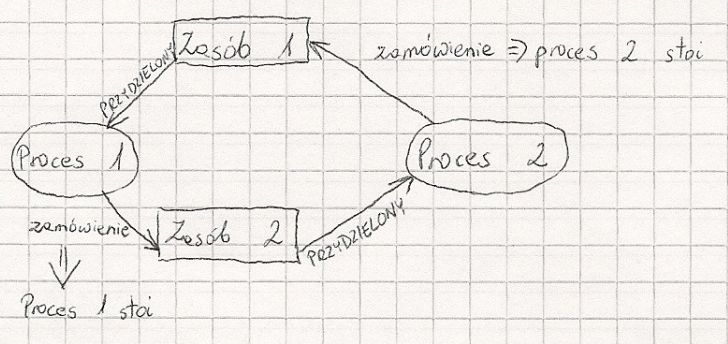
- Zaawansowane zagadnienia związane z wielodostępnymi systemami operacyjnymi.

23.02.2012
SO2
WYKŁAD 1

Problem zakleszczania (blokowania) procesów:

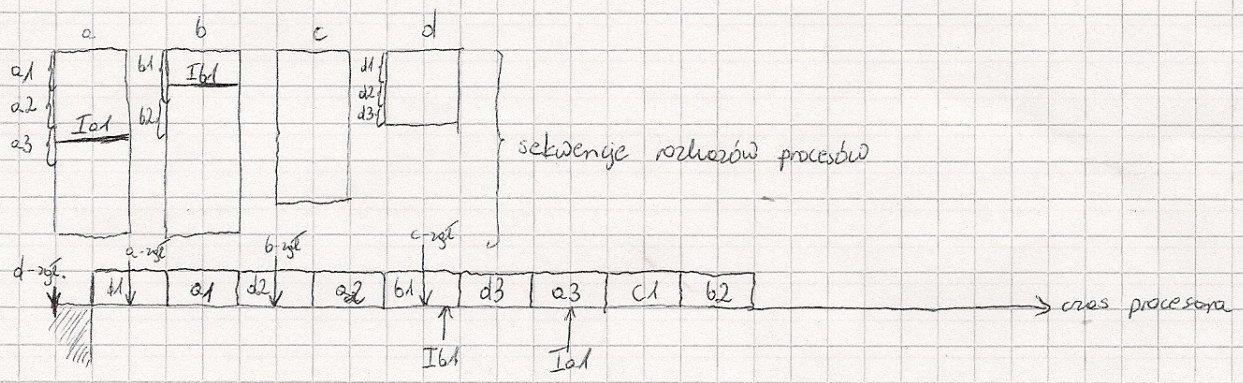


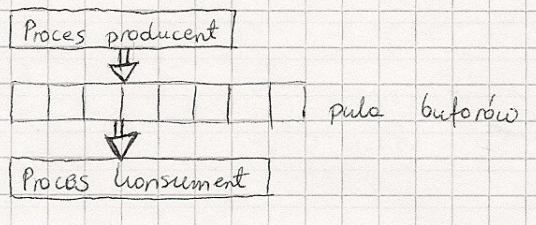
Efekt: 2 procesy stoją i jeśli się nic nie zrobi, to nigdy się nie wykonają.

Zakres tematyczny:

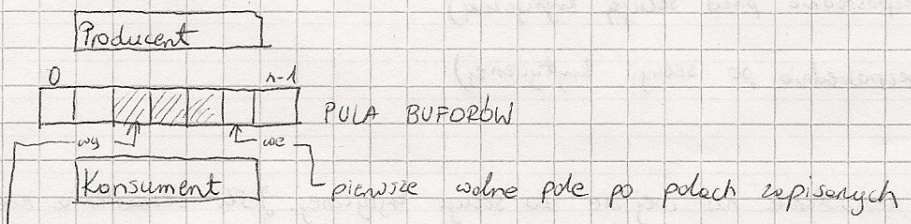
* Koordynacja procesów
* Komunikacja między procesami
* Blokady (zakleszczenia) procesów
* Wprowadzenie do systemów rozproszonych
* Wprowadzenie do sieci komputerowych (p. Aleksiejuk)

KOORDYNACJA PROCESÓW:





01.03.2012
SO2
WYKŁAD 2



zmienna licznik – liczba zapisanych buforów.

Mamy do czynienia z dzieloną pulą buforów, realizowaną jako tablica

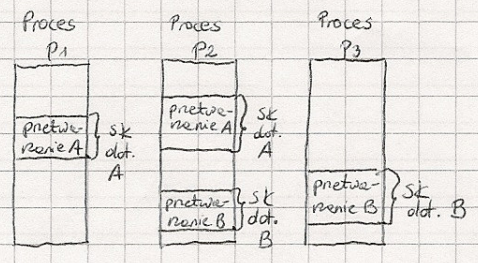
cykliczne z dwoma wskaźnikami logicznymi - we i wy.

Zmienna licznik służy do zatrzymania producenta, gdy wszystkie pule buforów są zapisane lub zatrzymania konsumenta, kiedy w puli buforów nic nie jest zapisane.

Zmienne dzielone - zmienne wspólne dla obu procesów. (tablica puli buforów)

Zmienne niezależne - lokalne dla danego procesu.

Występowanie błędów ma charakter losowy, ponieważ zależy od tego w jakiej kolejności zostaną wykonane rozkazy (kiedy nastąpi przełączenie procesora). Konieczne jest zsynchronizowanie procesów, w celu otrzymania prawidłowego wyniku.



P_1 i P_2 korzystają ze wspólnej zmiennej A.

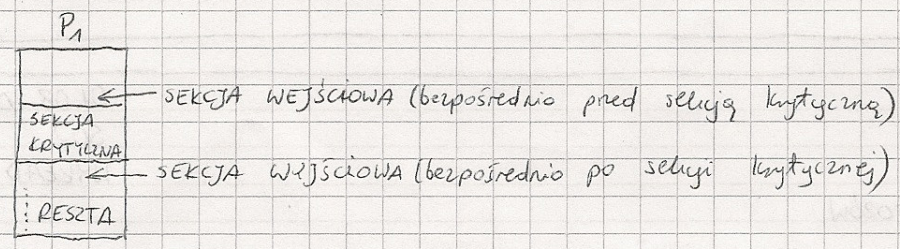
P_2 i P_3 korzystają ze wspólnej zmiennej B.

Sk (sekcja krytyczna) - segment kodu, w którym proces przetwarza wspólne zmienne.

Może być wiele sekcji krytycznych dotyczących wielu zmiennych w obrębie procesu, które mogą być wspólne dla wielu, różnych procesów.

Zasada wzajemnego wykluczania w czasie wykonania sekcji krytycznych procesów. Jeśli nastąpi przełączenie procesora w czasie wykonania sekcji krytycznej w procesie, to musi zostać wstrzymane wykonanie sekcji krytycznej w innym procesie, dotyczącej tej samej zmiennej do czasu aż sekcja krytyczna zostanie wykonana w pierwszym procesie.

Należy skonstruować protokół służący do organizacji współpracy procesów.



1) Sekcja wejściowa - prośba o zezwolenie na wejście do sekcji krytycznej. Jeśli zezwolenia nie ma, proces powinien czekać. Jeśli jest zezwolenie to sekcja krytyczna powinna zawierać instrukcje zatrzymujące inne procesy przed wejściem do sekcji krytycznej, dotyczącej tej

 samej zmiennej (lub obiektu)

2) Sekcja wyjściowa - sygnalizacja wyjścia z sekcji krytycznej. Powinna zawierać instrukcje sygnalizujące innym procesom, że mogą już wejść do sekcji krytycznej, dotyczącej tej samej zmiennej (lub obiektu).

01.03.2012
SO2
WYKŁAD 2

* Warunki rozwiązania problemu:

1 - wzajemne wykluczanie

2 - postęp

3 - ograniczone czekanie

Są to wymagania stawiane przed programistą, który musi rozwiązać problem sekcji krytycznej.

Ad. 1.

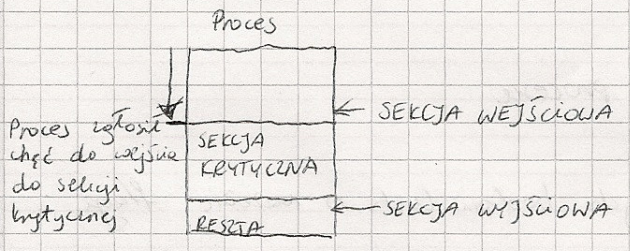
Jeśli pewien proces działa w sekcji krytycznej, to żaden inny proces w swojej sekcji krytycznej działać nie może.

Ad. 2.

Jeżeli żaden proces nie działa w sekcji krytycznej i istnieją procesy, które chcą wejść do swoich sekcji krytycznych, to tylko procesy nie wykonujące swoich reszt mogą kandydować do wejścia jako następne do tych sekcji krytycznych. Wybór tego nie może odwlekać w nieskończoność.

Ad. 3.

Po zgłoszeniu chęci wejścia danego procesu do sekcji krytycznej, liczba wejść innych procesów do ich sekcji krytycznych musi być ograniczona.



Warunek ograniczonego czekania ma zapobiec zjawisku głodzenia procesów (czyli proces, czekając na zgodę będzie czekał bardzo długo, nawet w nieskończoność)

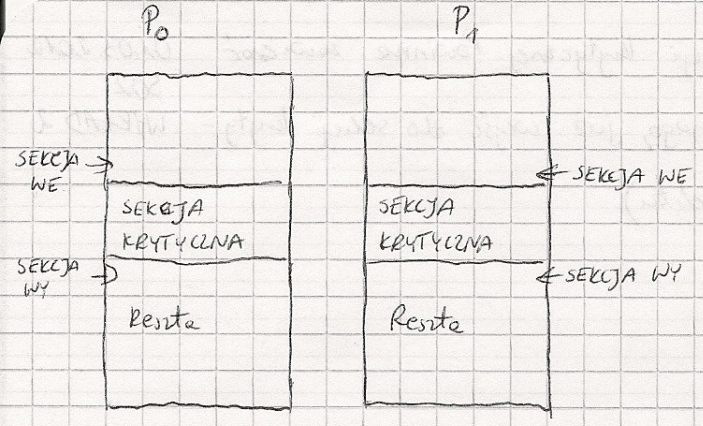
* Założenia do konstrukcji algorytmów

- niezerowa prędkość wykonywania procesów

- podstawowe rozkazy maszynowe wykonywane niepodzielnie

Rozwiązanie problemu sekcji krytycznej:

Założenie - mamy dwa współbieżne procesy P_0 i P_1 zawierające sekcje krytyczne.



