19. Opisz rejestry CPU oraz PSW.

1. Rejestry procesora  
a. licznik programu (PC) - adres rozkazu do pobrania  
b. rejestr rozkazu (IR) - kod rozkazu  
c. rejestr adresowy pamięci (MAR) - adres lokacji  
d. rejestr buforowy pamięci (MBR) - dane do/z pamięci  
e. rejestry PSW - słowo stanu programu, informacje o stanie

2. PSW(bity, flagi)  
a. znak - bit znaku ostatniej operacji  
b. zero - wynik operacji = zero  
c. przeniesienie - przeniesienie w wielokrotnej precyzji  
d. równość - wynik porównania logicznego  
e. przepełnienie  
f. zezwolenie/blokowanie przerwań  
g. nadzorca - tryb systemu lub tryb użytkownika

X20. Wymień kategorie urządzeń we/wy.

✦przeznaczone do odczytu przez człowieka (np. monitor ekranowy, wydruk, dźwięk)

✦przeznaczone do odczytu przez maszynę (np. dyski magnetyczne, taśmy, czujniki w robotach )

✦komunikacyjne (np. modem, karta sieciowa)

X21. Wymień i opisz sposoby realizacji we/wy.

✦programowane -dane są wymieniane między procesorem a modułem we/wy, procesor czeka na zakończenie operacji we/wy

✦sterowane przerwaniami -procesor wydaje operację we/wy i wykonuje dalsze rozkazy do momentu zakończenia operacji we/wy (przerwanie we/wy)

✦bezpośredni dostęp do pamięci (ang. direct memory access -DMA) -moduł we/wy wymienia dane bezpośrednio z pamięcią bez udziału procesora

22. Wymień i opisz metody dostępu do pamięci.

✦dostęp sekwencyjny (ang. sequential)

✦dostęp liniowy blok po bloku wprzód lub w tył

✦czas dostępu zależy od pozycji bloku względem pozycji bieżącej

✦np. taśmy

✦dostęp bezpośredni (ang. direct)

✦każdy blok ma unikalny adres

✦czas dostępu realizowany przez skok do najbliższego otoczenia i sekwencyjne przeszukiwanie

✦np. dysk magnetyczny

✦dostęp swobodny (ang. random)

✦każda adresowalna lokacja w pamięci ma unikatowy, fizycznie wbudowany mechanizm adresowania

✦czas dostępu nie zależy od poprzednich operacji i jest stały

✦np. RAM

✦dostęp skojarzeniowy (ang. associative)

✦dane są lokalizowane raczej na podstawie porównania z ich zawartością niż na podstawie adresu

✦czas dostępu nie zależy od poprzednich operacji i jest stały

✦np. pamięć podręczna (ang. cache)

X23. Wymień rodzaje pamięci ze względu na własności fizyczne.

✦półprzewodnikowa (ang. semiconductor)

✦RAM

✦magnetyczna

✦dysk & taśma

✦magneto-optyczna

✦CD & DVD

✦zanikanie, rozpad (ang. decay)

✦ulotność (ang. volatility )

✦wymazywalność (ang. erasable)

✦zasilanie do utrzymania zawartości

24. Opisz zasadę lokalności odniesień.

✦zasada lokalności odniesień (ang. locality of reference) oznacza, że w czasie wykonania programu odwołania do danych i rozkazów mają tendencję do gromadzenia się

✦przyczyna: programy zwykle zawierają tablice deklaracji zmiennych oraz stałych i wiele pętli iteracyjnych i podprogramów

✦lokalność przestrzenna –grupowanie odniesień do tych samych miejsc w pamięci

✦skłonność do sekwencyjnego sięgania po rozkazy lub dane (np. tablica)

✦lokalność czasowa -skłonność do odwołań do ostatnio wykorzystywanych miejsc w pamięci (np. pętla iteracyjna)

✦operacje na tablicach -dostęp do zgrupowanych słów

✦wykorzystanie zasady lokalności odniesień pozwala na zmniejszenie częstotliwości dostępu

✦przykład: pamięć podręczna (ang. cache, fr. cacher) procesora

25. Opisz działanie pamięci podręcznej.

✦Cache zawiera fragment pamięci głównej

✦Procesor sprawdza czy aktualnie potrzebne do wykonania rozkazu słowo z pamięci jest w cache’u

✦jeśli nie, to blok pamięci o ustalonej liczbie K słów zawierający potrzebne słowo jest ściągnięty do pamięci podręcznej

✦Cache zawiera znaczniki identyfikujące bloki pamięci głównej

26. Opisz sposoby mapowania dla pamięci podręcznej.

Mapowanie bezpośrednie

✦Mapowanie bezpośrednie (ang. direct mapping)

✦Każdy blok w pamięci głównej jest odwzorowywany na tylko jeden możliwy wiersz (ang. line) pamięci

✦tzn. jeśli blok jest w cache’u to tylko w ściśle określonym miejscu

✦Adres jest dzielony na dwie części

✦najmniej znaczących w-bitów identyfikuje jednoznacznie słowo(ang.word) lub bajt w pamięci

✦najbardziej znaczących s-bitów określa jeden z 2^s bloków pamięci

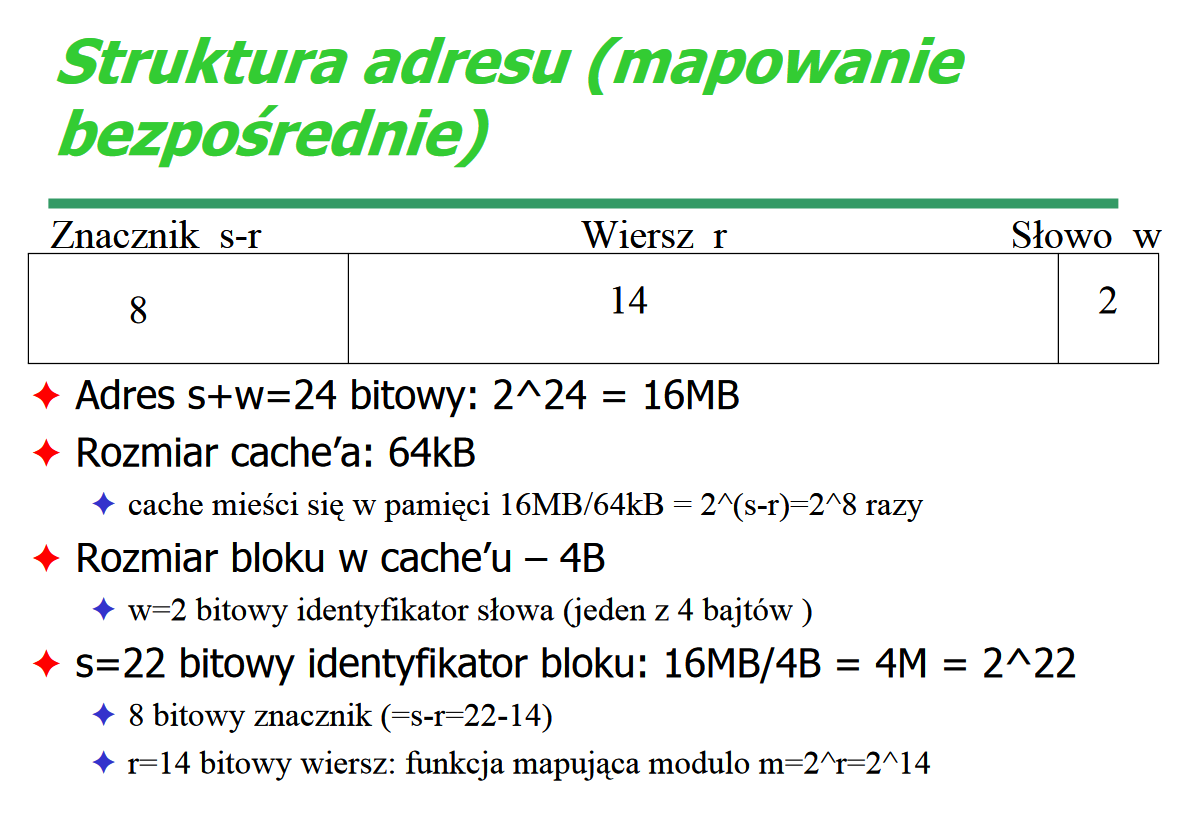
✦najbardziej znaczące bity są dzielone na pole wiersza złożone z r bitów oraz znacznik w postaci s-r bitów (najbardziej znacząca część)

✦Wiersz Przypisane bloki w pamięci głównej

✦0 0, 2^r, 2\*2^r, 3\*2^r,....., 2^s-2^r

✦1 1, 2^r+1, 2\*2^r +1,3\*2^r +1,...,2^s-2^r+1

✦2^r-1 2^r-1, 2\*2^r -1, 3\*2^r -1, 4\*2^r -1,.....,2^s-1



✦Długość adresu= (s + w) bitów

✦Liczba adresowalnych jednostek= 2^(s+w)słów lub bajtów

✦Rozmiar bloku= rozmiar wiersza= 2^w słów lub bajtów

✦Liczba wierszy w cache’u= m = 2^r

✦Liczba bloków w pamięci głównej= 2^(s+ w)/2^w = 2^s

Mapowanie skojarzeniowe

✦Rozmiar znacznika= (s –r) bitów

✦mapowanie skojarzeniowe (ang. associative mapping)

✦Blok pamięci może zostać załadowany do dowolnego wiersza w cache’u

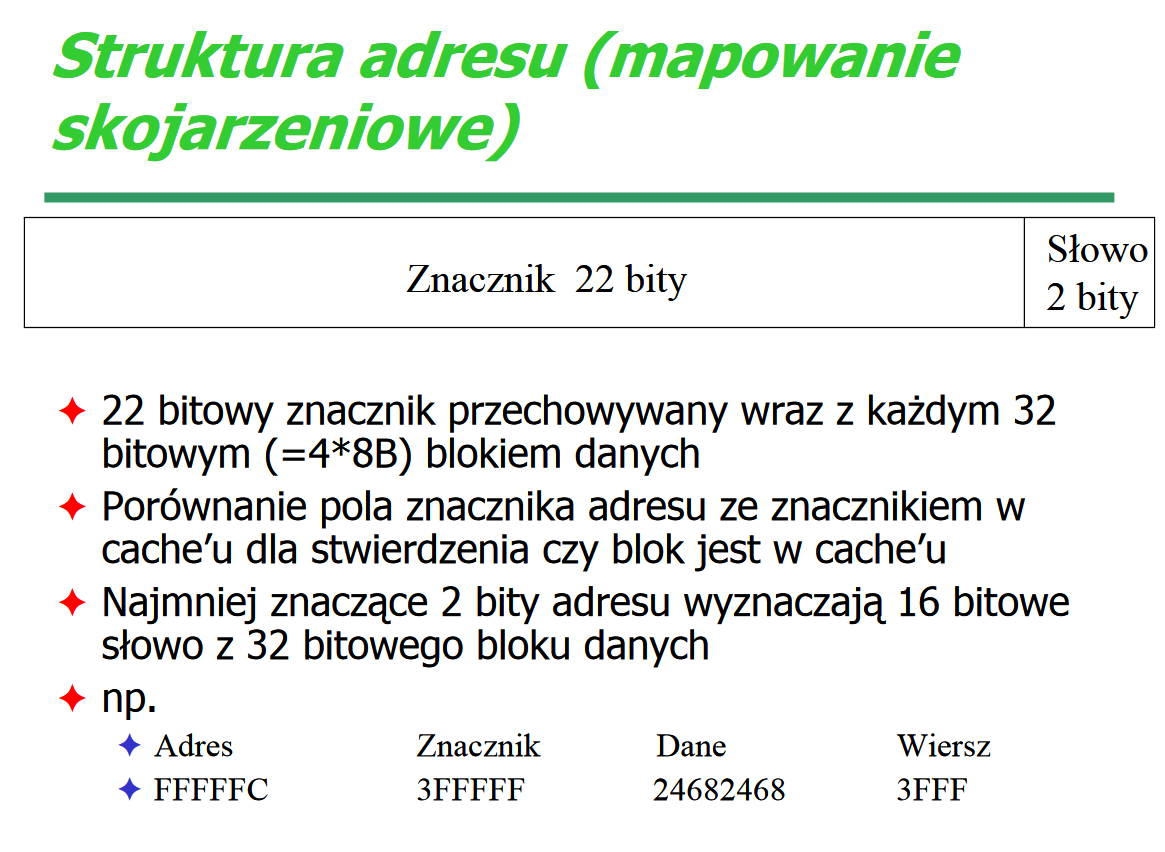
✦Adres dzielimy na dwie części: znacznik i słowo

✦znacznik jednoznacznie określa blok pamięci

✦Aby stwierdzić czy blok jest w cache’u trzeba zbadaćzgodność adresu ze znacznikiem każdego wiersza

✦Kosztowana metoda zwłaszcza gdy rozmiar cache’a jest duży

✦konieczność równoległego badania znaczników wszystkich wierszy wpamięci podręcznej (złożone układy)



✦Długość adresu= (s + w) bitów

✦Liczba jednostek adresowalnych= 2^(s+w)słów lub bajtów

✦Rozmiar bloku= rozmiar wiersza= 2^w słów lub bajtów

✦Liczba bloków w pamięci= 2^(s+w)/2^w = 2^s

✦Liczba wierszy w pamięci podręcznej=nieokreślona

✦Rozmiar znacznika= s bitów

Mapowanie sekcyjno-skojarzeniowe

✦k-drożne mapowanie sekcyjno-skojarzeniowe (ang. k-way set associative mapping)

✦Cache jest dzielony na v sekcji (ang. sets)

