Koordynacja procesów
w
środowisku rozproszonym

Systemy rozproszone

System rozproszony jest zbiorem luźno powiązanych ze sobą komputerów połączonych siecią komunikacyjną (*Silberschatz*).

Zasoby zdalne - zasoby lokalne

Dzielenie zasobów

Przyśpieszanie obliczeń

Niezawodność

Systemy rozproszone

- **Przemieszczanie danych**
- **Przemieszczanie obliczeń**
- **Przemieszczanie procesów**
  - ***Równoważenie załadowania***
  - ***Przyśpieszanie obliczeń***
  - ***Preferencje sprzętowe***
  - ***Preferencje oprogramowania***

Synchronizacja czasu

- **Dlaczego synchronizacja czasu jest ważna?**
- **Zależności czasowe w systemach operacyjnych**
- **Serwisy czasowe**
  - **serwis *time* (port 525)**
  - **serwis *daytime* (port 13)**
  - **serwis *ntp* (port 123)**

Źródła czasu

- Międzynarodowy czas atomowy *(International Atomic Time)*
- Uniwersalny czas skoordynowany (*Coordinated Universal Time - UTC*)
- Standard DCF77
- Sieciowe źródła czasu
- GPS

Synchronizacja zegarów

- Zegary opóźnione
- Zegary przyspieszone
- Metoda *Cristiana*
- Algorytm z *Berkeley*

NTP (*Network Time Protocol*)

- *Dostarczanie usług umożliwiających klientom sieci dokładną synchronizację z czasem UTC.*
- *Dostarczanie niezawodnych usług, zdolnych do przetrwania długich okresów braku połączenia.*
- *Umożliwianie klientom dostatecznie częstych resynchronizacji.*
- *Dostarczanie ochrony przed przypadkowym lub umyślnym zaburzaniem usługi.*

- Tryb rozsyłania (*multicast mode*)
- Tryb wywoływania procedur (*procedure-call mode*)
- Tryb symetryczny (*symmetric mode*)

Relacja uprzedniości zdarzeń Lamporta

1. Jeżeli A i B są zdarzeniami w tym samym procesie i A zaszło przed B, to zapisujemy:

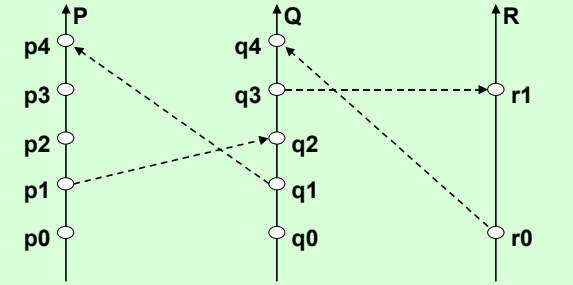
   $A \rightarrow B$

2. Jeżeli A polega na wysłaniu komunikatu przez proces P, a B na odebraniu tego komunikatu przez proces Q, to:

   $A \rightarrow B$

3. Jeżeli $A \rightarrow B$ i $B \rightarrow C$ to $A \rightarrow C$.

Zegary logiczne Lamporta



$p1 \rightarrow q2$, $r0 \rightarrow q4$, $q3 \rightarrow r1$, $p1 \rightarrow q4$, $q1 \rightarrow p4$

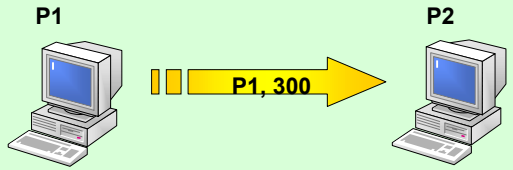
Niektóre pary zdarzeń równoczesnych:

(q0, p2) (r0, q3) (r0, p3), (q3, p3)

Zegary logiczne Lamporta

1. Zegar procesu jest zwiększany o 1 przed wystąpieniem każdego zdarzenia.
2. Jeśli proces wysyła komunikat, to umieszcza w nim wartość bieżącą swojego zegara logicznego.
3. Przy odbiorze komunikatu proces oblicza $z_o = \max(z_k, z_z)$ i zanim opatrzy tym znacznikiem zdarzenie odebrania komunikatu, stosuje regułę 1.

Zegary logiczne Lamporta



Przed odebraniem
LC(P2) = 245
Naruszenie monotoniczności !
LC(P2) := 301..

