ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа бакалавриата «Программная инженерия»

РЕШЕНИЕ ЗАДАЧИ О ЧИТАТЕЛЯХ И ПИСАТЕЛЯХ

Пояснительная записка

Исполнитель
студентка группы БПИ196
/Д.В. Еремина/
13 декабря 2020 г.

Москва 2020

СОДЕРЖАНИЕ

1. ПОСТАНОВКА ЗАДАЧИ

Вариант 8:

Базу данных разделяют два типа потоков – читатели и писатели. Читатели выполняют транзакции, которые просматривают записи базы данных, транзакции писателей и просматривают и изменяют записи. Предполагается, что в начале БД находится в непротиворечивом состоянии (т.е. отношения между данными имеют смысл). Транзакции выполняются в режиме «грязного чтения», то есть процесс-писатель не может получить доступ к БД только в том случае, если ее занял другой процесс-писатель, а процессы-читатели ему не мешают. Создать многопоточное приложение с потоками-писателями и потоками-читателями. Реализовать решение, используя семафоры, и не используя блокировки чтения-записи.

2. ПРОЦЕСС ВЗАИМОДЕЙСТВИЯ

По умолчанию базе данных есть три некоторых элемента.

Логика взаимодействия с базой организована следующим образом: по умолчанию потоки-писатели имеют приоритет над потоками-писателями, так как по условию писателю может помешать начать исполнение записи лишь другой поток-писатель.

Процесс чтения происходит следующим образом: пусть некоторый читатель хочет начать чтение. Тогда если сейчас есть активные или ожидающие записи писатели, этот читатель должен подождать. Иначе можно непосредственно начать процесс чтения. Также после завершения чтения необходимо проверить, остались ли ожидающие писатели. Если да, необходимо разрешить им начать запись.

Писатель же должен подождать, если есть активные читатели или писатели, так как только после этого он может начать обновлять базу данных, иначе он может начинать запись немедленно. Когда запись окончена, необходимо разбудить потоки, ожидающие очереди на запись, если таковые есть, иначе разбудить потоки чтения, если есть ожидающие.

3. ПРОТОКОЛ ВЗАИМОДЕЙСТВИЯ

В основной программе пользователь вводит *n_read*, *n_write*, *iter* – значения переменных, отвечающих за число читателей, писателей и итераций процесса соответственно (под одной итерацией понимается блок, в котором *n_read* читателей и *n_write* писателей совершат соответствующие действия).

Далее создаются массивы из соответствующих потоков (для работы с ними использована библиотека pthread.h), каждый из которых потом начинает работу.

Взаимодействие потоков происходит при помощи трех семафоров: *mutex*, отвечающего за начало критических секций, а также *writing* и *reading*, каждый из которых сигнализирует что нужно ожидать/начинать соответствующий процесс.

В соответствии с описанным в п.2 процессом за запуск соответствующего процесса отвечает вызов метода *sem_post(&…),* а за ожидание соответственно *sem_wait(&…).*

4. ДЕМОНСТРАЦИЯ РАБОТЫ ПРОГРАММЫ

```
Введите количество писателей:
101
Количество должно быть в диапазоне [1; 10]:
3
Введите количество читателей:
-1
Количество должно быть в диапазоне [1; 100]:
5
Введите количество производимых итераций:
100000
Количество должно быть в диапазоне [1; 100]:
2
Writer №3 writes value 53 to data[0]
Writer №3 writes value 57 to data[0]
Reader №2 gets value 0 from data[1]
Reader №2 gets value 0 from data[2]
Reader №2 gets value 0 from data[2]
Writer №3 writes value 45 to data[0]
Writer №3 writes value 75 to data[1]
Writer №3 writes value 3 to data[0]
Reader №3 gets value 0 from data[1]
Reader №4 gets value 0 from data[1]
Reader №4 gets value 0 from data[2]
Writer №3 writes value 51 to data[1]
Reader №2 gets value 3 from data[0]
Reader №5 gets value 0 from data[1]
Reader №5 gets value 0 from data[2]
Reader №3 gets value 0 from data[2]
```

Рисунок 1. Пример работы программы, демонстрация обработки некорректных входных данных

```
Введите количество писателей:
2
Введите количество читателей:
3
Введите количество производимых итераций:
1
Writer №1 writes value 55 to data[0]
Reader №2 gets value 0 from data[2]
Writer №2 writes value 2 to data[1]
Reader №3 gets value 2 from data[1]
Reader №3 gets value 55 from data[0]
```

Рисунок 2. Пример одной итерации процесса

```
Введите количество писателей:
1
Введите количество читателей:
1
Введите количество производимых итераций:
3
Reader №1 gets value 0 from data[0]
Writer №1 writes value 65 to data[0]
Reader №1 gets value 65 from data[0]
Reader №1 gets value 55 from data[0]
Writer №1 writes value 55 to data[0]
Writer №1 writes value 75 to data[0]
```

Рисунок 3. Пример работы одного писателя и одного читателя с одним элементом в базе

5. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) http://www.codenet.ru/progr/cpp/7/3.php
2) http://www.cs.kent.edu/~farrell/osf03/oldnotes/L15.pdf
3) http://softcraft.ru/edu/comparch/lect/07-parthread/multitreading.pdf