Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего
образования «Нижегородский государственный университет им. Н.И. Лобачевского»

Институт Информационных Технологий, Математики и Механики

Лабораторная работа

«Алгоритм Дейкстры поиска кратчайших путей в графе»

Выполнил:
студент группы 381506-3
Липатов И.Д.

Нижний Новгород

2018

Оглавление

Введение

Алгоритм Дейкстры — алгоритм на графах, находит кратчайшие пути от одной из вершин графа до всех остальных. Алгоритм работает только для графов без рёбер отрицательного веса. Алгоритм широко применяется в программировании и технологиях, например, его используют протоколы маршрутизации OSPF и IS-IS.

Постановка задачи

Разработать и реализовать программу для поиска кратчайшиз путей в графе с использованием алгоритма Дейкстры, используя стандарт OpenMP и библиотеку TBB.

Необходимо реализовать следующие модули:

1. Генератор тестов (по входным параметрам);
2. Последовательный метод решения задачи;
3. Параллельный метод с использованием OpenMP;
4. Параллельный метод с использованием TBB;
5. Checker.

Вспомогательные элементы

1. Checker

Проверяет корректность результата, сравнивая результаты работы последовательного алгоритма с результатами работы параллельных ревлизаций на основе тестов, которые создал генератор. Сравнения осуществляются по массиву кратчайших путей.

2. Generator

Генерирует матрицу смежности для графа по параметру число верших в графе.

Метод решения

Каждой вершине из V сопоставим метку — минимальное известное расстояние от этой вершины до a. Алгоритм работает пошагово — на каждом шаге он «посещает» одну вершину и пытается уменьшать метки. Работа алгоритма завершается, когда все вершины посещены.

Инициализация: метка самой вершины a полагается равной 0, метки остальных вершин — бесконечности. Это отражает то, что расстояния от a до других вершин пока неизвестны. Все вершины графа помечаются как непосещённые.

Шаг алгоритма: если все вершины посещены, алгоритм завершается. В противном случае, из ещё не посещённых вершин выбирается вершина u, имеющая минимальную метку. Мы рассматриваем всевозможные маршруты, в которых u является предпоследним пунктом. Вершины, в которые ведут рёбра из u, назовём соседями этой вершины. Для каждого соседа вершины u, кроме отмеченных как посещённые, рассмотрим новую длину пути, равную сумме значений текущей метки u и длины ребра, соединяющего u с этим соседом. Если полученное значение длины меньше значения метки соседа, заменим значение метки полученным значением длины. Рассмотрев всех соседей, пометим вершину u как посещённую и повторим шаг алгоритма.

Описание реализации последовательного алгоритма

В последовательной реализации для хранения чисел $d[i]$ используем массив типа int, а для хранения принадлежности элемента множеству U— массив массив типа bool.

В начале алгоритма расстояние для начальной вершины полагается равным нулю, а все остальные расстояния заполняются большим положительным числом (бóльшим максимального возможного пути в графе). Массив флагов заполняется нулями. Затем запускается основной цикл.

На каждом шаге цикла мы ищем вершину с минимальным расстоянием и флагом равным нулю. Затем мы устанавливаем в ней флаг в 1 и проверяем все соседние с ней вершины . Если в них (в) расстояние больше, чем сумма расстояния до текущей вершины и длины ребра, то уменьшаем его. Цикл завершается, когда флаги всех вершин становятся равны 1, либо когда у всех вершин с флагом 0 . Последний случай возможен тогда и только тогда, когда граф G несвязный.

Описание реализации параллельного алгоритма

Ключевые отличия от последовательной версии: цикл поиска минимальной непосещенной вершины разделен между потоками,каждый поток находит минимум на своем участе итераций цикла,при завершении цикла происходит редукция результатов работы потоков. Цикл обновления текущих кратчайших путей также разделен между потоками.

Результаты экспериментов по оценке масштабируемости

	1 поток,сек	2 потока,сек	4 потока,сек
Последовательный алгоритм	0.133	0.130	0.131
OpenMP	0.135	0.06	0.05
TBB	0.132	0.07	0.06

Результаты выполнения алгоритма для графа с 5000 вершин

	1 поток,сек	2 потока,сек	4 потока,сек
Последовательный алгоритм	1.56	1.51	1.51
OpenMP	1.63	0.89	0.72
TBB	1.76	0.97	0.84

Результаты выполнения алгоритма для графа с 20000 вершин

Для тестирования использовался процессор i7-4610M с 2 ядрами и 4 потоками,а также отключенной технологией Turbo Boost.

Вывод

Из полученных результатов можно сказать, что в рамках данной задачи использование параллелизма рационально.