$i=0 \Rightarrow j=1$

$i=1 \Rightarrow j=0$

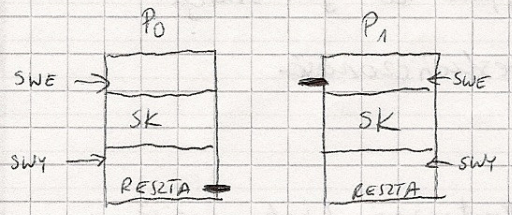
W celu rozwiązania wprowadzamy zmienną współdzieloną całkowitą, która może przyjmować wartość 0 lub 1. Zmienna numer.

Algorytm 1 - proces może wykonywać swoją sekcję krytyczną, kiedy zmienna numer ma taką wartość jak numer procesu.

Tylko jeden proces może wykonywać swoją sekcję krytyczną, zatrzymując drugi proces ⇒ spełniony warunek wzajemnego wykluczenia.

Natomiast nie jest spełniony warunek postępu!

numer = 0, P_1 gotowy do wejścia do sekcji krytycznej, P_0 wykonuje swoją resztę.



Proces P_0 jest już za sekcją wyjściową, czyli nie może już zmienić wartości zmiennej numer, czyli proces P_1 nigdy nie zostanie wykonany.

Algorytm zadziała tylko w sytuacji, gdy procesy P_0 i P_1 będą wykonywane naprzemiennie.

Problem - nie są przechowywane informacje o stanie procesu.

Algorytm 2 - wprowadzamy tablicę (wspólną) zmiennych boolowskich o nazwie flaga.

Spełniony warunek wzajemnego wykluczenia.

⇑ Co może się stać jeśli mamy 2 takie procesy: w procesie P_0 wykona się flaga[i] := true i nastąpi przełączenie do procesu P_1? ZASTANOWIĆ SIĘ.

Odp. Procesor będzie się przełączał między procesami, co spowoduje niespełnienie warunku postępu, bo obie flagi będą miały wartość "true".

⇑ z W2 Problem wykonania instrukcji:

T_0 : P_0 ustala flaga [0] = true;

→ przełączenie procesora.

T_1 : P_1 ustala flaga [1] = true;

Algorytm 3 - łączy koncepcje dwóch pierwszych algorytmów, spełnia trzy warunki rozwiązania problemu sekcji krytycznej.

* Sprzętowe środki synchronizacji - rozkazy sprzętowe pozwalające na sprawdzenie i zmianę zawartości słowa (słów) wykonywane jako niepodzielne jednostki.

Ważna cecha instrukcji: niepodzielne wykonanie!

W trakcie wykonywania tej funkcji zabronione jest przerwanie lub przełączenie procesora.

Albo wykona się w całości albo wcale.

* Semafory:

Semafor to zmienna całkowita dostępna tylko za pomocą standardowych, niepodzielnych operacji: czekaj (wait), sygnalizuj (signal).

- Zmienna współdzielona.
- Stanowi zasób systemu operacyjnego → zaimplementowane w systemie operacyjnym.
- Liczba zaimplementowanych semaforów jest skończona → trzeba więc z nich korzystać rozsądnie i oszczędnie, aby uniknąć sytuacji, kiedy nie będzie dostępnych semaforów.

1) Semafor trzeba zainicjalizować → zwrócić się do systemu operacyjnego z prośbą o użycie (udostępnienie) semafora i stwierdzić, jaką będzie miał nazwę.

2) Semaforowi trzeba nadać wartość początkową

Definicje operacji:

czekaj (S): while $S \leq 0$ do nic;

$S := S - 1$;

sygnalizuj (S): $S := S + 1$;

Semafor nie jest prostym rozwiązaniem, ponieważ jego prawidłowe działanie podlega wielu warunkom (ograniczeniom).

Ważna cecha: Niepodzielność wykonywania operacji.

Zastosowanie semaforów:

- do rozwiązania problemu sekcji krytycznej przy n procesach.

repeat

← sekcja wejściowa: czekaj(S)

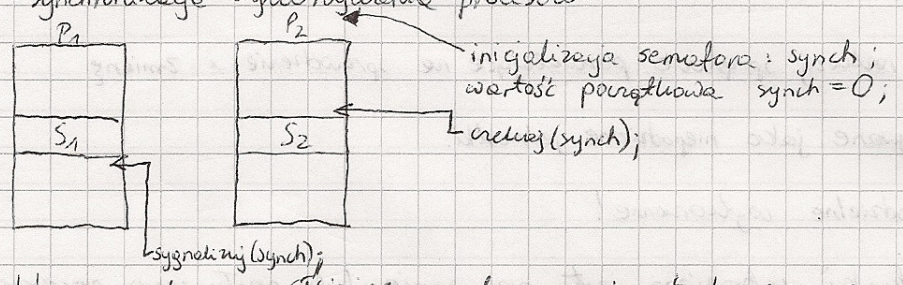
sekcja krytyczna

← sekcja wyjściowa: sygnalizuj(S)

reszta

until false

- synchronizacja wykonywania procesów



W przypadku współbieżnego wykonywania tych procesów bez użycia środków synchronizacji, instrukcje S_1 i S_2 mogą się wykonać w dowolnej kolejności, a reszta wymagana kolejność to - najpierw S_1, a potem S_2.

Zadanie 1. (dla wprawy)

Trzy współbieżne procesy: P_1, P_2, P_3

P_1 zawiera instrukcję S_1, P_2 - instrukcję S_2, P_3 - instrukcję S_3.

Stosując semafory, zapewnić, że instrukcje wykonają się w kolejności: S_3, S_1 i S_2.

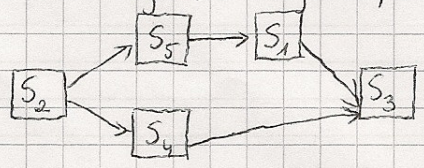
$S_3 \rightarrow S_1 \rightarrow S_2$ - graf określający kolejność wykonania tych instrukcji.

Zadanie 2. (egzaminacyjne)

Rozpatrzyć współbieżne procesy: P_1, P_2, P_3, P_4 i P_5

P_1 zawiera instrukcję S_1, P_2 - instrukcję S_2, P_3 - instrukcję S_3, P_4 - instrukcję S_4 i P_5 - instrukcję P_5.

Stosując semafory podać rozwiązanie zadania synchronizacji, polegającego na wykonaniu tych instrukcji w kolejności, określonej grafem.



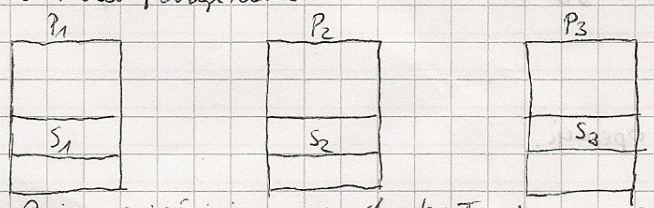
UWAGA: S_5 nie powinno czekać na S_4 i na odwrót,

S_1 nie czeka na S_4 i na odwrót.

Ad. 1 - SCHEMAT ROZWIĄZANIA

inicjalizacja semaforów: [2 semafory]

wartości początkowe:



[umieść odpowiednie sekcje wejściowe i wyjściowe]

⑥ Opis wyjaśniający sposób działania przy proponowanym rozwiązaniu (sprawdzić kombinacje przydziału

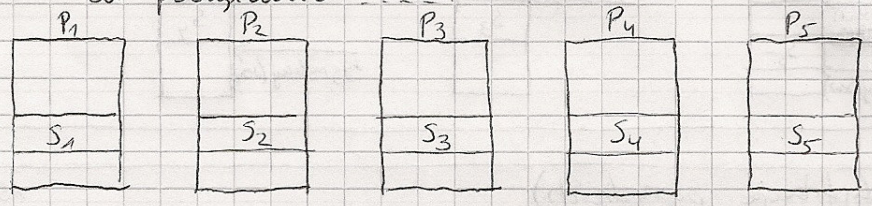
kwantu czasu dla poszczególnych procesów)

08.03.2012
SO2
WYKŁAD 3

Ad. 2 - SCHEMAT ROZWIĄZANIA

inicjalizacja semaforów: _____ [3 semafory]

wartości początkowe: _____



[naznaczyć odpowiednie sekcje wejściowe i wyjściowe]

Pamiętać o oszczędnym korzystaniu z semaforów!

Opis wyjaśniający sposób działania przy proponowanym rozwiązaniu.

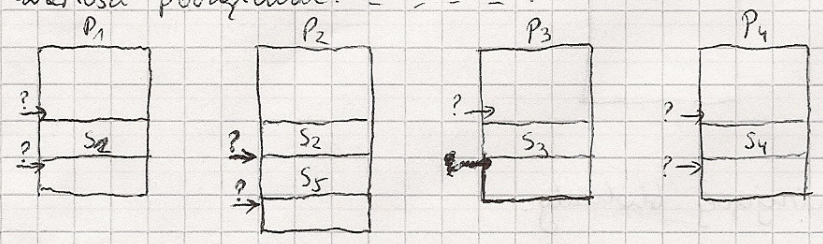
* Zadanie 3 (inny wariant) (graf taki jak w zadaniu 2)

Rozpatrzyć współbieżne procesy: P_1, P_2, P_3, P_4

P_1 - zawiera instrukcję P_1, P_2 - zawiera instrukcję S_2 i po niej S_5, P_3 - instrukcję S_3 i P_4 - inst. S_4

inicjalizacja semaforów: _ _ _ _

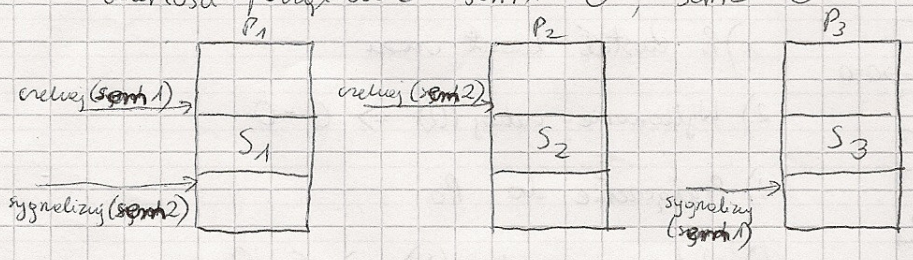
wartości początkowe: _ _ _ _ _



Zadanie 1

inicjalizacja semaforów: sem1 i sem2;

wartości początkowe: sem1 = 0, sem2 = 0

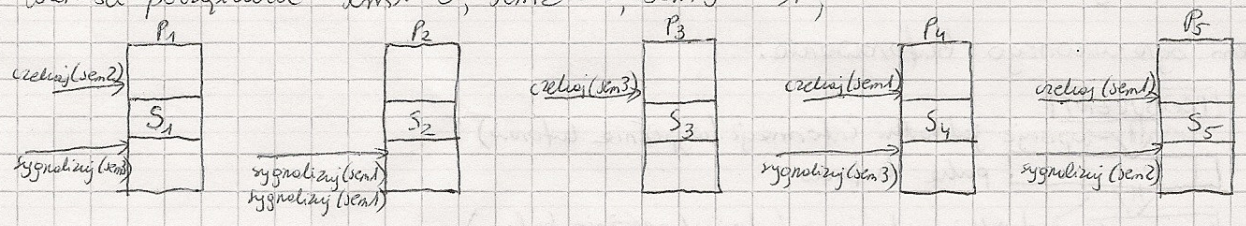


Najpierw wykonuje się S_3 i zwiększa synch 1 o 1, co odblokowuje S_1. Wykonuje się S_1 i zwiększa o 1 synch 2, co odblokowuje S_2. Wykonuje się S_2.

