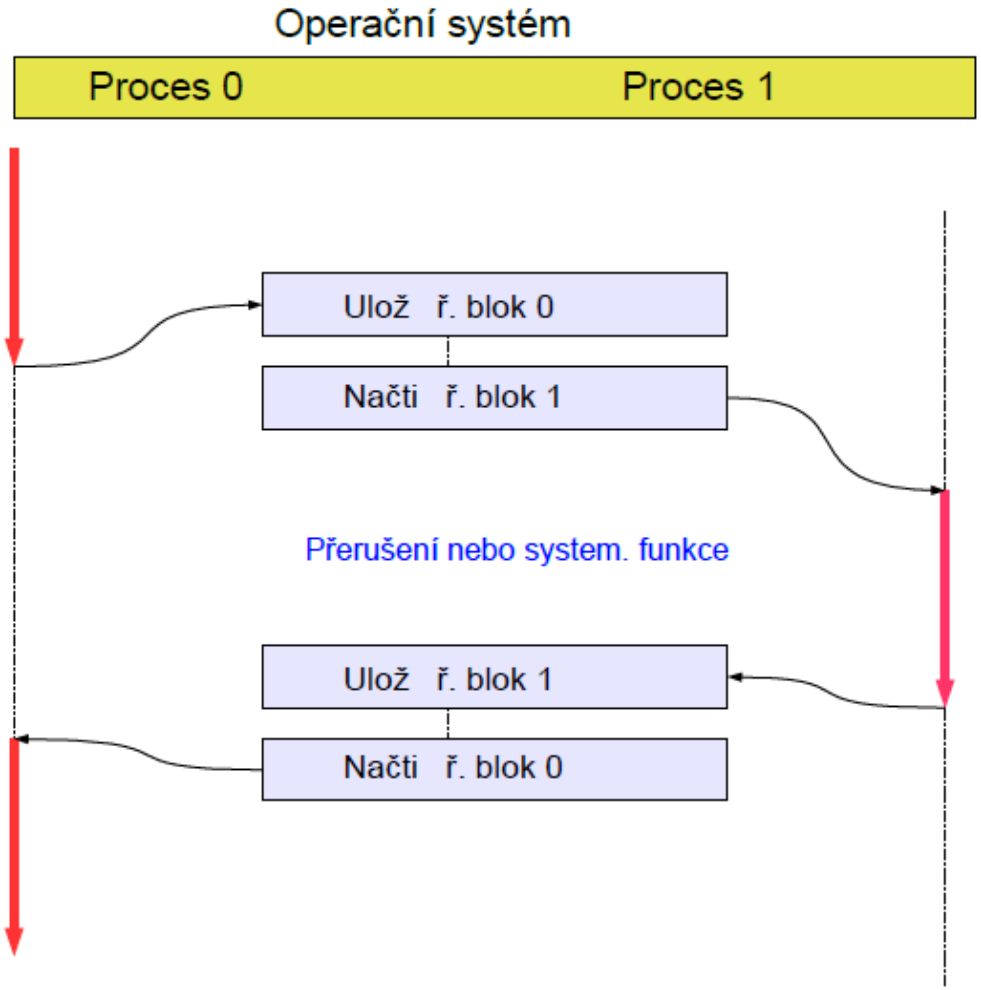
Operační systémy

3. Přepínání kontextu, plánovače OS a plánování CPU

3. ročník

Přepínání kontextu



Přepínání kontextu

- Operace přepnutí z jednoho běžícího procesu na druhý
  - Opakuje se mnohokrát za sekundu
    - Vytváří iluzi paralelního běhu
- Podporují všechny multitaskingové OS
- Dochází k němu i při obsluze přerušení nebo změně režimu (privilegovaný vs. uživatelský)
- Změnu provádí dispatcher na základě scheduleru
  - Plánování CPU -> plánovací algoritmy -> fronty

Přepínání kontextu

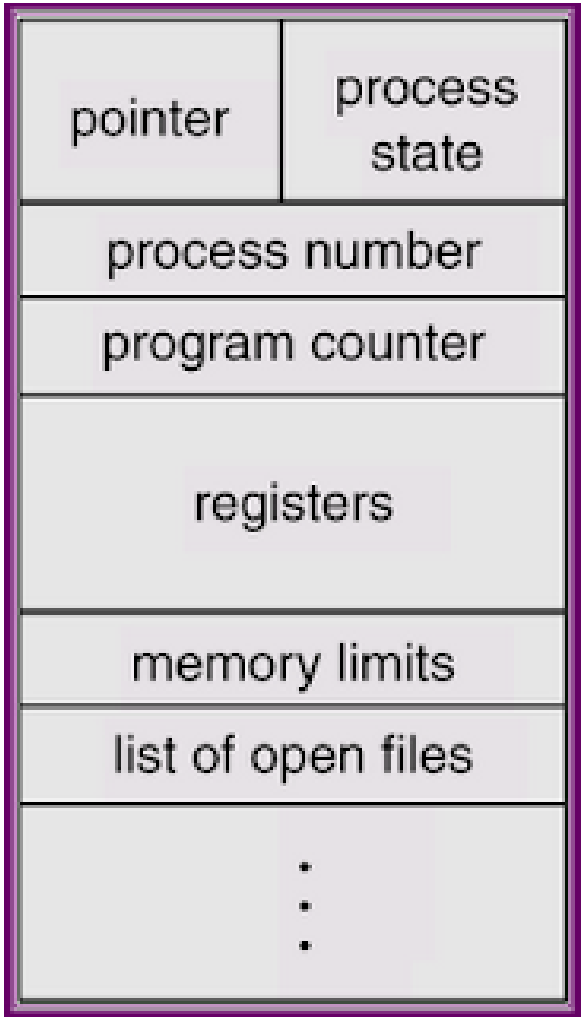
- HW varianta
  - Rychlejší
  - Problém se správným uložením všech registrů
- SW varianta
  - Nejčastější
  - Pomalejší -> vyšší režie
- Př.: Spočtěte dobu přepnutí kontextu, když víte, že proces běží 3ms a dispatcher zabere 10% celkového času. Nakreslete.
  - 3ms … 90%
  - Xms … 10%

