Петросян Акоб Аркадиевич
гр. Б01-814

Задача № 12

Travelling salesman.

Задача является задачей коммивояжёра

Так как между городами определены расстояния и выполняется нерав. треугольника, то наша задача коммивояжёра является метрической.

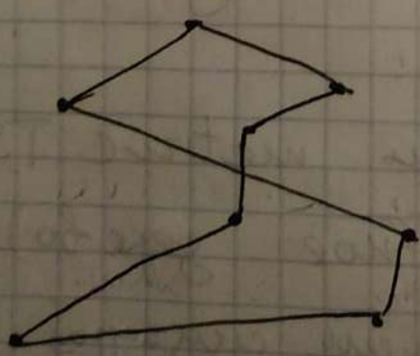
Ниже алгоритмы для нахождения оптимального маршрута:

Мы рассматриваем metric TSP → (минимальное остовное дерево), т.е остовное дерево [в нашем случае] неориентированного графа, имеющее минимальный возможный вес, где под весом понимается «длина» ребра.

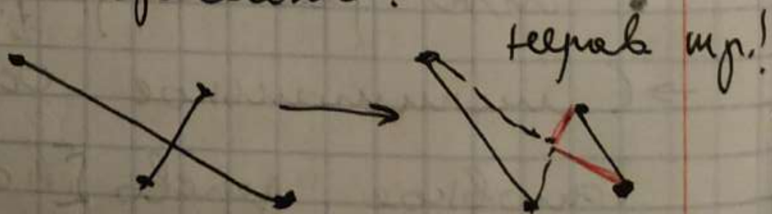
Алгоритмы:
- Nearest Neighbour
- Nearest Insertion
- Shortcuting (Christophides)

1) Nearest Neighbour $\frac{w(NN)}{w(OPT)} = O(\log n)$

Мы начинаем с точки N_0 и соединяем её ребром с ближайшей точкой N_x. Потом соединяем N_x с её ближайшей из оставшихся $\{N\}/N_1, N_x$ точек. Продолжаем до тех пор, пока все точки будут соед..



Пример неоптимальности. Есть пересечение!



Недостаток: т.к. алгоритм NN жадный, то у нас с очевидностью можем получится цикл неоптимальный.

Оптимизация:

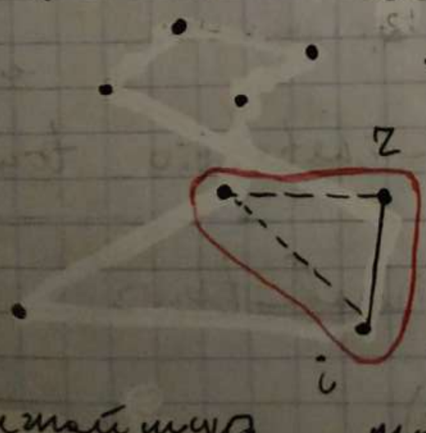
• замена пересекающихся отрезков на не пересек.

• [в общем случае] запускаем алгоритм несколько раз в разных начальных точках и выбираем условно оптимальный.

Недостаток: долгое выполнение.

Вывод: Этот алгоритм не подходит!

2) Nearest Insertion



2.1) выбираем кратчайшее ребро с одной вершиной в т. N_0

2.2) потом выбираем ближайшую точку к уже построенным двум.

2.3) (Шаг выбора) Для данного sub-tour найдём узел r, который не находится в sub-tour-е, ближайший к какому-либо узлу j в sub-tour, т.е с минимальным c_{rj}

2.4) (Шаг вставки)

Находим дугу (i, j) в sub-tour, который минимизирует $c_{ir} + c_{rj} - c_{ij}$. Вставим r между i и j.

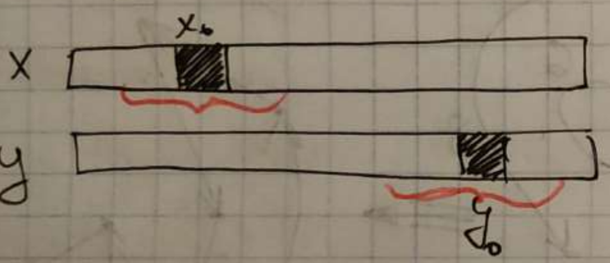
2.5) Если все узлы добавлены в sub-tour (цикл) - остановиться, иначе перейти к шагу 3.

$$\frac{\text{length\_of\_nearest-instertio\_tour}}{\text{length\_of\_optimal\_tour}} \le 2$$

Сложность: $O(n^2)$.