Globalne znaczniki czasu

(T_a, p_a) :

$(T_a, p_a) < (T_b, p_b) \Leftrightarrow T_a < T_b$ albo $T_a = T_b$ i $p_a < p_b$

Koordynacja - wzajemne wyłączanie

- Podejście scentralizowane
- Podejście rozproszone
- Metoda przekazywania żetonu

Wzajemne wyłączanie - podejście scentralizowane

- Proces wysyła do koordynatora *zamówienie*.
- Gdy proces otrzyma od koordynatora *odpowiedź* może wejść do swojej sekcji krytycznej.
- Po wyjściu z sekcji krytycznej, proces wysyła do koordynatora komunikat *zwalniający*.
- *Zamówienia* umieszczane są w kolejce, lub obsługiwane natychmiast, jeżeli kolejka jest pusta.
- Po otrzymaniu komunikatu ze *zwolnieniem*, koordynator pobiera komunikat z kolejki zamówień (jeżeli nie jest ona pusta) i wysyła *odpowiedź* do właściwego procesu.

Czy koordynator może nie wysyłać odpowiedzi ?

Wzajemne wyłączanie - podejście rozproszone

- Proces wytwarza znacznik czasowy T_s i wysyła *zamówienie* (P,T_s) do wszystkich innych procesów.
- Każdy z odbiorców *zamówienia* może wysłać *odpowiedź* natychmiast lub z opóźnieniem.
- Proces, który otrzyma *odpowiedzi* od wszystkich innych procesów, może wejść do sekcji krytycznej.
- W czasie pobytu w sekcji krytycznej musi on rejestrować wszystkie nadchodzące *zamówienia* (zwlekać z *odpowiedzią*).
- Po opuszczeniu sekcji krytycznej wysyła *odpowiedzi* do wszystkich procesów, od których ma zarejestrowane *zamówienia*.

Czy odbiorcy mogą nie wysyłać odpowiedzi ?

Wzajemne wyłączanie - podejście rozproszone

<u>Czynniki determinujące decyzję o zwłoce:</u>

- Jeżeli proces przebywa w sekcji krytycznej, to opóźnia *odpowiedź*.
- Jeżeli proces nie zamierza wejść do sekcji krytycznej, to *odpowiedź* wysyła natychmiast.
- Jeżeli proces też zamierza wejść do sekcji krytycznej (czeka), to porównuje znacznik czasowy swojego *zamówienia* ze znacznikiem *zamówienia* odebranego. Jeżeli jego znacznik czasowy jest większy, to wysyła *odpowiedź* natychmiast.

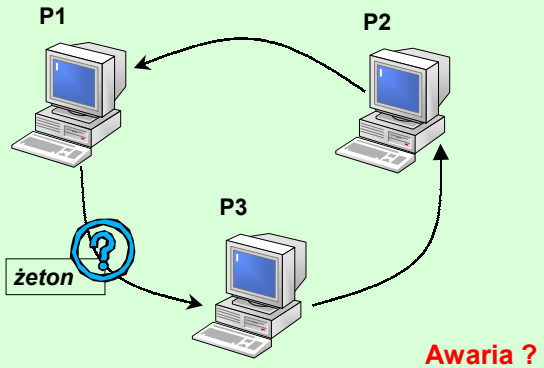
UWAGA: należy wykorzystywać znaczniki globalne

Wzajemne wyłączanie - podejście rozproszone

System składa się z procesów P1, P2, P3
P1 i P3 chcą wejść do sekcji krytycznych

- Procesy wysyłają zamówienia:
  Zamówienie(P1,10); *Zamówienie(P3,4)*
- Proces P2 odpowie natychmiast (nie wchodzi do sekcji krytycznej)
- Proces P1 odpowie natychmiast (jego znacznik (10) jest większy od znacznika w zamówieniu z procesu P3 (4))
- Proces P3 opóźni odpowiedź (w jego komunikacie znacznik jest mniejszy niż w odebranym)
- Proces P3 po otrzymaniu odpowiedzi od P1 i P2 wejdzie do sekcji krytycznej. Po wyjściu z niej wyśle komunikat odpowiedzi do P1.
- Proces P1 po otrzymaniu odpowiedzi od P3 wejdzie do sekcji krytycznej.