PCB – Process Control Block

- Řídící blok procesu
  - Task Control Block / Task Structure
- Dynamická datová struktura obsahující nejnutnější informace pro správu a běh procesu
- Každý proces má svou PCB
- Umístěna v privilegovaném režimu
  - Chráněna před přístupem uživatelů a dalšími procesy

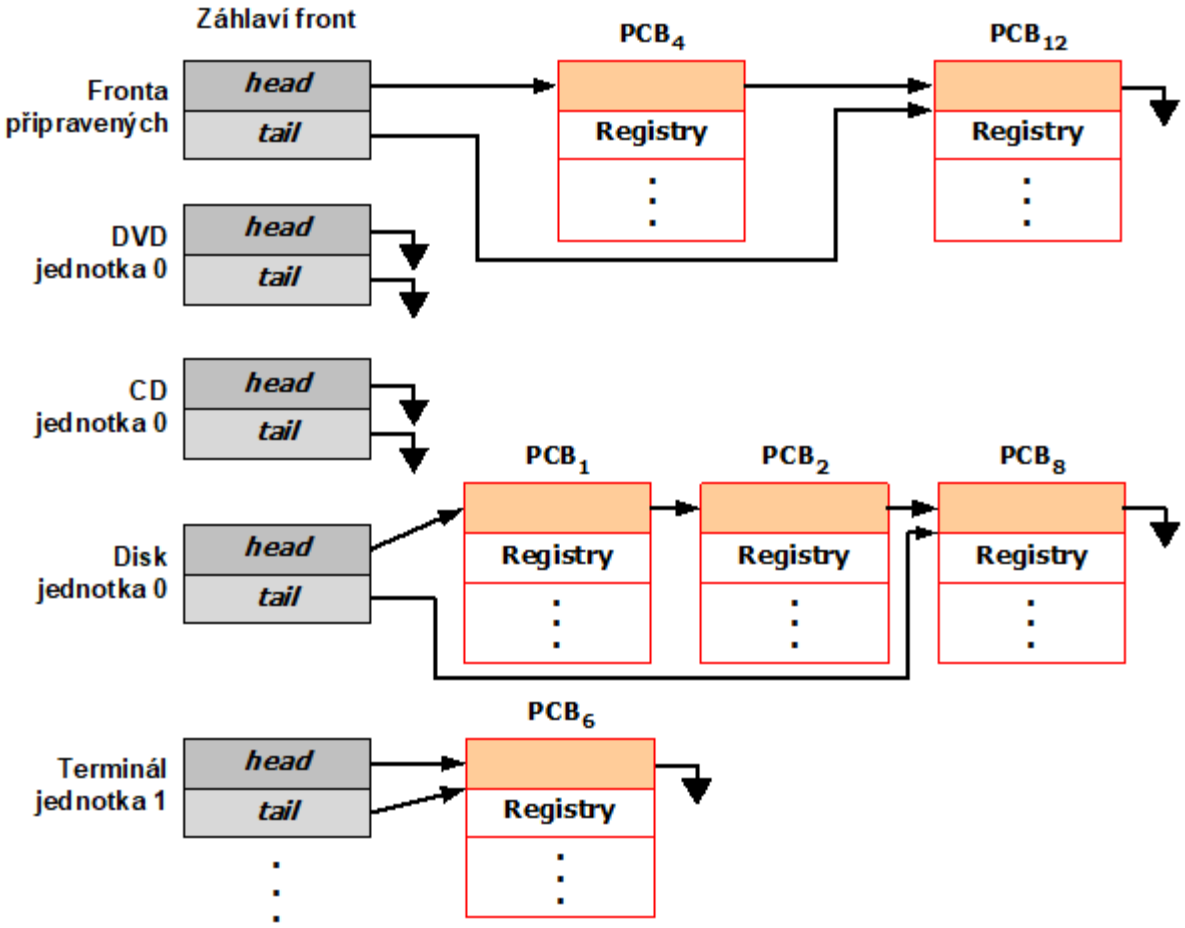
PCB – Process Control Block

- PID, UID
- Kopie registrů pro jejich pozdější načtení (EIP, ESP)
- Priorita
- Účtovací informace
  - Doba běhu, poslední spuštění
- Alokovaná IO zařízení