Zadanie 2:

inicjalizacja semaforów: sem1, sem2, sem3

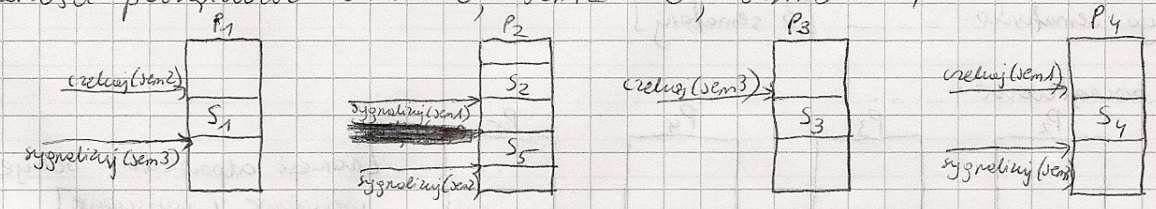
wartości początkowe: sem1 = 0, sem2 = 0, sem3 = -1;



Zadanie 3

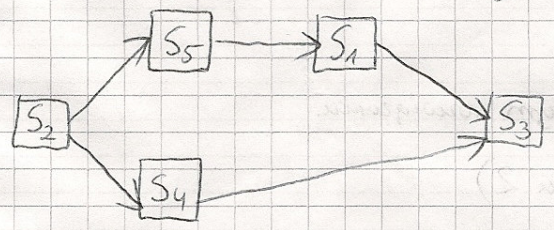
inicjalizacja semaforów: sem1, sem2 i sem3

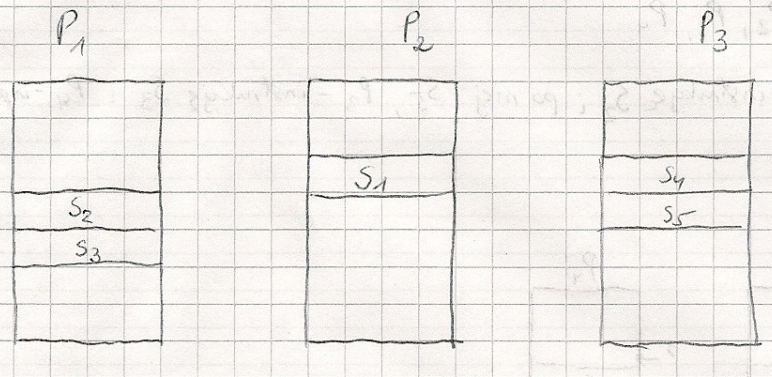
wartości początkowe: sem1 = 0, sem2 = 0, sem3 = -1



Zadanie o semaforach (modyfikacje poprzednich)

15.03.2012
SO2
WYKŁAD 4

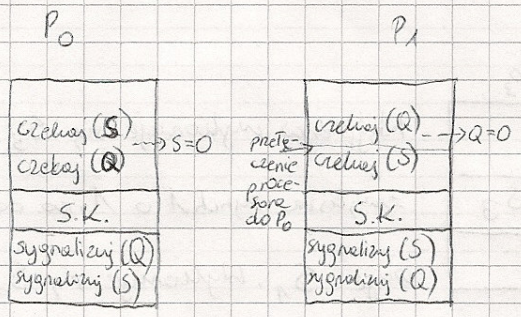




* Problem aktywnego czekania, winującej blokady.

* Problem blokowania.

Procesy P_0, P_1 semafory: S, Q wartości początkowe: $S=1$, $Q=1$



1.) P_1 dostał kwant czasu

2.) Wykonanie czekaj (Q) => Q = 0

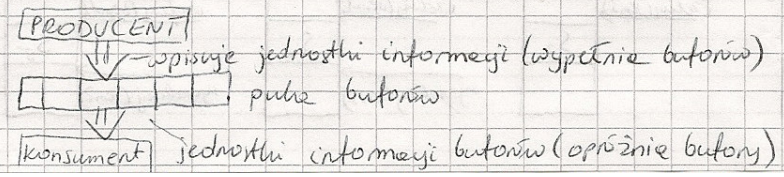
3.) Przełączenie do P_0

4.) Wykonanie czekaj (S) => S = 0

5.) Zablokowanie P_0 i P_1 bez względu na kolejność przełączeń procesora

* Wybrane problemy synchronizacji

+ Problem ograniczonego buforowania.



15.03.2012
SO2
WYKŁAD 4

- Zapewnienie wyłączności dostępu do puli buforów - semafor wzajemny
- Semafor pusty (wartość początkowa n) zawiera wartość pustych buforów w puli buforów.
- Semafor pełny (wartość początkowa 0) zawiera wartość pełnych buforów w puli buforów, zapisanych przez producenta.

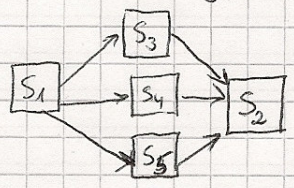
+ Problem czytelników i pisarzy

- Zmienna licz oznacza liczbę czytelników, odczytujących dany obiekt.
- Semafor wzajemny służy zapewnieniu wyłączności dostępu do zmiennej współdzielonej licz.
- Semafor pis jest wspólny dla pisarzy i czytelników. Zapewnia wyłączność dostępu do obiektu między pisarzami oraz między pisarzami a czytelnikami.

Zadanie:

22.03.2012
SO2
WYKŁAD 5

Mamy 4 procesy współbieżne. Proces P_1 ma instrukcje S1 i S2, P3 - S3, P4 - S4, P5 - S5. Stosując semafory, zapewnić kolejność wykonania tych instrukcji wg. schematu.



+ Problem posilających się filozofów:

* Konstrukcje synchronizacji w językach wysokiego poziomu:

Pomyłki programisty.

var. wzajemy : semafor

war. pocz wzajemy = 1;

P_i

sygnalizuj (wzajemy); wzajemy = 1

sekcja krytyczna

czekaj (wzajemy); wzajemy = 1

reszta

→ BRAK WZAJEMNEGO WYŁĄCZANIA SEKCJI KRYTYCZNEJ

var. wzajemy : semafor

war. pocz. wzajemy = 1;

P_i

czekaj (wzajemy); wzajemy = 0

sekcja krytyczna

czekaj (wzajemy);

reszta

→ PO WYKONANIU SEKCJI KRYTYCZNEJ (PIERWSZEJ) WSZYSTKIE PROCESY STOJĄ

+ Region krytyczny

+ Warunkowy region krytyczny.

+ Monitory

* Komunikacja między procesami:

- Określenie nadawców i odbiorców (komunikacja bezpośrednia).

- Wykorzystanie skrzynek pocztowych (portów) o jednoznacznej identyfikacji (komunikacja pośrednia)

Ilość tych skrzynek pocztowych jest ograniczona (skończona).

29.03.2012
SO2
WYKŁAD 6

- Własności skrzynki (procesu lub systemu operacyjnego)

1) specyficzna zmienna typu "skrzynka pocztowa". Proces, w którym ona by występowała, jest jej właścicielem. Może dać innym procesom dostęp do tej skrzynki.

PROBLEM: Zmienna istnieje tylko tak długo, jak istnieje proces, który ją powołuje.

2) skrzynka jest własnością samego procesu operacyjnego. Analogiczna idea jak dla semaforów. Proces musi "poprosić" o udostępnienie takiej skrzynki.

- Zagadnienia buforowania

Pojemność łącza:

1) tylko określona liczba komunikatów może w nim przebywać

2) kolejka komunikatów przyporządkowanych do łącza (jeśli może być ich kilka w łączu)

NADAWCA —ŁĄCZE→ ODBIORCA

- Sposoby implementacji kolejki - pojemność łącza:

1) pojemność zerowa (tylko jeden ~~proces~~ komunikat może być nadany i dopóki nie zostanie odebrany nie można nadać następnego

2) pojemność ograniczona (ilość komunikatów, które mogą być nadane po pierwszym nieodebranym)

3) pojemność nieograniczona (bufor jest na tyle duży, że procesy nie są w stanie go przekroczyć)

proces P wysyła komunikat do Q : P: nadaj (Q, komunikat)
odbierz (Q, komunikat)

Q : odbierz (P, komunikat)
nadaj (P, potwierdzenie_odbioru)

- Przykłady działań w sytuacjach wyjątkowych:

1) Zakończenie procesu, biorącego udział w komunikacji

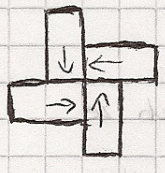
2) Utrata komunikatów

3) Zniekształcenie komunikatów

BLOKADY - ZAKLESZCZENIA PROCESÓW (deadlocks) 29.03.2012

SO2

Blokada - sytuacja, w której procesy wzajemnie przetrzymują zasoby, do których chciałyby uzyskać dostęp. W sytuacji takiej procesy są w stanie oczekiwania, z którego nie mogą wyjść. WYKŁAD 6



Zasoby:

- System zawiera skończoną liczbę zasobów różnego typu
- Mogą występować grupy równoważnych zasobów

Zasady korzystania z zasobów przez procesy:

- Zamówienie zasobu (żaden proces nie ma bezpośredniego dostępu do zasobu. Musi użyć odpowiedniej funkcji systemowej i "poprosić" system o przydzielenie zasobu)
- Użycie (kiedy system przydzieli procesowi zasób, to proces z niego korzysta. Jeśli nie, to proces musi stać)
- Zwolnienie zasobu (proces po zakończeniu korzystania z zasobu zwalnia dany zasób za pomocą odpowiedniej funkcji systemowej)

Przykłady zasobów:

- zasoby fizyczne: drukarki, napędy taśmy, cykle procesora, pamięć
- zasoby logiczne: pliki, semafory

Definicja: Zbiór procesów jest w stanie blokady, jeśli każdy proces z tego zbioru czeka na zdarzenie spowodowane przez inny proces z tego samego zbioru.

*Warunki konieczne wystąpienia blokady: (jeśli nie będą spełnione, to dana sytuacja nie wystąpi!)

1) Wzajemne wykluczanie:

co najmniej jeden zasób jest niepodzielny (tylko jeden proces może z niego korzystać w danym momencie), tylko jeden proces może korzystać z tego zasobu, inne procesy zamawiające ten zasób są opóźnione.

