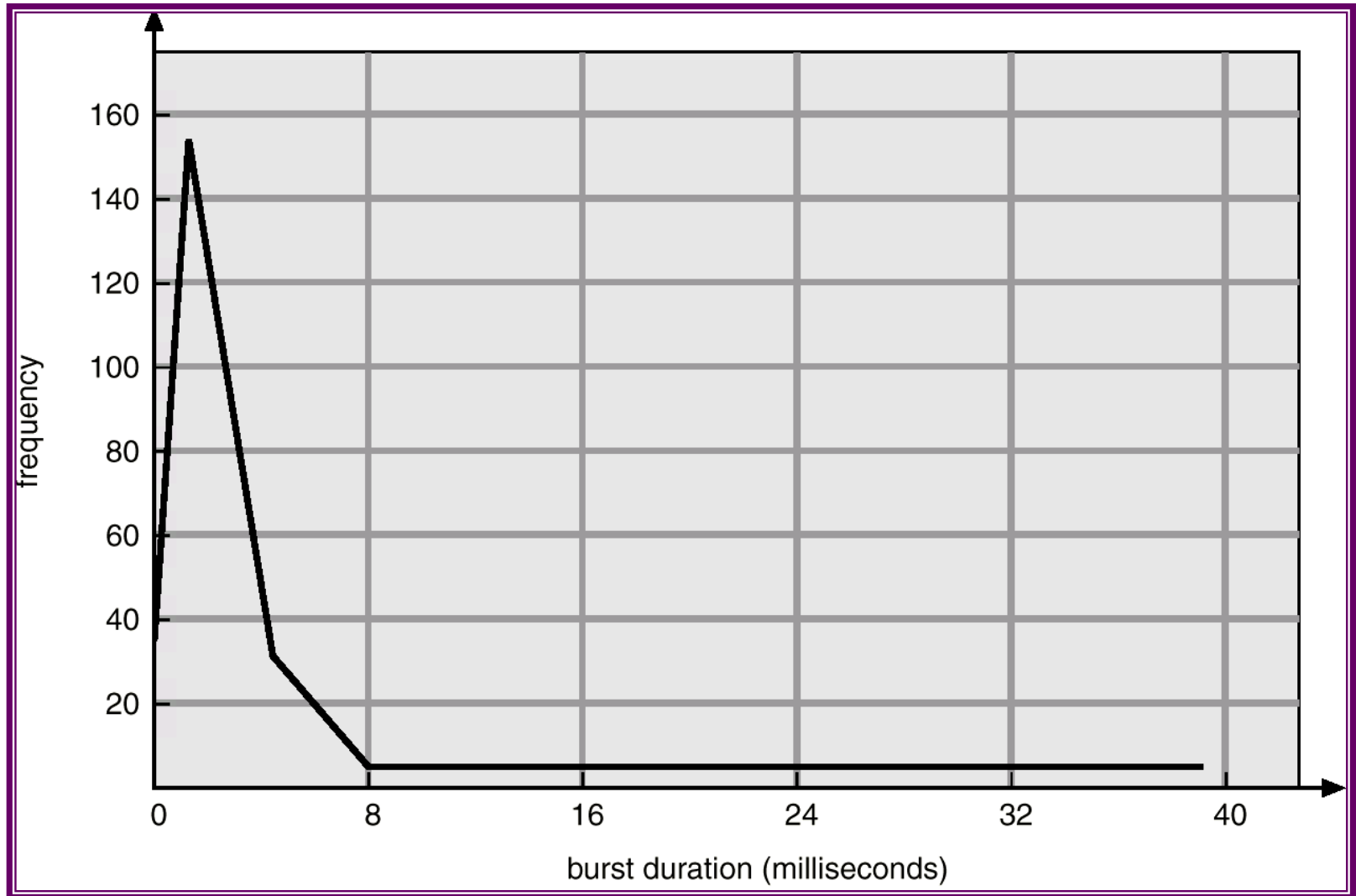
- ☞ **malo dugačkih-burst procesa**
- ☞ **mного kratkih-burst procesa**



Naizmenična sekvenca CPU i I/O Bursts

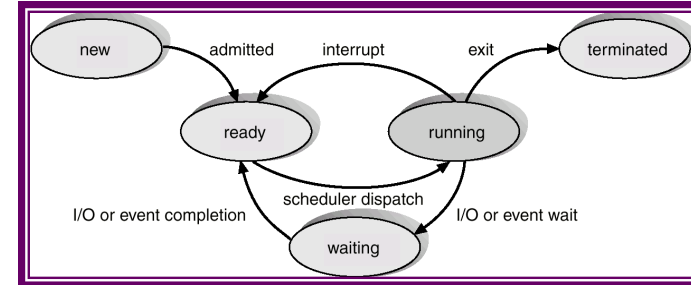


Histogram CPU-burst vremena



CPU raspoređivač (*CPU Scheduler*)

- **Selektuje** iz reda čekanja procese u memoriju
- koji su spremni da se izvrše,
- i **dodeljuje CPU jednom od njih**



- **CPU raspoređivanje se dešava kada** se proces:
 1. **Prebacuje iz stanja izvršavanja u blokirano stanje**
 2. **Prebacuje iz stanja izvršavanja u stanje spremnosti**
 3. **Prebacuje iz blokiranog stanja u stanje spremnosti**
 4. **Završi svoje aktivnosti**
- Raspoređivanje koje se dešava pod okolnostima 1 i 4 je **bez prekidanja (*non-preemptive*)**
- Raspoređivanje pod drugim okolnostima je **preemptive**

Dispečer (*Dispatcher*)

- **Dispečer** modul
 - ☞ daje kontrolu CPU procesu
 - ☞ koji je izabran od strane kratkoročnog raspoređivača

- **Funkcije** dispečera:
 - ☞ **prebacivanje konteksta**
 - ☞ prebacivanje u korisnički režim rada
 - ☞ skok na odgovarajuću lokaciju izabranog procesa koji će omogućiti njegovo izvršavanje

- **Kašnjenje dispečera (*Dispatch latency*):**
 - ☞ vreme koje je potrebno dispečeru
 - ☞ da zaustavi jedan proces i
 - ☞ startuje izvršavanje drugog.