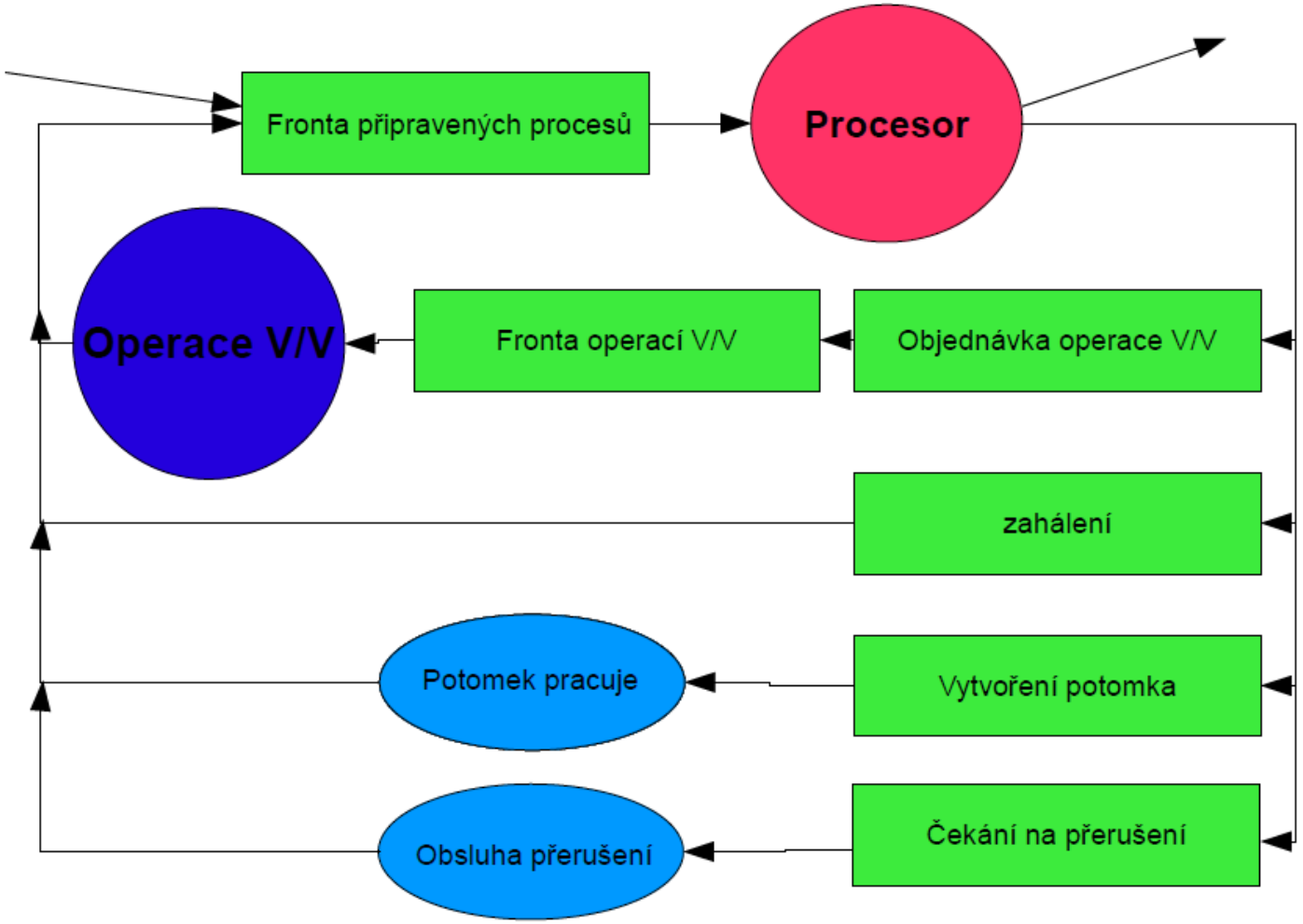


Fronty procesů

- Fronta ukazatelů na první a poslední PCB
- Procesy mezi fronty migrují
- Synchronizace a urychlení práce



