Wiktoria Szymańska 140790

Igor Mierzejewski 148199

Przetwarzanie Rozproszone

Sprawozdanie z projektu

I. Temat projektu: „Skrzaty zabójcy szczurów”

Są dwa rodzaje procesów - S skrzatów i G gnomów. Gnomy ubiegają się o jedną z A agrafek i C celowników. Kombinują agrafki z celownikiem tworząc broń. Skrzaty ubiegają się o broń. Po zdobyciu broni zabijają szczury i zwracają agrafki i celowniki do puli.

II. Algorytm (na podstawie alg. Lamporta)

Na potrzebę projektu wykorzystano zmodyfikowany algorytm Lamporta. W celu umożliwienia dostępu do zasobów dla wielu procesów jednocześnie, zmieniony został warunek wejścia do sekcji krytycznej. Dodatkowo, zmieniono sposób w jaki procesy reagują na wiadomość typu RELEASE, zależnie od tego, który proces ją wysyła. W podstawowej wersji algorytmu, wiadomość tego typu oznacza zwolnienie zasobu. Jednak w omawianym problemie zasoby nie wracają od razu do puli, tylko są dalej wykorzystywane przez inne procesy. Otrzymanie RELEASE nie gwarantuje więc, że inny proces uzyska dostęp do zasobu, bo może go brakować. Na przykład, gdy proces typu gnom otrzyma RELEASE od innego gnoma, oznacza to tylko, że zwolniło się miejsce w kolejce. Dopiero RELEASE otrzymany od skrzata oznacza, że zasoby zostały zwolnione.

Dla każdego procesu uruchomione są dwa wątki – wątek komunikacyjny, odpowiadający za odbieranie i reagowanie na wiadomości, oraz wątek główny odpowiadający za przechodzenie między stanami.

Wejście do sekcji krytycznej uzależnione jest od liczby dostępnych zasobów. Proces musi więc kontrolować aktualną liczbę dostępnych zasobów, na podstawie otrzymywanych komunikatów. Każdy proces utrzymuje lokalne kolejki do zasobów, posortowane według zegara Lamporta. Chociaż liczniki zasobów oraz kolejki są lokalne, to ze względu na równoczesne wykonywanie dwóch wątków w obrębie jednego procesu (wątek główny i wątek komunikacyjny), dostęp do tych zmiennych musi być zabezpieczony za pomocą mutexów.

Wykorzystane struktury i zmienne (lokalne dla każdego procesu):

1) queue_agr, queue_cel, queue_bron – kolejki żądań procesów oczekujących na dostęp do zasobów, posortowane według zegara Lamporta
2) agr_count , cel_count, bron_count – aktualne liczby dostępnych zasobów
3) ackAgrCount, ackCelCount, ackBronCount – liczba otrzymanych potwierdzeń od innych procesów, od momentu ubiegania się o dany zasób

*Warunek wejścia do sekcji krytycznej:

Jeżeli żądanie procesu jest na n-tym lub wcześniejszym miejscu w kolejce do danego zasobu, gdzie n to liczba dostępnych zasobów, i proces otrzymał od wszystkich pozostałych procesów wiadomość (związaną z tym zasobem) o starszej etykiecie czasowej, to proces ma dostęp do zasobu.

GNOM:

Reakcje na wiadomości:

a) REQ_AGR – dodaje gnoma, od którego otrzymał wiadomość, do kolejki do agrafek queue_agr i wysyła mu potwierdzenie ACK_AGR.
b) ACK_AGR – gnom zwiększa swój ackAgrCount o 1.
c) RELEASE_BIORE_AGR – usuwa żądanie tego gnoma z kolejki i zmniejsza liczbę agrafek o 1.
d) REQ_CEL – dodaje gnoma, od którego otrzymał wiadomość, do kolejki do celowników queue_cel i wysyła mu potwierdzenie ACK_CEL.
e) ACK_CEL – gnom zwiększa swój ackCelCount o 1.
f) RELEASE_BIORE_CEL – usuwa żądanie tego gnoma z kolejki i zmniejsza liczbę celowników o 1.
g) RELEASE_ODDAJE_AGR_CEL – zwiększa liczbę agrafek i celowników o 1.

