

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Департамент программной инженерии

**МИКРОПРОЕКТ № 2
ЗАДАЧА О БОЛТУНАХ**

Пояснительная записка

Исполнитель
Студент группы БПИ 196
Сахаров Никита Денисович

Москва, 2020

Содержание

1	Постановка задачи	2
2	Описание алгоритма	3
2.1	Описание структуры данных	3
2.1.1	enum class Statement	3
2.1.2	struct Talker	3
2.1.3	метод log	3
2.1.4	метод live	3
2.2	Формат входных данных	4
2.2.1	Аргументы командной строки	4
2.2.2	Файл с описанием болтунов	4
2.3	Формат выходных данных	4
3	Список литературы	5
4	Приложение 1. Граф состояний болтуна	6
5	Приложение 2. Тестирование программы	7

1 Постановка задачи

Задача о болтунах. 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 мутексы.

2 Описание алгоритма

Жизненный цикл каждого болтуна описывается одной функцией. В ней в бесконечном цикле с помощью switch-case каждый болтун реализует сценарий его текущего состояния (сценарии кратко описаны в описании функции).

Доступ к полям ботуна защищается его личным мьютексом. Кроме того для реализации состояний ожидания и выслушивания используются семафоры, которые опускаются им при ожидании и взводятся собеседником болтуна.

Запись логов защищена глобальным мьютексом. Запись истории разговоров защищена мьютексом, который выдается болтунам из пула мьютексов в начале разговора и возвращается в него по окончании.

2.1 Описание структуры данных

2.1.1 enum class Statement

Состояния болтунов

Состояние: **Описание:**

wait	Болтун ожидает звонка
call	Болтун Звонит другому болтуну
talk	Болтун говорит что-то собеседнику
listen	Болтун слушает собеседника
refresh	Болтун очищает свои данные для разговора

2.1.2 struct Talker

Структура описывающая болтуна.

Тип:	Поле:	Описание:
Statement	state	Состояние
int	number	Номер
string	name	имя
string	message	Уникальное сообщение
bool	wasCall	Флаг был ли он звонящим на прошлой итерации
bool	isWaiter	Флаг является ли он сейчас ожидающим звонка
int	talkInd	Номер разговора по счету
int	numberToCall	Номер болтуна которому хочет звонить
pthread_mutex_t	lock	Мьютекс для доступа к данным
sem_t	sem	Семафор ожидания
pthread_mutex_t	*talkLock	Ссылка на мьютекс разговора

2.1.3 метод log

Метод, вывода логов в консоль или файл. Выбор приемника логов зависит от аргументов командной строки.

Перегрузки

Аргументы :	int, const string	Номер болтуна и сообщение
Аргументы :	int, const string, int	Номер болтуна, сообщение и номер собеседника
Аргументы :	const string	Сообщение

2.1.4 метод live

Метод, описывающий жизнь болтуна, реализует смену состояний данного болтуна.

Логика состояний

- **wait:** Уменьшает свой семафор и ожидает, при выходе убирает флаг ожидания звонка. Когда ему дозвонились и взвели его семафор переходит в состояние listen.

- **call:** Выбирает случайный номер, проверяет доступность болтуна для разговора. Если доступен получает номер разговора и мьютекс из пула и передает их собеседнику, взводит его семафор и переходит в состояние `talk`, иначе ждет некоторое время и завершает итерацию без изменения состояния.
- **talk** Пишет в файл разговора сообщение. Если количество оставшихся фраз в разговоре ненулевое взводит семафор собеседника и переходит в состояние слушателя, иначе переходит в состояние `refresh`.
- **listen** Уменьшает свой семафор и ожидает пока собеседник договоит и взведет его. Если фразы в разговоре кончились переходит в состояние `refresh`, иначе в состояние `talk`.
- **refresh** Возвращает мьютекс в пул и очищает ссылки на него у участников разговора, переходит в состояние `call` или `wait` в зависимости от флага `wasCall`.

Аргументы : `void*` Передается структура описывающая болтуна

2.2 Формат входных данных

2.2.1 Аргументы командной строки

В аргументах командной строки передается 4 или 5 параметров.

- 4 : количество болтунов из файла, путь к файлу с описанием болтунов, количество фраз в разговоре, время моделирования.
- 5 : количество болтунов из файла, путь к файлу с описанием болтунов, имя файла с логом разговоров, количество фраз в разговоре, время моделирования.

2.2.2 Файл с описанием болтунов

В файле с расширением `.txt` построчно должна быть записана информация о болтунах.

Строка с информацией о болтуне должна иметь следующий формат: `"*name* *phrase* *state*"`

- **name** Строка имя (без пробелов).
- **name** Строка уникальной фразы (без пробелов).
- **state** 0 или 1, обозначающие состояния `wait` и `call`.

