Национальный исследовательский университет “Высшая Школа Экономики”,
Факультет компьютерных наук
Департамент программной инженерии

«Задача об обедающих философах»

Пояснительная записка

Исполнитель:
Студент группы БПИ199
Федченко Всеволод Александрович

Москва 2020

Оглавление

1. Текст задания

Задача об обедающих философах. Пять философов сидят возле круглого стола. Они проводят жизнь, чередуя приемы пищи и размышления. В центре стола находится большое блюдо спагетти. Спагетти длинные и запутанные, философам тяжело управляться с ними, поэтому каждый из них, что бы съесть порцию, должен пользоваться двумя вилками. К несчастью, философам дали только пять вилок. Между каждой парой философов лежит одна вилка, поэтому эти высококультурные и предельно вежливые люди договорились, что каждый будет пользоваться только теми вилками, которые лежат рядом с ним (слева и справа). Написать многопоточную программу, моделирующую поведение философов с помощью семафоров. Программа должна избегать фатальной ситуации, в которой все философы голодны, но ни один из них не может взять обе вилки (например, каждый из философов держит по одной вилке и не хочет отдавать ее). Решение должно быть симметричным, то есть все потоки-философы должны выполнять один и тот же код. [1][2]

2. Применяемые расчетные методы

2.1 Запуск программы

Для компиляции и запуска программы в консоль необходимо ввести строчку c++ ./avs_mp_2.cpp -o main -lpthread -std=c++11 && ./main <argv1>, где <argv1> опциональный аргумент, отвечающий за количество итераций.

2.2 Алгоритм

Для того, чтобы избежать ситуации, когда каждый философ (поток) взял по одной вилке и ждет, когда освободится еще одна вилка, и, следовательно, ждет вечно, так как никто не отдаст свою вилку до тех пор, пока не поест (так называемый deadlock[3] потоков) используется решение, впервые предложенное Дейкстрой. Алгоритм представляет собой присвоение частичного порядка ресурсам и установление соглашения, что ресурсы запрашиваются в указанном порядке, а возвращаются в обратном порядке.

Для решения поставленной задачи обозначенным выше алгоритмом, каждой вилке был присвоен свой номер. Далее, для каждого сидящего за столом философа была назначена левая и правая вилка таким образом, что левая вилка имеет наименьший номер из тех, что может взять данный филосов, а правая вилка — наибольший. Если философ проголодался, он должен сначала взять левую вилку (подождать до ее освобождения, если она занята), а затем правую. После окончания трапезы, которая длится час (секунду), философ должен вернуть на стол сначала правую вилку, а затем левую. Таким образом патовая ситуация (deadlock) невозможна. Если философ не успел проголодаться, по истечению сеанса размышлений в 1.5 часа (1.5 секунды), то он продолжит размышлять не прерываясь на еду.

Философы сидят за столом до тех пор, пока в очередной сеанс размышления или приема пищи им не надоест это делать (завершение программы после n итераций; n – введенное пользователем или заданное по умолчанию число).

2.2 Входные данные

Входными данными является количество итераций цикла размышления или приема пищи. Вводятся через консоль и являются опциональными. Если пользователем не был введен данный аргумент, то ему по умолчанию количество итераций будет равно 3.

Замечание: входные данные должны представлять собой целое положительное число типа int. При введении некорректных данных программа выдаст соответствующее сообщение об ошибке и завершит свою работу.

2.3 Выходные данные

Каждый философ выводит в консоль информацию о своем состоянии. Если он размышляет, проголодался и ждет возможности поесть или осуществляет прием пищи, соответствующая информация будет выведена в консоль. Для корректного вывода информации в консоль философ блокирует возможность вывода информации другими философами, пока данный философ не закончит вывод информации. После этого данный философ снимает ограничение на запись и продолжает свое существование.

3. Тестовые примеры

Корректная работа программы при одной итерации. Итерация всего одна, т.к при большем количестве консольный вывод слишком большой для наглядного демонстрирования с помощью скриншота. (см. рис 1)

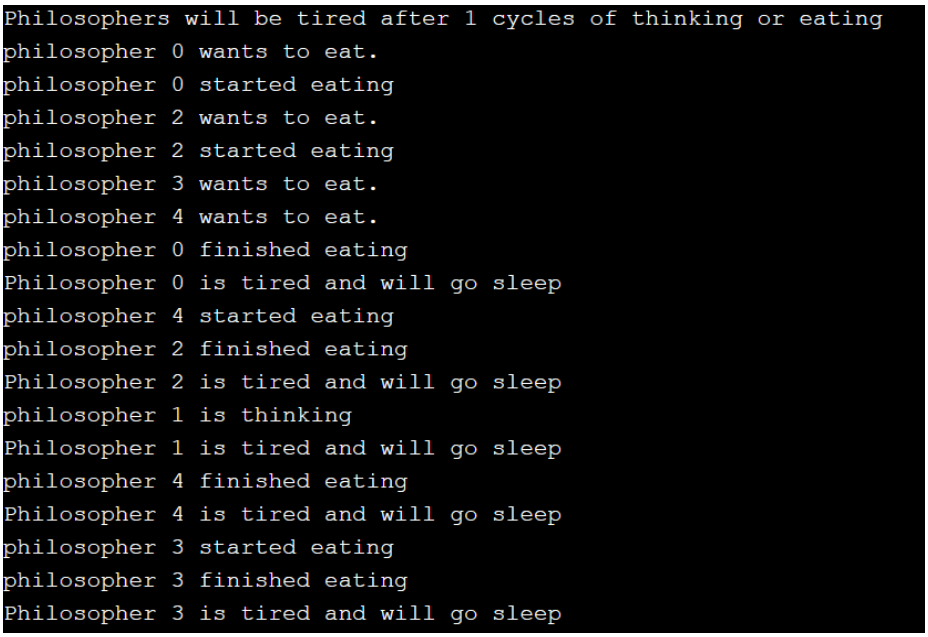

Рис 1. Работа программы

Обработка аргумента меньшего 1 (см. рис. 2).

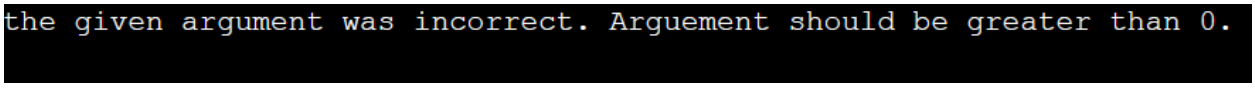

Рис 2. Работа программы

Обработка аргумента не являющегося целым чилом типа int (см. рис. 2).

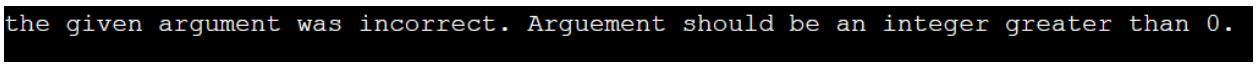

Рис 2. Работа программы

4. Список используемых источников

[1] Практические приемы построения многопоточных приложений. [Электронный ресурс]. // URL: http://softcraft.ru/edu/comparch/tasks/t03/ (Дата обращения: 12.12.2020, режим доступа: свободный)

[2] Dining philosophers problem. [Электронный ресурс]. // URL:https://en.wikipedia.org/wiki/Dining_philosophers_problem (Дата обращения: 12.12.2020, режим доступа: свободный)

[3] Deadlock. [Электронный ресурс]. // URL:https://en.wikipedia.org/wiki/Deadlock (Дата обращения: 12.12.2020, режим доступа: свободный)