Kriterijumi za raspoređivanje

■ Iskorišćenje CPU

- ☞ održavati CPU zauzetim (busy) kad god je moguće

■ Propusna moć (*Throughput*):

- ☞ broj procesa izvršenih u jedinici vremena

■ Vreme za kompletiranje procesa (*Turnaround time, TA, ATA*):

- ☞ ukupna količina vremena potrebna da se izvrši jedan proces

■ Vreme čekanja (*Waiting time, WT, AWT*) –

- ☞ ukupno vreme koje proces provede u redu čekanja spremnih procesa (ready queue)

■ Vreme odziva (*Response time*)

- ☞ vreme kada se uputi neki zahtev
- ☞ pa do pojave prvih rezultata procesa

Optimizacija kriterijuma

- **Maksimalno iskorišćenje CPU-a**
- **Maksimalna propusna moć**
- **Minimalno vreme za kompletiranje procesa**
- **Minimalno vreme čekanja**
- **Mininimalno vreme odziva**

Algoritmi

■ FCFS

■ SJF <-> SRTF

■ RR

■ Priority

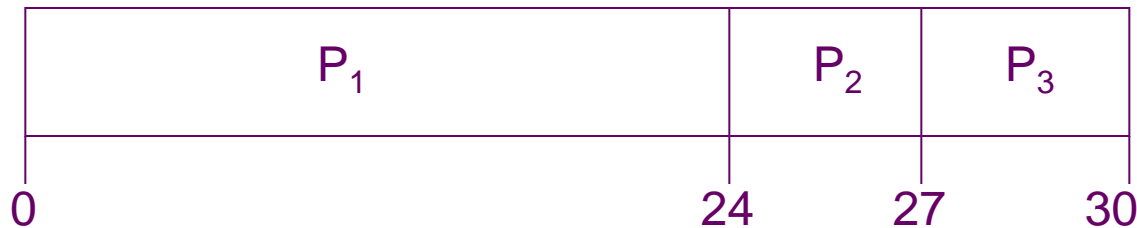
■ Multilevel

<u>Proces</u>	<u>Vreme izvršavanja</u>
<i>P1</i>	24
<i>P2</i>	3
<i>P3</i>	3

First-Come, First-Served (FCFS) raspoređivanje

<u>Proces</u>	<u>Vreme izvršavanja</u>
P_1	24
P_2	3
P_3	3

- Pretpostavimo da procesi dolaze sledećim redom: P_1 , P_2 , P_3
Gantt karta za raspoređivanje je:



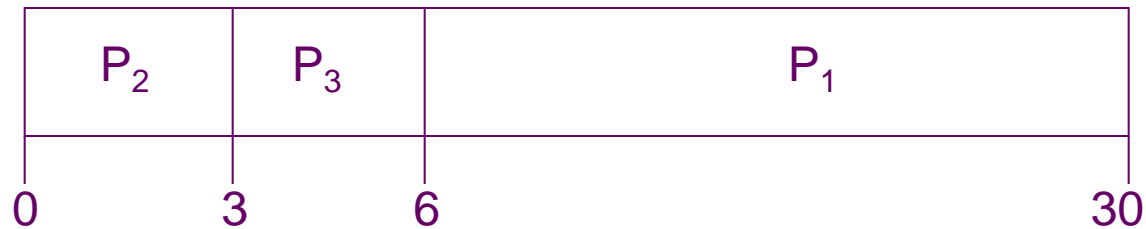
- **Vreme čekanja** za $P_1 = 0$; $P_2 = 24$; $P_3 = 27$
- **Srednje vreme čekanja:** $(0 + 24 + 27)/3 = 17$

FCFS raspoređivanje

Pretpostavimo da procesi dolaze sledećim redom:

P_2, P_3, P_1 .

- **Gantt karta** za raspoređivanje je:



- **Vreme čekanja** za $P_1 = 6$; $P_2 = 0$; $P_3 = 3$
- **Srednje vreme čekanja:** $(6 + 0 + 3)/3 = 3$
- **Mnogo bolje nego prošli slučaj (AWT je bio 17)**
- **Efekat konvoja:**
 - ☞ kratki procesi
📄 iza
 - ☞ dugačkih procesa

Shortest-Job-First (SJF) raspoređivanje

- Svaki proces određuje dužinu svog **sledećeg CPU ciklusa**
- Ove dužine služe za raspoređivanje procesa sa najkraćim vremenom
- **Dve šeme:**
 - ☞ **SJF: bez pretpražnjenja (*nonpreemptive*):**
 - ☞ kada je CPU dat procesu
 - ☞ **N** emoze biti oduzet
 - ☞ dok se proces ne izvrši do kraja.
 - ☞ **SRTF: sa pretpražnjenjem (*preemptive*):**
 - ☞ ako dođe novi proces
 - ☞ sa vremenom izvršavanja manjim od
 - ☞ preostalog vremena izvršavanja tekućeg procesa,
 - ☞ nastaje pretpražnjenje.
 - ☞ Ova šema je poznata kao **Shortest-Remaining-Time-First (SRTF)**
- **SJF je optimalan – daje najmanje srednje vreme čekanja za dati skup procesa.**

Određivanje dužine sledećeg CPU ciklusa

- **Može se samo proceniti** dužina

- **Procena se obavlja** tako što se

- ☞ na bazi svih prethodnih CPU ciklusa,

- ☞ odredi eksponencijalna srednja vrednost

1. t_n = aktuelna dužina n - tog CPU ciklusa

2. τ_{n+1} = procenjena vrednost sledećeg CPU ciklusa

3. $\alpha, 0 \leq \alpha \leq 1$

4. Formula:

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

Primer eksponencijalne srednje vrednosti

■ $\alpha = 0$

☞ $\tau_{n+1} = \tau_n$

☞ **Skorija istorija se neračuna** (stvarne dužine nemaju efekta)

■ $\alpha = 1$

☞ $\tau_{n+1} = t_n$

☞ **Samo poslednji CPU ciklus se računa**

■ Ako proširimo formulu dobijamo:

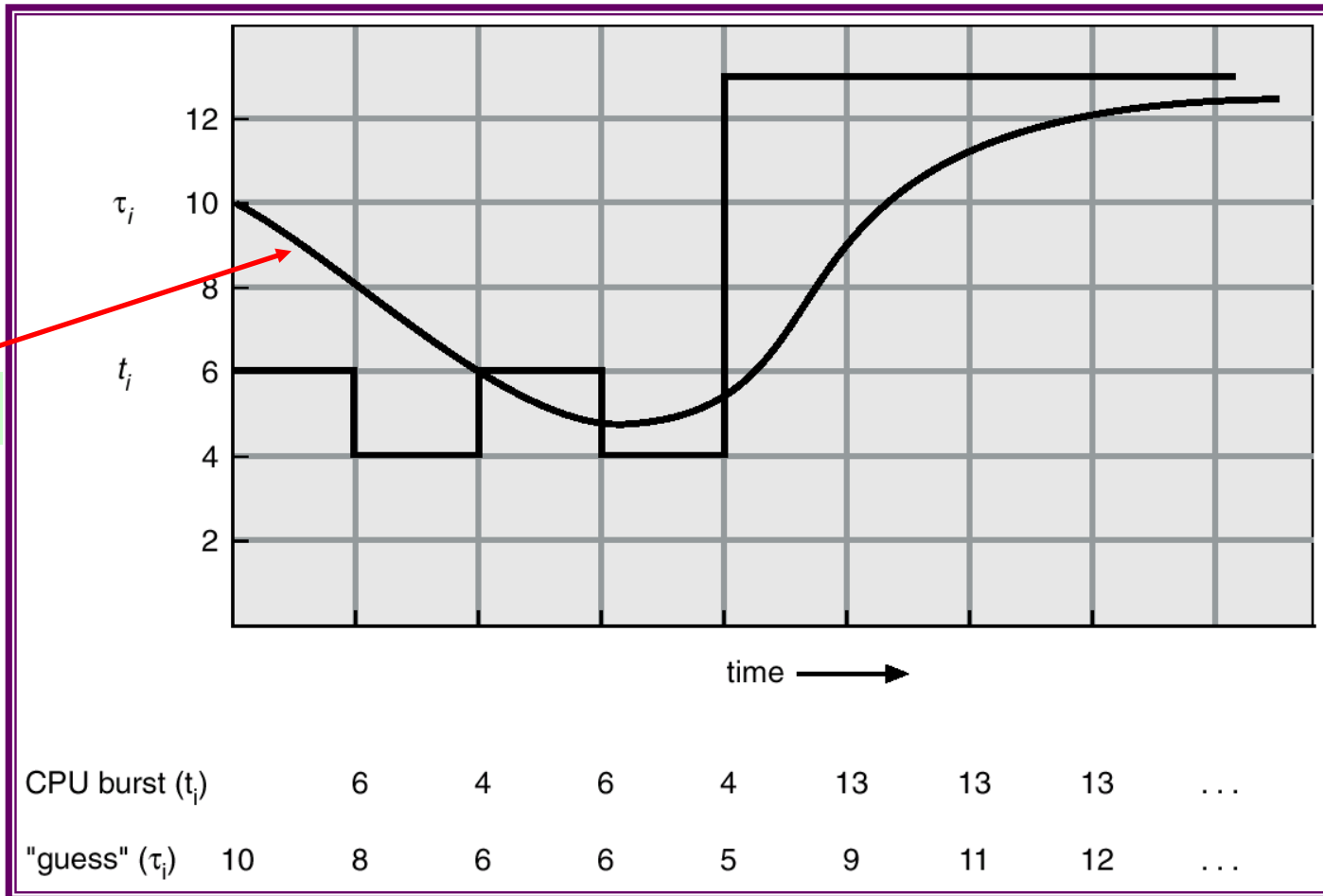
$$\begin{aligned}\tau_{n+1} = & \alpha t_n + (1 - \alpha) \alpha t_{n-1} + \dots \\ & + (1 - \alpha)^j \alpha t_{n-j} + \dots \\ & + (1 - \alpha)^n \alpha t_0 + (1 - \alpha)^{n+1} \tau_0\end{aligned}$$

■ Ako su oba α i $(1 - \alpha)$ manje od ili jednako 1,

☞ **svaki sledeći termin**

☞ **je manje težine od prethodnika**

Procena trajanja sledećeg CPU ciklusa



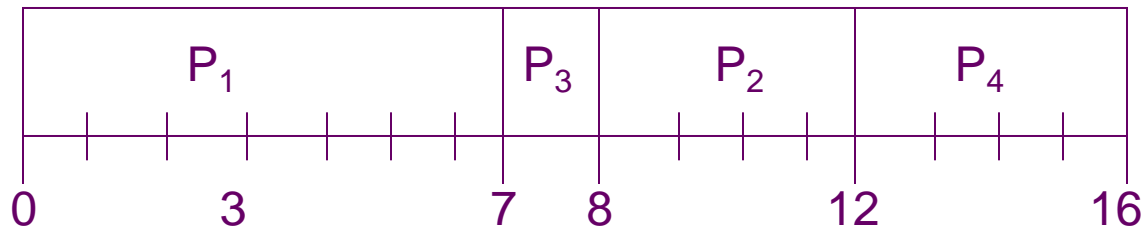
guess

Primer SJF bez pretpražnjenja

- **ново raspoređivanje: samo kad proces završi**

<u>Proces</u>	<u>Vreme nailaska</u>	<u>Vreme izvršavanja</u>
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- **SJF (non-preemptive)**



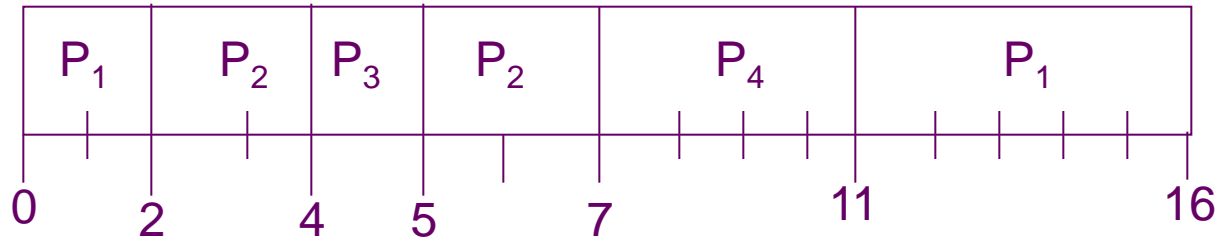
- **Srednje vreme čekanja** = $(0 + 6 + 3 + 7)/4 = 4$
 - ☞ (gleda se vreme nailaska)
 - ☞ za P_2 i P_4 , se primenjuje algoritam FIFO

Primer SJF sa pretpražnjenjem (SRFT)