Stany:

1. Stan początkowy **InRun**: gnom ubiega się o dostęp do agrafek. Zeruje swój licznik zgód ackAgrCount. Wysyła do innych gnomów żądanie REQ_AGR. Dodaje swoje żądania do lokalnej kolejki queue_agr. Przechodzi do kolejnego stanu.
2. Stan **InWantAgr**: gnom czeka na dostęp do agrafek. Sprawdza, czy spełnia warunek wejścia do sekcji krytycznej*. Jeżeli tak, to przechodzi do kolejnego stanu.
3. Stan **InSectionAgr**: gnom może wziąć agrafkę. Wysyła do pozostałych gnomów RELEASE_BIORE_AGR i zmniejsza liczbę agrafek o 1. Usuwa swoje żądanie z kolejki queue_agr.
4. Stan **InRunCel**: gnom ubiega się o celownik. Zeruje swój licznik zgód ackCelCount. Wysyłą do innych gnomów żądanie REQ_CEL. Dodaje swoje żądanie do lokalnej kolejki queue_cel.
5. Stan **InWantCel**: gnom czeka na dostęp do celowników. Sprawdza, czy spełnia warunek wejścia do sekcji krytycznej*. Jeżeli tak, to przechodzi do kolejnego stanu.
6. Stan **InSectionCel**: gnom może wziąć celownik. Wysyła do pozostałych gnomów RELEASE_BIORE_CEL i zmniejsza liczbę celowników o 1. Usuwa swoje żądanie z kolejki queue_cel.
7. Stan **InAddBron**: gnom tworzy nową broń z agrafki i celownika. Gnom wysyła do skrzatów wiadomość RELEASE_DODAJE_BRON.

SKRZAT:

Reakcje na wiadomości:

a) REQ_BRON – dodaje skrzata, od którego otrzymał wiadomość, do kolejki do broni queue_bron i wysyła mu potwierdzenie ACK_BRON.
b) ACK_BRON – skrzat zwiększa swój ackBronCount o 1.
c) RELEASE_BIORE_BRON – usuwa żądanie tego skrzata z kolejki i zmniejsza liczbę broni o 1.
d) RELEASE_DODAJE_BRON – zwiększa liczbę broni o 1.

Stany:

1. Stan początkowy **InRun**: skrzat ubiega się o dostęp do broni. Zeruje swój licznik zgód ackBronCount. Wysyła do innych skrzatów żądanie REQ_BRON. Dodaje swoje żądanie do lokalnej kolejki queue_bron. Przechodzi do kolejnego stanu.
2. Stan **InWantBron**: skrzat czeka na broń. Sprawdza, czy spełnia warunek wejścia do sekcji krytycznej*. Jeżeli tak, to przechodzi do kolejnego stanu.
3. Stan **InSectionBron**: skrzat może wziąć broń. Wysyła do pozostałych skrzatów RELEASE_BIORE_BRON i zmniejsza liczbę broni o 1. Usuwa swoje żądanie z kolejki queue_bron.
4. Stan **InReturnResources**: skrzat zabija szczura i zwraca agrafkę i celownik. Wysyła RELEASE_ODDAJE_AGR_CEL do gnomów.

III. Złożoność czasowa: 7 + 4 = 11 (rund)

Gnom bierze agrafkę -> 3 rundy

Gnom bierze celownik -> 3 rundy

Gnom dodaje broń -> 1 runda

Skrzat bierze broń -> 3 rundy

Skrzat oddaje broń -> 1 runda

IV. Złożoność komunikacyjna: 7G-6 + 4S-3

Gnom wysyłają wiadomości do pozostałych gnomów 6 razy -> 6G-6

Gnom wysyła wiadomość do wszystkich skrzatów 1 raz -> 1S

Skrzat wysyła wiadomość do pozostałych skrzatów 3 razy -> 3S-3

Skrzat wysyła wiadomość do wszystkich gnomów 1 raz -> 1G