Dispatcher a dlouhodobý plánovač



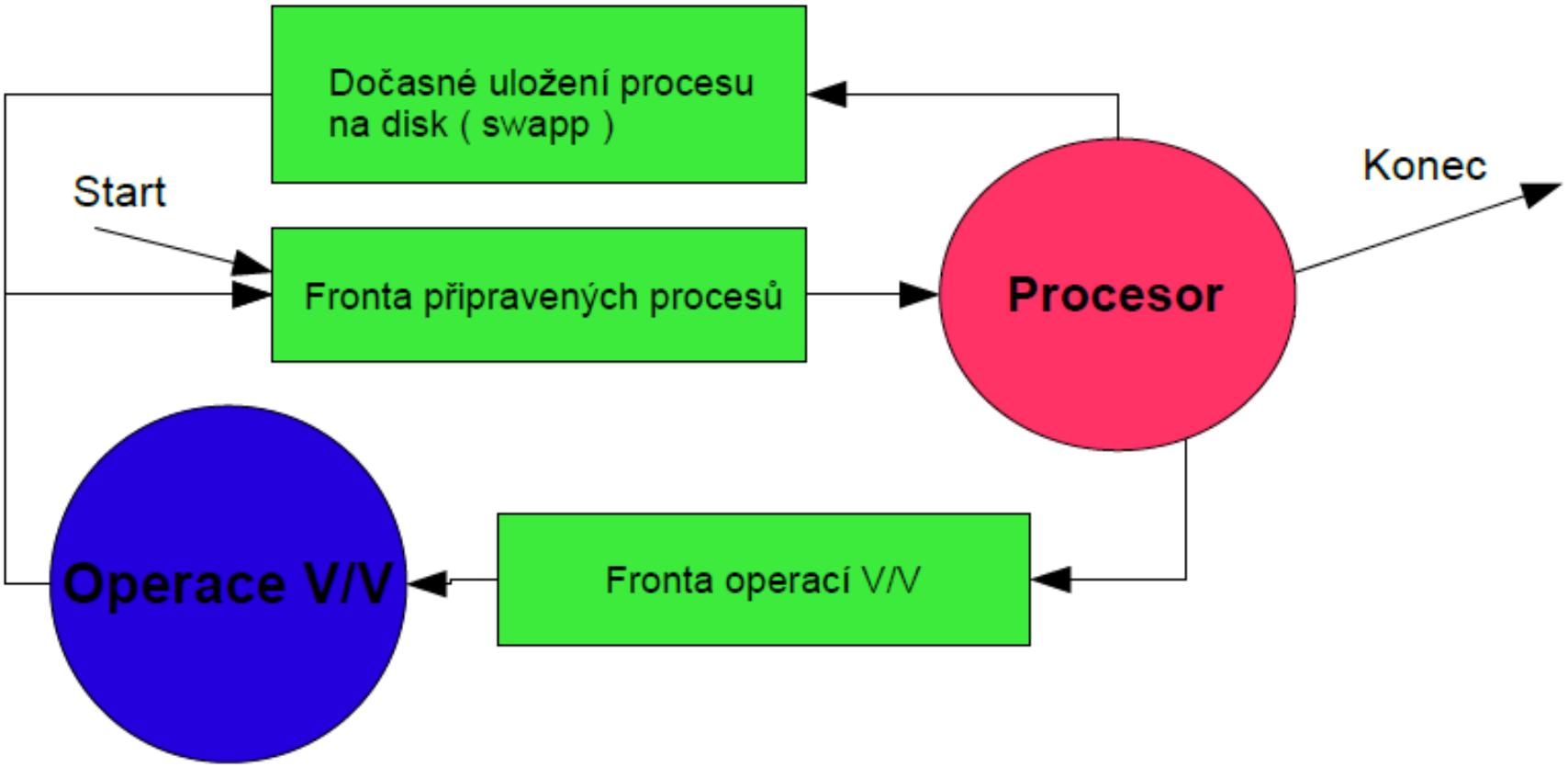
Krátkodobý plánovač

- Short Term, operační plánovač, plánování procesoru, dispatcher
- Vybírá, který proces bude využívat uvolněný CPU -> extrémně rychlý
- 2 základní fronty
  - Ready
  - IO Event

Dlouhodobý plánovač

- Long Term, plánovač úloh (job scheduler), strategický plánovač
- Aktivován zřídka
  - Sekund až minuty
  - Nemusí být rychlý
- Vhodná kombinace několika úloh náročných na IO operace a CPU
  - Dávkové zpracování
- V interaktivních systémech se prakticky nepoužívá a degraduje na přímé předání práce krátkodobému plánovači

