МИНОБРНАУКИ РОССИИ

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №3**

**по дисциплине «Построение и Анализ алгоритмов»**

Тема: Задача Коммивояжёра

|  |  |  |
| --- | --- | --- |
| Студент гр. 1384 |  | Соломин Д.А. |
| Преподаватель |  | Шевелева А.М. |

Санкт-Петербург

2023

**Задание.**

Дана карта городов в виде ассиметричного, неполного графа , где – это вершины графа, соответствующие городам; – это ребра между вершинами графа, соответствующие путям сообщения между этими городами.

Каждому ребру (переезд из города i в город j) можно сопоставить критерий выгодности маршрута (вес ребра) равный (натуральное число , если

Если маршрут включает в себя ребро , то , иначе .

Требуется найти минимальный маршрут (минимальный гамильтонов цикл):

**Цель и задачи работы.**

Определите проблему TSP и ее важность, а также познакомьтесь с концепцией динамического программирования.

Обзор алгоритма Хелда-Карпа: опишите ключевые идеи и этапы алгоритма, включая инициализацию матрицы C, вычисление подзадач и окончательное решение.

Примеры и приложения: приведите примеры того, как алгоритм Хелда-Карпа можно использовать для решения реальных задач, таких как маршрутизация в логистике и проектирование печатных плат.

Анализ временной и пространственной сложности: проанализируйте временную сложность и сложность по памяти алгоритма Хелда-Карпа и сравните его с другими алгоритмами, решающими TSP.

Реализация. Предоставьте пошаговое руководство по реализации алгоритма Хелда-Карпа на таком языке программирования и обсудите передовой опыт оптимизации производительности алгоритма.

**Введение.**

Задача коммивояжера (TSP) — это классическая задача оптимизации в компьютерных науках, которая включает в себя поиск кратчайшего возможного маршрута через набор городов, который посещает каждый город ровно один раз и возвращается в начальный город. Алгоритм Хелда-Карпа — это алгоритм динамического программирования, который может решить TSP за время , где n — количество городов. Однако сложность памяти алгоритма также составляет , что может быть ограничением для более крупных задач.

Динамическое программирование — это метод, используемый для решения сложных задач путем их разбиения на более мелкие подзадачи и решения каждой подзадачи только один раз. Алгоритм Хелда-Карпа использует динамическое программирование для решения TSP путем вычисления оптимального тура для подмножества городов и использования этой информации для создания оптимального тура для всего набора городов. Алгоритм основан на уравнении Беллмана-Хелда-Карпа, которое выражает оптимальную длину тура как рекурсивную функцию меньших подзадач.

Хотя алгоритм Хелда-Карпа может найти точное решение для TSP, его сложность во времени и памяти может быть недостатком для больших наборов данных (к примеру для вычисления оптимального маршрута для полного графа 35х35 потребуется около 4,375 терабайт оперативной памяти и около 24 часов). Тем не менее, алгоритм остается ценным инструментом для решения задач оптимизации и имеет множество приложений в таких областях, как логистика, инженерия и информатика.

**Решение.**

Алгоритм Хелда-Карпа — это подход к динамическому программированию для решения задачи коммивояжёра (TSP), целью которого является поиск кратчайшего возможного маршрута, проходящего через набор заданных городов и возвращающегося в начальный город.

TSP является NP-сложной задачей, что означает, что не существует известного эффективного алгоритма, который мог бы решить ее оптимально для всех возможных входных данных за полиномиальное время. Однако алгоритм Хелда-Карпа может найти оптимальное решение для небольших экземпляров проблемы, используя перекрывающуюся подструктуру проблемы и оптимальные свойства подструктуры.

Вот основные шаги алгоритма Хелда-Карпа:

1. Создайте матрицу A, где A[S][i] представляет собой длину кратчайшего пути, который проходит через все города множества S ровно один раз и заканчивается в городе i.
2. Инициализируйте A[{1}][1...n] для расстояний между городом 1 и каждым из других городов.
3. Запустите циклы:

Для каждого k от 2 до n:

Для каждого подмножества S размера k, содержащего город 1:

Для каждого i в S:

*If i is not 1, compute as the minimum of over all j in S where j is not i.*

1. Вычислите окончательное решение как для всех i в где представляет собой расстояние между городами i и j.