2) Przetrzymywanie i oczekiwanie

musi istnieć proces mający przydzielony pewien zasób (co najmniej jeden) i oze-

kujący na przydział dodatkowego zasobu przetrzymywanego przez inny proces.

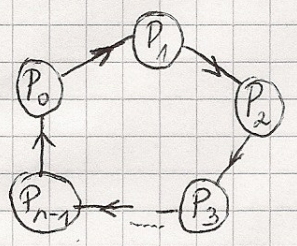
(Proces jednocześnie coś trzyma i czeka na coś, czego nie może dostać od razu)

3) Brak wywłaszczeń

tylko proces przetrzymujący określony zasób, może ten zasób zwolnić.

4) Czekanie cykliczne

musi istnieć zbiór oczekujących procesów $\{P_0, P_1, \dots, P_{n-1}\}$, takich że



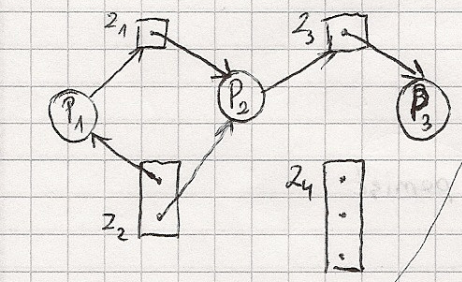
P_0 czeka na zasób przetrzymywany przez P_1 itd. aż P_{n-1} czeka na zasób przetrzymywany przez P_0 (tworzy się cykl!)

* Graf przydziału zasobów – graf skierowany opisujący stan systemu przydziału zasobów, umożliwia analizę sytuacji blokad.

Zbiór wierzchołków W składa się z podzbiorów: P – podzbiór procesów, Z – podzbiór typu zasobów

Zbiór krawędzi skierowanych K: $P_i \to Z_i$ proces P_i zamówił zasób Z_i, $Z_i \to P_i$ zasób Z_i został przydzielony procesowi P_i.

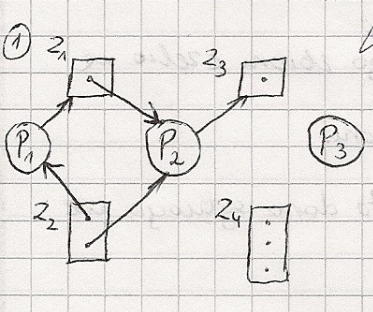
Strzałka zamówienia jest zawsze do brzegu zasobu Z_i



Istnieje możliwość wykonania wszystkich procesów w kolejności:

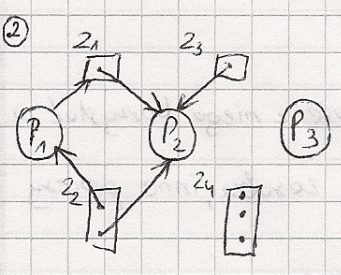
1) wykonanie P_3 i zwolnienie Z_3
2) przydział Z_3 do P_2
3) wykonanie P_2 i zwolnienie Z_1, Z_2, Z_3
4) przydział Z_1 do P_1
5) wykonanie P_1

(1)



* Warunek konieczny wystąpienia blokady – graf przydziału zasobów zawiera cykl (gdy nie ma cyklu = nie ma blokady, jest cykl = blokada może być lub nie). W przypadku, gdy każdy typ zasobów zawiera tylko jeden egzemplarz, to cykl zawarty w grafie przydziału zasobów jest warunkiem <u>koniecznym i dostatecznym</u>.

(2)



⋮

⇑ Narysować graf obrazujący następujący stan systemu przydziału zasobów: 29.03.2012
SO2
WYKŁAD 6

Zasoby: Z_1 posiada 4 egzemplarze, Z_2 posiada 2 egzemplarze,
Z_3 posiada 1 egzemplarz, Z_4 posiada 1 egzemplarz,
Z_5 posiada 2 egzemplarze

Procesy: • P_1 otrzymał dwa egzemplarze Z_1 i zamówił 1 egzemplarz Z_2 i 1 egzemplarz Z_3; P_2 otrzymał 2 egzemplarze Z_2 i zamówił 1 egzemplarz Z_4; P_3 otrzymał 1 egzemplarz Z_1 i 1 egzemplarz Z_3 i zamówił 1 egzemplarz Z_5; P_4 otrzymał 2 egzemplarze Z_5 i zamówił 1 egzemplarz Z_1 i 1 egzemplarz Z_3.

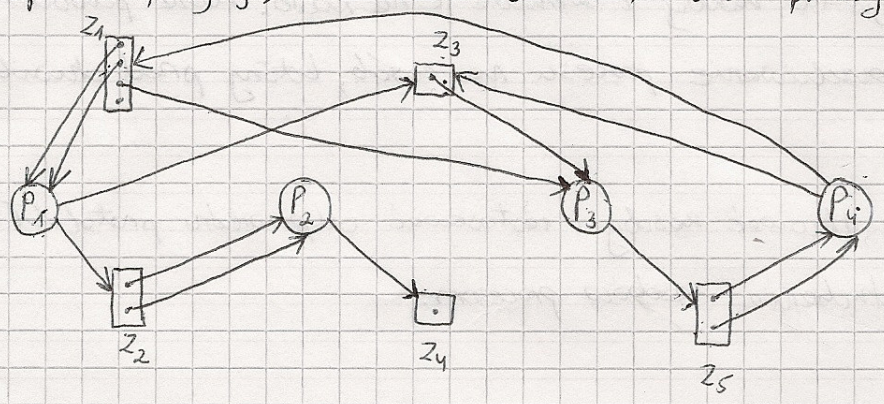
Co zrobić?

1.) Narysować graf

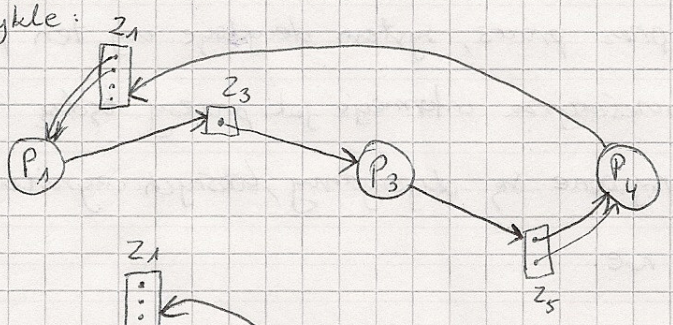
2) Znaleźć i policzyć cykle w tym grafie

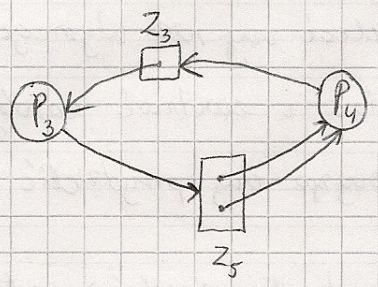
3) Sprawdzić, które z procesów mogą się wykonać i w jakiej kolejności

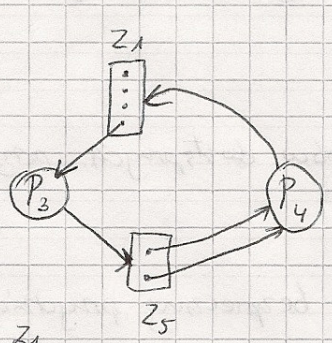
4) Sprawdzić, czy jest zakleszczenie (blokada) i które procesy tkwią w tym zakleszczeniu

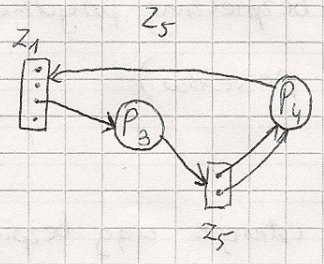


cykle:









Może się wykonać tylko proces P_2, pozostałe tkwią w zakleszczeniu:
P_3 czeka na zasób Z_5, który jest przydzielony do P_4, natomiast P_4 czeka na zasób Z_3, który jest przydzielony do P_3, w związku z czym P_1 również tkwi w zakleszczeniu.

- Sposoby rozwiązywania problemu blokady.

05.04.2012
SO2
WYKŁAD 7

· Zapobieganie wystąpieniu blokady
· Unikanie blokad
· Powodzić na wejście w stan blokady i spowodować jej usunięcie.

* Zapobieganie wystąpieniu blokady:

Wyeliminowanie co najmniej jednego z warunków koniecznych (wzajemnego wykluczenia, przetrzymywanie i oczekiwania oraz braku wywłaszczeń) czekania cyklicznego

Wzajemne wykluczenie – można wyeliminować tylko w przypadku zasobów podzielnych

Przetrzymywanie i oczekiwanie – zapewnienie, że proces zamawiający określony zasób nie powinien przetrzymywać innych. (*)

Brak wywłaszczeń – zastosować protokół wywłaszczeniowy.

(*) Przykład 1: na początku działania przydzielić procesowi wszystkie potrzebne zasoby

Przykład 2: przy zamówieniu kolejnego zasobu, proces zwalnia wszystkie przetrzymywane zasoby i zamawia je na nowo. Może powodować długie oczekiwanie procesu na zasób, który przed chwilą zwrócił.

Czekanie cykliczne – uporządkować zasoby i zastosować odpowiedni protokół przydzielania zasobów procesom.

* Unikanie blokad:

Idea: Przy każdym zamówieniu zasobów przez proces, system decyduje czy ten proces ma czekać czy nie. Wymagana jest ocześniejsza informacja jak procesy będą zamawiać i zwalniać zasoby. Wykonywane są algorytmy, których wynikiem jest decyzja czy przydzielić zasób czy nie.

Stan przydziału zasobów jest określony przez liczbę zasobów dostępnych, przydzielonych oraz przez maksymalne zapotrzebowania procesów.

· Stan bezpieczny – stan przydziałów dla którego istnieje bezpieczny przydział (ciąg procesów = kolejność wykonywania procesów)

- Stan zagrożenia – ciąg bezpieczny nie istnieje.

(14) Stan bezpieczny nie oznacza, że system jest wolny od blokad, ale istnieje ciąg bezpieczny.

* Rozwiązanie dla zasobów reprezentowanych wielokrotnie (algorytm bankiera): 12.04.2012 SO2 WYKŁAD 8

- Proces wchodzący do systemu musi zadeklarować maksymalną liczbę potrzebnych egzemplarzy każdego zasobu
- Za każdym razem, gdy proces zamawia zasób, system operacyjny decyduje czy przydzielić zasób, aby system przydziału zasobów był bezpieczny.

* Struktury danych przechowujące stan systemu przydziałów:

Relacja dominacji: $X \preceq Y$

wektor $X = \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}$ $Y = \begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}$ $Z = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

$$\underset{X}{\begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix}} \leq \underset{Y}{\begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}}$$

jaka jest relacja Z względem X? $\neg X \preceq Z$ $\neg Z \preceq Y$. SĄ NIEPORÓWNYWALNE W SENSIE TEJ RELACJI.