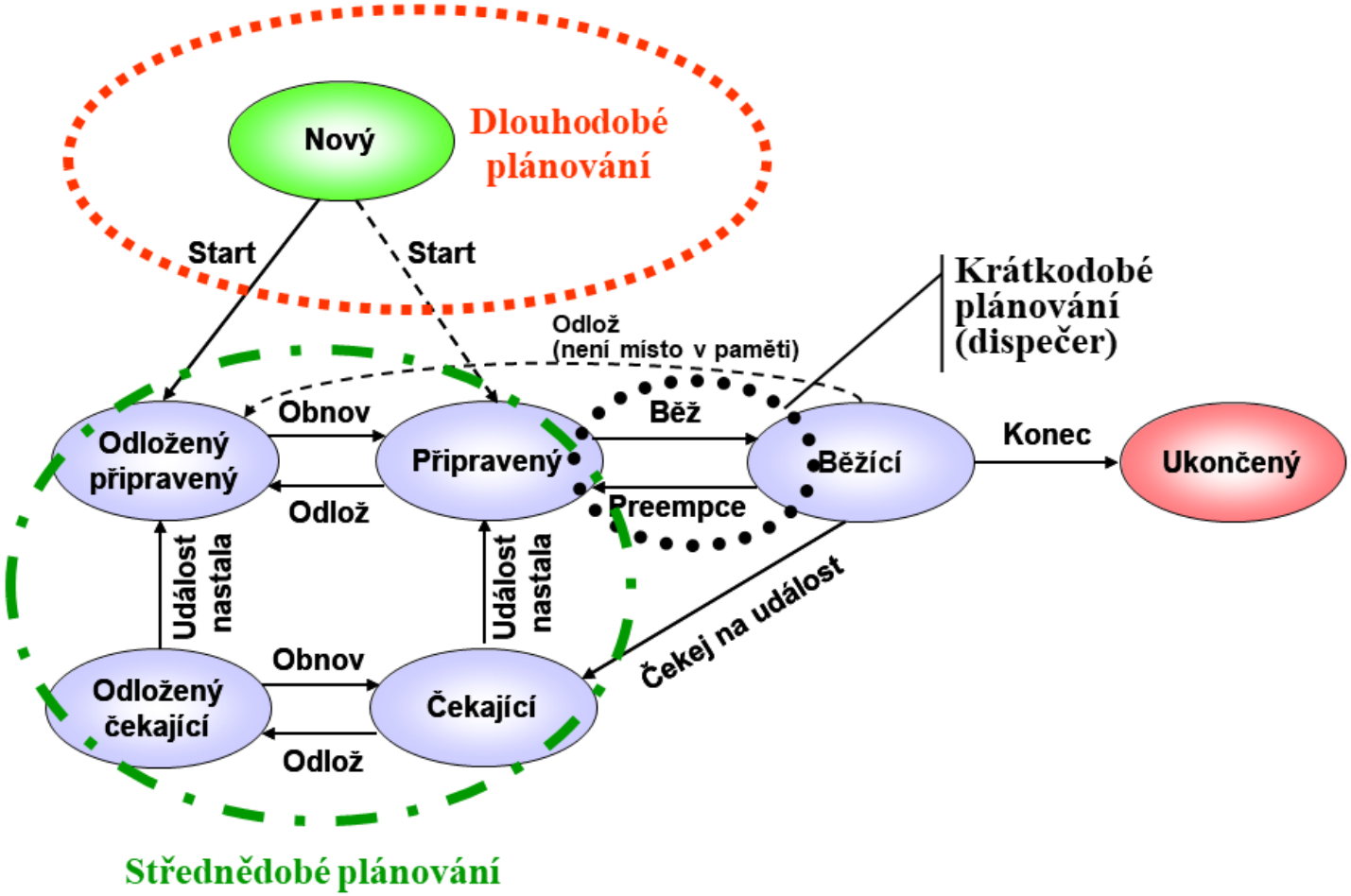
Střednědobý plánovač



Střednědobý plánovač

- Mid Term, taktický plánovač
- Vybírá, který proces odloží z OP na swap oddíl a naopak
  - Swap out / Swap in
  - Spolupracuje se správcem hlavní paměti
- Dva nové stavy
  - Swap Waiting
  - Swap Ready

Životní fáze procesu – 7 stavů



Plánování CPU

- Scheduler se rozhoduje, kterému procesu přidělí procesor na základě:
  a) Ukončení procesu
  b) Změně stavu procesu z Run do Ready
  c) Změně stavu procesu z Run do Wait
  d) Změně stavu procesu z Wait do Ready

- Preemptivní
  - a, b, d

- Nepreemptivní
  - a, c, d

Cíle plánování CPU

- Využití CPU
  - Maximalizace kontinuální činnosti CPU
- Propustnost
  - Maximalizace přirozeně ukončených procesů za jednotku času
- Doba čekání
  - Minimalizace doby čekání procesu na CPU
- Doba obrátky
  - Minimalizace potřebné doby pro provedení konkrétního procesu
- Doba odpovědi
  - Minimalizace doby, která uběhne od vyvolání požadavku na spuštění procesu po jeho první reakci (např. výpis na terminál)

Preemptivní vs. nepreemptivní

- Preemptivní
  - OS má plnou kontrolu nad procesem
    - Kdykoliv může odebrat CPU
    - Kontrola také nad všemi přidělenými prostředky
    - Algoritmy: SRTF, PS, RR, MFQS
  - Většinou po uplynutí přidělené doby (TQ)
    - Interní časovač -> přerušení
  - Složitější implementace OS
  - Nutnost HW podpory v CPU
  - Windows NT (32bit), Linux, Mac OS X, Unix
- Nepreemptivní
  - OS nemá plnou kontrolu nad procesem
    - Proces využívá CPU, jak dlouho potřebuje
    - OS nemůže odebrat CPU procesu
    - Problém s chybným programem -> nevrátí řízení OS -> zamrznutí
    - Algoritmy: FCFS, SJF
  - Využití v tzv. uzavřených systémech
    - Všechny procesy jsou předem známé, včetně jejich vlastností
  - Windows 3.x/95/98 (16bit), Mac OS

FCFS – First Come First Served

- Nepreemptivní
- Procesy jsou obsluhovány v pořadí, v jakém přišly do fronty Ready
- Nejjednodušší plánovací algoritmus
- Nevhodné pro OS se sdílením času
  - Vyžadována odpověď v přiměřené době
- Jeden proces může zablokovat ostatní na delší dobu
  - Efekt kolony
- Velká průměrná čekací doba
- Samostatně se nepoužívá
  - V kombinaci s RR -> MFQS