Wzajemne wyłączanie – algorytm pierścieniowy



Zapobieganie zakleszczeniom

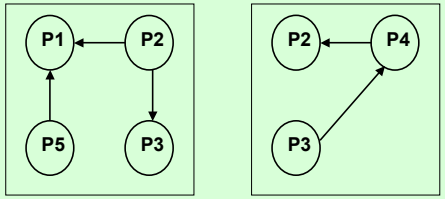
Każdy proces otrzymuje jednoznaczny priorytet. Proces P może czekać na Q jeżeli P ma wyższy priorytet. W przeciwnym przypadku P zostanie usunięty.

- *Czekanie albo śmierć (technika bez wywłaszczeń).* Jeżeli proces P zamawia zasób wykorzystywany przez proces Q, to procesowi P pozwala się czekać tylko wtedy, gdy jego znacznik czasowy jest mniejszy od znacznika czasowego procesu Q. W przeciwnym przypadku proces P jest usuwany (umiera).
- *Zranienie albo czekanie (technika wywłaszczeniowa).* Jeżeli proces P zamówi zasób przetrzymywany aktualnie przez proces Q, to proces P może czekać tylko wtedy, gdy ma większy znacznik czasowy niż Q. W przeciwnym razie proces Q jest usuwany (Q zostaje zraniony przez P).

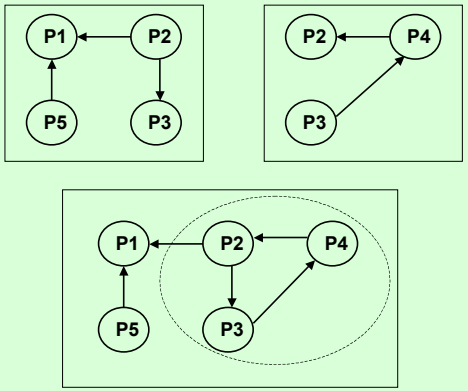
Wykrywanie zakleszczeń

Graf oczekiwań:

Jeżeli proces P potrzebuje zasobu użytkowanego na stanowisku przez proces Q, to wysyła do stanowiska komunikat z zamówieniem. Powoduje to dodanie do lokalnego grafu na stanowisku krawędzi P → Q.

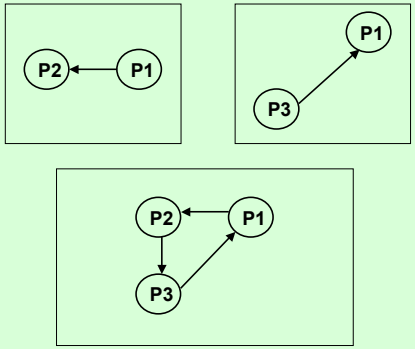


Wykrywanie zakleszczeń



Wykrywanie zakleszczeń

Podejście scentralizowane – fałszywe cykle



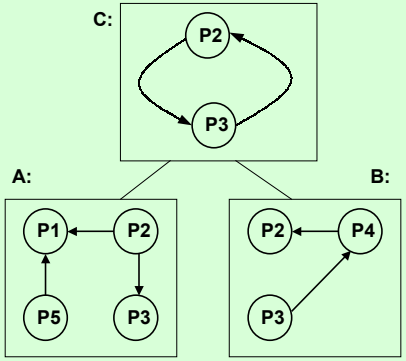
Wykrywanie zakleszczeń – podejście hierarchiczne

Jeżeli kontrolery A i B mają wspólnego (najniższego) przodka C i węzeł P występuje w grafach lokalnych A i B, to węzeł P musi również wystąpić w grafie:

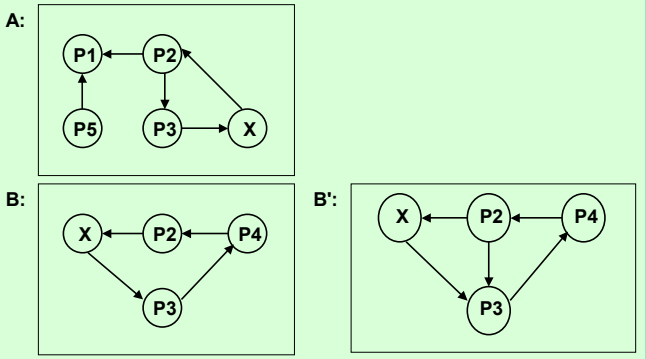
- **kontrolera C,**
- **każdego kontrolera na drodze od C do A,**
- **każdego kontrolera na drodze od C do B.**

Jeżeli węzły P i Q występują w grafie kontrolera D oraz w grafie jednego z potomków D istnieje droga z P do Q, to w grafie oczekiwań D musi wystąpić krawędź P Q.