✦Każda sekcja składa się z k wierszy

✦Dany blok B może zostać odwzorowany na dowolny wiersz jakiejś sekcji i

✦np.jeśli sekcja ma 2wiersze

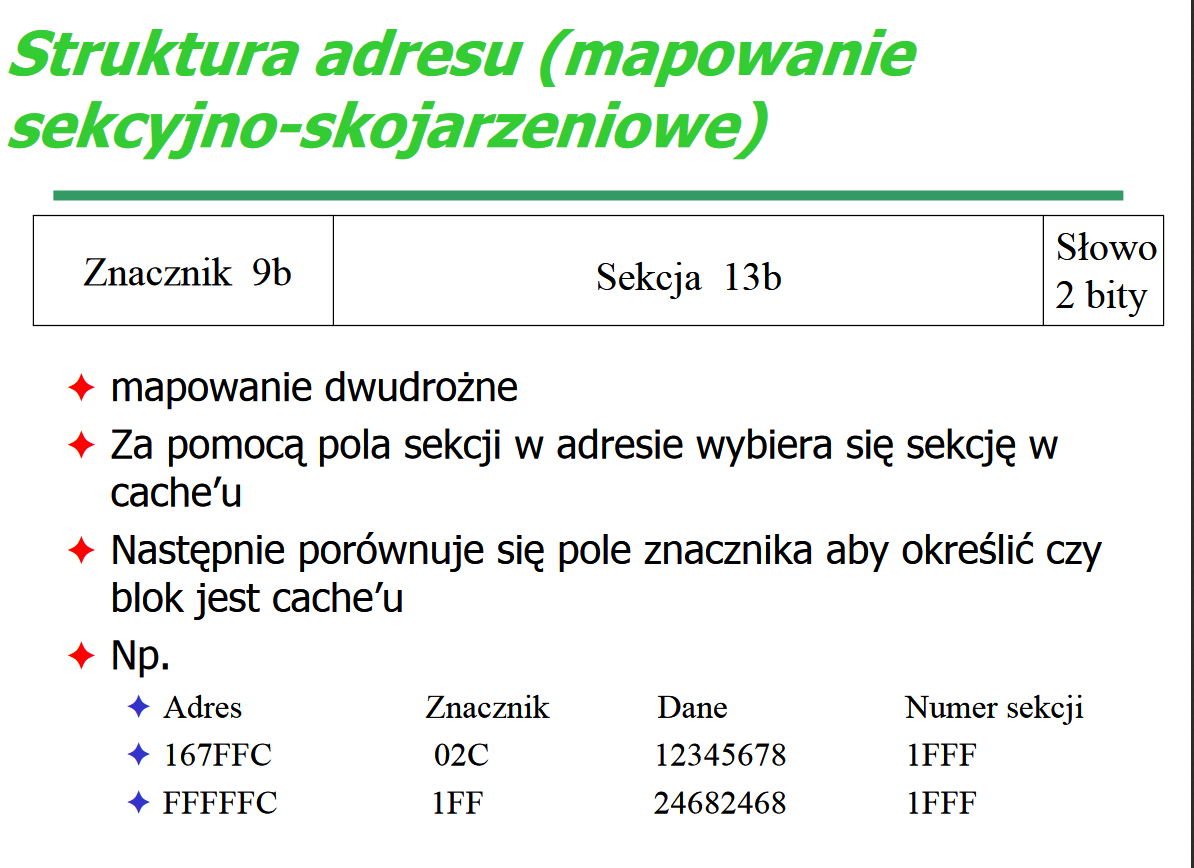
✦2 sposoby mapowania (mapowanie dwudrożne)

✦blok może zostać umieszczony w jednym z dwu wierszy w jednej sekcji

✦np. jeśli adres sekcji jest 13 bitowy

✦określa się numer bloku w pamięci modulo 213

✦bloki 000000, 008000, 018000,...., FF8000są mapowane na tę samąsekcję0



k-drożne mapowanie sekcyjno-skojarzeniowe

✦Długość adresu= (s + w) bitów

✦Liczba jednostek adresowalnych= 2^(s+w)bajtów lub słów

✦Rozmiar bloku = rozmiar wiersza= 2^w bajtów lub słów

✦Liczba sekcji = v = 2^d

✦Liczba bloków w pamięci = 2^(s+w)/2^w = 2^s

✦Liczba wierszy w sekcji = k

✦Liczba wierszy w cache’u= k\*v= k \* 2

✦Rozmiar znacznika = (s –d) bitów

27. Załóżmy, że mamy komputer z pamięcią główną oxM adresowalnych baj-tach oraz pamięć podręczną o pojemnościyKB i blokachzB mapują-ca bezpośrednio, skojarzeniowo,k-drożnie sekcyjnie-skojarzeniowo pamięćgłówną:

(a) jaka jest długość adresu tego komputera (w bitach)?

(b) jaki jest bitowy podział adresu pamięci na znacznik(tag), sekcje, oraz słowo?

(c) w którym wierszu pamięci podręcznej znajdą się następujące bajty(zapis adresu bajtu szesnastkowy), jaki jest ich tag i numer słowa:< ax1>, < ax2>, < ax3>, < ax4>, < ax5>, < ax6>, < ax7>?

(d) w którym wierszu pamięci podręcznej znajdą się następujące bajty(zapis adresu bajtu dziesiętny), jaki jest ich tag i numer słowa:<ad1>, < ad2>, < ad3>, < ad4>, < ad5>, < ad6>, < ad7>?

44. Wymień składowe systemu operacyjnego.

1. Zarządzanie procesorami  
2. Zarządzanie pamięcią operacyjną  
3. Zarządzanie plikami  
4. Zarządzanie systemem we/wy  
5. Zarządzanie pamięcią pomocniczą  
6. Praca sieciowa  
7. System ochrony  
8. System interpretacji poleceń

45. Wymień usługi systemu operacyjnego.

1. wykonanie programu - system powinien móc załadować program do pamięci i  
wykonać go,  
2. operacje we/wy – program użytkowy nie wykonuje operacji we/wy bezpośrednio więc  
musi to oferować system  
3. manipulowanie systemem plików - program musi mieć możliwość (pod kontrolą) do  
czytania, pisania, tworzenia i usuwania plików  
4. komunikacja – wymiana informacji pomiędzy procesami wykonywanymi na tym  
samym lub zdalnym komputerze  
a. np. za pomocą pamięci dzielonej lub przekazywania komunikatów  
5. wykrywanie błędów – zapewnienie prawidłowości działania komputera poprzez  
wykrywanie i obsługę wszystkich błędów w jednostce centralnej, pamięci operacyjnej,  
urządzeniach we/wy (np. błąd sumy kontrolnej) i w programie użytkownika (np.  
przekroczenie czasu)  
6. dodatkowe funkcje systemu nie są przeznaczone do pomagania użytkownikowi, lecz  
do optymalizacji działania samego systemu:  
a. przydzielanie zasobów dla wielu użytkownikowi i wielu zadań w tym samym  
czasie  
b. rozliczanie – przechowywanie danych o tym, którzy użytkownicy i w jakim  
stopniu korzystają z poszczególnych zasobów komputera (statystyka  
użytkowania)  
c. ochrona – zapewnienie aby cały dostęp do zasobów systemu odbywał się  
pod kontrolą  
i. np. dostęp przez modem po podaniu hasła

46. Co to są wywołania (funkcje) systemowe?

1. funkcje systemowe tworzą interfejs między wykonywanym programem a systemem  
operacyjnym,  
a. dostępne na poziomie języka maszynowego (asemblera)  
b. pewne języki zastępują asembler w oprogramowaniu systemowym i  
umożliwiają bezpośrednie wykonywanie funkcji systemowych (np. C, C++, Bliss, PL/360, PERL)  
d. win32 API (Application Programmer Interface) - wielki zbiór procedur  
dostarczanych przez Microsoft, które umożliwiają realizację funkcji  
systemowych

47. Wymień podstawowe metody przekazywania parametrów między procesem a systemem.

1. umieszczenie parametrów w rejestrach jednostki centralnej  
2. zapamiętanie parametrów w tablicy w pamięci operacyjnej i przekazanie adresu tej  
tablicy jako parametru w rejestrze  
3. składowanie przez program parametrów na stosie i zdejmowanie ze stosu przez  
system operacyjny

48. Wymień rodzaje wywołań (funkcji) systemowych.

1. nadzorowanie procesów  
2. operacje na plikach  
3. operacje na urządzeniach  
4. utrzymywanie informacji  
5. komunikacja

49. Wymień struktury systemów operacyjnych oraz przykłady ich realizacji.

50. Wyjaśnij zasadę maszyny wirtualnej.

1. maszyna wirtualna (ang. virtual machine) jest logiczną konsekwencją podejścia  
warstwowego: jądro systemu jest traktowane jako sprzęt  
2. maszyna wirtualna dostarcza identycznego interfejsu dla sprzętu  
3. system operacyjny tworzy wirtualne systemy komputerowe, każdy proces ma do  
dyspozycji własne(wirtualne) jądro, dyski, pamięć, drukarki  
4. zasoby fizycznego komputera są dzielone w celu utworzenia maszyn wirtualnych  
a. planowanie przydziału procesora jest tak wykorzystane, że użytkownik ma  
wrażenie jakoby miał do dyspozycji własny procesor  
b. spooling i system zarządzania plikami jest wykorzystany tak, że powstaje  
wrażenie użytkowania drukarki,czytnika na wyłączność  
c. zwykłe terminale użytkownika funkcjonują jak konsole operatorskie maszyny  
wirtualnej

61. Podaj przykłady uruchamiania procesów w systemie Unix.

1. Proces to program działający we własnej przestrzeni adresowej  
 a. proces wsadowy –polecenie batch, at, cron  
 b. procesy interakcyjne  
 i. procesy pierwszoplanowe – związane z terminalem (np. polecenie ls)  
 ii. procesy drugoplanowe –wykonywane w tle  
 c. demony (ang. daemons) – wystartowane serwisy  
 i. init, syslogd, sendmail, lpd, crond, getty, bdflush, pagedaemon,  
 swapper, inetd, named,routed, dhcpd, portmap, nfsd, smbd, httpd,  
 ntpd  
2. Limity zasobów procesów: polecenia limit, ulimit

X62. Podaj komendy do sterowania procesami w systemie Unix.

1. Uruchamianie w tle: &  
2. Zatrzymanie procesu pierwszoplanowego: ^Z  
3. Ponowne uruchamianie procesu w tle: bg  
4. Listowanie procesów w tle: jobs  
5. Odwołanie do procesu n w tle: %n  
a. Odwołanie do procesu xyz w tle: %?xyz  
6. Przenoszenie z tła na pierwszy plan: fg %

X63. Narysuj diagram stanów procesów.

✦Wykonujący się proces zmienia swój stan (ang. state)

✦nowy (ang. new): proces został utworzony

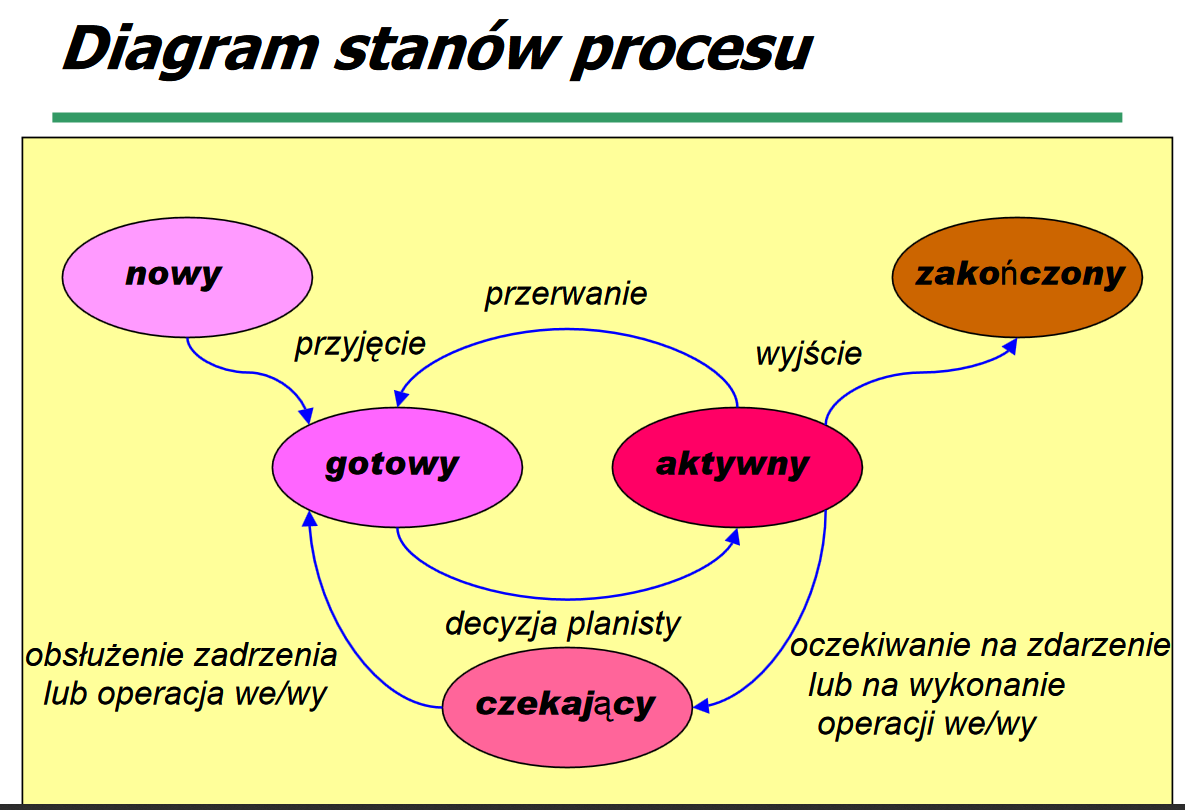
✦aktywny (ang. running): są wykonywane instrukcje

✦czekający(ang. waiting): proces czeka na zdarzenie (np. zakończenie we/wy)

✦gotowy (ang. ready): proces czeka na przydział procesora

✦zakończony (ang. terminated): proces zakończył działanie

✦Model pięciostanowy



✦aktywny

✦(user) –wykonywany w trybie użytkownika

✦(kernel) –wykonywany w trybie jądra

✦gotowy

✦w pamięci –gotowy do wykonywania gdy jądro go zaszereguje

✦swap out –gotowy do wykonywania po przeniesieniu do pamięci

✦uśpiony

✦w pamięci –gotowy do wykonywania po zajściu zdarzenia (zablokowany)

✦swap out –czekający na zdarzenie i przeniesiony do obszaru wymiany

✦wywłaszczony –proces, który przeszedł od trybu jądra do trybu użytkownika i pozbawiony został procesora w celu uruchomienia innego procesu

✦utworzony –utworzony, niegotowy do uruchomienia

✦zombie –potomek przestał istnieć, rodzic jest w pamięci

64. Opisz zawartość PCB.

✦Każdy proces w systemie operacyjnym jest reprezentowany przez blok kontrolny procesu (ang. process control block -PCB) zawierający

✦stan procesu -gotowy, nowy, aktywny, czekający, zatrzymany

✦licznik rozkazów -adres następnego rozkazu do wykonania w procesie

✦rejestry procesora -zależą od architektury komputera: akumulatory, rejestry (ogólne, bazowe, indeksowe) wskaźniki stosu przechowywane aby proces mógł byćkontynuowany po przerwaniu

✦stan procesora (PSW –IBM/370, EFLAGS –Pentium II)

✦informacje o planowaniu przydziału procesora -priorytet procesu, wskaźniki do kolejek porządkujących zamówienia

✦informacje o zarządzaniu pamiecią -zawartości rejestrów granicznych, tablice stron, tablice segmentów w zależności od systemu używanej pamięci

✦informacje do rozliczeń -ilość zużytego czasu procesora i czasu rzeczywistego, ograniczenia czasowe, numery kont, numery zadań

✦informacje o stanie we/wy -lista zaalokowanych urządzeń, wykaz otwartych plików



65. Omów przyczyny przełączania procesu.

✦Przerwanie (ang. interrupt) –zdarzenie zewnętrzne w stosunku do procesu

✦Pułapka (ang. trap) –zdarzenie wewnątrz aktualnie przetwarzanego procesu, błąd lub warunek wyjątku

✦Polecenie administracyjne (ang. supervisor call) –wywołanie funkcji systemu operacyjnego

X66. Omów rodzaje planistów i zadania przez nich realizowane.

