Semafory - zadanie 3

Mikołaj Garbowski

Polecenie

Mamy bufor FIFO na liczby całkowite.

- Procesy A1 generują kolejne liczby parzyste modulo 50, jeżeli w buforze jest mniej niż 10 liczb parzystych.
- Procesy A2 generują kolejne liczby nieparzyste modulo 50, jeżeli liczb parzystych w buforze jest więcej niż nieparzystych.
- Procesy B1 zjadają liczby parzyste pod warunkiem, że bufor zawiera co najmniej 3 liczby.
- Procesy B2 zjadają liczby nieparzyste, pod warunkiem, że bufor zawiera co najmniej 7 liczb.

W systemie może być dowolna liczba procesów każdego z typów. Zrealizuj wyżej wymienioną funkcjonalność przy pomocy semaforów.
Zakładamy, że bufor FIFO poza standardowym put() i get() ma tylko metodę umożliwiającą sprawdzenie liczby na wyjściu (bez wyjmowania) oraz posiada metody zliczające elementy parzyste i nieparzyste. Zakładamy, że semafory mają tylko operacje P i V.

Koncepcja rozwiązania

Główny program, napisany w C++

- Tworzy bufor FIFO i inicjuje wartości semaforów
- Przyjmuje jako parametr liczbę procesów z każdej z grup i tworzy je przez wywołanie `fork`
- Mapuje bufor i semafory do współdzielonej pamięci

Skoro bufor umożliwia zliczenie elementów parzystych i zliczenie elementów nieparzystych to również umożliwia zliczenie wszystkich elementów

Wykorzystywane semafory

- `mutex` - binarny, zapewnia wyłączny dostęp do bufora
  - na początku 1
  - opuszczany przed wejściem do sekcji krytycznej
  - podnoszony po wyjściu z sekcji krytycznej
- `canA1Save`
  - liczący
  - inicjowany wartością 10

    - ile liczb parzystych jeszcze można wstawić
    - opuszczany przez A1
    - podnoszony przez B1
- `canA2Save`
    - inicjowany wartością 0 bo na początku bufor jest pusty
    - liczba parzystych - liczba nieparzystych
    - podnoszony przez A1 - zwiększa się liczba parzystych
    - opuszczany przez B1 - zmniejsza się liczba parzystych
    - opuszczany przez A2 - zwiększa się liczba nieparzystych
    - podnoszony przez B2 - zmniejszza się liczba nieparzystych
- `canB1Read`
    - liczący
    - inicjowany wartością 0
    - podnoszony przez A1 warunkowo, jeśli liczba elementów większa lub równa 3
    - opuszczany przez B1
- `canB2Read`
    - liczący
    - inicjowany wartością 0
    - podnoszony przez A2 warunkowo, jeśli liczba elementów większa lub równa 7
    - opuszczany przez B2

W treści zadania nie ma mowy o ograniczeniu rozmiaru bufora. Gdyby był ograniczony należałoby wprowadzić dodatkowy semafor liczący `full` zabezpieczający przed zapisem do pełnego bufora

- inicjowany rozmiarem bufora
- opuszczany przy każdym zapisie
- podnoszony po każdym odczycie

Sekcje krytyczne

Dla każdego typu procesu, blok instrukcji, które wykonuje w pętli

- sprawdzenie liczb elementów, czy może wykonać operację
- zapis / odczyt do bufora

jest sekcją krytyczną

Każda musi być poprzedzona `mutex.p()` i zakończona `mutex.v()`