Архитектура вычислительных систем

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ПРАКТИЧЕСКОЙ РАБОТЕ НА ТЕМУ:

«Построение многопоточных приложений»

Работу выполнил:

студент группы БПИ207
Пендищук Владислав

Москва
2021 г.

ОГЛАВЛЕНИЕ

ОПИСАНИЕ ЗАДАНИЯ

Задача состояла в разработке консольного приложения с использованием библиотеки POSIX Threads языка программирования C или стандартной библиотеки языка программирования C++.

Требования к функционалу

В соответствии с полученным вариантом задания (18 – «Задача о болтунах») задача формулировалась следующим образом:

«N болтунов имеют телефоны, ждут звонков и звонят друг другу, чтобы побеседовать. Если телефон занят, болтун будет звонить, пока ему кто-нибудь не ответит. Побеседовав, болтун не унимается и или ждет звонка или звонит на другой номер. Создать многопоточное приложение, моделирующее поведение болтунов. Для решения задачи использовать мьютексы.»

Требования к запуску и вводу\выводу

Требований к запуску программы и вводу\выводу из неё приведено не было.

ИСПОЛЬЗУЕМАЯ МОДЕЛЬ ПОСТРОЕНИЯ МНОГОПОТОЧНОГО ПРИЛОЖЕНИЯ

Полученная программа была разработана на языке C с использованием библиотеки POSIX threads (*pthreads*).

Описание модели

В ходе разработки использовалась парадигма параллельного программирования, называемая «Управляющий и рабочие» – модель организации вычислений, при которой существует поток, координирующий работу всех остальных потоков. Как правило, управляющий поток распределяет данные, собирает и анализирует результаты. [2]

Эта парадигма часто применяется в задачах оптимизации и статистической обработки информации, при обработке изображений и других научных вычислениях с итеративными алгоритмами. [2]

Существует ряд подвидов данной парадигмы. Так, одной из них является парадигма «Круговой конвейер», которая подразумевает взаимодействие процессов-вычислителей через круговой конвейер, элементами которого являются сами процессы. Другим подвидом данной модели является парадигма «Взаимодействующие равные» с распределённой памятью ВС, в которой портфель задач (ряд задач, определённых для выполнения равноправными процессами) реализован в виде отдельного процесса (который таким образом выполняет роль управляющего потока).

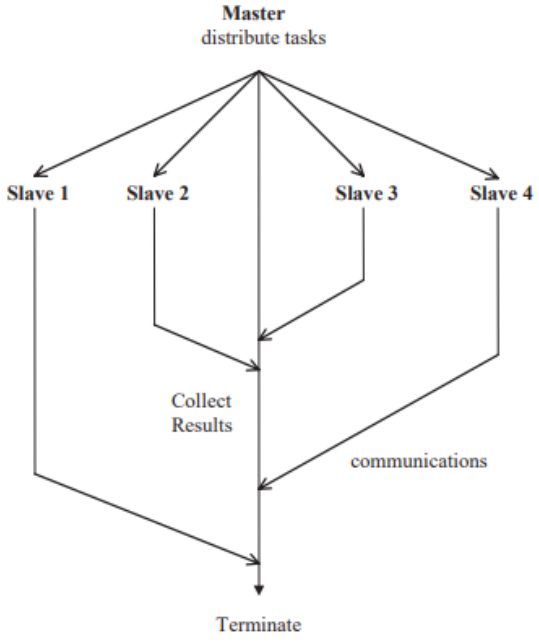

Рис. 1: модель «управляющий и рабочие (master & slaves)» [1]

Использование модели в программе

В разработанной программе модель взаимодействия потоков исполнения можно изобразить следующим образом (при условии, что в программе симулируется взаимодействие n болтунов):

Управляющий (master) поток, на котором запущена программа, содержит в себе объект (контейнер), хранящий в себе массив объектов, представляющих болтунов. Для каждого из этих объектов master-поток создаёт выделенный POSIX slave-поток, который выполняет функцию, симулирующую поведение болтуна заданное пользователем количество времени в контексте заданного массива болтунов как возможных собеседников.

При этом наблюдается классическая для данной парадигмы модель доступа к памяти: управляющим потоком осуществляется выделение и инициализация общей памяти (памяти массива болтунов), после чего адрес выделенной памяти передаётся при создании рабочим потокам. В процессе выполнения функции симуляции поведения болтуна рабочие потоки обращаются к данному массиву по предоставленному адресу (указателю) (для доступа к данным объектов-болтунов в массиве – имени и собеседнику болтуна). При этом, для синхронизации доступа к общей памяти используется общий mutex-объект, который блокируется тем потоком, который осуществляет доступ к общей памяти в данный момент. С использованием того же механизма синхронизации рабочие потоки изменяют данные (при назначении собеседников болтунам). Таким образом, управляющий поток передаёт данные рабочим потокам, а последние изменяют их синхронизировано между собой.

ОСНОВНЫЕ ХАРАКТЕРИСТИКИ ПО

Исходный код программы содержится в 7 файлах. 4 из них являются модулями реализации (файлами .c с определением программных объектов), 3 – интерфейсными модулями (заголовочными файлами языка С). Общий размер исходных текстов следующий:

	С форматированием и комментариями	*Без форматирования и комментариев*
С интерфейсными модулями	557	325
Без интерфейсных модулей		

Размер исполняемого файла, полученного после компиляции кода на ОС Linux, равен 34456 байт (33,6 килобайт).

Результаты тестов с использованием случайной генерации объектов контейнера следующие (временные настройки – максимум 5 секунд на разговор, минута на работу рабочего потока (симуляции болтуна)):

Количество элементов	*Время работы программы*
5 болтунов	98 секунд
1000 элементов	102 секунды
10000 элементов	99 секунд

Примечание: время, указанное для продолжительности работы одного болтуна, не совпадает с временем работы программы ввиду временных затрат на выполнение программы управляющим (основным) потоком и затратами на синхронизацию между потоками (потоки блокируются пока иной поток осуществляет доступ к данным

ПРИЛОЖЕНИЕ 1

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. High Performance Cluster Computing: Programming and Applications / Rajkumar Buyya - 2 изд. - Хобокен, США: Prentice Hall, 1999. - 664 с.
2. Основы многопоточного и параллельного программирования: учеб. пособие / Е. Д. Карепова. – Красноярск: Сиб. федер. ун-т, 2016. - 356 с.