- ❑ **ново raspoređivanje: kad proces završi i kad naidje novi proces**
- ❑ **pogledati u ready queue: svaki novi proces izaziva novo raspoređivanje**

Proces	<u>Vreme nailaska</u>	<u>Vreme izvršavanja</u>
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

■ SJF (sa pretpražnjenjem)




- **srednje vreme čekanja** = $(9 + 1 + 0 + 2)/4 = 3$
- **pogledati u ready queue: svaki novi proces izaziva novo raspoređivanje**

Prioriteno raspoređivanje

- **Prioritet** (ceo broj) se dodeljuje svakom procesu
- CPU alokira proces sa najvećim prioritetom (manji broj \equiv veći prioritet)
 - ☞ sa pretpražnjenjem
 - ☞ bez pretpražnjenja
- SJF je prioriteno raspoređivanje
 - ☞ gde je prioritet obrnuto proporcionalna funkcija od dužine trajanja sledećeg CPU ciklusa
- **Problem \equiv zakucavanje**
 - ☞ proces niskog prioriteta se možda nikad ne izvrši
- Rešenje \equiv **Aging**
 - ☞ sa vremenom čekanja raste prioritet procesa

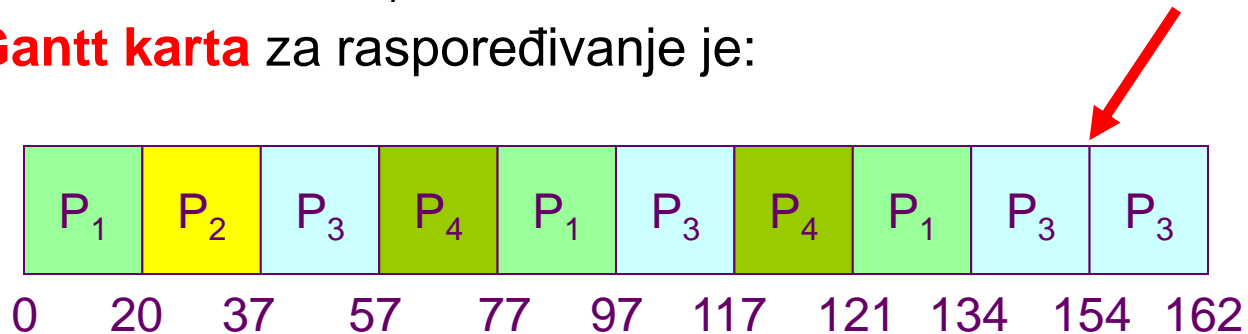
Round Robin (RR)

- **Svaki proces ima malu količinu vremena (vremenski kvantum)**
 - ☞ obično 10-100 milisekundi
- **Kada ovo vreme istekne**
 - ☞ proces se prekida
 -  i
 - ☞ ubacuje na kraju reda čekanja
- Ako imamo
- **n procesa u redu čekanja** i
- **ako je vremenski kvantum q, onda:**
 - ☞ svakom procesu pripada procesor u odnosu **1/n**
 - ☞ nijedan proces ne čeka više od **(n-1)*q** vremena na sledeći vremenski kvantum.
- **Performanse**
 - ☞ **veći q** ⇒ FIFO
 - ☞ **manji q** ⇒ maksimalno ravnomerna raspodela CPU, ali i **veoma često prebacivanje konteksta** između procesa, a to znači veliki gubitak korisnog vremena .

Primer RR gde je vremenskim kvantum = 20

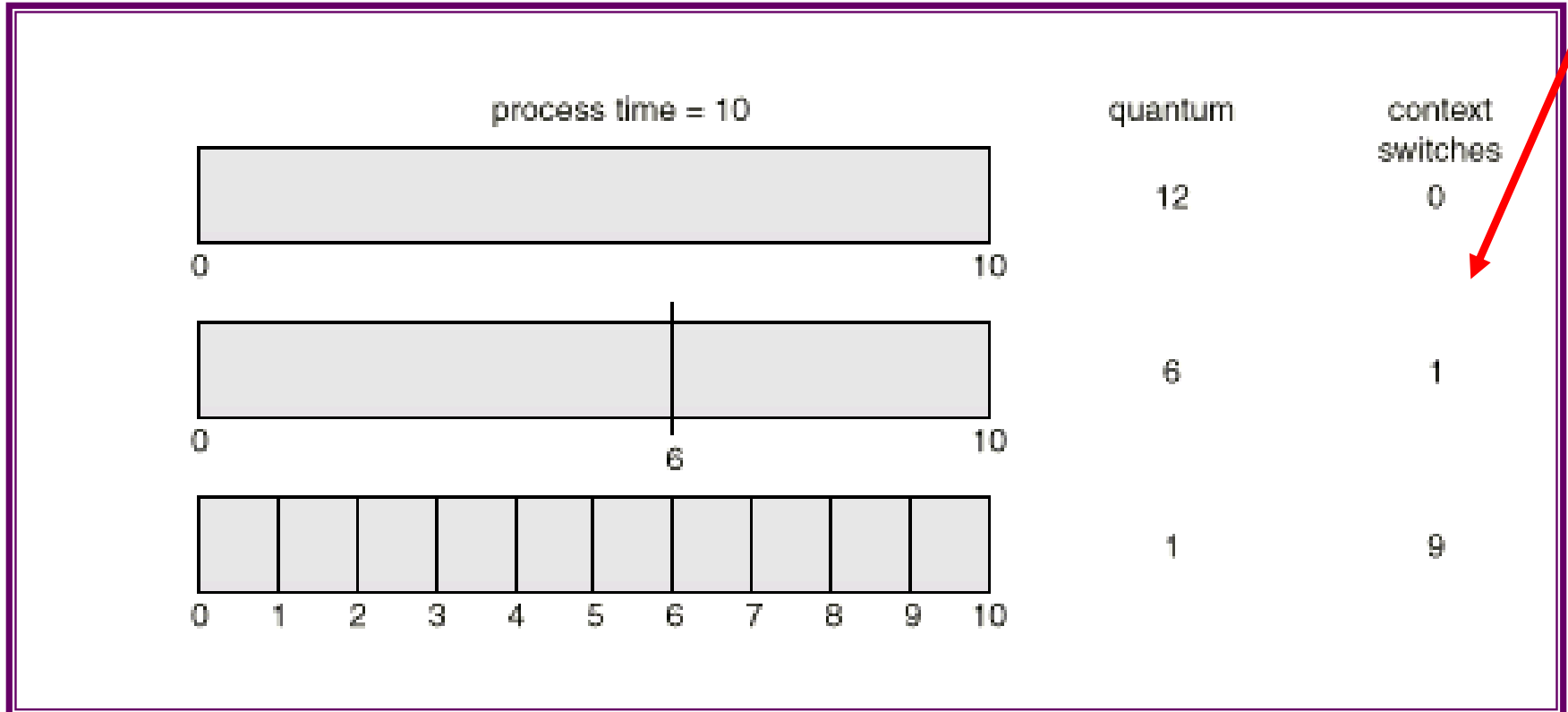
<u>Proces</u>	<u>Vreme izvršavanja</u>
P_1	53
P_2	17
P_3	68
P_4	24