✦Planista długoterminowy (ang. long-term scheduler) lub planista zadań (ang. job scheduler)

-wybiera procesy, które powinny być sprowadzone do pamięci z kolejek procesów (niegotowych)

✦zazwyczaj na dyskach

✦Planista krótkoterminowy (ang. short-term scheduler) lub planista przydziału procesora (ang. CPU scheduler)

-wybiera proces następny do wykonania z kolejki procesów gotowych i przydziela mu procesor

✦Planista krótkoterminowy jest wołany bardzo często (milisekundy) dlatego musi być bardzo szybki

✦raz na 100ms

✦Planista długoterminowy jest wołany rzadko (sekundy, minuty) dlatego może nie być szybki

✦Planista długoterminowy nadzoruje stopień wieloprogramowości (tzn. liczbę procesów w pamięci)

✦Proces może być opisany jako jeden z

✦ograniczony przez we/wy (ang. I/O bound) -więcej czasu zajmuje we/wy niż dostęp do procesora

✦ograniczony przez dostęp do procesora (ang. CPU bound) -więcej czasu zajmują obliczenia, we/wy sporadyczne

✦Planista długoterminowy powinien dobrać mieszankę procesów (ang. process mix) zawierającą zarówno procesy ograniczone przez we/wy jak i procesor

✦Planista średnioterminowy (ang. medium-term scheduler) –usuwa procesy z pamięci, zmniejszając stopieńwieloprogramowości, a później sprowadza je ponownie

✦uzyskanie lepszej mieszanki procesów

✦żądania przydziału pamięci przekraczają ilość dostępnej pamięci

✦postępowanie takie nazywamy wymianą (ang.swapping)

67. Omów przełączanie kontekstu.

✦Gdy procesor przełącza do innego procesu system musi zachować stan starego procesu i załadować zachowany stan nowego procesu

✦czynność tę nazywamy przełączaniem kontekstu (ang. context switch)

✦Przełączanie kontekstu jest ceną za wieloprogramowość

✦system operacyjny nie wykonuje wtedy żadnej użytecznej pracy

✦Czas przełączenia kontekstu zależy od sprzętu (zwykle od 1 do 1000 μs)

✦np. od ilości rejestrów (Sun UltraSPARC ma po kilka zbiorów rejestrów)

X68. Omów tworzenie procesu zombie i orphan.

✦Proces macierzysty (ang. parent process) tworzy nowy proces -potomka (ang. child process) i każdy nowy proces może tworzyć nowe procesy, które tworzą drzewo procesów

✦przy pomocy wywołania systemowego create-process

✦Dzielenie zasobów

✦proces macierzysty i potomek dzielą wszystkie zasoby

✦proces macierzysty i potomek dzielą część zasobów

✦proces macierzysty i potomek nie dzielążadnych zasobów

✦Drzewo procesów: polecenie pstre

✦Wykonanie

✦proces macierzysty i potomek działają współbieżnie

✦proces macierzysty czeka aż potomek zakończy

✦Przestrzeń adresowa nowego procesu

✦potomek jest kopią procesu macierzystego

✦potomek ładuje do przestrzeni adresowej nowy program

69. Omów komunikację pośrednią i bezpośrednią między procesami.

✦Komunikacja międzyprocesowa (ang. interprocess-communication -IPC) udogodnienia systemu pozwalające współpracującym procesom na kontaktowanie się ze sobą

✦łączność i synchronizacja

✦System przekazywania komunikatów (ang. message system) -sposób realizacji komunikacji międzyprocesorowej pozwalający nie odwoływać się do zmiennych dzielonych

✦Komunikacja międzyprocesowa dostarcza dwóch operacji

✦nadaj (komunikat) (ang. send (message))

✦odbierz(komunikat) (ang. receive(message))

✦Jeśli procesy P i Q chcą się skomunikować to

✦muszą ustanowićłącze komunikacyjne

✦nadawać i odbierać komunikaty

✦Implementacja

✦fizyczna (np. pamięć dzielona, szyna sprzętowa)

✦logiczna (np. własności logiczne)

Komunikacja bezpośrednia

✦Proces musi jawnie nazwać odbiorcę

✦nadaj(P,komunikat) -nadaj komunikat do procesu P

✦odbierz(Q,komunikat) -odbierz komunikat od procesu Q

✦Własności łącza

✦łącze jest ustanawiane automatycznie, do komunikowania wystarczy znajomość identyfikatorów

✦łącze dotyczy dokładnie dwóch procesów

✦między każdą parą procesów istnieje dokładnie jedno łącze

✦łącze jest zwykle dwukierunkowe, ale może być jednokierunkowe

Komunikacja pośrednia

✦Komunikaty są nadawane i odbierane za pomocą skrzynek pocztowych (ang. mailboxes) nazywanych także portami (ang. ports)

✦każda skrzynka ma swój unikalny identyfikator

✦procesy komunikują się jeśli mają wspólną skrzynkę

✦Własności łącza

✦łącze jest ustanawiane jedynie wtedy gdy procesy dzieląskrzynkę

✦łącze może być związane z więcej niż dwoma procesami

✦każda para procesów może mieć kilka łączy z których każdy odpowiada jakiejś skrzynce

✦łącze może być jedno-lub dwukierunkowe

✦System operacyjny dostarcza mechanizmów do

✦tworzenia nowej skrzynki

✦nadawania i odbierania komunikatów za pomocą skrzynki

✦likwidowania skrzynki

✦Dzielenie skrzynki : P, Q, R dzielą skrzynkę A

✦P nadaje; Q i R odbierają

✦który proces otrzyma komunikat nadany przez P?

✦Rozwiązanie

✦zezwalać jedynie na łącza między dwoma procesami

✦pozwalać najwyżej jednemu procesowi na odbiór

✦pozwalać aby system wybrał i poinformował odbiorcę

70. Wymień zasoby używane przez wątki.

✦Wątek (ang. thread) nazywany niekiedy procesem lekkim (ang. lightweight process -LWP) jest podstawową jednostką wykorzystania procesora. W skład tej jednostki wchodzą

✦licznik rozkazów

✦zbiór rejestrów

✦obszar stosu

✦Wątek współużytkuje z innymi równorzędnymi wątkami

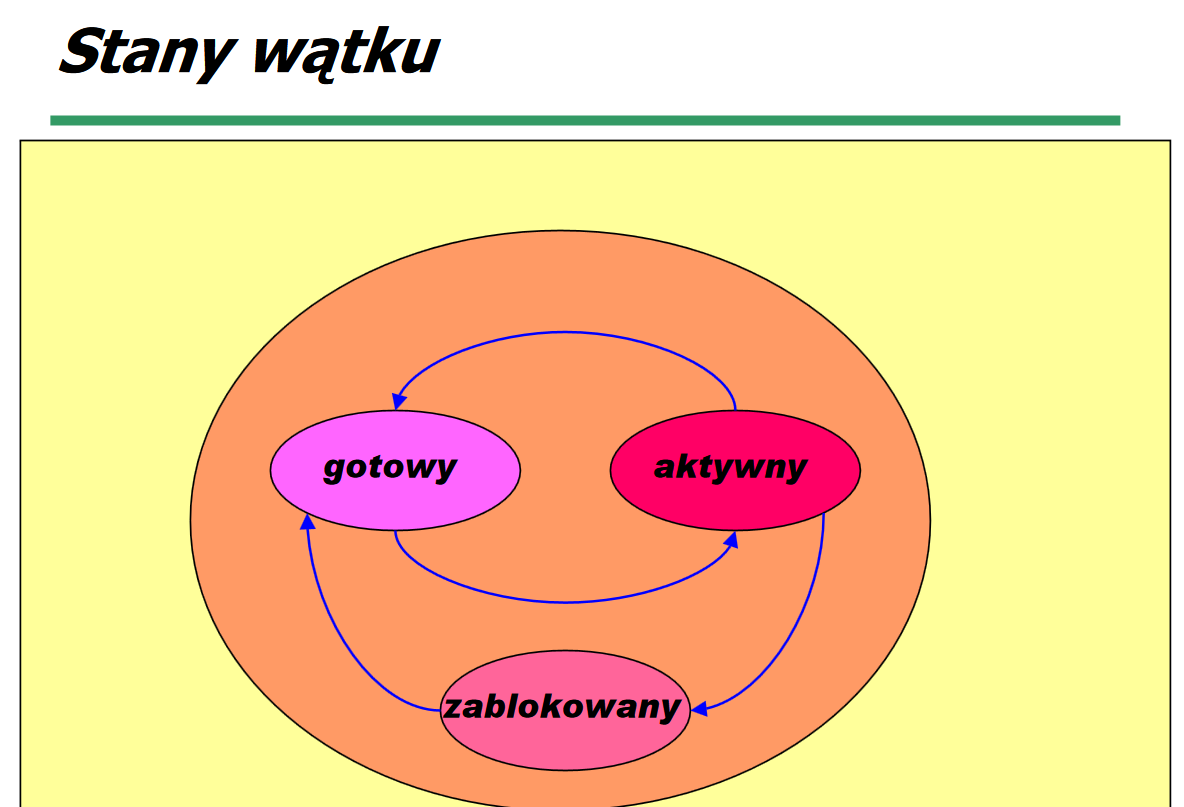
✦sekcję kodu

✦sekcję danych

✦zasoby systemu (takie jak otwarte pliki i sygnały) zwane wspólnie zadaniem (ang. task)

✦Proces tradycyjny lub ciężki (ang. heavyweight) to zadanie o jednym wątku

X71. Narysuj diagram stanów wątku.



72. Wymień typy wątków.

✦Wątki (ang. kernel-level, KLT) obsługiwane przez jądro

✦Mach, OS/2, Windows NT/2K/XP, Linux, BeOS, Mac OS X

✦Wątki tworzone na poziomie użytkownika (ang. user-level, ULT) za pomocą funkcji bibliotecznych

✦system Andrew

✦P-wątki (ang. Pthreads) -implementacje w postaci bibliotek

✦standard POSIX (IEEE 1003.1c) definiujący interfejs programów użytkowych (API) do tworzenia wątków oraz ich synchronizowania

✦Mach: C-threads,

✦Solaris 2: UI-threads

✦zaleta: szybsze przełączanie, wada: planowanie wątków

✦Hybrydowe podejście -Solaris 2

✦Wątki zarządzane przez JVM

X73. Wymień sposoby odwzorowań wątków.

✦Wiele do jednego (ang. Many-to-One)

✦biblioteka “green threads” Solaris 2

✦wykonanie blokującegowywołania systemowego blokuje proces

✦Jeden do jednego (ang. One-to-One)

✦OS/2, Windows NT/2K/XP, Linux

✦równoległe wykonywanie wątków (na procesorach)

✦ograniczona ilość wątków

✦Wiele do wielu (ang. Many-to-Many)

✦Solaris 2, IRIX, HP-UX, True 64 (Digital UNIX)

✦funkcjonalność w Windows NT/2K/XP w postaci biblioteki włókien(ThreadFiber)

✦Jeden do wielu (ang. One-to-Many)

✦wątek może migrować pomiędzy systemami (Clouds, Emerald)

X74. Wymień sytuacje w jakich planista przydziału procesora podejmuje decyzję o przydziale procesora.

✦Planista (krótkoterminowy) przydziału procesora (ang. CPU scheduler) wybiera jeden proces spośród przebywających w pamięci procesów gotowych do wykonania i przydziela mu procesor

✦Decyzje o przydziale procesora podejmowane są

✦1. gdy proces przeszedł od stanu aktywności do stanu czekania

✦np. z powodu operacji we/wy

✦np. czekanie na zakończnie potomka

✦2. gdy proces przeszedł od stanu aktywności do stanu gotowości

✦np. wskutek przerwania

✦3. gdy proces przeszedł od stanu czekania do stanu gotowości

✦np. po zakończeniu operacji we/wy

✦4. gdy proces kończy działanie

✦W pkt.2 i3 można dokonać wyboru procesu któremu przydzielić CPU

✦Planowanie w sytuacjach 1 i 4 nazywamy niewywłaszczającym (ang. nonpreemptive)

✦Algorytm planowania (szeregowania) nazywamy wywłaszczającym(ang. preemptive)w pozostałych sytuacjach

✦systemy Windows 9x, 2K, XP, Mac OS X

✦Planowanie bez wywłaszczeń: proces, który otrzyma procesor, zachowuje go tak długo aż nie odda go z powodu przejścia w stan oczekiwania lub zakończenia

✦planowanie to nie wymaga zegara

✦systemyWindows 3.xi Apple Macintosh OS

✦Planowanie wywłaszczające: drogie i ryzykowne

✦Co się stanie gdy wywłaszczony zostanie proces w trakcie wykonywania funkcji systemowej (np. zmiany danych w blokach opisu kolejki we/wy)?

✦UNIX czeka z przełączeniem kontekstu do zakończenia wywołania systemowego lub do zablokowania procesu na we/wy

✦model ten słabo nadaje się do przetwarzania w czasie rzeczywistym

✦Nie można wywłaszczać procesu gdy wewnętrzne struktury jądra są niespójne

✦Blokowanie przerwań przy wejściu do ryzykownych fragmentów kodu jądra

✦włączanie i wyłączanie przerwań jest kosztowne (np. wiele procesorów)

X75. Wymień warunki planowania bez wywłaszczeń.

1. planowanie bez wywłaszczeń : proces, który otrzyma procesor, zachowuje go tak  
długo aż nie odda go z powodu przejścia w stan oczekiwania lub zakończenia (nie  
wymaga zegara)  
2. planowanie w sytuacji 1 i 4 nazywamy nie wywłaszczeniowym – z poprzedniego  
pytania

76. Co to jest dispatcher oraz dispatch latency?

✦Ekspedytor (ang. dispatcher)jest modułem, który faktycznie przekazuje procesor do dyspozycji procesu wybranego przez planistękrótkoterminowego;obowiązki ekspedytora to:

✦przełączanie kontekstu

✦przełączanie do trybu użytkownika

✦wykonanie skoku do odpowiedniej komórki w programie użytkownika w celu wznowienia działania programu

✦Opóźnienie ekspedycji (ang. dispatch latency) to czas, który ekspedytor zużywa na wstrzymanie jednego procesu i uaktywnienie innego

X77. Co to jest przepustowość oraz wykorzystanie CPU?

