|  |
| --- |
| Wojciech Ganobis Lista 6  Redaktor: |

1. Pierwszym opóźnieniem jest czas propagacji sygnału w atmosferze. Następnie może nastąpić opóźnienie kolizyjne podczas gdy maszyny z odbiornikami WWV walczą aby dostać się do Ethernetu. Po trzecie, istnieje czas propagacji w sieci LAN. Po czwarte, istnieje opóźnienie na każdym z procesorów po otrzymaniu pakietu, powodem czego jest procedura obsługi przerwań oraz wewnętrzne opóźnienia kolejek.
2. Drugi zegar tyka 999,000 razy na sekundę, powodując błąd wielkości 10ms na sekundę. W czasie minuty odchylenie to wzrasta do 600ms. Drugim sposobem obliczeń może być przyjęcie, że drugi zegar jest o 1% wolniejszy, czyli po minucie odchyli się on o 0.01 x 60s, lub 600ms.
3. Jednymi z typowych przykładów mogą być sporty oraz opieka zdrowotna. W dzisiejszych czasach istnieją BAN-y bazujące na technologii GPS, pozwalające osobie trenującej na zewnątrz monitorować tempo treningu. Sieci te często są poszerzone o urządzenia monitorujące puls, które następnie mogą być podłączone do komputera aby pobrać dane zebrane z sensorów w celach dalszej analizy. Inny przykład tworzony jest przez sprzęt służący do nawigacji samochodowej, który bazowany jest na odbiornikach GPS.
4. Oczywistym powodem jest możliwość wystąpienia błędu w aktualnym wyniku pomiaru. Zakładając, że zegary mogą być stopniowo regulowane, jedną z możliwości jest wzięcie pod uwagę ostatnich *N* wartości i obliczenie ich mediany lub średniej. Jeśli zaobserwowana wartość znajdzie się poza obecnym interwałem, nie jest brana pod uwagę (ale jest dodana do listy). Jednakowoż, nowa wartość może być obliczona poprzez użycie średniej ważonej lub algorytmu starzenia się.
5. Rozwiązanie nie może zawierać 0 gdyż zostałoby uporządkowane. Z tego powodu musi być to wiadomość od 1 do 2 lub od 2 do 1. Jeżeli opuści lub dotrze z 1 przed 16, zostanie uporządkowana względem *A*, dlatego musi ona opuścić lub dotrzeć przed 16. Istnieją dwie możliwości: wiadomość opuszczająca proces 2 w chwili 0 i docierająca do procesu 1 w chwili 8, lub wiadomość opuszczająca proces 1 w chwili 0 i docierająca do procesu 2 w chwili 10. Obydwa rozwiązania są współbieżne z *A*.
6. Nie, multicast (multiemisja) jest wystarczający, jeśli wiadomość posiada znacznik czasu większy od znacznika czasu odebranej wiadomości. Wymogiem dostarczenia wiadomości *m* do aplikacji jest odebranie innej wiadomości posiadającej duży znacznik czasu z każdego procesu. Gwarantuje to brak wiadomości w toku z małym znacznikiem czasu.
7. Wyobraźmy sobie transfer dużego obrazu, który został podzielony na bloki. Każdy z bloków jest identyfikowany przez jego pozycję w oryginalnym obrazie oraz jego szerokość i wysokość. W tym przypadku porządkowanie FIFO nie jest potrzebne ze względu na to, że odbiornik może wkleić przychodzące bloki na poprawne pozycje.
8. W algorytmie scentralizowanym bardzo często występuje jeden stały proces, który pełni rolę koordynatora. Dystrybucja przeprowadzana jest przez inne procesy działające na pozostałych maszynach. W algorytmach rozproszonych bez stałego koordynatora jest on wybierany wśród procesów tworzących algorytm. Fakt istnienia koordynatora nie powoduje, że algorytm jest mniej rozproszony.
9. Żądania mogą zostać powiązane z poziomami priorytetu w zależności od ich ważności. Koordynator mógłby następnie zezwalać na żądania z największym priorytetem.
10. Załóżmy, że algorytm jest skonstruowany w taki sposób, że odpowiada natychmiastowo na każde żądanie, pozytywnie lub negatywnie. Jeśli nie istnieją żadne procesy korzystające z zasobów oraz żadne procesy nie czekają w kolejce, wtedy awaria nie jest krytyczna. Kolejny proces, który zażąda pozwolenia nie otrzyma odpowiedzi i będzie w stanie rozpocząć wybór nowego koordynatora. System może być solidniejszy poprzez posiadanie koordynatora, który będzie przechowywał każdy przychodzące żądanie na dysku *przed* wysłaniem odpowiedzi. W ten sposób nawet w przypadku załamania systemu nowy koordynator będzie w stanie odbudować listę udostępnionych zasobów i kolejki przez wczytanie pliku z dysku.