- **Gantt karta** za raspoređivanje je:

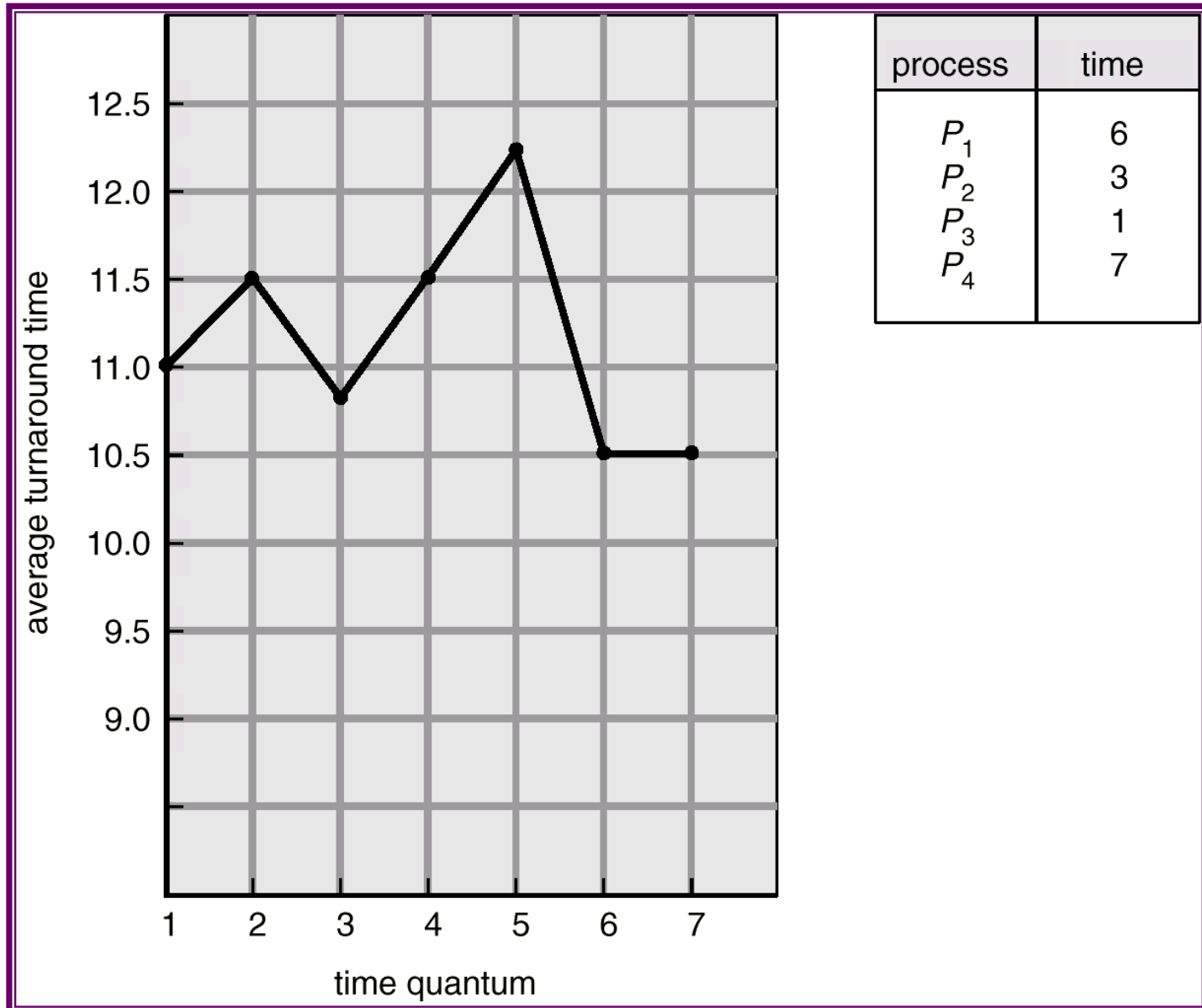


- Karakteristično,
 - ☞ **veće srednje vreme završetka procesa u odnosu na SJF,**
 - ☞ **ali i bolje vreme odziva**

Time Quantum and Context Switch Time



Turnaround Time Varies With The Time Quantum



Raspoređivanje u više redova

- **Red čekanja je podeljen u različite redove:**

- ☞ **foreground** (interactive)

- ☞ **background** (batch)

- **Svaki red čekanja ima poseban algoritam za raspoređivanje:**

- ☞ **foreground – RR**

- ☞ **background – FCFS**

- **Raspoređivanje između redova može biti:**

- ☞ **Prioriteno raspoređivanje;**

- 📄 (npr., opslužuje sve iz prvog plana a onda iz drugog plana).

- 📄 Mogućnost **zakucavanja**.