✦Wykorzystanie procesora (ang. CPU utilization) –procent czasu, przez który procesor pozostaje zajęty

✦najlepiej by było gdyby procesor był nieustannie zajęty pracą

✦powinno się mieścić od 40% (słabe obciążenie systemu) do 90% (intensywna eksploatacja)

✦Przepustowość (ang. throughput) -liczba procesów kończących w jednostce czasu

✦długie procesy -1 na godzinę, krótkie -10 na sekundę

X78. Podaj definicje czasu (cyklu) przetwarzania, czasu oczekiwania, czasu odpowiedzi.

✦Czas cyklu przetwarzania (ang. turnaroundtime)–czas między nadejściem procesu do systemu a chwilązakończeniaprocesu

✦suma czasów czekania na wejście do pamięci, czekania w kolejce procesów gotowych, wykonywania procesu przez CPU i wykonywania operacji we/wy

✦Czas oczekiwania(ang. waiting time)-suma okresów, w których proces czeka w kolejce procesów gotowych do działania

✦Czas odpowiedzi lub reakcji (ang. response time) -ilośćczasu między wysłaniem żądania a pojawieniem sięodpowiedzi bezuwzględnienia czasu potrzebnego na wyprowadzenie odpowiedzi (np. na ekran).

✦czas odpowiedzi jest na ogół uzależniony od szybkości działania urządzenia wyjściowego

✦miara zastępująca miarę czasu cyklu przetwarzania w systemach interakcyjnych

✦np. kliknięcie myszą obiektu –mniej niż 0,1s

X79. Wymień kryteria optymalizacji algorytmu planowania.

✦Maksymalnewykorzystanie procesora

✦Maksymalna przepustowość

✦Minimalny czas cykluprzetwarzania

✦Minimalny czas oczekiwania

✦Minimalny czas odpowiedzi

✦optymalizujemy miary średnie, maksymalne (ew. minimalne)

✦minimalizowanie wariancji czasu odpowiedzi zamiast średniego czasu odpowiedzi w systemach z podziałem czasu

✦mało algorytmów minimalizujących wariancję

✦pożądany system z sensownym i przewidywalnym czasem odpowiedzi zamiast systemu o lepszym średnio czasie odpowiedzi i bardzo zmiennym

80. Opisz algorytm FCFS.

Planowanie metodą FCFS

✦Pierwszy zgłoszony-pierwszy obsłużony (ang. first-come, first-served -FCFS)

✦Implementuje sięłatwo za pomocą kolejek FIFO -blok kontrolny procesu dołączany na koniec kolejki, procesor dostaje PCB z czoła kolejki

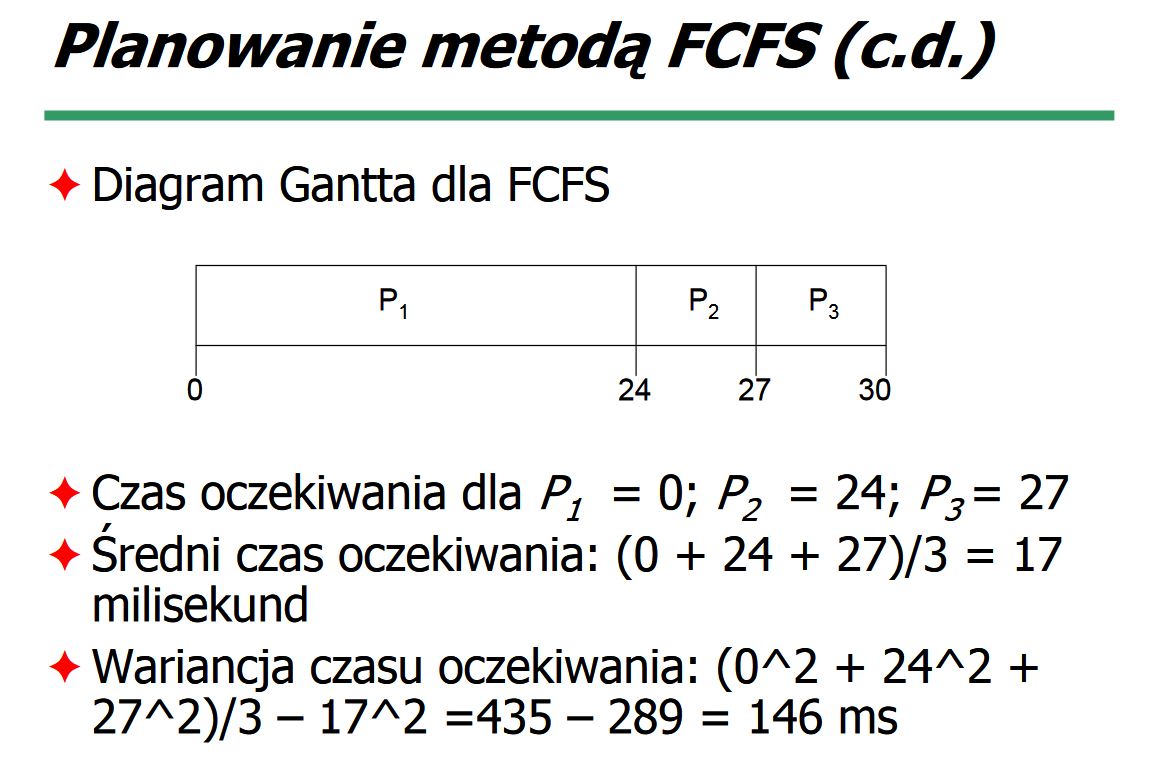
✦Przykład: Proces Czas trwania fazy

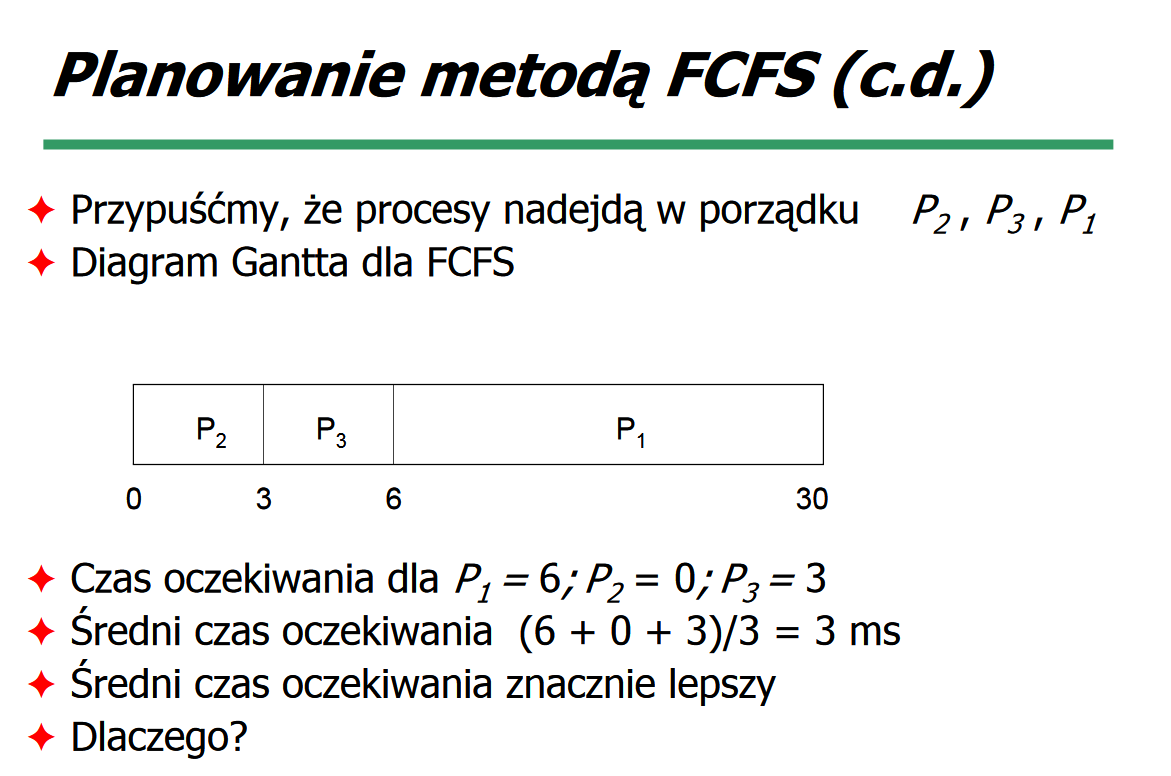
P1 24

P2 3

P3 3

✦Przypuśćmy, że procesy nadejdą w porządku: P1, P2, P3





✦Założmy, ze mamy jeden proces P ograniczony przez procesor i wiele procesów ograniczonych przez we/wy (Q,R,..) wtedy może dojść do realizacji następującego scenariusza:

✦proces P uzyskuje procesor i procesy Q,R,.. kończą we/wy

✦urządzenia we/wy są bezczynne

✦proces P kończy swoją fazę procesora, procesy Q,... zadziałają szybko (mają krotkie fazy procesora bo są ograniczone przez we/wy)

✦proces P uzyskuje procesor, procesy Q,.. kończą we/wy...

✦Efekt konwoju (ang. convoy effect) -procesy czekają ażwielki proces odda procesor

✦Algorytm FCFS jest niewywłaszczający -proces utrzymuje procesor do czasu aż zwolni go wskutek zakończenia lub zamówi operację we/wy

✦Niewydajne wykorzystanie CPU (efekt konwoju dla procesora) oraz we/wy (efekt konwoju dla urządzeńwe/wy)

✦Krzywdzący dla procesów krótkich oraz ograniczonych przez we/wy bowiem faworyzuje dłuższe zadania

✦Proces zawsze dostanie się do CPU (po pewnym czasie) tj. nie ma groźby zagłodzenia procesów

✦Algorytm FCFS jest kłopotliwy w systemach z podziałem czasu bowiem w takich systemach ważne jest uzyskiwanie procesora w regularnych odstępach czasu

81. Opisz algorytm SJF wywłaszczający.

✦Algorytm napierw najkrótsze zadanie (ang. shortest-job-first -SJF) wiąże z każdym procesem długość jego najbliższej z przyszłych faz procesora; gdy procesor staje się dostępny wówczas zostaje przydzielony procesowi o najkrótszej następnej fazie procesora (gdy fazy są równe to stosujemy planowanie FCFS)

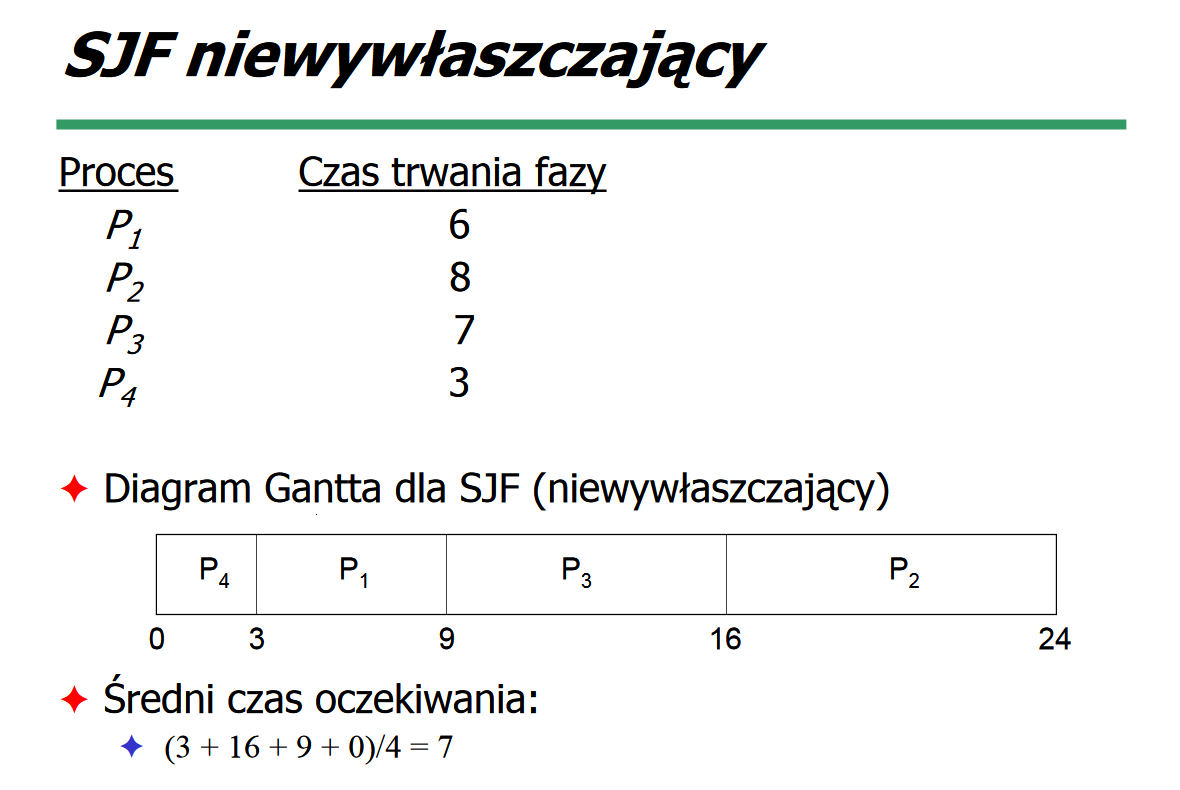
✦Algorytm SJF może być:

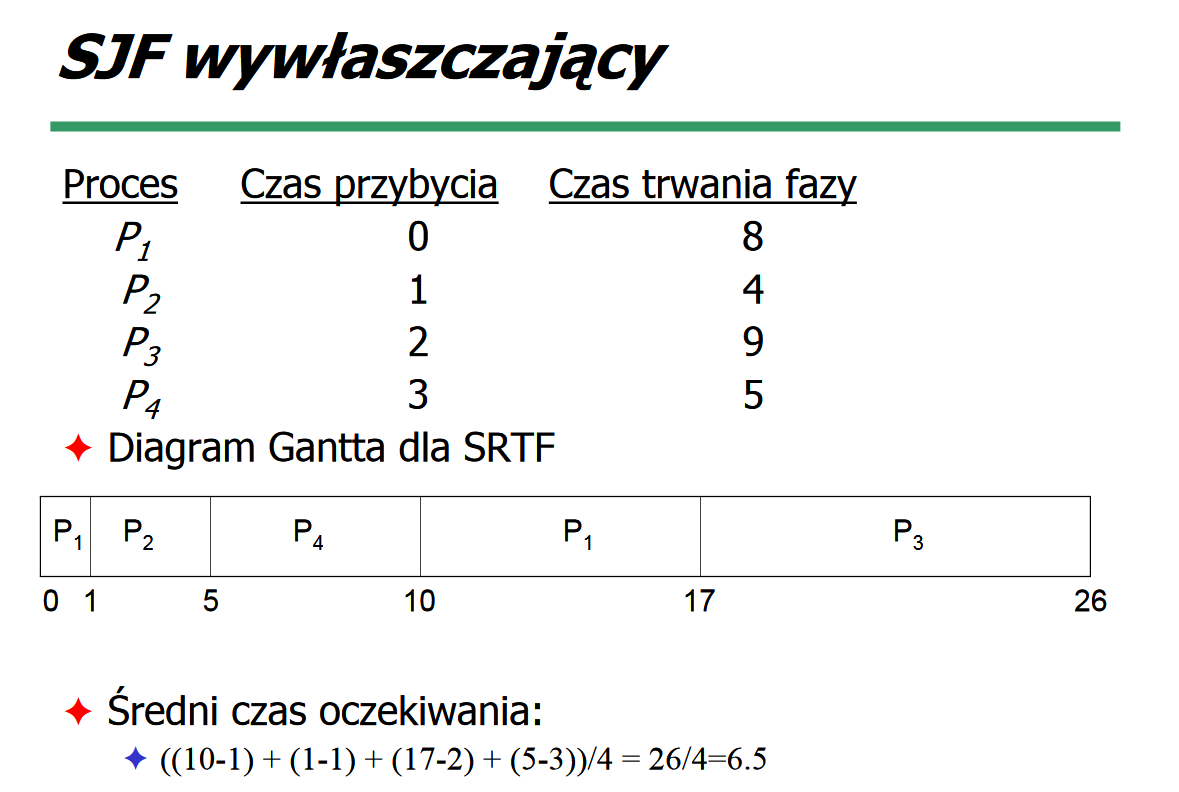
✦wywłaszczający -SJF usunie proces jeśli nowy proces w kolejce procesów gotowych ma krótszą następną fazę procesora od czasu do zakończenia procesu

✦gdy w kolejce procesów gotowych są procesy o jednakowych fazach to stosujemy FCFS

✦algorytm SRTF (ang. shortest-remaining-time-first) -najpierw najkrótszy pozostały czas

✦niewywłaszczający -pozwól procesowi zakończyć





✦SJF charakteryzuje to, że

✦ma dobry czas odpowiedzi dla krótkich procesów

✦jest krzywdzący dla procesów długich

✦może powodować zagłodzenie procesów

✦Planowanie metodą SRTF zwykle daje lepszy czas przetwarzania niż SJF

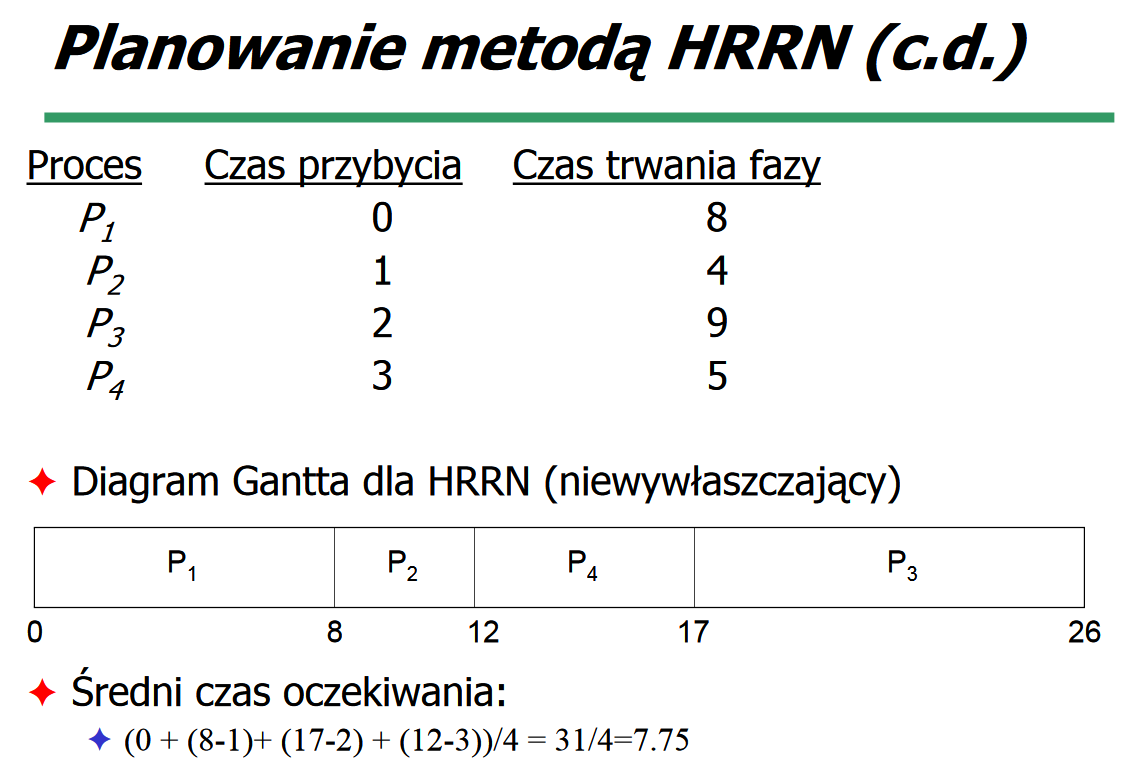
82. Opisz algorytm HRRN.

✦Def. Stosunkiem reaktywności (ang. response ratio) nazywamy liczbę R = 1+ w/t, gdzie w oznacza czas oczekiwania na procesor zaśt -fazę procesora

✦Największy stosunek reaktywności jako następny (ang. highest response ratio next -HRRN)

✦Faworyzuje krótkie zadania, lecz po pewnym czasie oczekiwania dłuższy proces uzyska CPU

✦Podobnie jak SJF i SRTF również algorytm HRRN wymaga oszacowaniadla następnej fazy procesora



✦Faworyzuje krótkie zadania jednak oczekiwanie dłuższych zadań zmienia ich współczynnik i w konsekwencji pozwala im uzyskać dostęp do CPU

✦Ma dobry czas odpowiedzi

✦Proces zawsze dostanie się do CPU (po pewnym czasie)

✦tj. nie ma groźby zagłodzenia procesów

83. Podaj wzór na oszacowanie następnej fazy procesora.

✦SJF jest optymalny: daje minimalny średni czas oczekiwania dla danego zbioru procesów

✦Nie ma sposobu na poznanie długości następnej fazy, możemy ją jedynie oszacować

✦Można tego dokonać wyliczając średnią wykładnicząpoprzednich faz procesora

✦t(n) = długość n-tej fazy procesora

✦a -liczba z przedziału [0,1], zwykle 0.5

✦Definiujemy średnią wykładniczą jako:  gdzie s(n+1) = przewidywana długość następnej fazy a s(n) przechowuje dane z minionej historii; s(0) –stała (np. średnia wzięta z całego systemu)

✦a=0

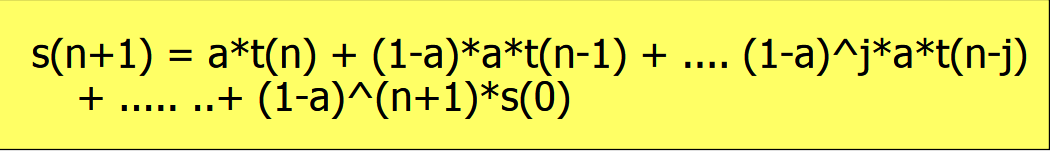
✦s(n+1) = s(n)

✦niedawna historia nie ma wpływu

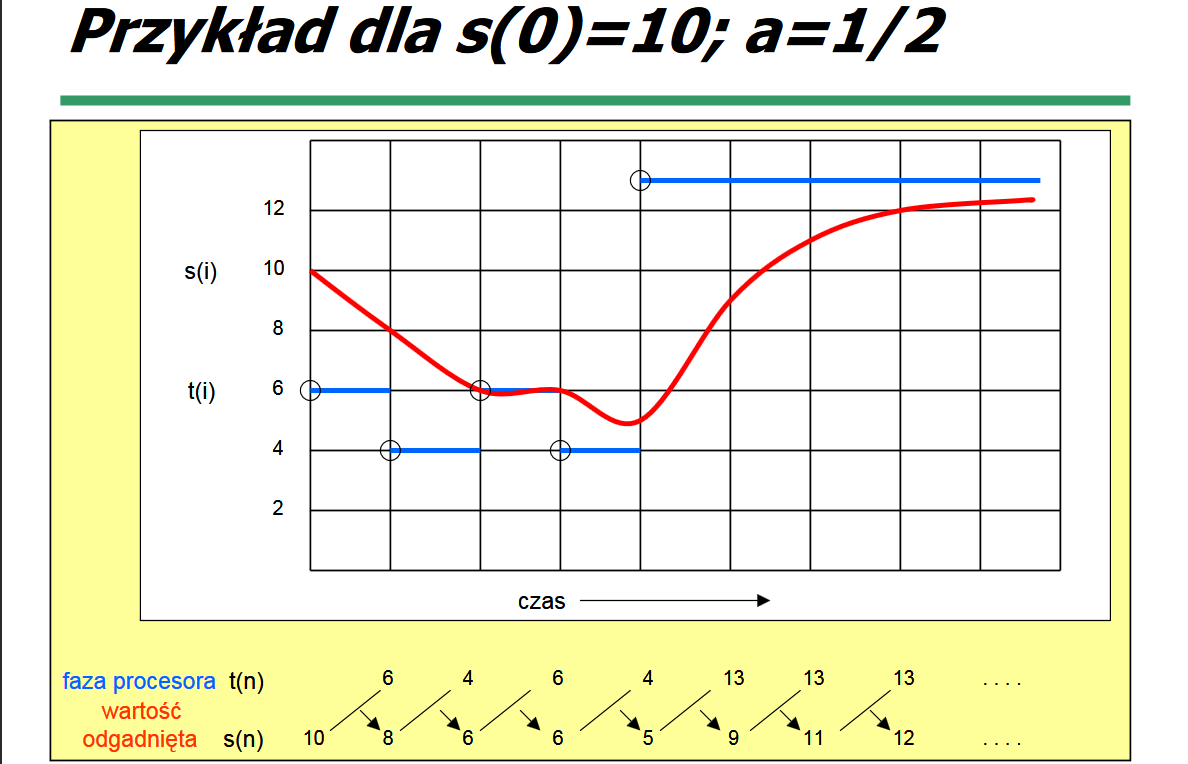
✦a=1

✦s(n+1) = t(n)

✦jedynie najnowsze notowanie długości fazy ma wpływ

✦a\*(1-a)≠0i rozwiniemy wzór to

✦Ponieważ a i (1-a) są mniejsze od 1 to starsze składniki (przeszłość) mają coraz mniejszą wagę



84. Opisz planowanie priorytetowe.

✦SJF jest przykładem planowania priorytetowego (ang. priority scheduling) w którym każdemu procesowi przypisuje się priorytet (liczbę)

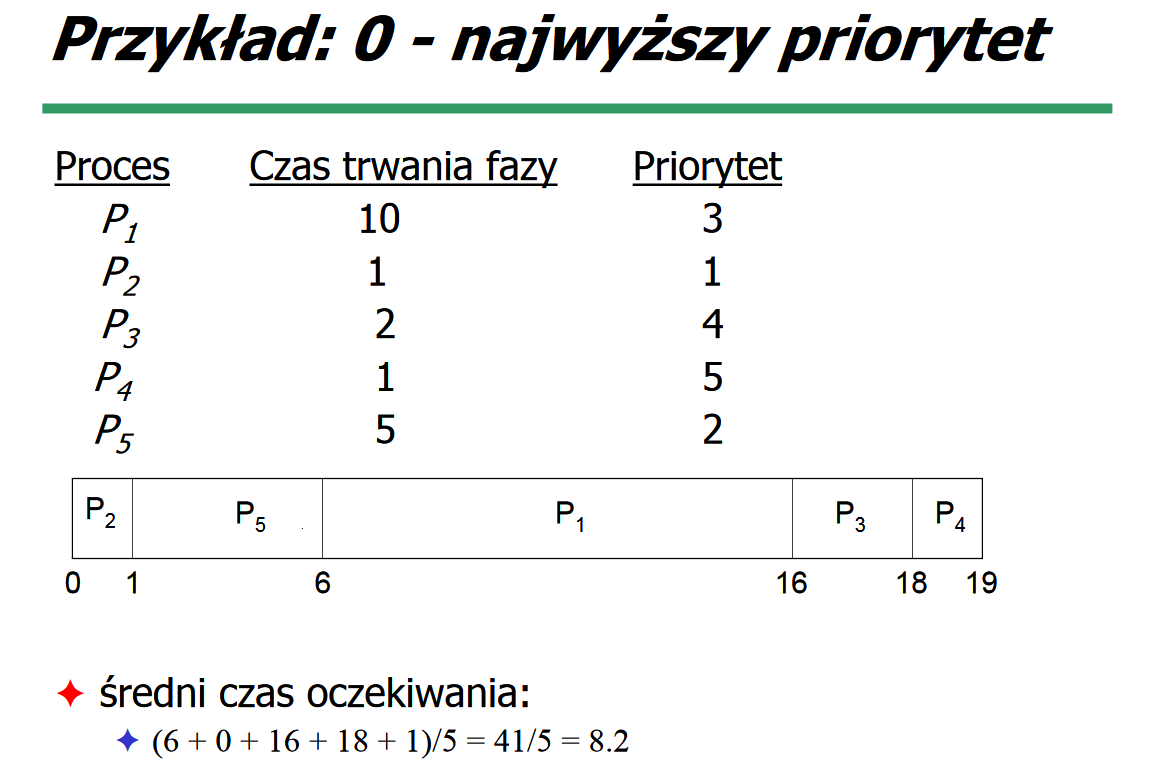
✦Priorytety należą do pewnego skończonego podzbioru liczb naturalnych np. [0..7], [0,4095] ✦Prz. nice {+|-n}polecenie

✦Procesor przydziela się procesowi o najwyższym priorytecie (jeśli priorytety są równe to FCFS)

✦planowanie priorytetowe wywłaszczające

✦planowanie priorytetowe niewywłaszczające

✦SJF -priorytet jest odwrotnością następnej fazy



✦Problem: nieskończone zablokowanie (ang. indefinite blocking) lub głodzenie (ang. starvation) -procesy o małym priorytecie mogą nigdy nie dostać czasu procesora

✦Krąży taka pogłoska, że gdy w 1973 r. wycofywano z eksploatacji w MIT komputer IBM 7094 wykryto zagłodzony niskopriorytetowy proces przedłożony do wykonania jeszcze w 1967 r.

