Przetwarzanie rozproszone, Temat: Wojna

Adrian Karolewski 145327, Dominik Ziębicki 145295

1. Przedstawienie problemu

Program odpowiada za naprawy N okrętów użytych do prowadzenia potyczek, po pewnym czasie okręt kończy walkę i wraca do bazy z różnymi wartościami uszkodzenia. Naszym zadaniem jest przydzielenie do okrętów wymagających naprawy jednego z K doków i x z M mechaników, ilość w zależności od wartości otrzymanych obrażeń. Po skończeniu naprawy mechanicy są dostępni dla innych jednostek wymagających naprawy a dok zostaje zwolniony. Okręt znów wyrusza na walkę.

1. Rozwiązanie

Rozwiązanie problemu wymaga od nas implementacji symulacji, toczenia bitew – co jakiś czas proces ( okręt będzie wymagał naprawy ze zmienna wartością obrażeń )

Potrzebujemy też mechanizmu dostępu do określonej liczby doków oraz gdy proces będzie miał przydzielony dok mechanizm dostępu do zmiennej M – reprezentującą ilość dostępnych mechaników, dla procesów znajdujących się w sekcji krytycznej K.

Implementacja dostępu do sekcji krytycznej będzie opierać się na algorytmie **Ricarta-Agrawali.** Każdy z procesów będzie miał swój zegar Lamporta który będzie służył nam za wyznacznik kolejkowania w dostępie do zasobów współdzielonych. Każdy z procesów będzie posiadał tablicę która odwzorowywać będzie kolejkę do wymaganych zasobów naprawczych. Początkowo każdy proces ( okręt wymagający naprawy ) będzie starał się uzyskać dostęp do doków, później do sekcji mechaników. Po wykonaniu napraw proces zwolni zasoby.

Algorytm realizacji dostępu do doków.

* Wątek ubiegający się o dostęp do doków

- Proces chcący ubiegać się o dostęp do doków wysyła wiadomość do siebie (wątku obsługującego komunikacje) z Tagiem REQ\_FOR\_K zawierająca jego zegar lamporta

- Proces chcący zwolnić dok wysyła do siebie ( wątku obsługującego komunikacje) wiadomość z tagiem ACK\_FOR\_K

* Wątek obsługujący komunikację

- Odbierze od swojego procesu wiadomość z tagiem REQ\_FOR\_K

Doda do swojej kolejki proces z aktualnym stanem zegara lamporta, wyśle broadcast wiadomość z tagiem REQ\_FOR\_K i wartością swojego zegara

- Odbierze od swojego procesu wiadomość z tagiem ACK\_FOR\_K

Podniesie wartość swojego zegara, wyśle ACK\_FOR\_K dla każdego procesu który jest za nim w jego kolejce i usunie je, usunie siebie z kolejki

- Odbierze od innego procesu wiadomość z tagiem REQ\_FOR\_K

Podniesie wartość swojego zegara, doda proces do kolejki, jeżeli dodany proces będzie miał lepszą pozycje w kolejce odeśle mu ACK\_FOR\_K

- Odbierze od innego procesu wiadomość z tagiem ACK\_FOR\_K

Podniesie wartość swojego zegara, sprawdź czy ack jest dla wysłanego przez niego aktualnego req, jeżeli proces od którego dostaliśmy ack będzie miał lepsza pozycje w kolejce usuwamy go z kolejki, inkrementujemy ilość swoich ack

Algorytm realizujący dostęp do mechaników działa na bardzo podobnej zasadzie. Dostęp do mechaników uzyskamy gdy dostaniemy ilość ack = ilośc mechaników \* (ilość procesów - 2) + ilość mechaników wymaganych przez dany proces. Gwarantuje nam to dostęp do sekcji krytycznej o wielkość równej = ilość mechaników z czego każdy proces może zajmować różną wielkość tej sekcji. Komunikacja wygląda podobnie jak przy uzyskaniu dostępu do sekcji krytycznej doków z tą różnicą iż przy odebraniu REQ\_M od innego procesu odsyłamy ack = ilości mechaników jeżeli proces jest lepiej usytuowany od nas w kolejce lub ack = ilość mechaników – ilość mechaników wymaganych przez nasz proces jeżeli proces od którego dostaliśmy REQ\_M jest dalej w kolejce do sekcji krytycznej. Przy odebraniu ACK\_M od swojego procesu rozsyłamy dla każdego procesu za nami w kolejce ack = ilość mechaników wymaganych przez nasz proces – co uzupełnia brakującą ilość miejsca w sekcji krytycznej. Gdy odbieramy ACK od zewnętrznego procesu inkrementujemy nasze wewnętrzne ack o wartość przysłaną w komunikacie.

1. Środowisko

Algorytm funkcjonuje w pełni asynchronicznym środowisku, nie mamy wspólnych zegarów, zasobów. Łącza są niezawodne a procesy nie ulegają awarii. Kanały komunikacyjne dostarczają wiadomości zgodnie z architekturą FIFO, a powinny pomieścić minimum 2 wiadomości mówimy tu o kanale jednokierunkowym P­1 -> P2  ( REQ , ACK ) .

1. Złożoność komunikacyjna i czasowa rozwiązania problemu

Złożoność komunikacyjna wynosi 2n – 2 ( dostęp do doków ) + 2n - 2 ( dostęp do mechaników) zaś czasowa: 4 .