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

««Программа моделирования задачи об острове сокровищ с использованием библиотеки OpenMP»

Пояснительная записка

Исполнитель:
Студент группы БПИ199
Федченко Всеволод Александрович

Москва 2020

Оглавление

1. Текст задания

Вторая задача об Острове Сокровищ. Шайка пиратов под предводительством Джона Сильвера высадилась на берег Острова Сокровищ. Несмотря на добытую карту старого Флинта, местоположение сокровищ по-прежнему остается загадкой, поэтому искать клад приходится практически на ощупь. Так как Сильвер ходит на деревянной ноге, то самому бродить по джунглям ему не с руки. Джон Сильвер поделил остров на участки, а пиратов на небольшие группы. Каждой группе поручается искать клад на нескольких участках, а сам Сильвер ждет на берегу. Группа пиратов, обшарив одну часть острова, переходит к другой, еще не обследованной части. Закончив поиски, пираты возвращаются к Сильверу и докладывают о результатах. Требуется создать многопоточное приложение с управляющим потоком, моделирующее действия Сильвера и пиратов. При решении использовать парадигму портфеля задач. [1]

2. Применяемые расчетные методы

2.1 Алгоритм

При запуске программы создается массив типа bool, имитирующий участки острова. Далее рандомно выбирается участок, в котором будет находиться сокровище (в массиве по этому индексу устанавливается значение true). В качестве команд пиратов выступают потоки, каждый из которых пытается найти «сокровище» на выделенных ему участках. Все потоки выполняют одну функцию: `searchForTreasure`, входным параметром которой является количество потоков. В ней и происходит распараллеливание задачи. Каждый поток имеет общую переменную типа bool *globalFlag* и приватную переменную типа bool *localFlag*. Переменная *globalFlag* отвечает за то, было ли найдено сокровище одной из поисковых команд. Если значение *false*, то сокровище найдено не было и все команды продолжают поиск. Если значение *true*, то какая-то из команд уже нашла сокровище, поэтому все остальные команды, как только видят этот флаг прекращают поиски. Переменная *localFlag* является внутренней для каждого потока и нужна для вывода данных.

Важное замечание: библиотека OpenMP[2] не позволяет завершать распараллеленный цикл *for* командой *break*, поэтому де-факто даже после того, как *globalFlag* становится равен *true*, потоки продолжат выполняться, однако тело цикла не выполняется.

2.2 Входные данные

Входными данными являются количество участков на острове и количество ищущих команд (потоков). Вводятся они через аргументы консоли в формате a.exe <кол-во секторов> <кол-во потоков>. Оба аргумента должны быть типа int32, количество секторов должно быть больше 0, количество потоков должно быть больше 0 и не больше количества секторов. При введение некорректных данных работоспособность программы не гарантируется.

Команда для компиляции: c++ avs_hw_4.cpp -fopenmp -std=c++11

Команда для запуска: a.exe <кол-во секторов> <кол-во потоков>

2.3 Выходные данные

В консоль выводится количество участков на острове, количество команд пиратов, ищущих сокровища, а так же номер участка на котором зарыто сокровище. Далее каждая команда отчитывается о начале работы, после чего начинается поиск. При нахождении сокровища команда, которая его нашла выводит номер участка, на котором оно было найдено и сообщает об этом Сильверу, чтобы тот мог отозвать остальные команды.

3. Тестовые примеры

Компиляция и запуск (см рис 1.1, 1.2)

```
C:\Users\vollfed\source\repos\avs_hw_4\avs_hw_4>c++ avs_hw_4.cpp -fopenmp -std=c++11
```

Рис 1.1. Компиляция. Пример 1

```
C:\Users\vollfed\source\repos\avs_hw_4\avs_hw_4>a.exe 1000 8
```

Рис 1.2. Запуск. Пример 2

Примеры работы на корректных данных (см. рис 2.1, 2.2)

```
number of sectors = 1000
number of teams = 8
treasure index = 562
Team 1 started searching
Team 2 started searching
Team 5 started searching
Team 3 started searching
Team 4 started searching
Team 6 started searching
Team 0 started searching
Team 7 started searching
Team 4 has found treasure in sector 562 and stopped searching
Team 6 stops searching for treasure because other team has already found it
Team 0 stops searching for treasure because other team has already found it
Team 7 stops searching for treasure because other team has already found it
Team 1 stops searching for treasure because other team has already found it
Team 2 stops searching for treasure because other team has already found it
Team 5 stops searching for treasure because other team has already found it
Team 3 stops searching for treasure because other team has already found it
```

Рис 2.1. Консольный вывод (корректные данные). Пример 1

```
C:\Users\vollfed\source\repos\avs_hw_4\avs_hw_4>a.exe 1234567 13
number of sectors = 1234567
number of teams = 13
treasure index = 693644
Team 1 started searching
Team 4 started searching
Team 2 started searching
Team 3 started searching
Team 5 started searching
Team 8 started searching
Team 9 started searching
Team 6 started searching
Team 7 started searching
Team 10 started searching
Team 0 started searching
Team 11 started searching
Team 12 started searching
Team 7 has found treasure in sector 693644 and stopped searching
Team 3 stops searching for treasure because other team has already found it
Team 8 stops searching for treasure because other team has already found it
Team 2 stops searching for treasure because other team has already found it
Team 0 stops searching for treasure because other team has already found it
Team 9 stops searching for treasure because other team has already found it
Team 11 stops searching for treasure because other team has already found it
Team 10 stops searching for treasure because other team has already found it
Team 1 stops searching for treasure because other team has already found it
Team 12 stops searching for treasure because other team has already found it
Team 5 stops searching for treasure because other team has already found it
Team 6 stops searching for treasure because other team has already found it
Team 4 stops searching for treasure because other team has already found it
```