FCFS – First Come First Served

Process	AT	BT	CT	TAT	WT	VT
P0	0	4				
P1	2	7				
P2	5	2				
P3	6	1				
P4	8	3				

- AT = Arrival Time
- BT = Burst Time
- CT = Completion Time
- TAT = Turn Around Time
  - CT–AT = BT + WT
- WT = Waiting Time
  - TAT – BT = CT – AT
- VT = Visiting Time
- RT = Reading Time
  - VT – AT

FCFS – First Come First Served

Process	AT	BT	CT	TAT	WT	VT
P0	0	4				
P1	2	7				
P2	5	2				
P3	6	1				
P4	8	3				

FCFS – First Come First Served

Process	AT	BT	CT	TAT	WT	VT
P0	0	4				
P1	2	7				
P2	5	2				
P3	6	1				
P4	8	3				

FCFS – First Come First Served

Process	AT	BT	CT	TAT	WT	VT
P0	0	4	4	4	0	0
P1	2	7	11	9	2	4
P2	5	2	13	8	6	11
P3	6	1	14	8	7	13
P4	8	3	17	9	6	14

- AVG TAT = 7,6ms | AVG WT = 4,2ms

SJF – Shortest Job First

- Nepreemptivní
  - Shortest Job Next
- Přednost mají ty úlohy u nichž je předpoklad nejkratšího běhu
  - V případě stejného BT je vybrán dřívější proces
- Hrozí hladovění
- Mnohem menší doba čekání než u FCFS

SJF – Shortest Job First

- Nelze implementovat v systémech s krátkodobým plánováním -> dlouhodobé plánování
  - Odhadování délky běhu

- Velmi závislé na dobrém odhadu
  - $t_{n+1} = \alpha t_n + (1-\alpha)t_{n-1}$

Předpokládaná doba pro minulé spuštění.

Odhadovaná délka následujícího běhu.

Skutečná doba při minulém spuštění.

Jak důvěryhodný je odhad <0,1>

SJF – Shortest Job First

Process	AT	BT	CT	TAT	WT	VT
P0	0	4				
P1	2	7				
P2	5	2				
P3	6	1				
P4	8	3				

SJF – Shortest Job First

Process	AT	BT	CT	TAT	WT	VT
P0	0	4	4	4	0	0
P1	2	7	11	9	2	4
P2	5	2	14	9	7	12
P3	6	1	12	6	5	11
P4	8	3	17	9	6	14

- AVG TAT = 7,4ms | AVG WT = 4ms

Shortest Remaining Time First

- Preemptivní SJF -> SRTF
- Pokud má nově příchozí úloha nejkratší BT přeruší a nahradí se aktuálně probíhající proces
- V případě dvou stejných BT vybráno dle FCFS

Shortest Remaining Time First

Process	AT	BT	BTR	CT	TAT	WT	VT
P0	0	7					
P1	1	5					
P2	2	3					
P3	3	1					
P4	4	2					
P5	5	1					

Shortest Remaining Time First

Process	AT	BT	BTR	CT	TAT	WT	VT
P0	0	7	6				0
P1	1	5	4				1
P2	2	3	2				2
P3	3	1	0	4	1	0	3
P4	4	2					
P5	5	1					

Shortest Remaining Time First

Process	AT	BT	BTR	CT	TAT	WT	VT
P0	0	7	6, 0	19	19	12	0
P1	1	5	4, 0	13	12	7	1
P2	2	3	2, 0	6	4	1	2
P3	3	1	0	4	1	0	3
P4	4	2	0	9	5	3	7
P5	5	1	0	7	2	1	6

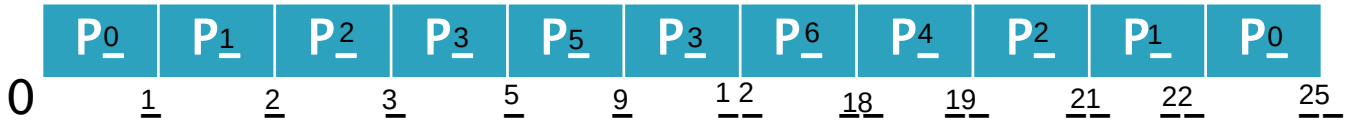
- AVG TAT = 7,1ms | AVG WT = 4ms

PS – Priority Scheduling

- Možnost, jak preemptivního, tak nepreemptivního řešení
- Plánování podle priority
  - Čím nižší číslo u priority, tím vyšší má přednost nebo naopak
  - Proces s nejvyšší prioritou vybrán jako první
    - V případě stejné priority vybráno dle FCFS
- Interní vs. externí priorita
  - Měřitelné hodnoty odvozené ze samotného procesu
    - Paměťové nároky, počet používaných souborů
  - Vnější kritéria
    - Důležitost procesu
- Hrozí umoření / hladovění
  - Procesy s malou prioritou
  - Řešením je umělé navýšení priority po určité době