2.3 Формат выходных данных

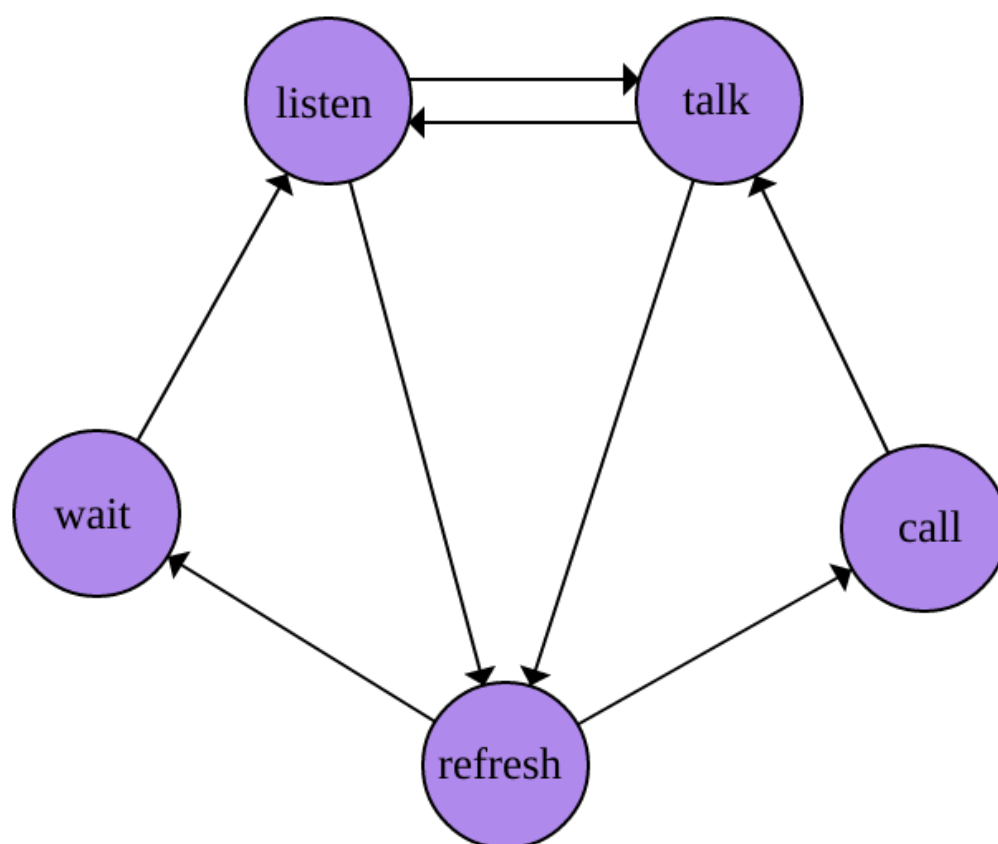
Программа формирует:

1. Каталог с файлами, описывающими разговоры между болтунами.
2. Лог сообщений от всех потоков в консоли или указанном файле.

3 Список литературы

- Пункт 4 из предложенного списка литературы по многопоточности. Грегори Р. Эндрюс. Основы многопоточного, параллельного и распределенного программирования. - М.: Издательский дом "Вильямс 2003.
- Семафоры в pthreads [Электронный ресурс] / learnc.info (дата обновления: 15.01.2018). - Электрон. текстовые дан. - Режим доступа: https://learnc.info/c/pthreads_semaphores.html (дата обращения: 08.12.2020).

