Правительство Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет «Высшая школа экономики»

Факультет компьютерных наук

Департамент программной инженерии

Отчет к домашнему заданию По дисциплине «Архитектура вычислительных систем»

Работу выполнил:

Студент группы БПИ-196 Мосолков Е.Н.

Москва 2020

Содержание

1. ЗАДАЧА

16. *Задача об инвентаризации по рядам.* После нового года в библиотеке университета обнаружилась пропажа каталога. После поиска и наказания виноватых, ректор дал указание восстановить каталог силами студентов. Фонд библиотека представляет собой прямоугольное помещение, в котором находится M рядов по N шкафов по K книг в каждом шкафу. Требуется создать многопоточное приложение, составляющее каталог. При решении задачи использовать метод «портфель задач», причем в качестве отдельной задачи задается составление каталога одним студентом для одного ряда.

2. РЕШЕНИЕ

Способ входных данных – аргументы командной строки.

Формат аргументов командной строки:

1. Путь к файлу
2. Путь к входному тесту (в формате .txt)
3. Путь к выходному файлу (в формате .txt)
4. Количество потоков

Входной файл содержит 3 строки с числами – m, n, k

Выходной файл содержит информацию о книге с уникальным идентификатором,

Программа считывает входные данные и создает catalog (тип vector<string>) с размером m в котором хранятся все записи каталога в формате строк. При этом catalog – это глобальная переменная (что сделано для доступа из разных задач)

Способ распределения задач – «Портфель задач». Каждый поток получает задачу по номеру (глобальная переменная taskCount), выполняет задачу и добавляет результат в catalog, затем проверяет остались ли еще задачи, и если да, то выполняет следующую задачу.

Для ввода и вывода данных в файл используем библиотеку fstream

После формирования catalog, происходит вывод в консоль и в файл.

Используется библиотека OpenMP для многопоточного распределения задач на число потоков указанное в коммандной строке.

В консоль выводиться информация о задаче и потоке который ее выполняет

КОД ПРОГРАММЫ

```
#include <iostream>
#include <vector>
#include <string>
#include <fstream>
#include <thread>
#include <algorithm>


using namespace std;

int m, n, k, taskCount = 0;
int threadCount;

/**
 * Variable to set book ids
 * Increments after every book counted;
 */
int bookId = 100001;

vector<string> catalog;

vector<string> getArguments(ifstream &in);

/**
 * A task method that fills catalog
 */
void task() {
    int row_num;
    // write info about where current task is running
#pragma omp critical
    {
        row_num = taskCount++;
        cout << "task: " << row_num + 1 << " thread: " << this_thread::get_id() << endl;
    }
    while (row_num < m) {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < k; j++) {
                string line = "Book ID: " + to_string(bookId++) +
                              +"\n\trow: " + to_string(row_num + 1)
                              + "\n\tbook case: " + to_string(i + 1);
                // pushing info about book to catalog
#pragma omp critical
                {
                    catalog.push_back(line);
                }
            }
        }
#pragma omp critical
        {
            row_num = taskCount++;
            cout << "task: " << row_num + 1 << " thread: " << this_thread::get_id() << endl;
        }
    }
}

/**
 * This method reads a text file and returns a vector of m, n, k string values
 */
vector<string> getArguments(ifstream &in) {
    vector<string> args;
    string arg;
```

```
    while (!in.eof()){
        getline(in, arg);
        args.push_back(arg);
    }
    return args;
}

int main(int argc, char** argv) {
    // Check if there are invalid arguments in command prompt
    if (argc != 4) {
        cout << "Command prompt arguments are class path, input file path, "
                "output file path and count of threads\nThree and only four arguments
allowed";
        return -1;
    }
    ifstream in;
    ofstream out;
    in.open(argv[1]);
    threadCount = argv[3]
    vector<string> args = getArguments(in);
    in.close();
    m = stoi(args[0]);
    n = stoi(args[1]);
    k = stoi(args[2]);
    if (args.size() != 3) {
        cout << "Invalid number of arguments in test file";
        return -1;
    }
    if (m <= 0) {
        cout << "M can't be equals 0 or negative";
        return -1;
    }
    else if (n <= 0) {
        cout << "N can't be equals 0 or negative";
        return -1;
    }
    else if (k <= 0) {
        cout << "K can't be equals 0 or negative";
        return -1;
    }
    catalog.reserve(m * n * k);
    out.open(argv[2]);
    // run in 4 parallel threads all rows and write Book info to output file
#pragma omp parallel for num_threads(threadCount)
    for (int i = 0; i < threadCount; ++i) task();
    sort(catalog.begin(), catalog.end());
    out << "Books count: " << size(catalog) << endl;
    for (auto const &line: catalog) {
        out << line << endl;
    }
}
```

3. ТЕСТИРОВАНИЕ

3.1. Параметры командной строки

При вводе некорректного числа аргументов, программа завершается с кодом -1, и выводит сообщение в консоль о том, что количество аргументов командной строки недопустимо.

3.2. Корректные данные

При тестировании проверяем количество строк которые выводятся в ответ. Они корректны на всех тестах. Test1 проверяет работу программы в целом. Test2 проверяет ее работу с большим числом книг полок и рядов. Test3 проверяет программу с большими входными параметрами.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. YouTube канал «#SimpleCode» плейлист «Многопоточное программирование»: https://www.youtube.com/watch?v=NawpxG81RRk&list=PLQOaTSbfxUtAc_RpyDiWCHq0YTzLtVSD0
2. SoftCraft «Практические приемы построения многопоточных приложений»: http://www.softcraft.ru/edu/comparch/tasks/t03/
3. Парадигмы параллельного программирования: https://pro-prof.com/forums/topic/parallel-programming-paradigms