✦Rozwiązanie: postarzanie (ang. aging) polegające na podwyższeniu priorytetu procesów oczekujących już zbyt długo

✦Prz. Proces ma priorytet 127, co 15 min zwiększamy priorytet o 1 więc w najgorszym przypadku (tzn. jeśli nie dostanie się do CPU) po 32 godzinach proces będzie miałnajwyższy priorytet równy 0 (co wcale nie oznacza, że dostanie się wtedy do CPU )

85. Opisz algorytm RR.

✦Planowanie rotacyjne, RR (ang. round-robin, time-slicing) zaprojektowano dla systemów z podziałem czasu

✦Każdy proces otrzymuje małą jednostkę czasu, nazywaną kwantem czasu (ang. time quantum, time slice) zwykle od 10 do 100 milisekund. Gdy ten czas minie proces jest wywłaszczany i umieszczany na końcu (ang. tail) kolejki zadańgotowych (FCFS z wywłaszczeniami)

✦średni czas oczekiwania jest stosunkowo długi

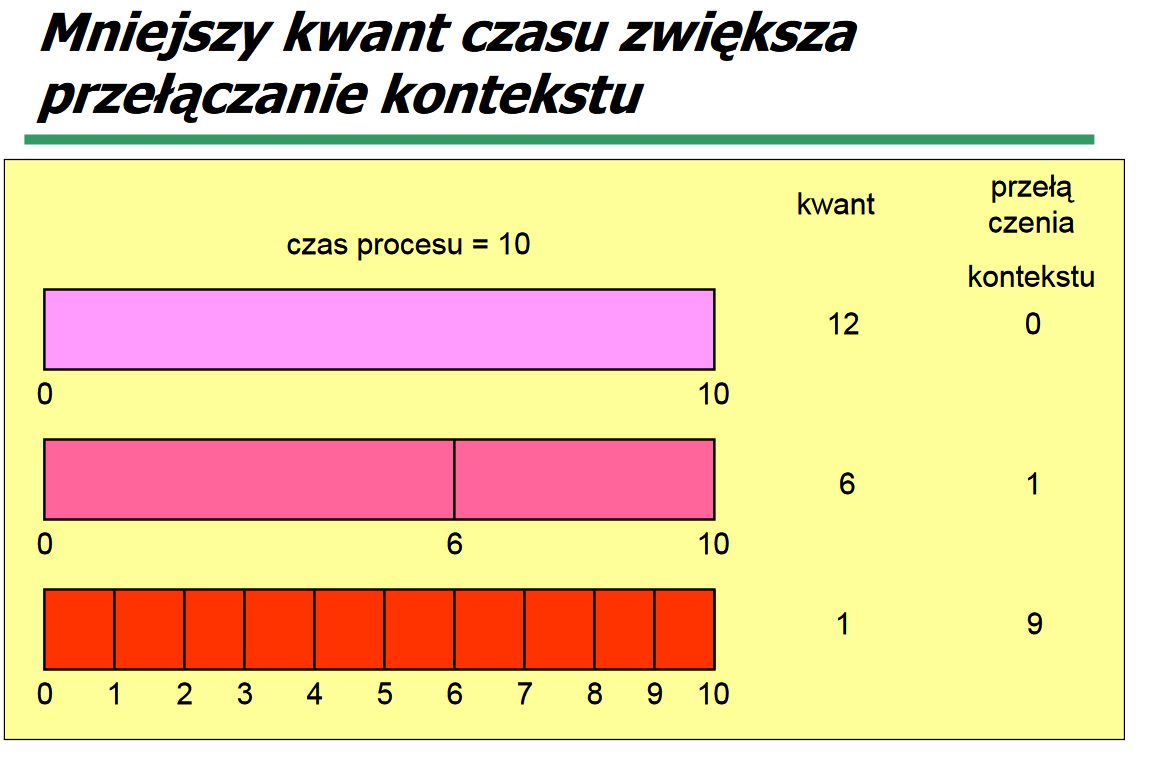
✦Jeśli jest n procesów w kolejce procesów gotowych a kwant czasu wynosi q to każdy proces otrzymuje 1/n czasu procesora porcjami wielkości co najwyżej q jednostek czasu.

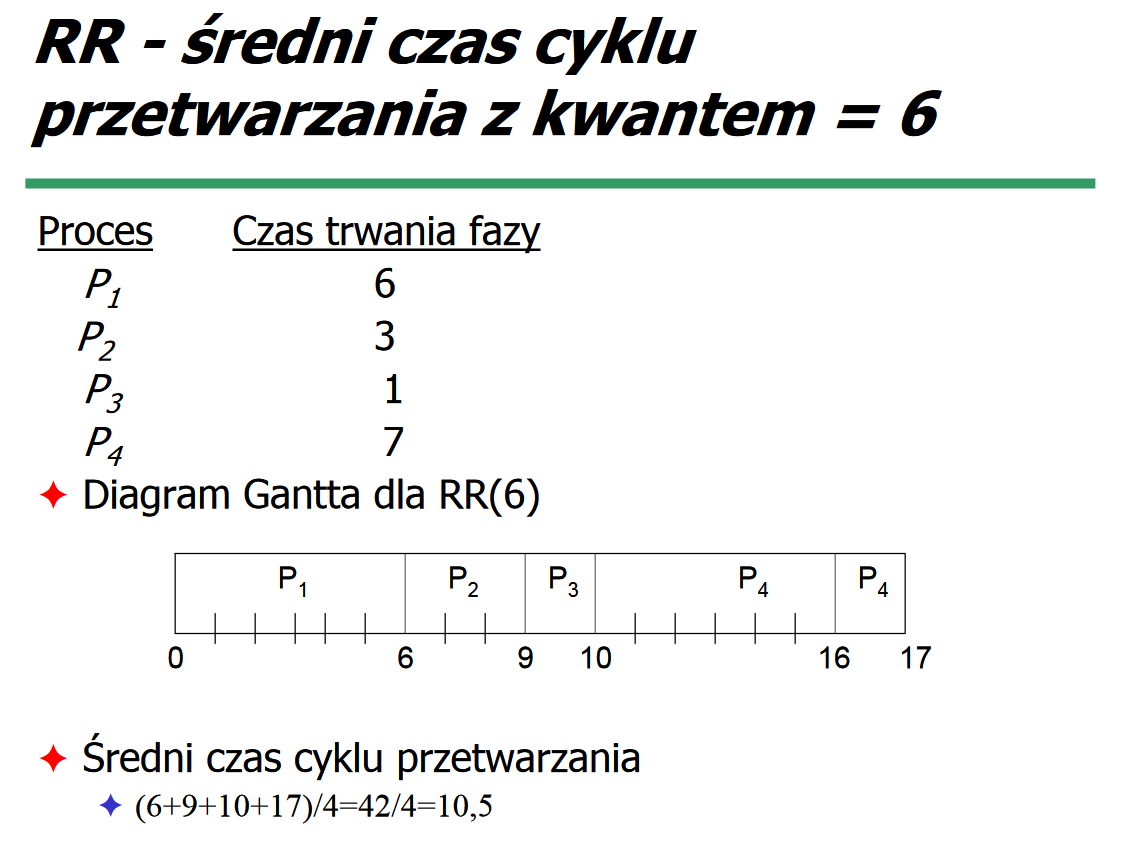
✦Każdy proces czeka nie dłużej niż (n-1)\*q jednostek czasu

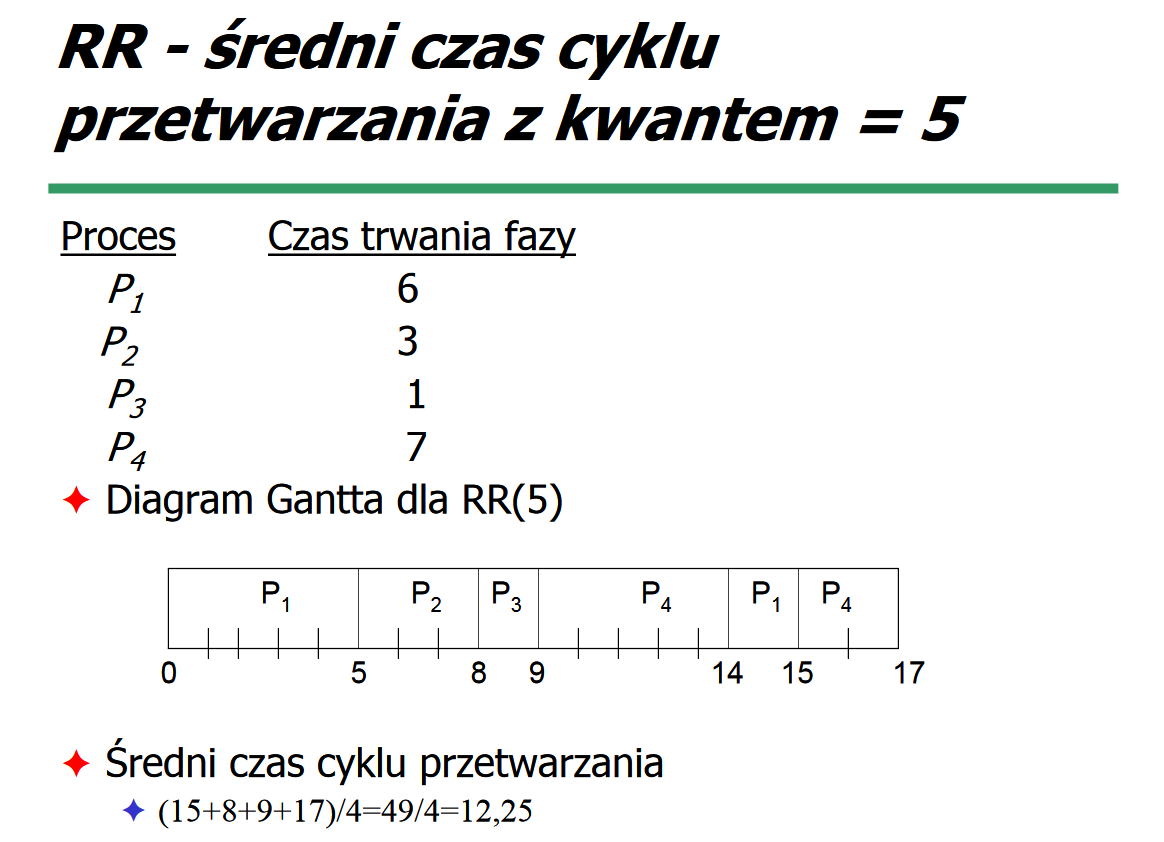
✦Wydajność algorytmu

✦gdy q duże to RR(q) przechodzi w FCFS

✦gdy q małe to mamy dzielenie procesora (ang. processor sharing) ale wtedy q musi być duże w stosunku do przełączania kontekstu (inaczej mamy spowolnienie)









✦Dobry czas odpowiedzi dla krótkich procesów

✦Efektywny w systemach z podziałem czasu

✦Sprawiedliwe traktowanie procesów

✦Kwant powinien być nieco dłuższy od czasu wymaganego na typową interakcję

✦Procesy ograniczone przez CPU są faworyzowane kosztem procesów ograniczonych przez we/wy co prowadzi do nieefektywnego wykorzystania we/wy

✦Nie powoduje zagłodzenia procesów

86. Rozważmy procesy:P1, P2, P3, P4, P5o następujących czasach trwania fazy(ms):

f1, f2, f3, f4, f5, które przybyły o czasiet1, t2, t3, t4, t5i priorytetach:p1, p2, p3, p4, p5

przy czym najwyższy priorytet mapi.

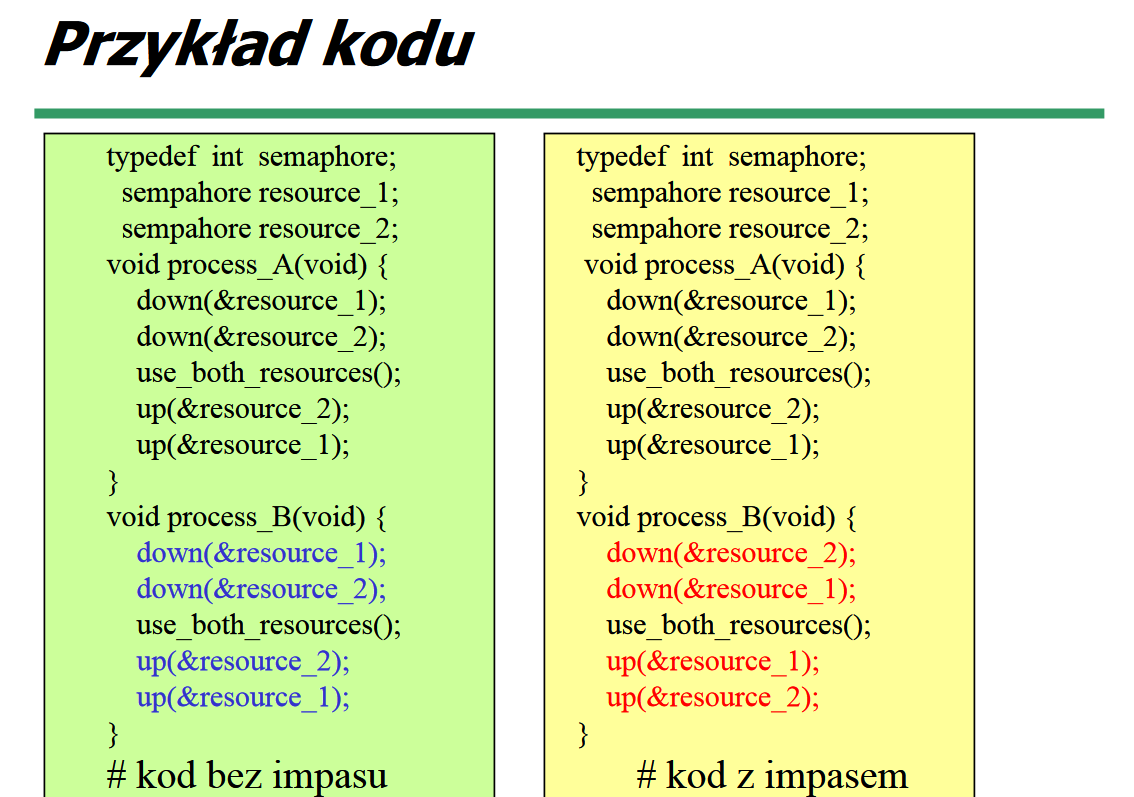
Narysuj diagramyGantta dla algorytmów FCFS, HRRN, SJF, SRTF, priorytetowy z i bezwywłaszczeń, RR (kwant=q)

oraz oblicz ich średnie czasy cyklu przetwa-rzania i oczekiwania. Uwaga: jeśli nie są podane

t1, t2, t3, t4, t5 to przyjmujemy, że t1=t2=t3=t4=t5= 0 oraz że procesy przybyły w porząd-kuP1, P2, P3, P4, P5

jeśli zaś nie są podanep1, p2, p3, p4, p5to przyjmujemyp1=p2=p3=p4=p5

X105. Podaj przykład kodu programu z impasem i bez impasu.