Wykrywanie zakleszczeń – podejście hierarchiczne



Wykrywanie zakleszczeń – podejście rozproszone



Wykrywanie zakleszczeń – podejście rozproszone

Ograniczenie ruchu komunikatów

Gdy stanowisko wykryje cykl z węzłem X w postaci:

$$X \rightarrow P1 \rightarrow P2 \ldots Pk \rightarrow X$$

to wyśle komunikat tylko wtedy, gdy ID(P1) > ID(Pk)

Stanowisko A:

cykl: $X \rightarrow P2 \rightarrow P3 \rightarrow X$ ID(P2) < ID(P3)

Stanowisko B:

cykl: $X \rightarrow P3 \rightarrow P4 \rightarrow P2 \rightarrow X$ ID(P3) > ID(P2)

Odporność systemu na uszkodzenia

Najczęstsze błędy w systemach rozproszonych:

- awaria łącza,
- awaria stanowiska,
- utrata komunikatu.

Wykrywanie awarii - procedura uzgadniania (*handshaking procedure*):

- Wysyłanie komunikatów *jestem sprawny*
- Wysyłanie komunikatów *czy jesteś sprawny?*
- Limit czasu na odpowiedź.

Odporność systemu na uszkodzenia

Rekonfiguracja - uruchomienie procedury, która umożliwi rekonfigurację systemu i kontynuowanie obliczeń.

- Przekazanie informacji do innych stanowisk.
- Wytypowanie nowego koordynatora (jeżeli taki istniał i uległ awarii) - algorytmy elekcji.
- Skonstruowanie nowego pierścienia logicznego (jeżeli awarii uległo stanowisko będącego elementem łańcucha).
- Inne ...

Odporność systemu na uszkodzenia

Wychodzenie z awarii

- Włączenie po naprawie uszkodzonego łącza lub stanowiska do systemu, bez zakłócania jego pracy.
- Powiadomienie innych stanowisk.

Elekcja – algorytm tyrana

- Proces Pi wykrył awarię koordynatora i próbuje siebie obrać nowym koordynatorem.
- Proces Pi wysyła komunikat o elekcji do wszystkich procesów z wyższymi priorytetami. Czeka określony czas na nadejście przynajmniej jednej odpowiedzi.
- Jeżeli brak jest odpowiedzi, to proces siebie wybiera koordynatorem.
  - Wznawia kopię koordynatora.
  - Wysyła komunikat informujący o swoim wyborze do wszystkich procesów z mniejszymi od niego priorytetami.
  - KONIEC.

Elekcja – algorytm tyrana cd.

- Jeżeli odpowiedź nadejdzie, to proces Pi rozpoczyna czekanie przez określony czas na komunikat o wyborze nowego koordynatora.
- Jeżeli komunikat o wyborze nie nadejdzie, to powinien wznowić algorytm.
- Jeżeli komunikat o wyborze nadejdzie, to powinien odnotować tą informację i KONIEC.
- Jeżeli jakikolwiek proces Pi otrzyma od procesu Pj (J<i) komunikat o rozpoczęciu elekcji, to wysyła odpowiedź do Pi i zaczyna wykonywać swój algorytm elekcji.
- Każdy proces, który wrócił do działania po awarii, również rozpoczyna realizację algorytmu elekcji.

Elekcja – algorytm pierścieniowy

- Jeżeli proces Pi wykryje awarię koordynatora, to tworzy nową listę aktywną (pustą). Wysyła komunikat *elekcja(i)* do sąsiada a swój identyfikator dodaje do swojej listy aktywnej.
- Jeżeli proces Pi otrzyma komunikat *elekcja(j)* to:
  - Jeśli jest to pierwszy komunikat o elekcji, to tworzy nową listę aktywną z numerami *i* oraz *j*. Wysyła komunikat *elekcja(i)* a potem *elekcja(j)*.
  - Jeśli i ≠ j, to proces Pi dodaje *j* do swojej listy aktywnej i przekazuje komunikat dalej.
  - Jeśli i = j, to koniec. Proces może teraz wybrać z listy najwyższy numer i wskazać go jako koordynatora.