Замечание: ближайшие точки можно выбрать такими способами

- quad tree.
- деление координат x и y



– массива коорд. всех вершин

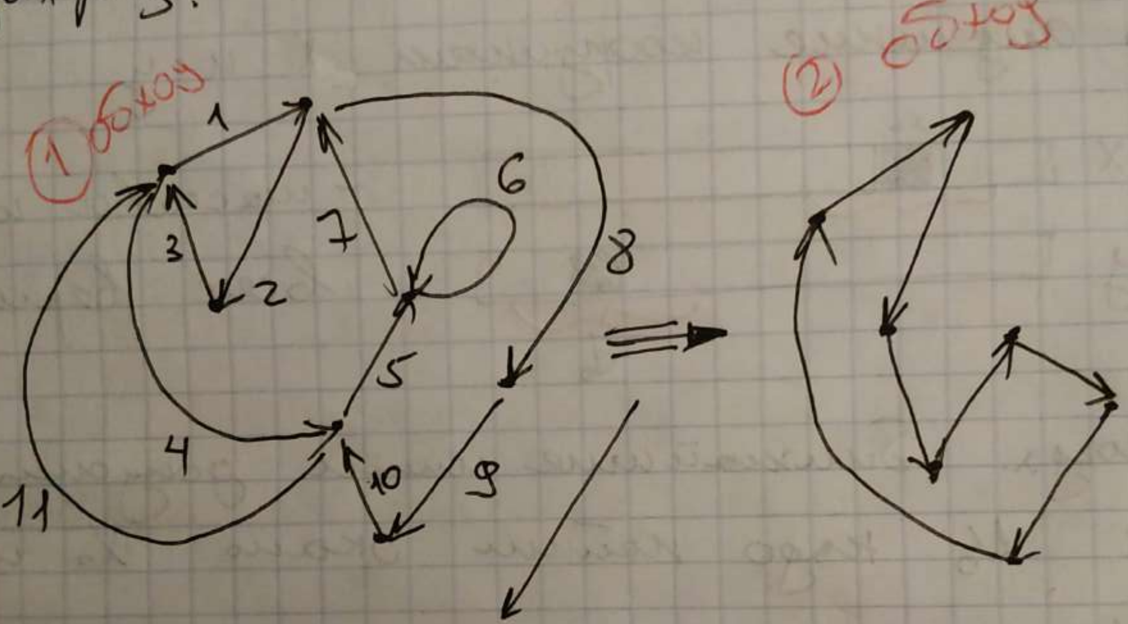
тогда ближайшие точки допустим к N_0 надо найти около x_0 и y_0

Итог: $\dfrac{w(NI)}{w(OPT)} \le 2$

Теоретический NI лучше NN, но на практике нельзя сказать что NI выиграет на 100%.

3) Последний алгоритм Shortcutting

Допустим у нас есть какой-то обход нашего графа, который посещает каждую верш. хотя бы один раз.



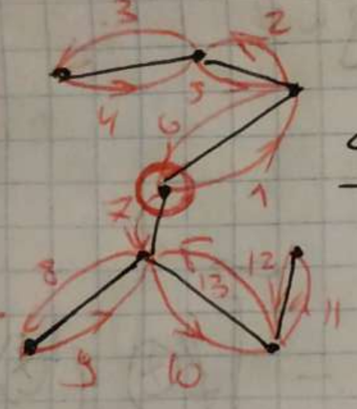
строим гамильтовый цикл
[спрямление (нерав. треугольника)]

$w(1) \geq w(2)$ (из з. нерав. треугол)

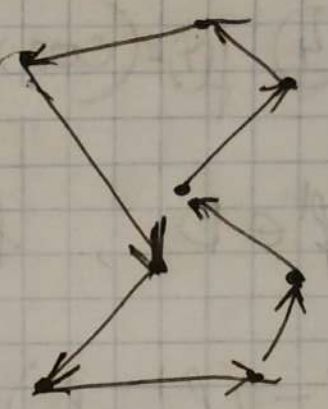
Оценивание:

задача кратчайшего остовного дерев.

3



спрямление



4

MST
(Алгоритм
Прима)

гамильт-ый
цикл.