- ☞ **Time Slice (Deo vremena)**

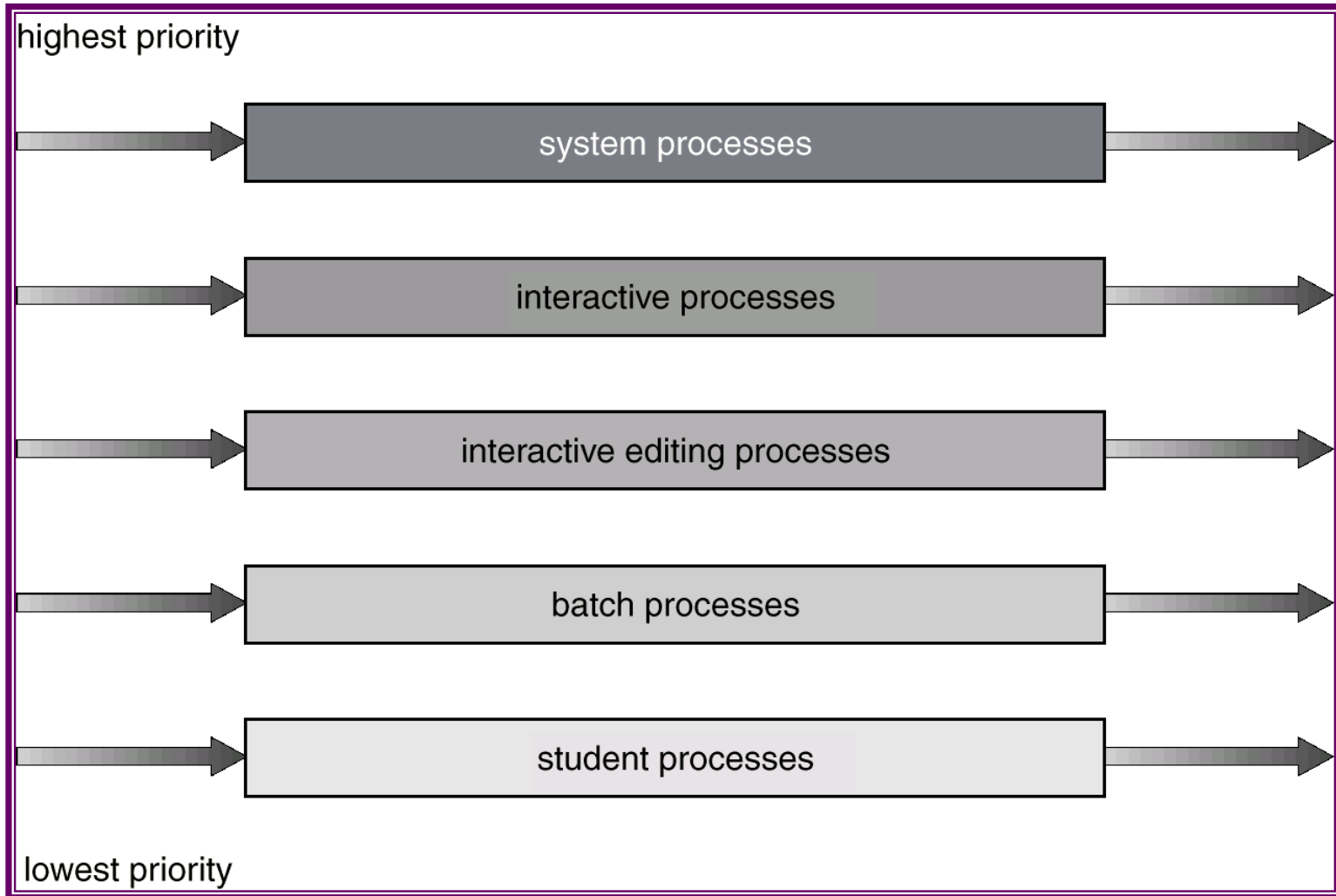
- 📄 **svaki red dobija procesor na neko vreme**

- 📄 **a u tom vremenu selektuju se procesi iz tog reda;**

- 📄 npr., 80% za interaktivno red (RR)

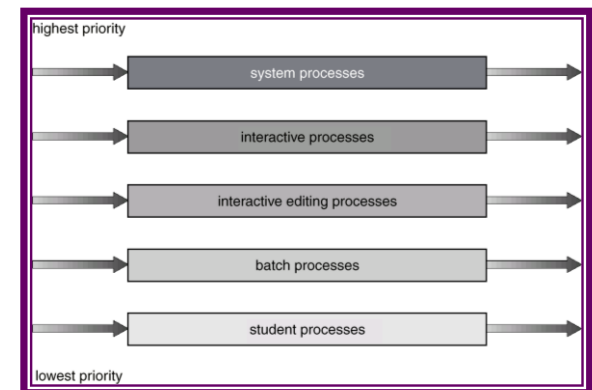
- 📄 20% za pozadinski red (FCFS)

Raspoređivanje u više redova



Povratna sprega između redova čekanja

- Proces se može pomerati između različitih redova;
 - ☞ aging može biti implementiran na sledeći način:
- **Multilevel-feedback-queue** raspoređivač je definisan sledećim parametrima:
 - ☞ broj redova
 - ☞ raspoređivanje algoritama za svaki red
 - ☞ metod za određivanje kada proces može da pređe u red višeg prioriteta
 - ☞ metod za određivanje kada proces može da pređe u red nižeg prioriteta
 - ☞ metod koji određuje u koji će red proces ući po kreiranju



Primer Multilevel Feedback Queue

■ Tri reda:

- ☞ Q_0 – vremenski kvantum 8 milisekunde (RR)
- ☞ Q_1 – vremenski kvantum 16 milisekunde (RR)
- ☞ Q_2 – FCFS