**Пример.**

Вот пример того, как использовать алгоритм Хелда-Карпа для поиска оптимального тура для набора из 4 городов:

Предположим, что набор городов равен {1, 2, 3, 4}, и его матрица расстояний:

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10 | 15 | 20 |
| 10 |  | 35 | 25 |
| 15 | 35 |  | 30 |
| 20 | 25 | 30 |  |

1. Инициализируем C

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| {1} | 0 | 10 | 15 | 20 |

1. Посчитаем C &

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| {1} |  | 10 | 15 | 20 |
| {1, 2} |  |  | 45 | 35 |
| {1, 3} |  | 45 |  | 45 |

1. Посчитаем

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| {1} |  | 10 | 15 | 20 |
| {1, 2} |  |  | 45 | 35 |
| {1, 3} |  | 45 |  | 45 |
| {1, 4} |  | 35 | 45 |  |
| {1,2,3} |  | 55 | 10 | 60 |
| {1,2,4} |  | 45 | 60 | 55 |
| {1,3,4} |  | 60 | 55 | 50 |

1. Вычислите окончательное решение, которое равно 80.

Следовательно, оптимальный тур для этого набора городов с общим расстоянием 80.

**Тесты.**

Алгоритм работает достаточно быстро и считает полный граф 20х20 за 1.8сек в среднем. Запустим несколько тестов для проверки (также есть в исходном кода). Все тесты верные. Программы работает корректно.

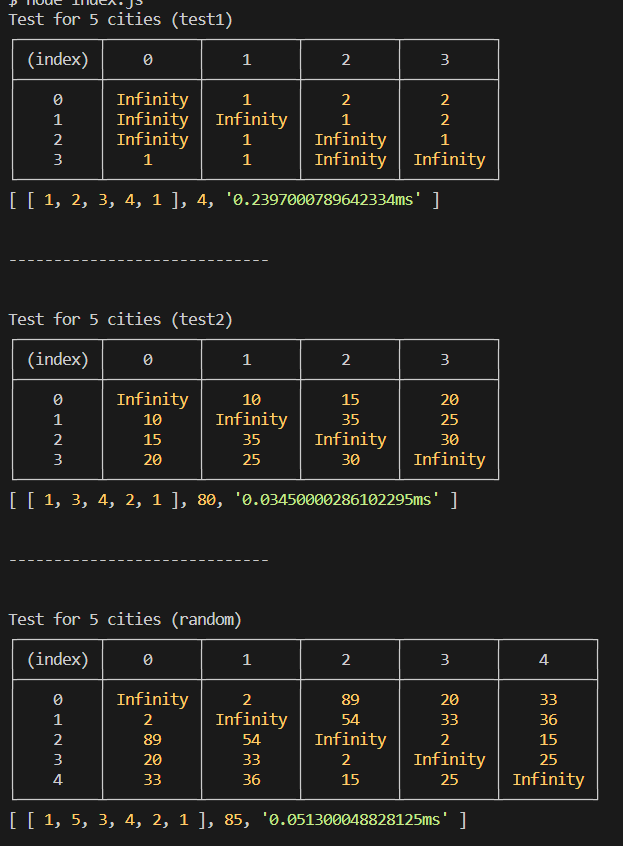


Рисунок 1 – Матрицы 5х5.