X106. Podaj definicję impasu.

✦Kilka procesów blokuje jakiś zasób i czeka na otrzymanie innego zasobu blokowanego przez jakiś proces z tego zbioru

✦Efekt -może się zdarzyć, że oczekujące procesy nigdy już nie odmienią swego stanu, ponieważ zamawiane przez nie zasoby są przetrzymywane przez inne procesy

✦Nastąpił impas, zakleszczenie, blokada (ang. deadlock)

✦We współczesnych systemach operacyjnych

✦z jednej strony wzrasta liczba procesów, aplikacji wielowątkowych i zasobów

✦z drugiej strony mamy coraz więcej długowiecznych serwerów ( np.plików i baz danych)

✦coraz większe prawdopodobieństwo wystąpienia impasu

✦brak środków do zapobiegania impasom

✦Przykład 1

✦system ma trzy przewijaki taśmowe

✦procesy P1, P2 i P3przetrzymują po jednym przewijaku i żądają któregoś z dwu pozostałych

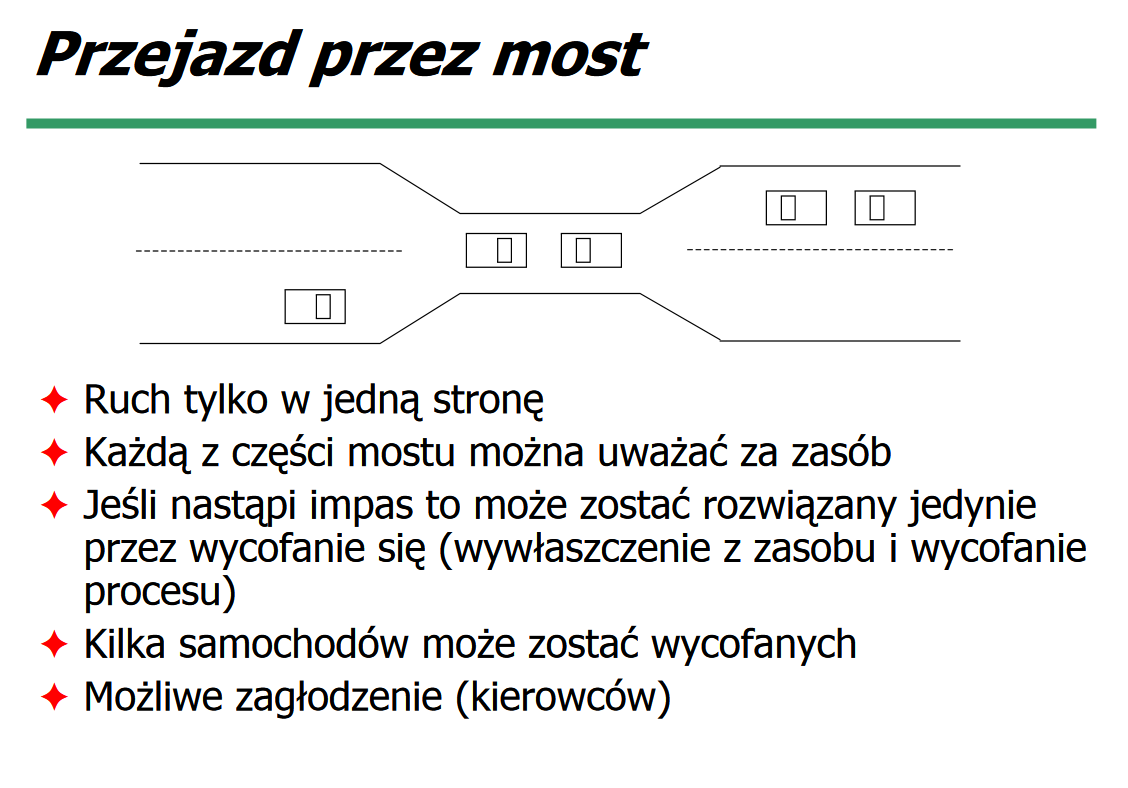
✦Przykład 2

✦proces pierwszy pisze na drukarkę (D), drugi na taśmę (T) jednocześnie proces pierwszy żąda taśmy a drugi drukarki

P1 P2

write(D); write(T)

open(T); open(D)



107. Kiedy może wystąpić impas w systemie?

✦Rodzaje (typy) zasobów R1, R2, . . ., Rmcykle procesora, obszar pamięci, urządzenia we/wy

✦Każdy typ zasobuRima Wiegzemplarzy

✦Każdy proces wykorzystuje zasób w następującym porządku

✦zamówienie (ang. request) –jeśli zamówienie nie może byćspełnione, to proces musi zaczekać

✦użycie (ang. use) –proces korzysta z zasobu

✦zwolnienie (ang. release) –proces oddaje zasób

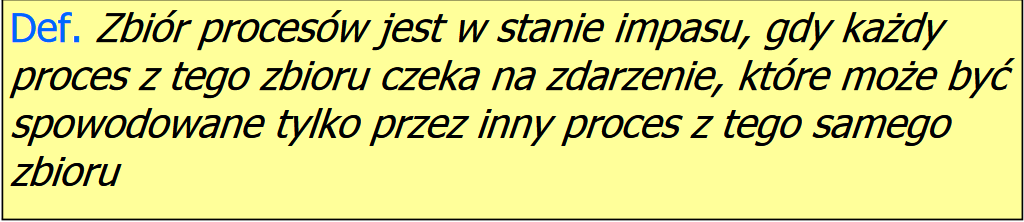
✦Zasoby można zamawiać i zwalniać za pomocą funkcji systemowych

✦urządzenia –{request| release} device

✦pliki –{open| close} file

✦pamięć –{allocate| free} memory

✦semaforowych operacji czekaj i sygnalizuj



✦zasoby fizyczne: drukarki, taśmy, pamięć, cykle procesora

✦zasoby logiczne: pliki, semafory, monitory

✦komunikacja : międzyprocesorowa

✦Do impasów może dochodzić wtedy, kiedy w systemie zachodzą jednocześnie cztery warunki

✦wzajemne wykluczanie (ang. mutual exclusion)

✦przetrzymywanie i oczekiwanie (ang. hold and wait)

✦brak wywłaszczeń (ang. no preemption)

✦czekanie cykliczne (ang. circular wait)

✦Warunek konieczny wystąpienia impasu

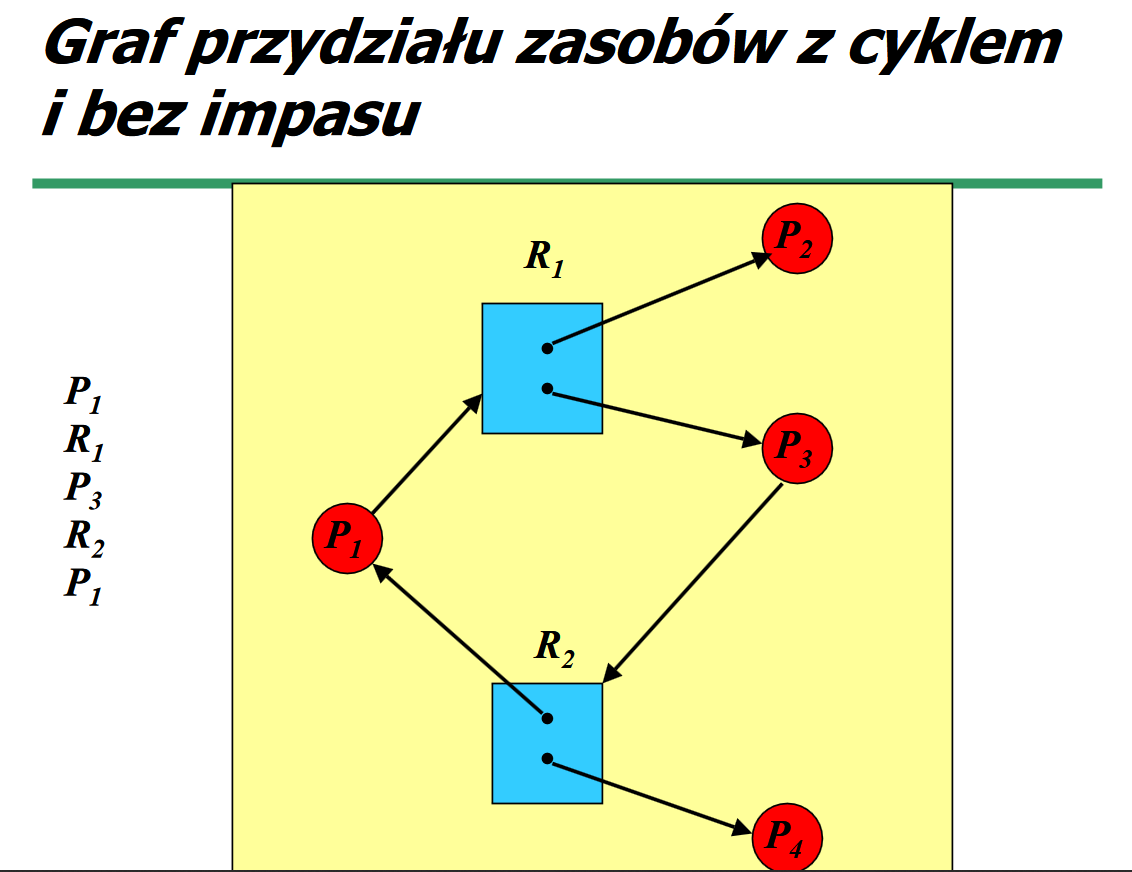
✦Wzajemne wykluczanie zajdzie wtedy gdy przynajmniej jeden zasób jest niepodzielny; zasobu w danym czasie może używać tylko jeden proces

✦Przetrzymywanie i oczekiwanie zajdzie wtedy jeśli istnieje proces, któremu przydzielono co najmniej jeden zasób i który oczekuje na przydział dodatkowego zasobu, przetrzymywanego przez inny proces

✦Brak wywłaszczeń zajdzie wtedy gdy zasób może zostać zwolniony jedynie z inicjatywy przetrzymującego go procesu (np. po zakończeniu)

✦Czekanie cykliczne zajdzie wtedy jeśli istnieje zbiór czekających procesów{ P1, P2, P3..., Pn }, takich że proces P1czeka na zasób trzymany przez proces P2, proces P2czeka na zasób trzymany przez proces P3, ..., proces Pn–1czekana zasób trzymany przez proces Pn, oraz proces Pnczeka na zasób trzymany przez proces P1

X108. Podaj przykład grafu przydziału zasobów z cyklem i bez impasu.



109. Opisz sposób zapobiegania impasom.

✦Stosowanie protokołu gwarantującego, że system nigdy nie wejdzie w stan impasu

✦Pozwala się systemowi na impas, po czym podejmuje się działania w celu jego usunięcia

✦Zlekceważenie problemu i założenie, że stan impasu nigdy się nie pojawi w systemie

✦większość systemów operacyjnych w tym system UNIX

✦Zapobieganie impasom (ang. deadlock prevention) -zbiór metod zapewniających, że co najmniej jeden z warunków koniecznych do wystąpienia impasu nie będzie spełniony

✦Unikanie impasu (ang. deadlock avoidance) -zapewnienie aby system zawczasu dysponował dodatkowymi informacjami o zasobach wykorzystywanych przez proces

✦Jeśli system nie korzysta z algorytmu zapobiegania impasom ani ich unikania to może mieć miejsce impas

✦w takim razie system powinien korzystać z

✦algorytmu sprawdzana stanu

✦algorytmu likwidowania impasu

✦Zapewnić, aby nie mógł wystąpić przynajmniej jeden z czterech warunków koniecznych:

✦Wzajemne wykluczanie -dotyczy jedynie zasobów niepodzielnych (np. drukarek); nie można zaprzeczyć temu warunkowi bowiem niektóre zasoby są z natury niepodzielne

✦Przetrzymywanie i oczekiwanie -aby zapewnić, że warunek ten nigdy nie wystąpi w systemie trzeba zagwarantować, że gdy proces zamawia zasób to nie powinien trzymać innych zasobów

✦każdy proces zamawia i otrzymuje wszystkie swoje zasoby przed rozpoczęciem działania (np. najpierw wywołania systemowe dotyczące zasobów, potem pozostałe)

✦proces może zamówić zasób o ile nie ma żadnych zasobów

✦zanim proces zamówi zasób musi wpierw wszystkie oddać

✦np. kopiowanie danych z taśmy na dysk, sortowanie i drukowanie

✦wady: słabe wykorzystanie zasobów, możliwość głodzenia

✦Brak wywłaszczeń -trzeba zapewnić, że zasoby nie ulegają wywłaszczeniu jedynie z inicjatywy przetrzymującego je procesu

✦jeśli proces żąda zasobu i nie może go otrzymać to traci wszystkie swoje zasoby i czeka na nie

✦jeśli proces zamawia jakieś zasoby to się sprawdza czy sąone dostępne

✦jeśli są dostępne to się je przydziela

✦jeśli zasoby te są trzymane przez proces, który sam czeka na jakieś zasoby to się mu te zasoby te odbiera i przydziela zamawiającemu

✦w przeciwnym razie proces czeka i może utracić zasoby tylko wtedy gdy inny proces ich zażąda

✦proces zostanie wznowiony gdy otrzyma nowe zasoby i odzyska utracone

✦Czekanie cykliczne -sposobem aby warunek ten nigdy nie wystąpił w systemie jest przyporządkowanie wszystkim zasobom danego typu kolejnych liczb i żądanie, aby

✦każdy proces zamawiał zasoby we wzrastającym porządku numeracji

✦alternatywnie można wymagać aby proces zamawiający zasób miał zawsze zwolnione zasoby o numeracji wyższej od zamawianego

✦Porządek używania zasobów

✦np. taśmy -1, dyski -5, drukarki -12

110. Opisz algorytm unikania impasu dla zasobów reprezentowanych jednokrotnie.

✦Wymaga dodatkowej informacji o tym jak będzie następowało zamawianie zasobów

✦W najprostszym i najbardziej użytecznym modelu zakłada się, że system zadeklaruje maksymalną liczbęzasobów każdego typu, którą mógłby potrzebować

✦Stan przydziału zasobów (ang. resource-allocation state) jest określony przez liczbę dostępnych (ang. available) i przydzielonych (ang. allocated) zasobów oraz przez maksymalne zapotrzebowanie procesów

✦Algorytm unikania impasu (ang. deadlock-avoidance) sprawdza dynamicznie stan przydziału zasobów aby zapewnić, że nigdy nie dojdzie do spełnienia warunku czekania cyklicznego

111. Podaj definicję stanu bezpiecznego.

✦Kiedy proces żąda wolnego zasobu system musi zadecydować czy przydzielenie zasobu pozostawi system w stanie bezpiecznym

✦System jest w stanie bezpiecznym (ang. safe) jeśli istnieje taki porządek przydzielania zasobów każdemu procesowi, który pozwala na uniknięcie impasu

✦System jest w stanie bezpiecznym jeśli istnieje bezpieczny ciąg procesów

✦System jest w stanie zagrożenia (ang. unsafe) jeśli nie jest w stanie bezpiecznym

✦Ciąg procesów <P1, P2, ..., Pn> jest bezpieczny jeśli dla każdego procesuPi , jego potencjalne zapotrzebowanie na zasoby może byćzaspokojone przez bieżąco dostępne zasoby + zasoby użytkowane przez wszystkie procesy Pj , przy czym j<i

✦jeśli Piżąda zasobów, które nie są natychmiast dostępne, to Pimoże poczekać aż skończą się wszystkie procesy Pj

✦jeśli Pjskończą, Pimoże otrzymać wszystkie potrzebne zasoby, dokończyć pracę, zwolnić zasoby i zakończyć

✦jeśli Pizakończy, Pi+1może otrzymać niezbędne zasoby itd.