* Algorytm działań.

* Algorytm bezpieczeństwa

Zadanie: Dany stan systemu przydziału zasobów:

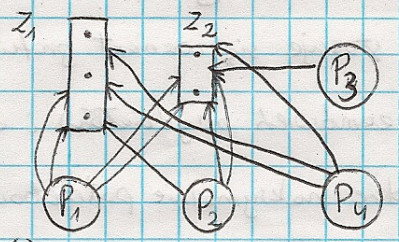
Zasoby:	Z_1	Z_2
egzemplarze	3	2

Procesy:	P_1	P_2	P_3	P_4
potrzebuje	$2 \times Z_1$	$1 \times Z_1$	$1 \times Z_2$	$2 \times Z_1$
	$2 \times Z_2$	$2 \times Z_2$		$1 \times Z_2$

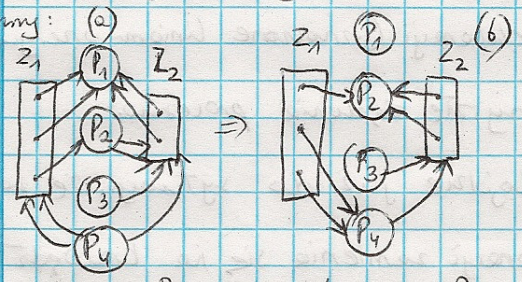
Pokazać 2 różne stany blokady, znaleźć ciąg bezpieczny przydziału zasobów, pokazać stan zagrożenia.

Rozwiązania przedstawić, wykorzystując grafy oraz podać uzasadnienie.

Sytuacja początkowa:



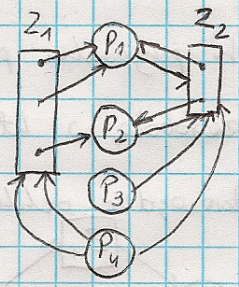
Ciąg bezpieczny:



wykonanie P_1 wykonanie P_2

$\Rightarrow$ c.d. na następnej stronie

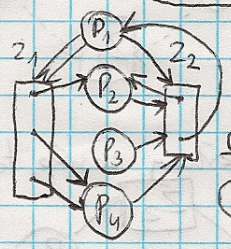
Blokada 1:



Wszystkie procesy stoją. Procesy P_1 i P_2 wzajemnie przetrzymują zasoby, jednocześnie nie mogąc się wykonać (czekanie cykliczne)

$P_1 \to Z_2 \to P_2 \to Z_2$

Blokada 2:



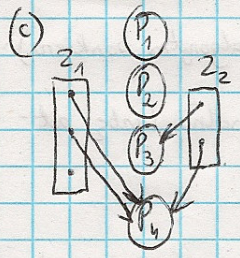
— " —

Z_1

$P_4 \to Z_2 \to P_2 \to Z_2 \to P_1$

$\to Z_2$

(c)



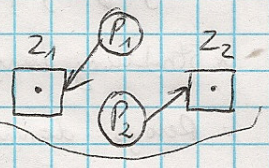
Kolejność wykonania procesów: P_1, P_2, P_3 i P_4

wykonanie P_3 i P_4

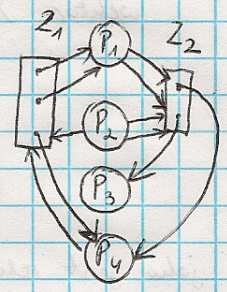
W ogólnym przypadku:

1) System może być wolny od blokad

przykład:



2) Może być taki stan przydziału zasobów, że nie będzie w ogóle ciągu bezpiecznego. przykład:



Po wykonaniu P_3, proces P_3 zwolni 1 egzemplarz zasobu Z_2, ale nastąpi blokada, bo jest wolny tylko 1 egzemplarz Z_2 i bez względu na sposób jego przydzielenia, żaden proces nie będzie mógł się wykonywać.

⇑ Dany jest stan systemu przydziału zasobów

Zasoby: $Z_1 \times 4$, $Z_2 \times 4$.

Procesy:	P_1	P_2	P_3	P_4
potrzebuje	$2 \times Z_2$	$3 \times Z_1$ $2 \times Z_2$	$2 \times Z_1$ $3 \times Z_2$	$2 \times Z_1$ $3 \times Z_2$

Pokazać 2 różne stany blokady, znaleźć i pokazać ciąg stanów bezpiecznych, pokazać stan zagrożenia.

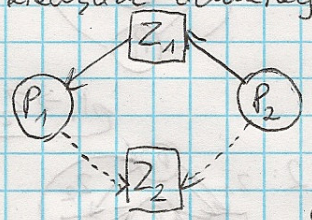
! Stosować grafy, podać uzasadnienie.

DODATKOWE: (nie ma na slajdach)

* Algorytmy unikania blokady w przypadku pojedynczych zasobów

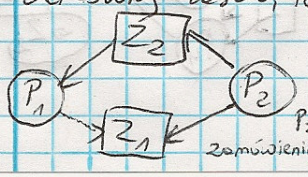
Algorytm bankiera: złożoność pesymistyczna $m \times n^2$ operacji do rozstrzygnięcia, czy badany stan jest bezpieczny.

W przypadku pojedynczych zasobów stosuje się wariant grafu przydziału zasobów, w którym są krawędzie zamówień przydziałów i dodatkowo krawędzie deklaracji (to, co proces będzie maksymalnie potrzebował).



Krawędzie deklaracji (oznaczone liniami przerywanymi) oznaczają przyszłe żądania procesów.

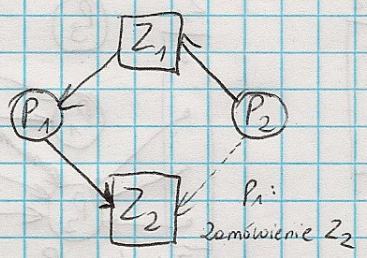
Gdy proces dojdzie już do sytuacji, że zamawia określony zasób, to krawędź deklaracji zamienia się na krawędź przydziału.



P_2: zamówienie Z_2

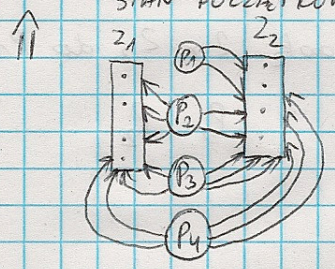
W algorytmie sprawdza się, co by się stało, gdyby to nowe zamówienie zostało spełnione. (jaki powstałby stan)

W grafie tym uruchamiamy algorytm szukania cyklu. Jeśli powstałby- nie przydzielać, gdyby nie było cyklu – można przydzielić.

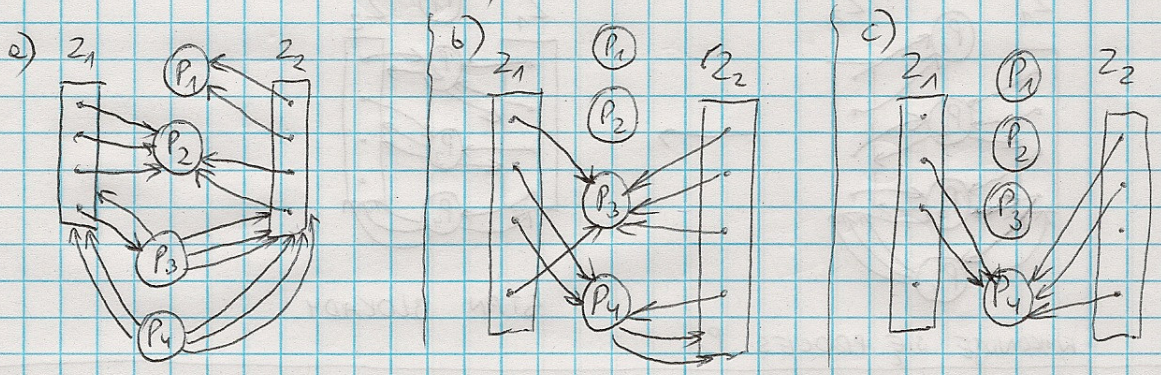


Nie powstałby cykl, więc można spełnić zamówienie P_1.

STAN POCZĄTKOWY



CIĄG STANÓW BEZPIECZNYCH: (1)

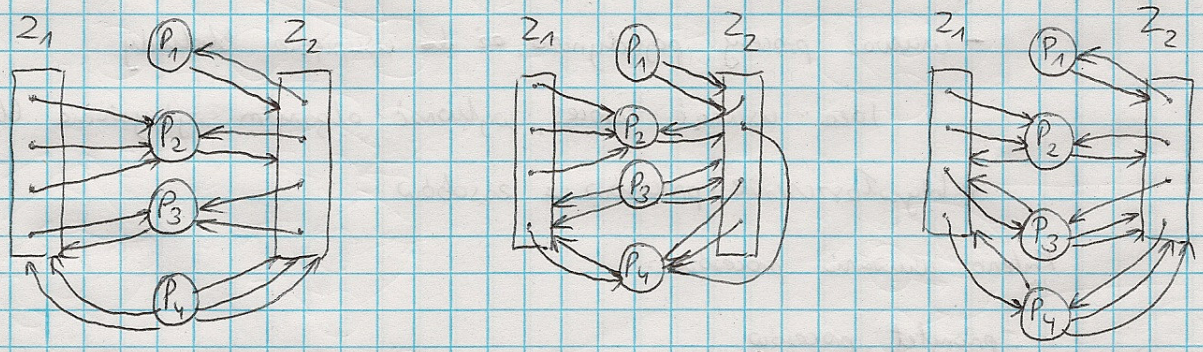


PRZYDZIELAMY ZASOBY PROCESOM:

$2 \times Z_2$ do P_1, $2 \times Z_2$ do P_2 i $3 \times Z_1$ do P_2 oraz $1 \times Z_1$ do P_3.	$1 \times Z_1$ do P_3 i $2 \times Z_1$ do P_4 i $1 \times Z_2$ do P_4	$2 \times Z_2$ do P_4
WYKONUJE SIĘ PROCES: P_1 i P_2	P_3	P_4

CIĄG BEZPIECZNY: P_1 i P_2, P_3, P_4

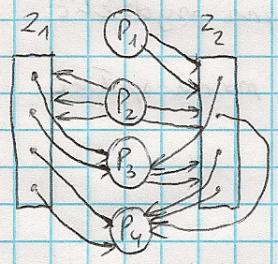
PRZYKŁADY BLOKADY:



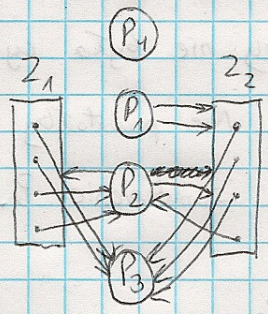
cykl: $P_3 \to Z_1 \to P_2 \to Z_2$ $\quad$ $P_2 \to Z_2 \to P_4 \to Z_2$ $\quad$ $P_1 \to Z_2 \to P_2 \to Z_1 \to P_3 \to Z_2 \to Z_4 \to Z_2$