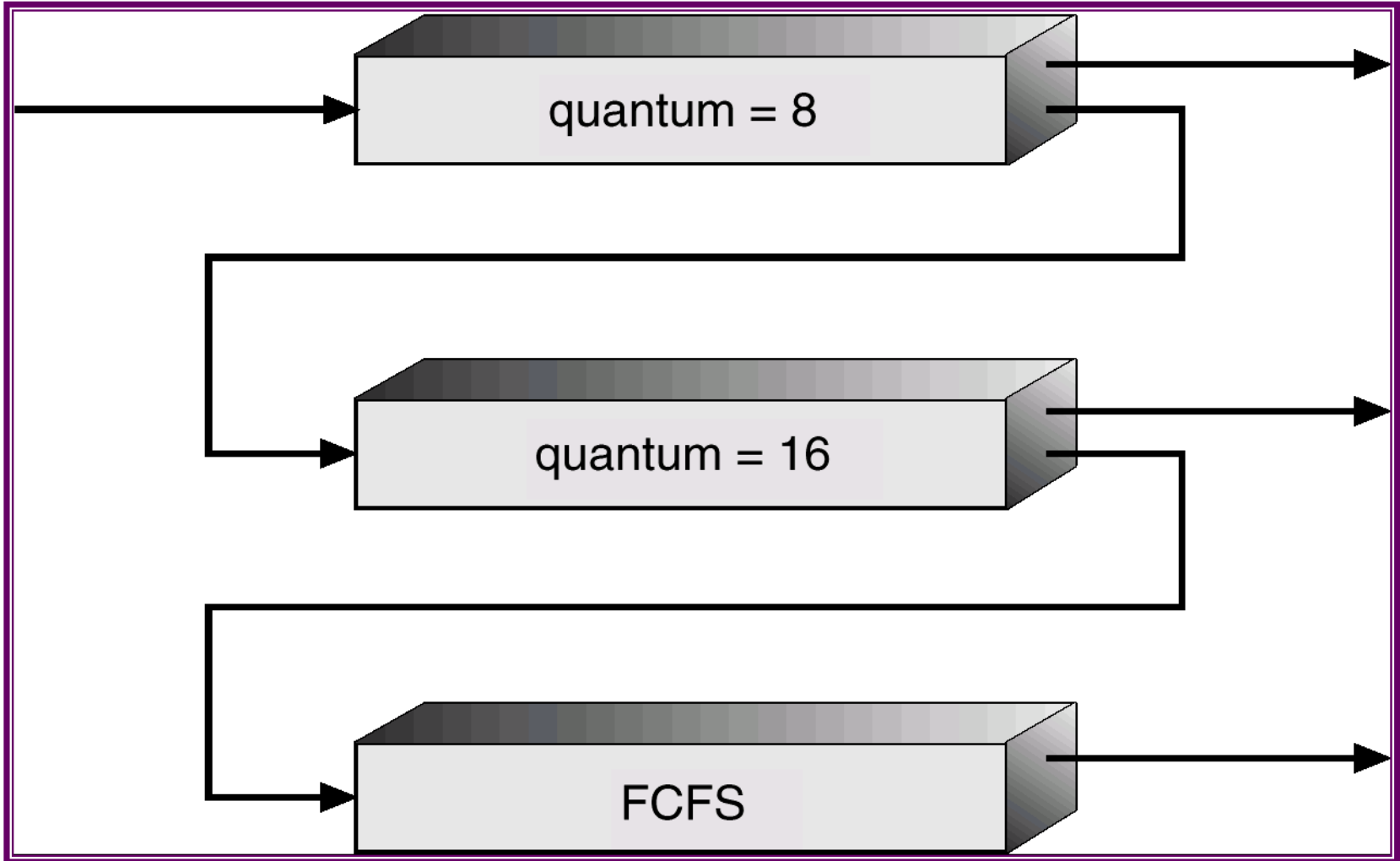
■ Raspoređivanje

- ☞ **Novi posao ulazi u red Q_0** koji je serviran uz pomoć FCFS.
- ☞ Kada dobije CPU, posao ima 8 milisekunde.
- ☞ Ako nije završio za 8 milisekunde, **posao se pomera u red Q_1**