Рис 2.2. Консольный вывод (корректные данные). Пример 2

Примеры работы при некорректных данных (см рис. 3.1, 3.2, 3.3)

```
C:\Users\vollfed\source\repos\avs_hw_4\avs_hw_4>a.exe 0 10
incorrect input. Number of sectors should be more than 0
```

Рис 3.1. Консольный вывод (некорректные данные). Пример 1

```
C:\Users\vollfed\source\repos\avs_hw_4\avs_hw_4>a.exe 10 0
incorrect input. Number of threads should be more than 0 and not more than number of sectors
```

Рис 3.2. Консольный вывод (некорректные данные). Пример 2

```
C:\Users\vollfed\source\repos\avs_hw_4\avs_hw_4>a.exe 10 11
incorrect input. Number of threads should be more than 0 and not more than number of sectors
```

Рис 3.3. Консольный вывод (некорректные данные). Пример 3

4. Список используемых источников

[1] Практические приемы построения многопоточных приложений. [Электронный ресурс]. // URL: http://softcraft.ru/edu/comparch/tasks/t03/ (Дата обращения: 16.11.2020, режим доступа: свободный)

[2] OpenMP. [Электронный ресурс]. // URL: https://www.openmp.org/(Дата обращения: 28.11.2020, режим доступа: свободный)

5. Текст программы

```
#include <iostream>
#include <thread>
#include <random>
#include <omp.h>
#include <vector>
#include <string>

// Вариант: 26
// Выполнил: Федченко Всеволод Александрович БПИ199

// Вторая задача об Острове Сокровищ.
// Шайка пиратов подпредводительством Джона Сильвера высадилась на берег Острова Сокровищ.
// Несмотря на добытую карту старого Флинта, местоположение сокровищ по - прежнему остается загадкой,
// поэтому искать клад приходится практически на ощупь. Так как Сильвер ходит на деревянной ноге,
// то самому бродить по джунглям ему не с руки. Джон Сильвер поделил остров на участки, а пиратов на
небольшие группы.
// Каждой группе поручается искать клад на нескольких участках, а сам Сильвер ждет на берегу.
// Группа пиратов, обшарив одну часть острова, переходит к другой, еще не обследованной части.
// Закончив поиски, пираты возвращаются к Сильверу и докладывают о результатах.
// Требуется создать многопоточное приложение с управляющим потоком, моделирующее действия Сильвера и
пиратов.
// При решении использовать парадигму портфеля задач.


// Программа компилируется командой "c++ avs_hw_4.cpp -fopenmp -std=c++11" (проверено только на Windows
10).
// Команда для запуска: a.exe <кол-во секторов> <кол-во потоков>


using namespace std;

//Кол-во участков.
int numOfSectors;
// Кол-во потоков.
int numOfThreads;

int treasureIndex = -1;

vector<bool> islandSectors;
volatile bool globalFlag = false;

void searchForTreasure(int numOfThreads)
{
	bool localFlag = false;

#pragma omp parallel private(localFlag) shared(globalFlag) num_threads(numOfThreads)
	{
#pragma omp critical
		{
			cout << "Team " << omp_get_thread_num() << " started searching " << endl;
		}

#pragma omp for
		for (int i = 0; i < numOfSectors; i++)
		{
			if (!localFlag)
			{
#pragma omp critical
				{
					if (globalFlag)
					{
						cout << "Team " << omp_get_thread_num() << " stops
searching for treasure because other team has already found it" << endl;
						localFlag = true;
					}

					if (islandSectors[i] && !globalFlag)
					{
						globalFlag = true;
						cout << "Team " << omp_get_thread_num() << " has found
treasure in sector " << i << " and stopped searching" << endl;
						localFlag = true;
```

```
                                        }
                                }
                        }
                }
        }
}

int main(int argc, char* argv[])
{
        numOfSectors = stoi(argv[1]);
        numOfThreads = stoi(argv[2]);

        // Обрабтка некорректного ввода.
        if (numOfSectors < 1)
        {
                cout << "incorrect input. Number of sectors should be more than 0";
                return -1;
        }
        if (numOfThreads < 1 || numOfThreads > numOfSectors)
        {
                cout << "incorrect input. Number of threads should be more than 0 and not more than
number of sectors";
                return -1;
        }

        islandSectors.reserve(numOfSectors);

        for (int i = 0; i < numOfSectors; i++)
        {
                islandSectors.push_back(0);
        }

        //Создание рандомайзера.
        std::random_device rd;
        std::mt19937 rng(rd());
        std::uniform_int_distribution<int> uni(0, numOfSectors);

        auto random_integer = uni(rng);

        // Рандомно задать сектор с сокровищем.
        treasureIndex = random_integer;
        islandSectors[treasureIndex] = true;
        cout << "number of sectors = " << numOfSectors << endl;
        cout << "number of teams = " << numOfThreads << endl;
        cout << "treasure index = " << treasureIndex << endl;


        searchForTreasure(numOfThreads);
}
```