CIĄG STANÓW BEZPIECZNYCH (2)

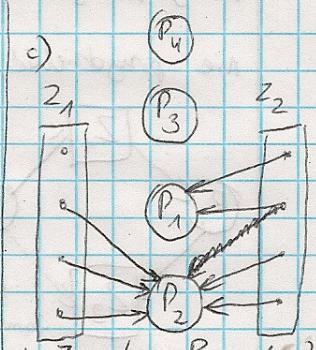
a)



b)



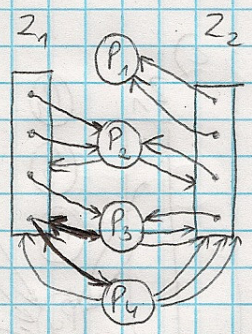
c)

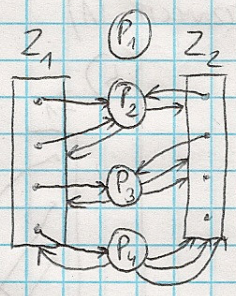


	a)	b)	c)
PRZYDZIELANIE:	$1 \times Z_2$ do P_3, $2 \times Z_1$ do P_3	$2 \times Z_2$ do P_3,	$1 \times Z_2$ do P_2, $1 \times Z_1$ do P_2
	$2 \times Z_1$ do P_4, $3 \times Z_2$ do P_4	$2 \times Z_1$ do P_2 i $1 \times Z_2$ do P_2	$2 \times Z_2$ do P_1
WYKONUJE SIĘ:	P_4	P_3	P_1, P_2

CIĄG BEZPIECZNY: $P_4 \rightarrow P_3 \rightarrow P_1$ i P_2

STAN ZAGROŻENIA:





STAN BLOKADY.

WYKONUJE SIĘ PROCES P_1

19.04.2012
SO2
WYKŁAD 3

* WYKRYWANIE I WYCHODZENIE Z BLOKADY:

- Wykrywanie blokady polega na wykrywaniu pętli w grafie oczekiwań. Niezbędne jest przechowywanie i aktualizowanie na bieżąco grafu oczekiwań oraz wykonywanie algorytmu szukającego pętli w grafie.
- Wychodzenie z blokady:
  * Usunięcie jednego lub kilku procesów w celu przerwania cyklicznego czekania
  * usunąć wszystkie procesy w blokadzie
  - usuwać procesy pojedynczo, aż do usunięcia blokady.

    Idea: usunąć proces, wykonać algorytm wykrywania blokady
  * Wywłaszczenie procesów z zasobów

Kryteria usuwania procesów:

- priorytet procesów
- czas wykonanych do tej pory obliczeń przez ten proces
- ilość zasobów przetrzymywanych przez dany proces

- ilość zasobów potrzebnych do wykonania procesu

19.04.2012
SO2
WYKŁAD 9

- ile procesów trzeba przenieść

Te same kryteria możemy brać pod uwagę przy wyborze procesów do wywłaszczenia z zasobów. Jednocześnie trzeba zagwarantować, że po wywłaszczeniu proces nie będzie głodzony tzn. że długo czekałby na ponowne przydzielenie mu jednego lub więcej zasobów.

* Rozwiązanie problemu blokad w praktyce:
  - zasoby wewnętrzne systemu - zapobieganie blokadom przez uporządkowanie zasobów
  - pamięć główna - wywłaszczenie
  - zasoby zadania - uniknięcie blokad (np. algorytm bankiera)
  - obszar wymiany - zastosowanie wstępnego przydziału, proces od razu musi dostać miejsce w obszarze wymiany.

PODSUMOWANIE Z ZAKRESU ZAKLESZCZEŃ (BLOKAD)

26.04.2012
SO2
WYKŁAD 10

1) Blokada dotyczy sytuacji, kiedy procesy chcą korzystać z pewnych zasobów przydzielonych przez system operacyjny, a procesy wzajemnie przetrzymują swoje zasoby i jednocześnie nie mogą się wykonywać.

2) Warunki konieczne wystąpienia blokady: wzajemne wykluczenie, przetrzymywanie i oczekiwanie, brak wywłaszczeń, czekanie cykliczne.

3) 4) opis warunków koniecznych.

5) Metody zapobiegania blokadom: wykorzystać warunki konieczne i przydzielać tak, aby co najmniej jeden warunek konieczny nie był spełniony (zastosowanie protokołu)

WPROWADZENIE DO SYSTEMÓW ROZPROSZONYCH

System rozproszony - układ niezależnych komputerów, który sprawia wrażenie na jego użytkownikach, że jest jednym komputerem.

Podstawowy problem: wymagane jest oprogramowanie odmienne niż w systemach scentralizowanych.

Przykład: 10 000 procesorów każdy działa z prędkością 50 MIPS ⇒ 500.000 MIPS

pojedynczy procesor o tej samej zdolności obliczeniowej musiałby wykonywać

1 instrukcję w czasie $\frac{2}{2000}$ ns $= 2 \cdot 10^{-12}$ s co jest niemożliwe do osiągnięcia.

Tolerowanie awarii - nawet gdy dojdzie do uszkodzenia jednego z procesorów (komputerów) system ma dalej działać - użytkownik ma nie odczuć awarii

Dostępność systemu - procent czasu, gdy system jest dostępny do prawidłowego wykonywania zadań. 99,999% aktualny poziom (czyli w ciągu roku system ma prawo być niedostępny przez około 7 minut)

* Klasyfikacja systemów rozproszonych: (Flynn) - klasyfikacja z punktu widzenia liczby strumieni instrukcji i liczby strumieni danych.

SISD - 1 strumień instrukcji, 1 strumień danych (standardowe komputery)

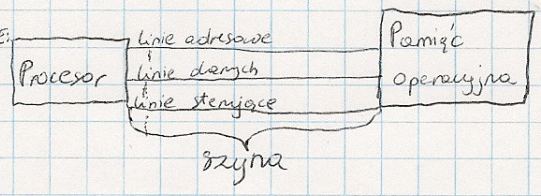
SIMD - 1 strumień instrukcji, wiele strumieni danych (komputery macierzowe)

- MIMD - grupa niezależnych komputerów z własnymi licznikami rozkazów, programami i danymi.
- Systemy MIMD:
  - podział ze względu na budowę:
    - wieloprocesory - układy, w którym jest wiele procesorów, ale ma jedną wspólną pamięć operacyjną (przestrzeń adresową pamięci operacyjnej) do której wszystkie te procesory mają bezpośredni dostęp.
    - multikomputery - układy w którym jest wiele procesorów, ale każdy z nich ma własną pamięć operacyjną. Jednak żaden procesor nie ma bezpośredniego (swobodnego) dostępu do żadnej pamięci operacyjnej poza swoją własną.
  - podział ze względu na architekturę sieci powiązań:
    - szynowe (bus) - (sygnał przechodzi za każdym razem w ten sam sposób.) jeden szkielet połączeń (stały), jeden nośnik łączący wszystkie komputery.
    - przełączane (switched) - (łącze jest dopiero tworzone w wyniku licznych przełączeń) trasa wytyczona jest w wyniku jawnie określonych przełączeń, nie ma stałego szkieletu połączeń.
    - systemy ściśle powiązane - jeśli czas związany z przekazaniem komunikatu jest krótki w porównaniu z czasem wykonania. (jeśli jest długi - system luźno powiązany)

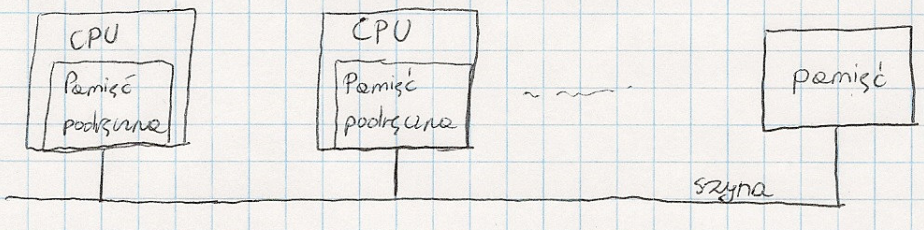
Wieloprocesory szynowe:

26.04.2012
SO2
WYKŁAD 10

PRZYPOMNIENIE:



Pamięć podręczna - przechowuje skończoną ilość ostatnio używanych słów przez procesor. Każdy procesor ma własną pamięć podręczną



Spójność pamięci - jeśli procesor A zapisuje słowo o określonym adresie, a chwilę później procesor B odczytuje to słowo o tym adresie to otrzyma ostatnio zapisaną zawartość tego słowa.

1) Przechowanie ostatniego zawartości słowa w pamięci podręcznej.
2) Przechodzą przez nią wszystkie zamówienia
3) Jeśli potrzebne słowo w niej jest, to procesor z tego korzysta (nie wysyła zapytania przez szynę)

Hit rate - współczynnik trafienia np. 98 % (oznacza, że na 100 odwołań do pamięci, 98 kończy się na pamięci podręcznej, tylko 2 są wysyłane przez szynę)

*Zapewnienie spójności pamięci podręcznej:

pamięć podręczna:

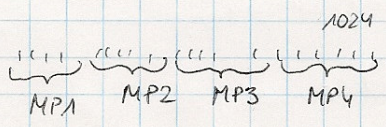
- przepisywalna (write-through) - jeśli wartość słowa jest zapisana w pamięci podręcznej, to przesyła wynik do pamięci operacyjnej
- podglądająca, posłuchująca (snoopy) - wszystkie pamięci podręczne podsłuchują co się dzieje w szynie.

Wieloprocesory przełączane: (pamięć podzielona na moduły)

10.05.2012
SO2
WYKŁAD 11

- Wybierak krzyżowy:

adresowe



przedziały adresów poszczególnych modułów

- Sieć przełączająca omega ← 1000 procesorów ⇒ 1000 000 przełączników – problem!

idea, mająca na celu zmniejszenie liczby przełączników poprzez użycie przełączników wielokrotnych. (przełączniki poczwórne - dwa wejścia i dwa wyjścia) Przełączniki działają na podstawie adresów w linii adresowej

n-procesorów n - modułów przełączających

liczba stopni przełączających $\log_2 n$

w każdym stopniu potrzeba $\frac{n}{2}$ przełączników

Przykład:

czas wykonania instrukcji : $\frac{1s}{100 MIPS} = \frac{1s}{100 \cdot 10^6} = \frac{10^{8\,8}\,ns}{\cancel{100 \cdot 10^6}} = 10\,ns$

$$1s = 10^3\,ms = 10^6\,\mu s = 10^9\,ns = 10^{12}\,ps$$

Czas opóźnienia w przepływie informacji nie powinien przekraczać czasu wykonania instrukcji określonego przez zdolność obliczeniową. W zadaniach będziemy pomijali czas odczytu w pamięci operacyjnej.

czas przełączania przełącznika $\leq$ (czas wykonania instrukcji / 2 · liczbę stopni przełączających)

w zadaniu: $\frac{10\,ns}{2 \cdot 10} = 0{,}5\,ns = 500\,ps$ = czas przełączania przełącznika

ZADANIE EGZAMINACYJNE:

(A) $n = 2048$ procesorów połączonych w sieci Omega.