- ☞ **U red Q_1** posao je opet serviran FCFS i dobija 16 dodatnih milisekundi

- ☞ **Ako još uvek nije završio, on je preempted i pomera se u red Q_2 .**

Multilevel Feedback Queues-example



kernel mode priorities

priority level

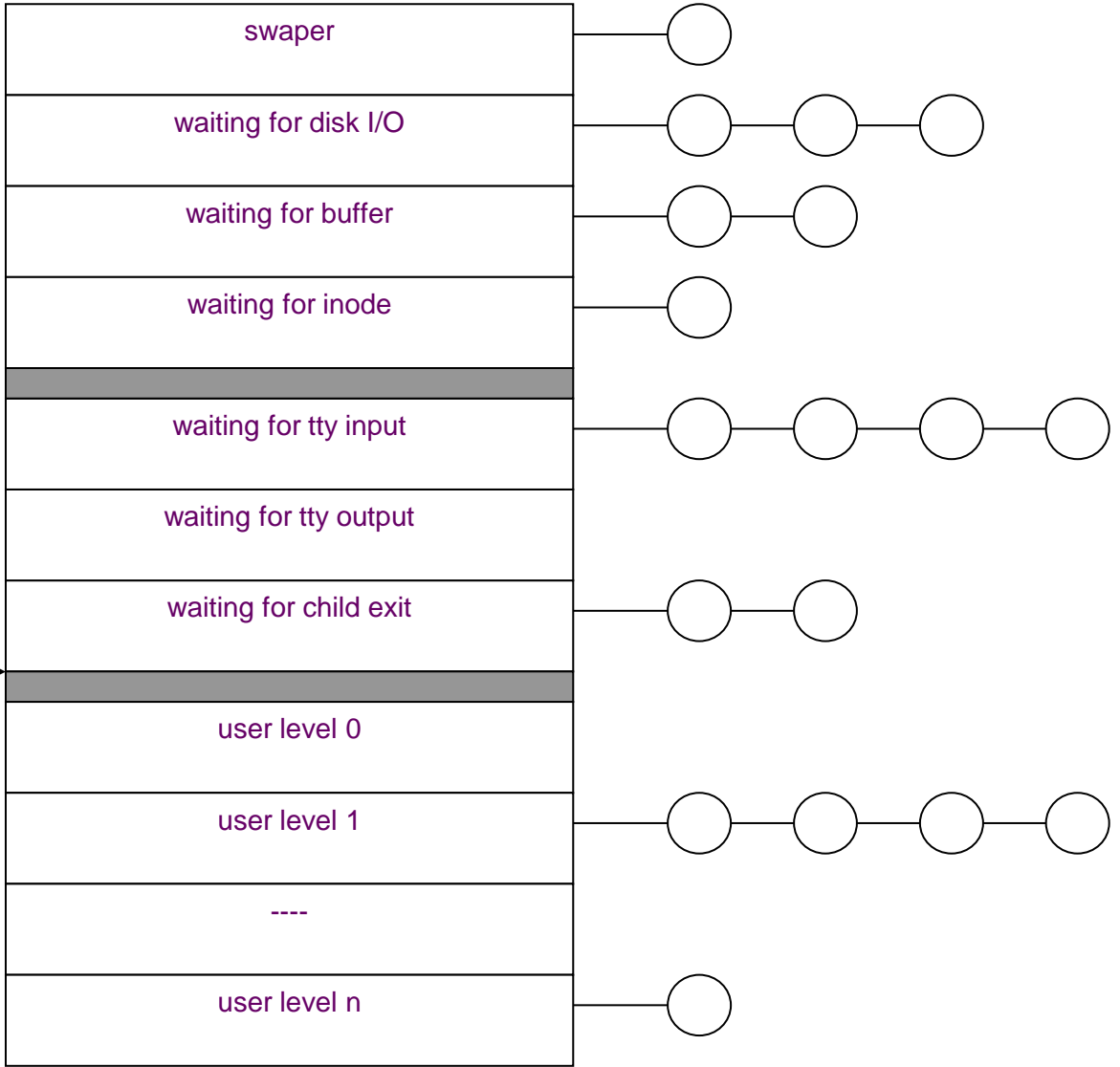
processes

not interruptable

interruptable

threshold priority

user mode priorities



Višeprocorsko raspoređivanje

- CPU raspoređivanje je kompleksnije kad je dostupno više CPU-a
- **SMP: Homogeni procesori su u SMP**
 - ☞ Svi CPU su identični
 - ☞ Pitanje je?
 - ☞ jedan isti ready queue (za sve procesore)
 - ☞ ili
 - ☞ poseban ready queue (za svaki CPU)
- **Load sharing**
 - ☞ Svaki CPU = poseban ready queue,
 - ☞ jedan CPU može biti neaktivan
 - ☞ drugi veoma zauzet=>isti ready queue
 - ☞ U istoj oblasti podataka, svaki CPU mora biti programiran veoma pažljivo
- **Asimetrično multiprocesiranje:**
 - ☞ svaki CPU ima specifičan cilj (jedan od njih je master)
 - ☞ samo jedan procesor ima pristup strukturama sistema podataka,
 - ☞ olakšava potrebu za deljenjem podataka

Rapoređivanje u realnom vremenu

■ **Hard real-time systems:**

- ☞ zahteva da izvršavaju kritične poslove
- ☞ u **garantovanom vremenskom roku**
- ☞ **Samo u sistemima bez diskova i virtuelne memorije**

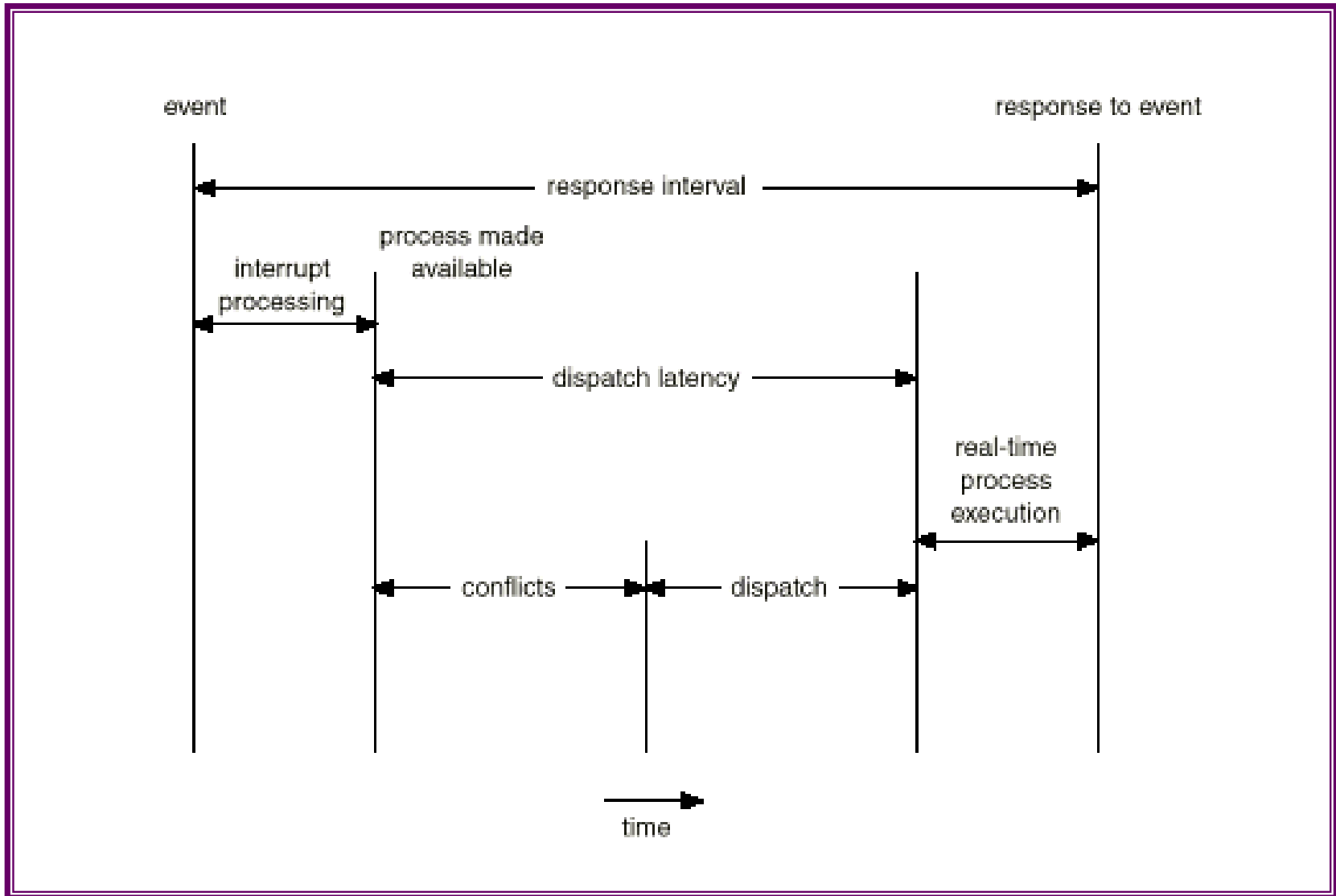
■ **Soft real-time systems:**

- ☞ zahtevaju da im se
- ☞ **kritični procesi izvršavaju prioritetnije**
- ☞ **nego drugi**

■ Zahteva prioritarno raspoređivanje

■ **Dispečersko kašnjenje mora da bude veoma malo**

Vreme za dispečer (Dispatch Latency)



Ispitivanje algoritama

■ Simulacija: Deterministic modeling:

- ☞ uzima poseban predeterminisan **workload**
- ☞ i
- ☞ definiše performanse svakog algoritma
- ☞ za taj workload

■ Modeli za redove:

- ☞ Broj redova, prioritet, pravila
- ☞ Dužina
- ☞ Nailazak procesa
- ☞ Algoritam

■ Implementacija

Procena CPU raspoređivanja simulacijom