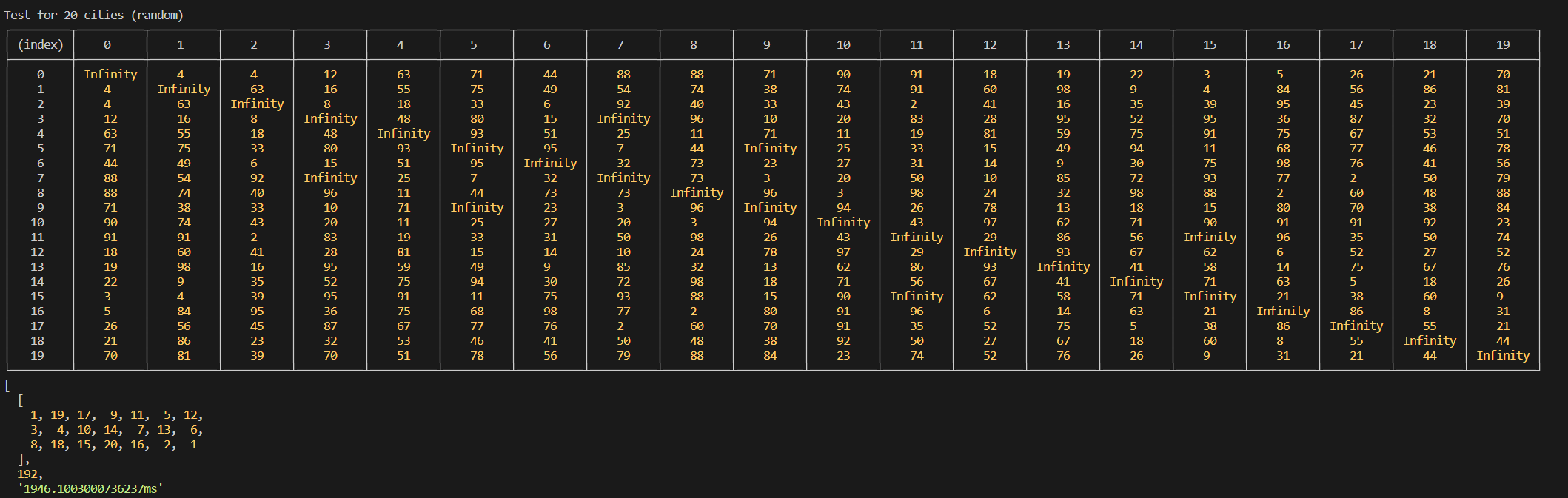


Рисунок 2 – Матрица 20х20

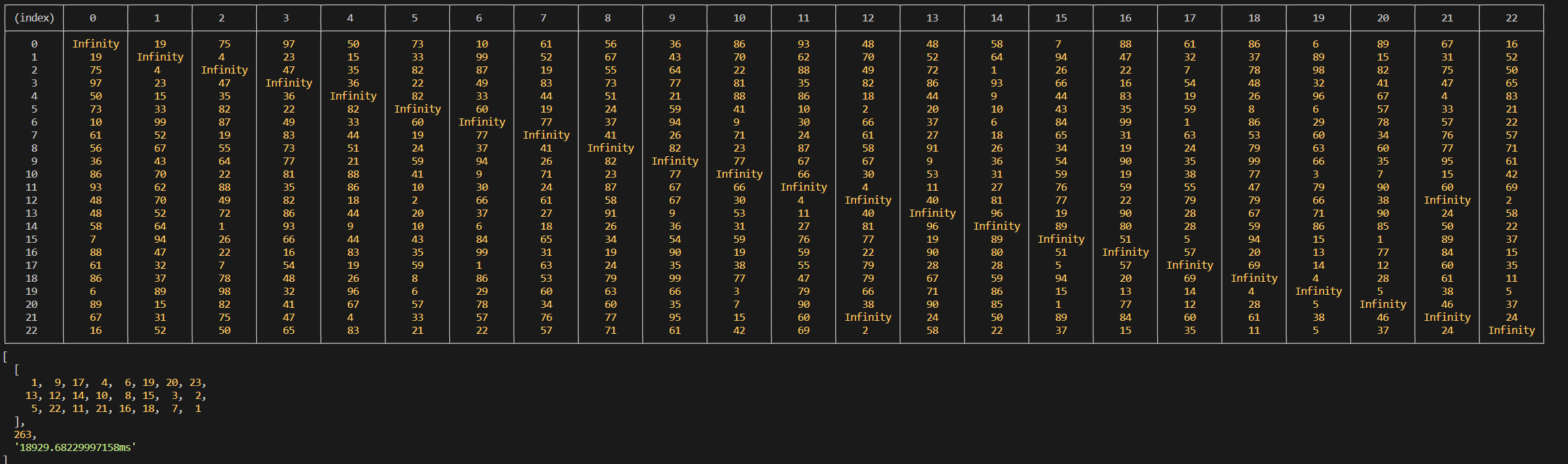


Рисунок 3 – Матрица 23х23

**Выводы.**

В заключение отметим, что алгоритм Хелда-Карпа является мощным инструментом для решения задачи коммивояжёра (TSP), которая включает в себя поиск кратчайшего маршрута через набор городов, который посещает каждый город ровно один раз и возвращается в начальный город. Алгоритм основан на динамическом программировании, которое включает разбиение задачи на более мелкие подзадачи и решение каждой подзадачи только один раз.

Одним из ключевых преимуществ алгоритма Хельда-Карпа является то, что он может найти **точное** решение TSP, в то время как многие другие алгоритмы могут дать только приблизительные решения. Это делает алгоритм особенно полезным в приложениях, где важно точное решение, например, в логистике и управлении цепочками поставок, где его можно использовать для оптимизации маршрутов доставки и минимизации времени и затрат в пути.

Однако алгоритм Хелда-Карпа имеет некоторые ограничения с точки зрения временной сложности и сложности по памяти. Алгоритм имеет плохую временную сложность , что может быть чрезмерно медленным для больших наборов данных. Кроме того, алгоритму требуется большой объем памяти для хранения матрицы C, что также может быть проблемой для более крупных задач.

Несмотря на эти ограничения, алгоритм Хелда-Карпа остается ценным инструментом для решения TSP и имеет множество приложений в самых разных областях. Понимая ключевые идеи и этапы алгоритма, исследователи и практики могут применять его для решения сложных проблем и повышения своей эффективности и производительности. Кроме того, текущие исследования изучают способы оптимизации производительности алгоритма и уменьшения его временной и пространственной сложности, что может еще больше расширить его применение и влияние.