PS – Priority Scheduling

Process	Priority	AT	BT	BTR	CT	TAT	WT	VT	RT
P0	2	0	4	3, 0	25	25	21	0	0
P1	4	1	2	1, 0	22	21	19	1	0
P2	6	2	3	2, 0	21	19	16	2	0
P3	10	3	5	3, 0	12	9	4	3	0
P4	8	4	1	0	19	15	14	18	14
P5	12	5	4	0	9	4	0	5	0
P6	9	6	6	0	18	12	6	12	6
						15	11.43		



- Pozn.: Čím vyšší číslo u priority, tím má proces vyšší přednost

PS – Priority Scheduling

Process	Priority	AT	BT	BTR	CT	TAT	WT	VT	RT
P0	2	0	4	3,				0	
P1	4	1	2	1,				1	
P2	6	2	3	2,				2	
P3	10	3	5	3,				3	
P4	8	4	1						
P5	12	5	4					5	
P6	9	6	6						

- Pozn.: Čím vyšší číslo u priority, tím má proces vyšší přednost

PS – Priority Scheduling

Process	Priority	AT	BT	BTR	CT	TAT	WT	VT	RT
P0	2	0	4	3, 0	25	25	21	0	0
P1	4	1	2	1, 0	22	21	19	1	0
P2	6	2	3	2, 0	21	19	16	2	0
P3	10	3	5	3, 0	12	9	4	3	0
P4	8	4	1	0	19	15	14	18	14
P5	12	5	4	0	9	4	0	5	0
P6	9	6	6	0	18	12	6	12	6

15ms 11,43ms

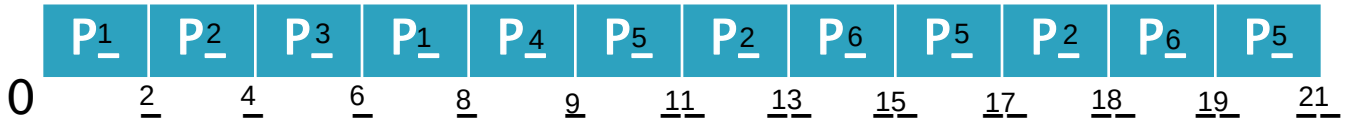
- Pozn.: Čím vyšší číslo u priority, tím má proces vyšší přednost

RR – Round Robin

- Plánování cyklickou obsluhou
  - Rotační plánování
- Vhodný pro OS se sdílením času
- Základ tvoří TQ – Time Quantum
  - Maximálně po tuto dobu může proces využívat CPU
    - Poté zařazen na konec fronty -> fair play planning
  - Skončí-li dříve, nastoupí pustí se dříve další proces
  - Efektivita -> velké vs. malé TQ
- Preemptivní FCFS s TQ

RR – Round Robin ; TQ = 2ms :3

Process	AT	BT	BTR	CT	TAT	WT	VT	RT
P1	0	4	2, 0	8				
P2	1	5	3, 1, 0					
P3	2	2	0	6				
P4	3	1	0	9				
P5	4	6	4, 2					
P6	6	3	1, 0					



- Fronta: P1, P2, P3, P1, P4, P5, P2, P6, P5, P2, P6, P5

RR – Round Robin ; TQ = 2ms

Process	AT	BT	BTR	CT	TAT	WT	VT	RT
P1	0	4	2,				0	
P2	1	5	3,				2	
P3	2	2	0	6	4	2	4	2
P4	3	1						
P5	4	6						
P6	6	3						

- Fronta: ~~P1, P2, P3~~, P1, P4, P5, P2, P6,

RR – Round Robin ; TQ = 2ms

Process	AT	BT	BTR	CT	TAT	WT	VT	RT
P1	0	4	2, 0	8	8	4	0	0
P2	1	5	3,				2	
P3	2	2	0	6	4	2	4	2
P4	3	1	0	9	6	5	8	5
P5	4	6	4,				9	
P6	6	3						

- Fronta: ~~P1, P2, P3, P1, P4, P5,~~ P2, P6, P5,

RR – Round Robin ; TQ = 2ms

Process	AT	BT	BTR	CT	TAT	WT	VT	RT
P1	0	4	2, 0	8	8	4	0	0
P2	1	5	3, 1, 0	18	17	12	2	1
P3	2	2	0	6	4	2	4	2
P4	3	1	0	9	6	5	8	5
P5	4	6	4, 2, 0	21	17	11	9	5
P6	6	3	1, 0	19	13	10	13	7
					10,83ms	7,33ms		

- Fronta: ~~P1, P2, P3, P1, P4, P5, P2, P6, P5, P2, P6, P5~~

Procvičování – PS a RR

- Priority Scheduling
  - Stejné zadání (tabulka), ale obrátit prioritu
    - Čím nižší číslo u priority, tím má proces větší přednost
- Round Robin
  - Stejné zadání (tabulka), ale TQ = 4ms

