Домашнее задание для курса OTUS C++ Basic

Пишем многопоточный подсчет самых частых слов

Введение

Многопоточное программирование добавляет нам дополнительные возможности по ускорению наших программ на C++. В данном задании вам предстоит переписать однопоточную программу, которая подсчитывает самые частые слова в файлах так, чтобы она получила выигрыш от использования нескольких потоков.

Однопоточный вариант

Ниже приведен пример программы в однопоточном варианте. Полный исходный код вы найдете в архиве к ДЗ.

```
int main(int argc, char *argv[]) {
   if (argc < 2) {
       std::cerr << "Usage: topk_words [FILES...]\n";
       return EXIT_FAILURE;
   }

   auto start = std::chrono::high_resolution_clock::now();
   Counter freq_dict;
   for (int i = 1; i < argc; ++i) {
       std::ifstream input{argv[i]};
       if (!input.is_open()) {
           std::cerr << "Failed to open file " << argv[i] << '\n';
           return EXIT_FAILURE;
       }
       count_words(input, freq_dict);
   }

   print_topk(std::cout, freq_dict, TOPK);
   auto end = std::chrono::high_resolution_clock::now();
   auto elapsed_ms = std::chrono::duration_cast<std::chrono::milliseconds>(end - start);
   std::cout << "Elapsed time is " << elapsed_ms.count() << " ms\n";
}
```

Листинг 1. Функция main

Данная программа печатает 10 самых частых слов (константа задача в полном листинге программы), которые встретили во всех входных файлах и печатает время

своей работы в миллисекундах. При подсчете слов регистр не учитывается, различные знаки препинания не игнорируются, а являются частью слова (любая непробельная последовательность символов). Для тестирования вы можете использовать текст книги “Великий Гетсби” https://www.gutenberg.org/files/64317/64317-0.txt . 3 самых частых слова в этом тексте:

- the
- and
- a

Это вполне ожидаемый результат, поскольку в английском языке артикли употребляются очень часто. А союзы часто употребляются во множестве различных языков.
Программа принимает на вход произвольное число файлов. Чтобы имитировать такое поведение вы можете скопировать текст книги любое число раз и подавать их все на вход программе или использовать любые другие текстовые файлы.

Указания к решению

Время исполнения, которое печатает программа может варьироваться от запуска к запуску. Чтобы удостовериться, что у вас получилась программы быстрее, либо медленнее запустите ее 5 раз и возьмите минимальное время исполнения.

Не забывайте про потенциальные проблемы при работе в многопоточной среде. Используйте правильные примитивы синхронизации и возможности стандартной библиотеки, чтобы избежать ситуации гонок.

Подумайте над стратегией распараллеливания. Обычно существует несколько способов распараллелить программу. Например, можно обрабатывать каждый файл в отдельном потоке и использовать общий словарь для подсчета встречаемости слов. Но такой словарь нужно будет защищать с помощью мьютекса. Другой подход предполагает, что у каждого потока будет свой словарь, но потом нужно их слить воедино. Эти подходы могут отличаться по времени исполнения кода.