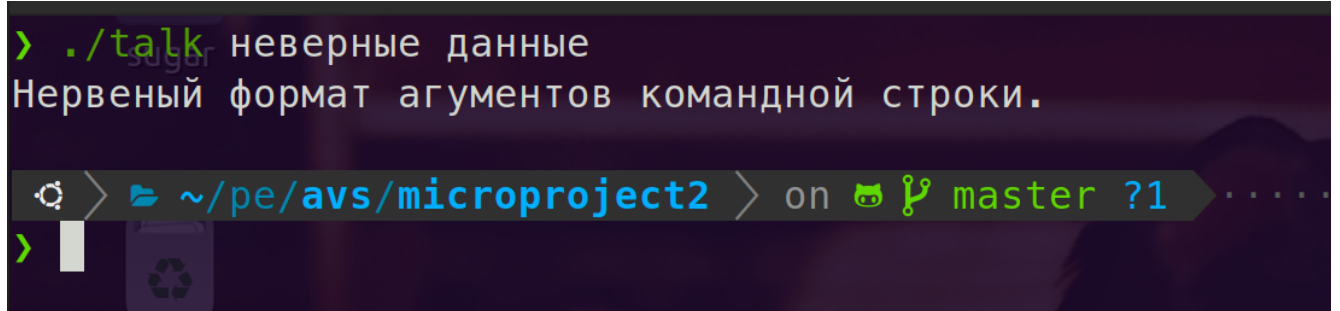
4 Приложение 1. Граф состояний болтуна



5 Приложение 2. Тестирование программы

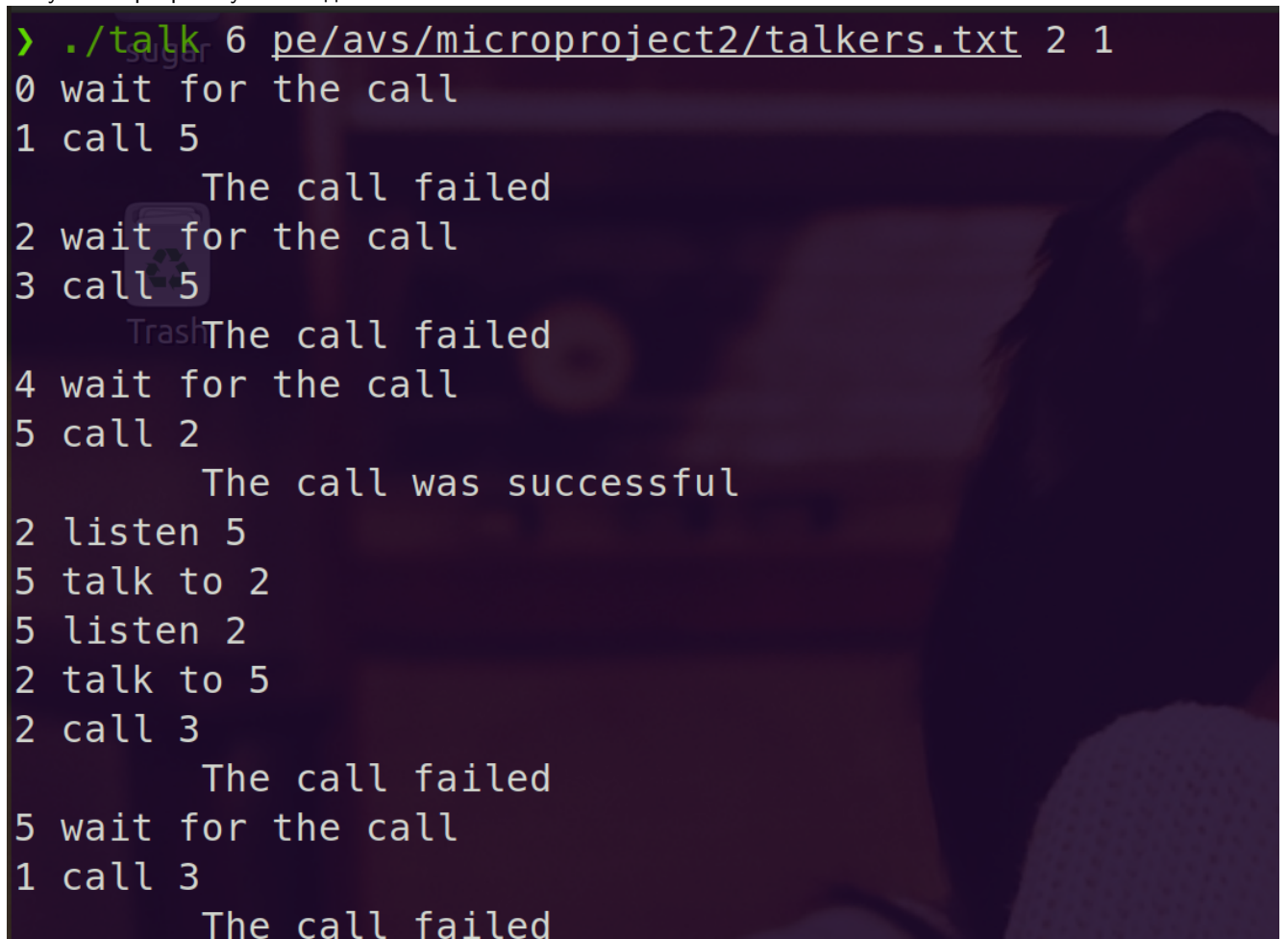
Введем аргументы не по формату :

```
> ./talk неверные данные
Нервный формат аргументов командной строки.
```



Запустим программу с выводом лога в консоль :

```
> ./talk 6 pe/avs/microproject2/talkers.txt 2 1
0 wait for the call
1 call 5
    The call failed
2 wait for the call
3 call 5
    The call failed
4 wait for the call
5 call 2
    The call was successful
2 listen 5
5 talk to 2
5 listen 2
2 talk to 5
2 call 3
    The call failed
5 wait for the call
1 call 3
    The call failed
```



Результаты остальных тестов можно найти в соответствующих папках репозитория на github.

<https://github.com/Sugarhl/avs/tree/master/microproject2>

Аргументы командной строки для запуска тестов :

1. 3 ../talkers.txt log.txt 4 4
2. 5 ../talkers.txt log.txt 4 4
3. 12 ../talkers.txt log.txt 4 4
4. 10 newtalkers.txt log.txt 2 10