MQS – Multilevel Queue Scheduling

- Plánování pomocí více front
  - Procesy se mezi frontami nemohou přesouvat
- Rozdělení procesů do skupin
  - Interaktivní procesy -> vyšší priorita
  - Procesy na pozadí -> nižší priorita
  - Každá skupina má různé nároky na dobu odezvy
    - Plánování v rámci front
- Nutnost existence plánování mezi frontami
  a) Podle priority s využitím preempce
     - Pokud není předchozí fronta prázdná, nedostane se proces z nižší fronty k CPU
  b) Časové intervaly pro každou frontu
- Nejvyšší prioritu mají systémové procesy, pak interaktivní, dávkové a nakonec uživatelské

MFQS – Multilevel Feedback Queue Scheduling

- Plánování pomocí více front se zpětnou vazbou
  - Zpětnovazební plánování
  - Možnost přesunu procesů mezi frontami
- Jeden z nejvýznamnějších algoritmů plánování CPU
- Většinou využita kombinace RR-RR-FCFS
  - Procesy v první frontě mají nejkratší TQ
    - Do této fronty vstoupí každý proces
    - V následující frontě pak mají procesy delší TQ
  - Procesy v nižší frontě mohou být vykonány pokud je předchozí fronta prázdná

MFQS – Multilevel Feedback Queue Scheduling

- Velmi komplexní
- Hrozí stárnutí, tzv. Aging
  - Řešením je umělé zvýšení priority
    - Přesun procesu do vyšší fronty
- Proces z vyšší fronty může preemptivně ukončit proces z nižší fronty
  - Platí i pro frontu FCFS

MFQS – Multilevel Feedback Queue Scheduling ; TQ1 = 10ms, TQ2 = 20ms

Process	AT	BT	BTR	CT	TAT	WT	VT	RT
P1	0	12						
P2	8	25						
P3	21	33						
P4	30	2						

- Fronta Q1:
- Fronta Q2:
- Fronta Q3:

MFQS – Multilevel Feedback Queue Scheduling ; TQ1 = 10ms, TQ2 = 20ms

Process	AT	BT	BTR	CT	TAT	WT	VT	RT
P1	0	12	2, 1,				0	0
P2	8	25	15, 0				10	2
P3	21	33						
P4	30	2						

- Fronta Q1: ~~P1, P2,~~ P3,
- Fronta Q2: ~~P1,~~ P2, P1,
- Fronta Q3:

MFQS – Multilevel Feedback Queue Scheduling ; TQ1 = 10ms, TQ2 = 20ms

Process	AT	BT	BTR	CT	TAT	WT	VT	RT
P1	0	12	2, 1,				0	0
P2	8	25	15, 0	48	40	15	10	2
P3	21	33	23,				21	0
P4	30	2	0	33	3	1	31	1

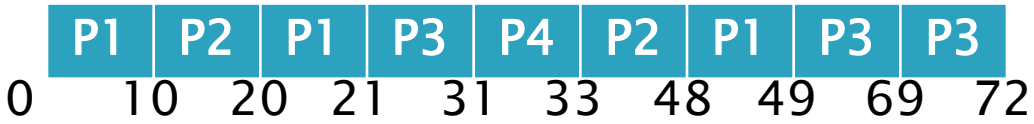
P1	P2	P1	P3	P4	P2	P1	P_	P_

0 10 20 21 31 33 48 __ __ __

- Fronta Q1: ~~P1, P2, P3, P4~~
- Fronta Q2: ~~P1, P2,~~ P1, P3
- Fronta Q3:

MFQS – Multilevel Feedback Queue Scheduling ; TQ1 = 10ms, TQ2 = 20ms

Process	AT	BT	BTR	CT	TAT	WT	VT	RT
P1	0	12	2, 1, 0	49	49	37	0	0
P2	8	25	15, 0	48	40	15	10	2
P3	21	33	23, 3, 0	72	51	18	21	0
P4	30	2	0	33	3	1	31	1



- Fronta Q1: ~~P1, P2, P3, P4~~
- Fronta Q2: ~~P1, P2, P1, P3~~
- Fronta Q3: ~~P3~~

Dodatek

- Guaranteed Scheduling
  - Také znám jako „Fair Share“
  - Při n uživatelích je čas využití CPU rozdělen jako 1/n
- Lottery Scheduling
  - Každému procesu je přidělen tiket
  - Periodické losování
  - CPU je přiděleno tomu procesu, jež má výherní tiket
  - Důležité procesy mohou mít více tiketů
  - Kooperativní procesy si mohou předávat tikety

KONEC

Zdroje

- http://labe.felk.cvut.cz/vyuka/A3B33OSD/Tema-03-ProcesyVlakna-OSD-4.pdf [19. 5. 2020]
- http://labe.felk.cvut.cz/vyuka/A3B33OSD/Tema-04-Planovani-OSD.pdf [21. 5. 2020]
- https://www.youtube.com/watch?v=HIB3hZ-5fHw [21. 5. 2020]
- https://www.youtube.com/watch?v=wDmLvYiXzXI [21. 5. 2020]
- https://www.youtube.com/watch?v=1w9FybdNi_Y [21. 5. 2020]
- http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-sched-mlfq.pdf [21. 5. 2020]
- https://publi.cz/books/11/05.html [21. 5. 2020]