11. Przyjmijmy, że proces załamuje się po odmowie zgody. Żądający proces myśli, że drugi proces wciąż działa, jednak odpowiedź z zezwoleniem nigdy nie nadejdzie. Jednym z wyjść jest uśpienie procesu na ustalony przedział czasu zamiast blokowania, po czym bada wszystkie procesy, które odmówiły zgody sprawdzając czy wciąż działają.
12. Zmienią się jedynie wpisy dotyczące rozproszonych przypadków. Ze względu na to, że wysłanie wiadomości punkt-punkt jest tak samo wymagające jak transmisja, musimy wysłać jedynie jedną wiadomość transmitowaną do wszystkich procesów żądających dostępu do zasobu. Tak samo, potrzebna jest tylko jedna uwalniająca wiadomośc transmitowana. Opóźnienie wynosi 1 + (*n* - 1): jedno pochodzące z wiadomości transmitowanej oraz dodatkowe *n* - 1, ponieważ wciąż musimy otrzymać wiadomość z każdego innego procesu zanim dostaniemy zezwolenie dostępu do zasobu.
13. Jeśli procesy mają dostęp do zasobów ściśle sekwencyjny, to znaczy gdy proces przechowujący zasób nie próbuje uzyskać dostępu do innego zasobu, wtedy nie ma możliwości, żeby mógł się zablokować podczas korzystania z zasobu, który ma być wykorzystany przez inny proces. System jest wtedy wolny od zakleszczeń. Z drugiej strony, jeśli proces 0 może wstrzymać zasób A i próbować uzyskać dostęp do zasobu B, zakleszczenie może wystąpić jeśli jakiś inny proces spróbuje uzyskać dostęp w odwrotnej kolejności. Sam algorytm Ricart i Agrawala nie przyczynia się do zakleszczeń, ponieważ każdy zasób obsługiwany jest niezależnie od wszystkich pozostałych.
14. Każdy z wyżej-numerowanych procesów otrzyma dwie wiadomości WYBORU, ale zignoruje drugą z nich. Proces elekcji zostanie przeprowadzony w standardowy sposób.
15. Gdy proces odbierze komunikat WYBORU sprawdza kto je zaczął. Jeśli sam go uruchomił (tj. Jego numer znajduje się na początku listy) zmienia on wiadomość w wiadomość KOORDYNATORA, tak jak opisano w tekście. Jeśli nie uruchomi żadnego komunikatu WYBORU, dodaje swój numer do procesu i przesyła go wzdłuż pierścienia. Jednak jeżeli wcześniej wysłał on własny komunikat WYBORU i odkrył konkurenta, porównuje on numer procesu inicjatora z własnym. Jeśli drugi proces ma niższą liczbę, odrzuca wiadomość zamiast ją przekazywać. Jeśli numer jest wyższy, wiadomość jest przekazywana w zwykły sposób. Dzięki temu, jeśli uruchomionych zostaje wiele wiadomości WYBORU, ten którego pierwszy wpis jest najwyższy zostaje zachowany. Reszta zostaje odrzucona na trasie.
16. Ponieważ użytkownicy aktualnie potrzebują natychmiastowego dostępu do plików online, co uniemożliwia przechowywanie zapisów jako taśmę magnetyczną.
17. Dopuszczalne wartości to 1, 2 i 3,a niedopuszczalne to 4, 5 i 6. 4 osiągamy poprzez uruchomienie pierwszej transakcji, a następnie nieprawidłowe przeplatanie pozostałych dwóch. 5 osiągniemy poprzez działanie zgodne z harmonogramem 3. Natomiast 6 ustawiając x na 0 i zwiększając go trzykrotnie.
18. Można ustawić blokadę na każdy indeks aby zapobiec aktywności dopóki wszystkie indeksy nie zostaną zastąpione.
19. Jeśli plik nie jest zablokowany to, próba zablokowania zawsze kończy się powodzeniem. Jeśli plik zostanie odczytany jako zablokowany, a próba dotyczy kolejnej blokady również się powiedzie. Natomiast w każdym innym przypadku np. Plik zablokowany do odczytu itp. Kończy się niepowodzeniem.
20. Blokadę odczytu można przekonwertować na blokadę zapisu tylko wtedy, gdy nie ma innych czytników. Wykonanie tej konwersji skutecznie zwalnia blokadę, a następnie natychmiast próbuje odzyskać ją jako blokadę zapisu. Ta operacja kończy się niepowodzeniem, jeśli istnieją inne procesy oczekujące. Zmniejszenie blokady z zapisu do odczytu jest zawsze legalne i zawsze powinno działać.
21. Tak.
22. Jest bardziej ograniczające, ponieważ przy użyciu optymistycznego sterowania, jeżeli transakcja próbuje się wykonać i zauważy że inna transakcja zmodyfikowała plik to zostanie przerwana.
23. Tak. Całkowicie szereguje transakcje w kolejności znaczników czasu, albo przerywa niektóre z nich.
24. Jeśli np system odpowiadał na obróbkę drewna, to to przerwanie transakcji nie przywróci nam zużytego surowca.