✦Jeśli system jest w stanie bezpiecznym to nie ma impasu

✦Jeśli system jest w stanie zagrożenia istnieje możliwość impasu

✦Unikanie impasu to zapewnienie, że system nigdy nie wejdzie w stan zagrożenia

112. Opisz algorytm bankiera.

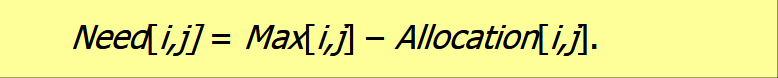
✦n -liczba procesów w systemie; m-liczba typów zasobów

✦Dostępne (ang. Available) : wektor długości m. Jeśli Available[j] = k, to jestkegzemplarzy zasobu typu Rjdostępnych w systemie

✦Maksymalne (Max) : macierz n x m. Jeśli Max[i,j] = k, to proces Pimoże zażądać co najwyżejk egzemplarzy zasobu typu Rj.

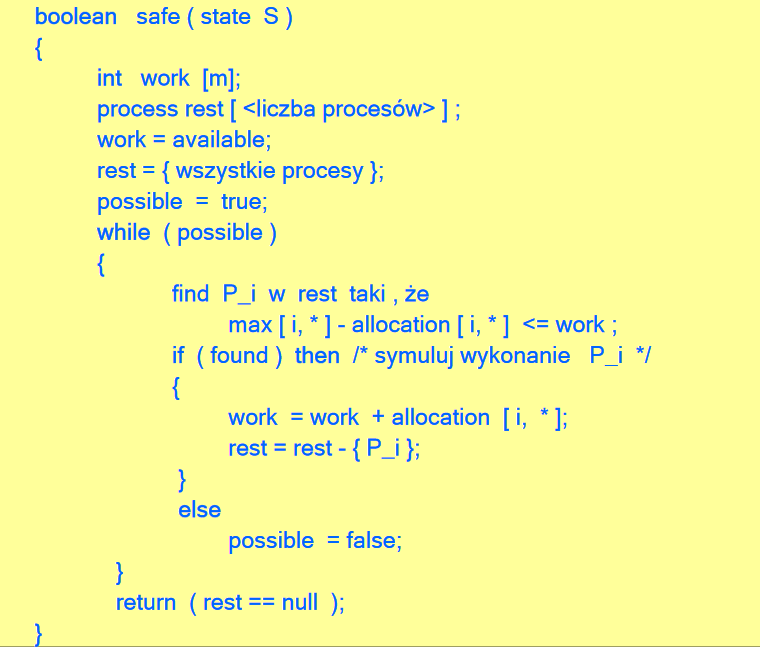
✦Przydzielone (ang. Allocation) : macierz n x m. Jeśli Allocated[i,j] = kto procesPima przydzielonych kegzemplarzy zasobu o typie Rj.

✦Potrzebne (ang. Need): macierz n x m. Jeśli Need[i,j] =k, to procesPimoże jeszcze potrzebowaćkdodatkowych egzemplarzy zasobu typu Rjaby zakończyć



✦X≤Ygdy X[i]≤Y[i] dla i=1,2,..n

1.Niech Work i Finishbędą wektorami długościmi n, odpowiednio. PrzypiszmyWork = AvailableFinish [i] =false fori= 1,2,3, ..., n.2.Znajdujemy i takie, że zarówno (a) Finish[i] == false(b) Needi≤Workw przeciwnym razie wykonujemy krok 43.Work= Work + AllocationiFinish[i] =true goto 2.4.Jeśli Finish[i] == true dla i<n+1, to stan bezpieczny



Cały algorytm w wykładzie 10 str. 50

113. Opisz algorytm zamawiania zasobów.

Requesti - wektor żądań Pi. Jeśli Requesti[j] == kto Pi chce k egzemplarzy zasobu typu Rj

1. Jeśli Requesti≤Needi goto 2. W przeciwnym razie błąd

2.JeśliRequesti≤Available, goto 3. W przeciwnym razie Pi czeka

3.System próbuje zaalokować zasoby Pimodyfikującstan

Available = Available –Requesti ;

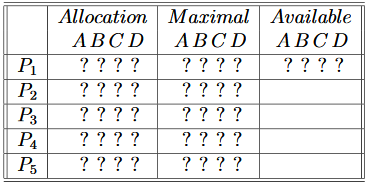
Allocationi= Allocationi+ Requesti ;

Needi=Needi–Requesti ;

•Jeśli stan safe ⇒alokuj zasoby do Pi .

•Jeśli stan unsafe ⇒Pi musi czekać na realizacjęi poprzedni stan przydziału zasobów jest odtwarzany

114. Dana jest migawka systemu w chwilit0:(tu zdjęcie)



Posługując się algorytmem bankiera określ czy system jest w stanie bez-piecznym. Jeśli procesP?złoży zamówienie (?,?,?,?) to czy jest możliwejego spełnienie?

115. Opisz algorytm wykrywania impasu dla zasobów reprezentowanych jedno-krotnie.

✦W systemie, w którym nie stosuje się algorytmu zapobiegania impasom ani ich unikania może wystąpić impas

✦Musz istnieć w systemie algorytmy

✦wykrywania impasu

✦likwidowania impasu

✦Zasoby reprezentowane jedno-i wielokrotnie

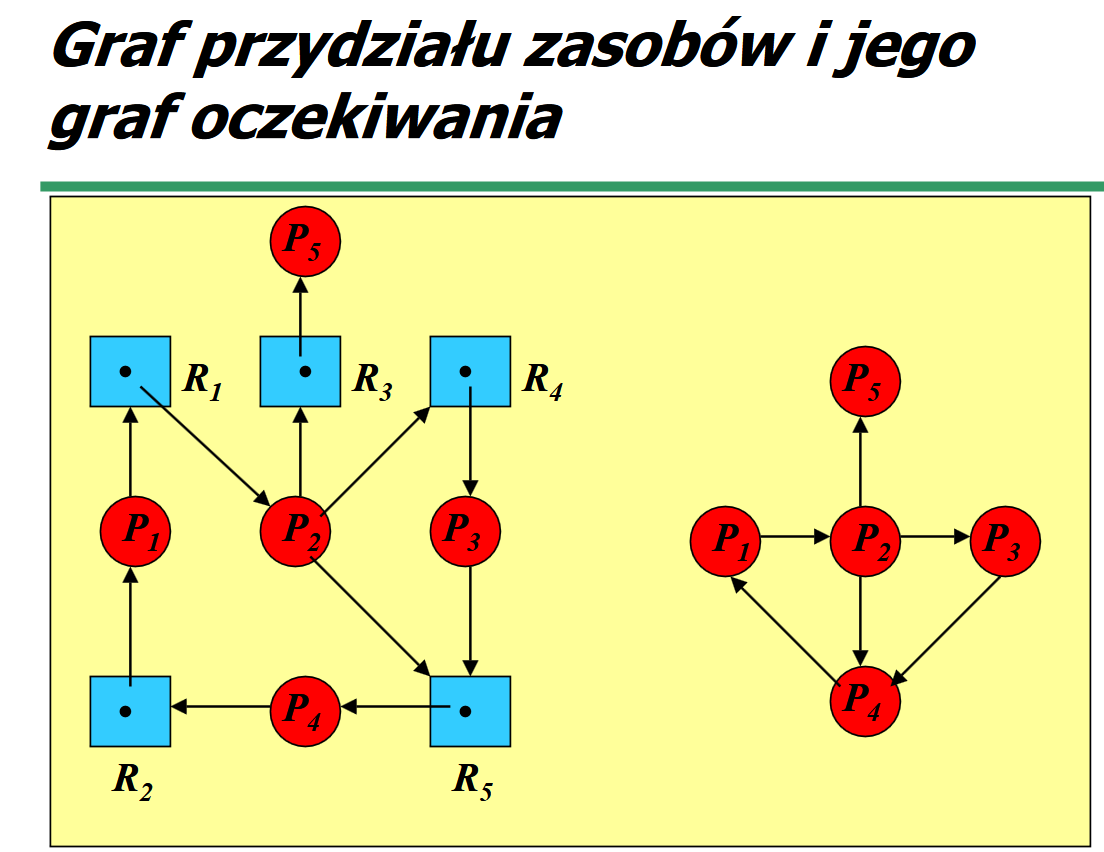
✦Tworzymy graf oczekiwania (ang. wait-for graph)

✦węzły grafu są procesami

✦Pi→Pj gdy Piczeka naPj.

✦Okresowo wykonujemy algorytm szukający cyklu w grafie oczekiwania

✦Rząd algorytmu wykrywania cykli w grafie oczekiwania wynosi n2, przy czym njest liczbąwierzchołków grafu



116. Opisz algorytm wykrywania impasu dla zasobów reprezentowanych wielo-krotnie.

✦Metoda grafu oczekiwania jest bezużyteczna do wykrywania impasu gdy każdy typ zasobu ma wiele egzemplarzy

✦Algorytm wykrywania impasu bada czy istnieje ciąg bezpieczny dla procesów, które trzeba dokończyć

✦korzysta ze struktur danych algorytmu bankiera

✦Available:wektor długości moznacza liczbę dostępnych zasobów każdego typu

✦Allocation:macierz n x mdefiniuje liczbę zasobów każdego typu aktualnie zaalokowanych do każdego z procesów

✦Request:macierz n x mokreśla bieżące zamówienie każdego procesu. Jeśli Request [i,j] = k, to procesPizamawia dodatkowokegzemplarzy zasobu typu Rj.

1.Niech Work i Finish wektory długości m i n, odpowiednio. Przypiszmy:

(a) Work = Available

(b)For i 1,2, ...,n if Allocationi ≠ 0then Finish[i] = false;

otherwise, Finish[i] := true

2.Znajdujemy indeks i taki, że jednocześnie:

(a)Finish[i] ==false

(b)Requesti≤Work

Gdy nie istnieje takie i, goto 4.

3.Work= Work+ Allocationi

Finish [i ] = true

goto 2.

4.Jeśli Finish[i] == false dla pewnego i, 1 ≤i≤n, to system jest w stanie impasu.

Co więcęj, jeśli Finish [i ] ==false, to Pi jest w impasie

✦Rząd liczby operacji tego algorytmu, potrzebny do wykrycia czy system jest w stanie impasu wynosi m x n2

Przykład algorytmu w wykładzie 10 str. 63

117. Opisz sposoby likwidowania impasu.

✦Kiedy i jak często należy wywoływać algorytm wykrywania impasu? Odpowiedź zależy od tego

✦jak często może wystąpić impas

✦ile procesów popadnie w impas w przypadku jego wystąpienia

✦Jeśli algorytm impasu jest wywoływany losowo to może występować wiele cykli w grafie zasobów i w efekcie identyfikacja procesu, który spowodował impas może być utrudniona

✦Zaniechanie (abort) wszystkich zakleszczonych procesów

✦Usuwanie procesów (wywłaszczanie) pojedyńczo, aż do wyeliminowania cyklu impasu

✦Na kolejność zaniechania (abort) ma wpływ

✦priorytet procesu

✦długość czasu wykonania i czas pozostały do zakończenia

✦ilość zasobów danego typu użytkowanych przez proces

✦zasoby i typy potrzebne do zakończenia

✦ilość procesów do przerwania

✦interakcyjność lub wsadowość

✦Usuwanie procesów (wywłaszczanie)

✦Wybór ofiary -minimalizowanie kosztów

✦Wycofanie (ang. rollback) -wycofanie procesu do bezpiecznego stanu, od którego można go będzie wznowić

✦Głodzenie -ten sam proces może być zawsze ofiarą, podobnie jak i ten sam proces może byćciągle wycofywany. Trzeba zadbać aby proces mógł być delegowany do roli ofiary tylko skończoną liczbę razy

✦Podejście mieszane

✦Podstawowe strategie

✦zapobieganie (ang. prevention)

✦unikanie (ang. avoidance)

✦wykrywanie (ang. detection)

✦Strategie stosowane osobno nie nadają się do rozwiązywania problemów z przydzielaniem zasobów

✦Podział wszystkich zasobów na hierachicznie uporządkowane klasy

✦W obrębie klas dobieramy najodpowiedniejsze techniki pokonywania impasu

✦System składa się z klas zasobów

✦zasoby wewnętrzne (bloki kontrolne)

✦pamięć główna

✦zasoby zadania (urządzenia, pliki)

✦wymienny obszar pamięci pomocniczej

✦Strategie postępowania z impasami w obrębie każdej z klas

✦porządkowanie zasobów

✦wywłaszczanie

✦unikanie impasu

✦wstępny przydział

137. Wymień zalety pamięci wirtualnej.

138. Wymień sposoby implementacji pamięci wirtualnej.

139. Opisz stronicowanie na żądanie.

140. Opisz procedurę obsługi braku strony.

141. Podaj wzór na obliczenie sprawności stronicowania.

142. Opisz procedurę zastępowania stron.

143. Opisz algorytm zastępowania stron FIFO.

144. Skonstruuj przykład ilustrujący anomalię Belady’ego.

145. Opisz algorytm zastępowania stron LRU.

146. Opisz sposoby implementacji algorytmu LRU.

147. Opisz algorytm zastępowania stron OPT.

148. Scharakteryzuj algorytmy stosowe.

149. Opisz algorytm dodatkowych bitów odniesienia.

150. Opisz zegarowy algorytm drugiej szansy (CLOCK).

151. Rozważmy następujący ciąg odniesień:?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?Ile braków stron wystąpi gdy mamy x, y lub z ramek dla algorytmów FIFO,OPT, LRU, CLOCK ?

152. Opisz ulepszony algorytm drugiej szansy.

153. Opisz schematy przydziału ramek.