1) z zasady konstrukcji sieci Omega wynika ($2048 = 2^{11}$) że sygnał od procesora do pamięci przechodzi przez $\log_2 2^{11} = 11$ stopni przełączających (11 przełączników) w jedną stronę i wraca też przez 11 przełączników

2) średni czas wykonywania instrukcji $= \frac{1s}{50 MIPS} = \frac{1s}{50 \cdot 10^6} = \frac{10^{9\,8^2}\,ns}{5\cancel{0} \cdot \cancel{10^6}} = 20\,ns$

3) czas zadziałania przełącznika $\leq \frac{20\,ns}{\cancel{2} \cdot 11} = \frac{10\,ns}{11} \approx 0{,}909\,ns$

(B) $n = 4096$

czas zadziałania przełącznika $= 0{,}5\,ns$

liczba stopni przełączających $= \log_2 4096 = 12$

czas opóźnienia $= 0{,}5\,ns \cdot 2 \cdot 12 = 12\,ns$

średni czas instrukcji > czas opóźnienia

X - szybkość procesora

$\frac{1s}{X}$ = czas instrukcji > 12 ns

$$X < \frac{1s}{12\,ns} = \frac{1s}{12 \cdot 10^{-9}\,s} = \frac{10^9}{12} = \frac{1000 \cdot 10^6}{12} = \frac{1000}{12}\,MIPS \approx 83\,MIPS$$

Zadanie 1: Wieloprocesor ma 8192 procesory połączone w sieci Omega z pojedynczymi przełącznikami. Działanie pojedynczego przełącznika wynosi 0,4 ns. Jak szybkie mogą być procesory, aby zamówienie zdążyło dojść do pamięci i wrócić w czasie jednego cyklu rozkazowego? Odpowiedź należy podać w liczbie MIPS, prezentując również sposób obliczenia. (cykl rozkazowy = czas wykonania jednej instrukcji)

10.05.2012
SO2
WYKŁAD 11

Zadanie 2: Wieloprocesor ma 4096 procesorów o szybkości 200 MIPS połączonych w sieci Omega z pojedynczymi przełącznikami. Jakie powinny być czasy przełącznikowe, aby zamówienie zdążyło dojść do pamięci operacyjnej i wrócić w czasie jednego cyklu rozkazowego? Należy dokładnie podać sposób rozwiązania i uzasadnienie.

* Rozwiązanie NUMA (Non Uniform Memory Access):

Jeśli mamy przestrzeń adresową pamięci, to pamięć jest również podzielona na moduły, ale część modułów ma charakter lokalny dla danych procesorów. A więc czas dostępu do procesora lokalnego jest krótki, a do innych – dłuższy.

ZADANIE 1

$8192 = 2^{13}$ czyli $\log_2 2^{13} = 13$ stopni przełączających (przełączników) w jedną stronę.

Średni czas wykonywania instrukcji $= \frac{1s}{x} >$ czas opóźnienia $= 0{,}4\,ns \cdot 2 \cdot 13 = 10{,}4$

$10{,}4\,x\,ns < 1s$

$$x < \frac{1s}{10ns} = \frac{1s}{12 \cdot 10^{-9}s} = \frac{10^9}{12} = \frac{1000 \cdot 10^6}{12} = \frac{\cancel{1000}^{250}}{\cancel{12}_3} MIPS \approx 86{,}15\ MIPS$$

odp. procesor nie może być szybszy niż 86 MIPS

ZADANIE 2

$4096 = 2^{12}$ czyli $\log_2 2^{12} = 12$ stopni przełączających (przełączników w jedną stronę)

średni czas wykonywania instrukcji $= \frac{1s}{200\,MIPS} = \frac{10^8\,ns}{200 \cdot 10^6} = \frac{1000}{200}\,ns = 5\,ns$

czas rozdzielenia przełącznika $= \frac{\cancel{10}^{5}\,ns}{7 \cdot \cancel{12}_6} = \frac{5}{6}\,ns = 0{,}833\,ns$

WZORY

17.05.2012
SO2
WYKŁAD 12

średni czas instrukcji = sr = $\frac{1s}{\text{szybkość procesora}}$

czas opóźnienie = op = czas zadziałania przełącznika * przejścia * lb. stopni przełączających

sr > op

- Multikomputery szynowe
- Multikomputery przełączane

OPROGRAMOWANIE

- Sieciowe systemy operacyjne
  * Stacje robocze połączone siecią LAN
  * Każda maszyna ma własny system operacyjny (każda może mieć inny)
  * Podstawowe usługi: logowanie na innej maszynie, kopiowanie plików między maszynami

    rlogin id_maszyny nazwa_użytkownika hasło; // zdalne logowanie

    rcp id_maszyny1:plik1 id_maszyny2:plik2; // zdalne kopiowanie

    Te działania są możliwe i jawne, ale wymagają od użytkownika wiedzy na temat identyfikatora maszyny, struktury plików na maszynie, na której użytkownik chce zdalnie pracować.
  * Serwer plików realizuje globalny system plików dzielonych

    Możliwy różny obraz zasobów widzianych przez klientów w systemie sieciowym przy pracy na dwóch różnych maszynach.
- Prawdziwe systemy rozproszone
  * Wiele komputerów połączonych siecią
  * Wrażenie jednolitego systemu (systemy operacyjne mają działać tak, żeby użytkownik podczas pracy miał wrażenie, że pracuje na systemie scentralizowanym - z jednym procesorem) (ten sam obraz systemu plików)
  * Wymaganie:
    - Jednolity, globalny system komunikacji między procesami itd.

- Systemy wieloprocesorowe z podziałem czasu

17.05.2012
SO2
WYKŁAD 12

* Wiele jednostek centralnych z pamięcią podręczną
* Wspólna pamięć dzielona
* Wspólny dysk / dyski
* Połączenie szyną
* Jedna kolejka uruchomień procesów (algorytm rotacyjny)

ZADANIE: Wieloprocesor : 2 procesory

Algorytm rotacyjny z kwantem czasu 10s

Priorytety procesów: jednakowe.

Zgłoszono : 4 procesy (A, B, C, D)

W chwilach: A - 0s, B - 10s, C - 20s, D - 30s

Czas wykonywania przez procesor: A - 30s, B - 20s, C - 20s, D - 20s

Procesor 1 jest wolny od 15s, Procesor 2 - od 30 s.

Wyznaczyć czasy: czas cyklu przetwarzania procesów (średni)

Pokazać na diagramie czasowym przetwarzanie procesów przez te procesory

- Elastyczność:

  1) Idea Jądra Monolitycznego

  2) Idea Mikrojądra

31.05.2012
SO2
WYKŁAD) 14

SIECI KOMPUTEROWE:

* Adresowanie: numer IP - cztery jednostki ośmio-bajtowe.

Struktura numeru IP: część sieciowa i część hostowa

Klasy sieci:

klasa A - część sieciowa - pierwszych 8 cyfr (bitów)

klasa B - —"— —"— 16 cyfr (bitów)

klasa C —"— —"— 24 cyfr (bitów)

klasa D —"— —"— 32 cyfr (bitów) czyli całość.

Maska sieci - liczba 32-bitowa. Klasy:

A - maska standardowa 255. 0. 0. 0 IP/8

B - —"— 255. 255. 0. 0 IP/16

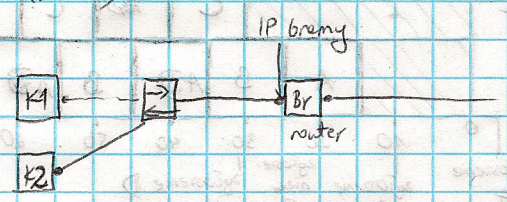
C - —"— 255. 255. 255. 0 IP/24

Maski niestandardowe - liczba jedynek nie jest wielokrotnością liczby 8

przykład: 255. 255. 255. 240

* Odwzorowanie adresów protokołowych (ARP)

1) IP komputera
2) Maska
3) IP bramy - adres routera
4) IP broadcastu



wszystkie komputery w danej sieci 1) IP Sieci

służy do wysłania do wszystkich 2) IP Broadcastu

Adresy sieci są używane w tablicach routingu

		host
IP Sieci	net	000
IP Broadcastu	net	111
IP Komputera	net	0 i 1 (nie może mieć samych zer lub samych jedynek)
IP Bramy	jeden z tych adresów	

AND
IP sieć net 0000
M 1111 0000
bez zmian

IP komp net 0 i 1
M 1111 0000
IP sieć net 0000

IP komp net 0 i 1
M 1111 0000
IP broadcast net 1111

normalne AND

AND i NEGACJA

31.05.2012
SOZ
WYKŁAD 14

ZADANIE

IP 192.168.10.45

M 255.255.255.240

	11000000				00101101			
AND	11111111				11110000			
	11000000				00100000			

WYNIK IP sieci: 192 . 168 . 10 . 32

00101111 ← zamiana 4 ostatnich zer na jedynki

IP Broadcastu 192.168.10.47

IP Bramy 192.168.10.46

IP Komputerów 192.168.10.33-45

ZADANIE 2

IP 130.10.55.110

M 255.255.0.0

IP Sieci 130.10.0.0.

IP Broadcastu 130.10.255.255

IP Bramy 130.10.255.254

IP Komputerów: 130.10.0.1 – 130.10.255.253

/n	Maska decymalna	Maska binarnie	Wielkość sieci	Il. podsieci
/24	255.255.255.0	11111111 (24) ... 00000000	256(-3)	1
/25	255.255.255.128	11111111 (24) ... 10000000	128(-3)	2
/26	255.255.255.192	11111111 (24) ... 11000000	64(-3)	4
/27	255.255.255.224	11111111 (24) ... 11100000	32(-3)	8
/28	255.255.255.240	11111111 (24) ... 11110000	16(-3)	16
/29	255.255.255.248	11111111 (24) ... 11111000	8(-3)	32
/30	255.255.255.252	11111111 (24) ... 11111100	4(-3)	64
/23	255.255.254.0	11111111 (23) ... 00000000	512(-3)	0,5
/22	255.255.252.0	11111111 (22) ... 00000000	1024(-3)	0,25

ZADANIE 3

IP 192.168.10.128

M 255.255.255.128

10.128 — 10.255

$2^{N_1} + 2^{N_2} = 128 = \overset{2^6}{64} + \overset{2^6}{64}$

LP	IP Sieci	MASKA	IP Broadcastu	IP bramy	IP komputerów
1	192.168.10.128	255.255.255.192	192.168.10.191	192.168.10.190	192.168.10.129-189
2	192.168.10.192	255.255.255.192	192.168.10.255	192.168.10.254	192.168.10.193-253

- Systemy wieloprocesorowe z podziałem czasu

17.05.2012
SO2
WYKŁAD 12

* Wiele jednostek centralnych z pamięcią podręczną
* Wspólna pamięć dzielona
* Wspólny dysk / dyski
* Połączenie szyną
* Jedna kolejka uruchomień procesów (algorytm rotacyjny)

ZADANIE: Wieloprocesor : 2 procesory

Algorytm rotacyjny z kwantem czasu 10s

Priorytety procesów: jednakowe.

