Synchronizacja procesów z wykorzystaniem semaforów

Autor: Halyna Polekha

Problem

Producent-konsument, przy następujących założeniach:

- występuje jeden "producent", który generuje zadania
- oraz trzech "konsumentów" A, B, C, którzy czytają i usuwają elementy z bufora
- do komunikacji jest wykorzystywany 1 bufor
- element jest usuwany z bufora, jeżeli zostanie przeczytany przez albo obu konsumentów A i B, albo przez obu konsumentów B i C
- konsument A nie może przeczytać elementu, jeżeli został on już przez niego wcześniej przeczytany albo został przeczytany przez konsumenta C i na odwrót.

Założenia wstępne

- nie dopuścić do czytania z pustego bufora,
- nie dopuścić do zapisu do pełnego bufora,

Opis struktur danych reprezentujących bufory komunikacyjne

Do obsługi kolejki dla wybranego bufora będę stosować strukturę oraz *enum* czytelników do modyfikacji flagi:

```
typedef struct FIFOQUEUE
{
        short int start, end;   // Wskazniki na head i tail kolekcji (zainicjalizowane
                                                                zerami)
        int size;       // Wskazniki na rozmiar kolekcji (zainicjalizowany zerem )
        int *prd;               // Wskaźnik na dynamicznie tworzoną tablicę przechowującą
                                                                kolejkę elementów
        reader qr;      // Flaga „kro przeczytał”

}Queue;


typedef enum reader{ NoNe, A, B, C}reader;
```

Taka struktura będzie opisywała kolejkę elementów. Znając indeks początku i indeks końca oraz o liczbę elementów kolejki – można łatwo określić, czy bufor jest pusty, czy pełny.

Opis struktur reprezentujących semafory

```
typedef struct ProjectSemaphores
{

    sem_t write;      // Mutex regulujący zapisywanie do kolejki
    sem_t read_A;     // Mutex do regulacji przerywań czytania przez konsumenta A
    sem_t read_B;     // Mutex do regulacji przerywań czytania przez konsumenta B
    sem_t read_C;     // Mutex do regulacji przerywań czytania przez konsumenta C
    sem_t MUTEX;      // Mutex zapewniający wyłączny dostęp do kolejki

}Semaphores;
```

Semafor **MUTEX** będzie zainicjalizowany wartością 1 – od początku życia programu możliwe jest jego zablokowanie.

Semafory **write, read_A, read_B, sem_t read_C** będą zainicjalizowane wartością 0, ponieważ żaden z konsumentów nie jest w stanie oczekiwania kontynuacji swojego działania w sekcji krytycznej.

Działanie programu

Producent będzie zgłaszał zadanie, które będzie padało do kolejki, jeśli ta nie jest pełna.

Każdy z producentów będzie zgłaszał zadanie, które będzie padało do kolejki o najmniejszej ilości elementów.

Te zadania będzie odbierał jeden z trzech konsumentów. Przy czym w taki sposób, że będzie pobierał zadanie z kolejki wyłącznie po poprzednim przeczytaniu (modyfikacji flagi) przez odpowiedniego konsumenta.

Pseudokod przedstawiający synchronizację procesów

Producent

1. Poczekaj na podniesienie semafora **MUTEX**, opuść
2. Czy kolejka nie jest pełna? Jeżeli jest
    a) zgłoś oczekiwanie na kontynuacje działania przez modyfikacje flagi **wait_for_write = true**;
    b) wyjdź z sekcji krytycznej (podniesienie **MUTEX**)
    c) czekaj na podniesienie semafora **write**, opuść i kontynuuj działanie
3. Ustaw **flagę wait_for_write** na wartość domyślną **false**
4. Wstaw element do kolejki (na koniec)
5. Podnieś jeden z innych semaforów, jeżeli jest żądanie do kontynuacji działania
6. Opuść sekcję krytyczną przez podniesienie semafora **MUTEX**
7. Po wysłaniu odpowiedniej liczby sygnałów – zakończ program

Konsument

1. Poczekaj na podniesienie semafora **MUTEX**, opuść
2. Czy kolejka nie jest pusta? Jeżeli jest
    a) zgłoś oczekiwanie na kontynuacje działania przez modyfikacje odpowiedniej flagi **wait_for_read = true**;
    b) wyjdź z sekcji krytycznej (podniesienie **MUTEX**)

c) czekaj na podniesienie odpowiedniego semafora **read**, opuść i kontynuuj działanie

3. Sprawdź flagę elementu z głowy kolejki i odpowiednio do warunków zadania usuń element, ustaw flagę lub ignoruj modyfikację.
4. Podnieś jeden z innych semaforów, jeżeli jest żądanie do kontynuacji działania
5. Opuść sekcję krytyczną przez podniesienie semafora **MUTEX**
6. Po obsłużeniu określonej liczby sygnałów – zakończ działanie

Main

1. Producent zaczyna działanie
2. Konsumenci zaczynają działanie
3. Poczekaj na zakończenie wszystkich procesów potomnych
4. zakończ