Zgłoszono : 4 procesy (A, B, C, D)

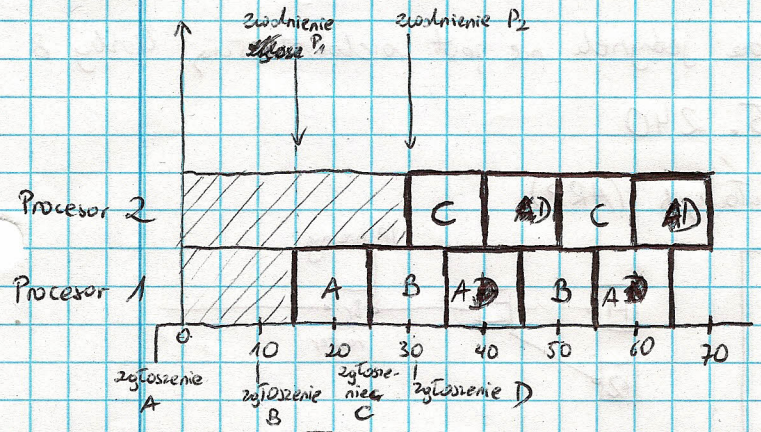
W chwilach: A - 0s, B - 10s, C - 20s, D - 30s

Czas wykonywania przez procesor: A - 30s, B - 20s, C - 20s, D - 20s

Procesor 1 jest wolny od 15s, Procesor 2 - od 30 s.

Wyznaczyć czasy: czas cyklu przetwarzania procesów (średni)

Pokazać na diagramie czasowym przetwarzanie procesów przez te procesory



średni czas przetwarzania procesów:

$$\frac{(65s-0s)+(55s-10s)+(60s-20s)+(70s-30s)}{4}=47{,}5s$$

(przeprowadzić analizę stanów kolejki, co 10s lub co 5s.)

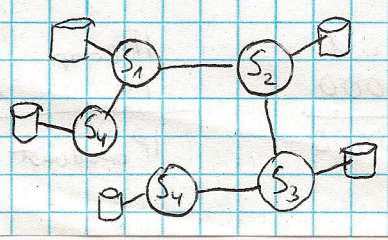
- Porównanie klas systemów operacyjnych z wieloma jednostkami centralnymi

24.05.2012
SO2
WYKŁAD 13

- Podstawowe wymagania (określenia) wobec systemów rozproszonych.

* Przezroczystość - dostępu i położenia, współbieżności, zwielokrotnienia, współbieżności i działań równoległych.

Przezroczystość zwielokrotnienia:



- Elastyczność:

1) Idea Jądra Monolitycznego

2) Idea Mikrojądra

31.05.2012
SO2
WYKŁAD) 14

SIECI KOMPUTEROWE.

* Adresowanie: numer IP - cztery jednostki ośmio-bajtowe.

Struktura numeru IP: część sieciowa i część hostowa

Klasy sieci:

klasa A - część sieciowa - pierwszych 8 cyfr (bitów)

klasa B - —"— —"— 16 cyfr (bitów)

klasa C —"— —"— 24 cyfr (bitów)

klasa D —"— —"— 32 cyfr (bitów) czyli całość.

Maska sieci - liczba 32-bitowa. Klasy:

A - maska standardowa 255. 0. 0. 0 IP/8

B - —"— 255. 255. 0. 0 IP/16

C - —"— 255. 255. 255. 0 IP/24

Maski niestandardowe - liczba jedynek nie jest wielokrotnością liczby 8

przykład: 255. 255. 255. 240

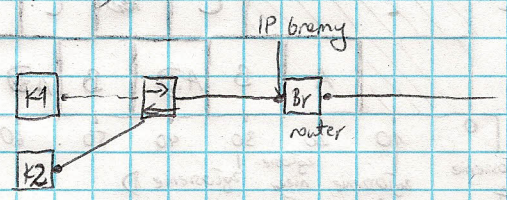
* Odwzorowanie adresów protokołowych (ARP)

1) IP komputera

2) Maska

3) IP bramy - adres routera

4) IP broadcastu



wszystkie komputery w danej sieci 1) IP Sieci

służy do wysłania do wszystkich 2) IP Broadcastu

Adresy sieci są używane w tablicach routingu

		host
IP Sieci	net	000
IP Broadcastu	net	111
IP komputera	net	0 i 1 (nie może mieć samych zer lub samych jedynek)
IP Bramy	jeden z tych adresów	

AND) IP sieci net 0000
M 1111 0000
bez zmian

IP komp net 0 i 1
M 1111 0000
IP sieci net 0000

IP komp net 0 i 1
M 1111 0000
AND i NEGACJA
IP broadcast net 1111

31.05.2012
SO2
WYKŁAD 14

ZADANIE

IP 192.168.10.45

M 255.255.255.240

	11000000	00101101
AND	11111111	11110000
	11000000	00100000

WYNIK IP sieci: 192 . 168 . 10 . 32

00101111 — zamiana 4 ostatnich zer na jedynki

IP Broadcastu 192.168.10.47

IP Bramy 192.168.10.46

IP Komputerów 192.168.10.33-45

ZADANIE 2

IP 130.10.55.110

M 255.255.0.0

IP Sieci 130.10.0.0.

IP Broadcastu 130.10.255.255

IP Bramy 130.10.255.254

IP Komputerów: 130.10.0.1 – 130.10.255.253

/n	Maska decymalna	Maska binarnie	Wielkość sieci	Il. podsieci
/24	255.255.255.0	11111111 (24) ... 00000000	256(-3)	1
/25	255.255.255.128	11111111 (24) ... 10000000	128(-3)	2
/26	255.255.255.192	11111111 (24) ... 11000000	64(-3)	4
/27	255.255.255.224	11111111 (24) ... 11100000	32(-3)	8
/28	255.255.255.240	11111111 (24) ... 11110000	16(-3)	16
/29	255.255.255.248	11111111 (24) ... 11111000	8(-3)	32
/30	255.255.255.252	11111111 (24) ... 11111100	4(-3)	64
/23	255.255.254.0	11111111 (23) ... 0 00000000	512(-3)	0,5
/22	255.255.252.0	11111111 (22) ... 00 00000000	1024(-3)	0,25

ZADANIE 3

IP 192.168.10.128

M 255.255.255.128

10.128 — 10.255

$2^{N_1} + 2^{N_2} = 128 = \overset{2^6}{64} + \overset{2^6}{64}$

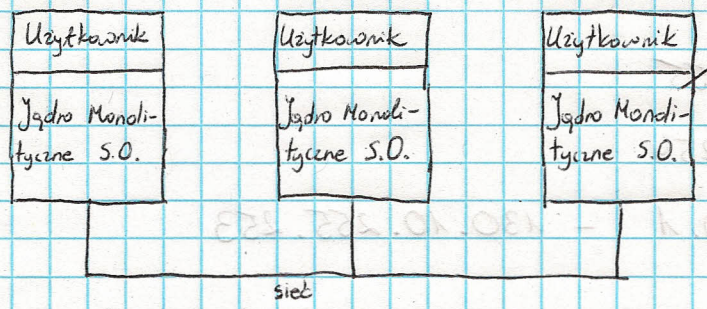
LP	IP Sieci	MASKA	IP Broadcastu	IP bramy	IP komputerów
1	192.168.10.128	255.255.255.192	192.168.10.191	192.168.10.190	192.168.10.129-189
2	192.168.10.192	255.255.255.192	192.168.10.255	192.168.10.254	192.168.10.193-253

14.05.2012
SO2
WYKŁAD 15

Podstawowe wymagania (oczekiwania) wobec systemów rozproszonych:

- Przezroczystość (dostępu, wzdrożenia, wielokrotnienia, współbieżności, działań równoległych)
- Elastyczność (Idea jądra monolitycznego, idea mikrojądra)
- Niezawodność, dostępność – % czasu gdy system prawidłowo wykonuje usługi (jest dostępny)
- Integralność danych, bezpieczeństwo
- Tolerowanie awarii
- Wydajność (czas odpowiedzi, ziarnistość obliczeń)
- Skalowalność
- Idea algorytmów zdecentralizowanych

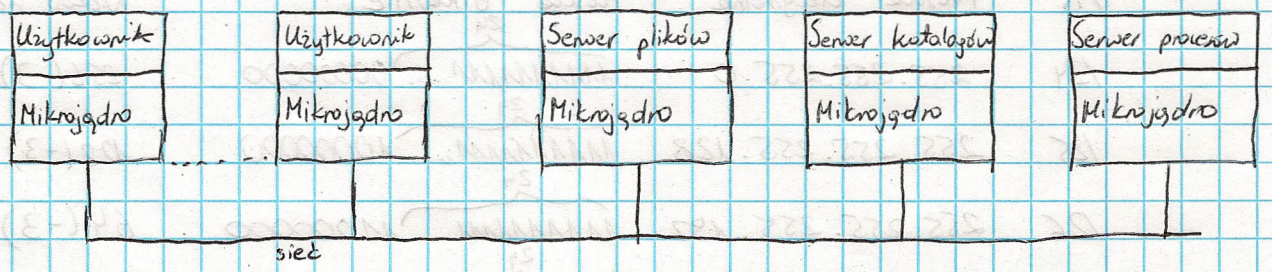
Idea Jądra Monolitycznego



Zawiera usługi dotyczące:
- plików
- katalogów
- procesów

zgodnie z ideami przezroczystości

Idea Mikrojądra



1) Mechanizm komunikacji między procesami
2) Zarządzanie pamięcią
3) Niskopoziomowe zarządzanie procesami (w tym planowanie)
4) Niskopoziomowe usługi wejścia-wyjścia

Cechy algorytmów zdecentralizowanych:

1) Żadna maszyna nie ma kompletu informacji o systemie
2) Maszyny podejmują decyzje na podstawie lokalnych danych
3) Uszkodzenie 1 maszyny nie zdezorganizuje pracy całego algorytmu
4) Nie ma niejawnych założeń o istnieniu globalnego zegara.

* Rozproszone środowisko operacyjne (DCE – Distributed Computing Environment)

efekt: Pakiety DCE