Обход в глубину

длина цикла (3) не больше чем длина исходного обхода, а исходный обход это удвоенная длина мин. ост. дер. MST

(*) $w(4) \leq 2 \cdot w(MST) = w(\text{исх обхода})$

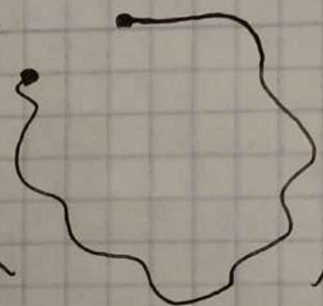
Учитывая, то что:

$$w(MST) \leq w(OPT\ TSP) \quad (**)$$

Доказательство

□

Если взять и удалить ребро $\forall$ гамильтонова цикла получим гамильт. цепь что является остовным деревом.

$\omega(MST) \le \omega\Big($  $\Big)$

Min ST

ST

■

Зам.: Чтобы алгоритм был хорошим нам нужны и нижние и верхние оценки.

Учитывая (*) и (**) получаем

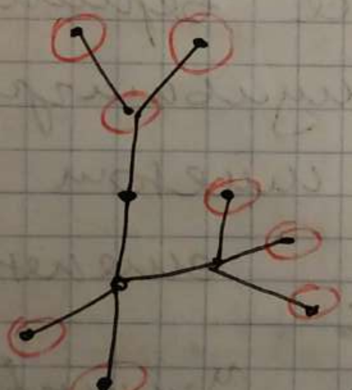
Итог: $\dfrac{\omega(4)}{\omega(OPTTSP)} \le 2 = k.$

На пути стремления можно улучшить оценку до $\frac{3}{2} = k$. Это делается с помощью алгоритма Christofides.

Оптимизация:

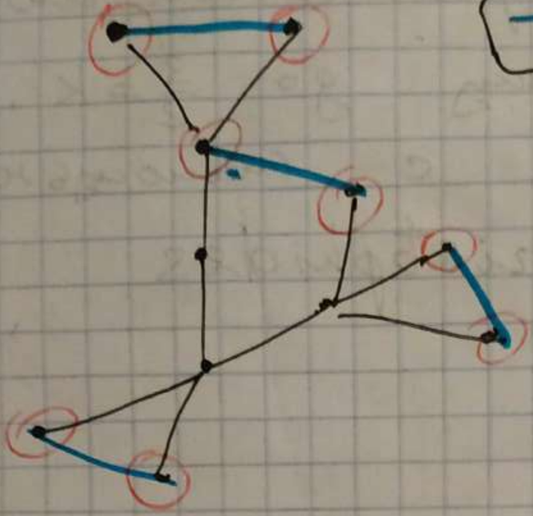
<u>Шаг 1</u>

строим MST, отмечаем вершины нечётной степени.



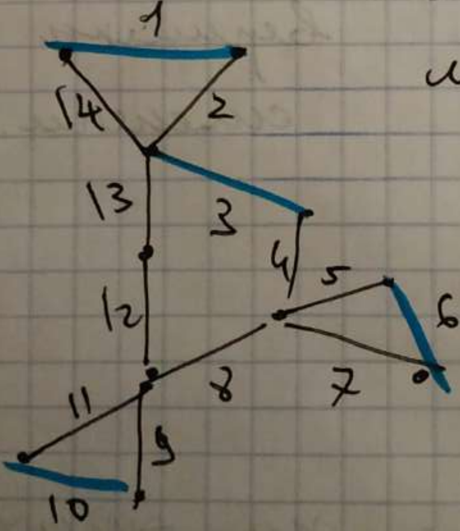
<u>Шаг 2</u> дальше на этих вершинах строим совершенное паросочетание минимального веса.

— паросочетание



количество вершин нечётной степени всегда чётно -> это можно сделать всегда!

Шаг 3



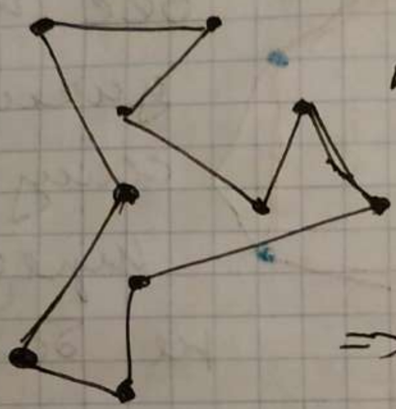
все вершины мультиграфа имеют чётную степень. (*#)

Учитывая (*#) и то что мультиграф связный можем сказать что это эйлеров цикл.
Дальше нумерация.
Мы построим эйлеров обход в мультиграфе.

Шаг 4 спрямляем эйл. обход

цикл получен. из. объед. MST и паросочет. =>

S



паросочет.

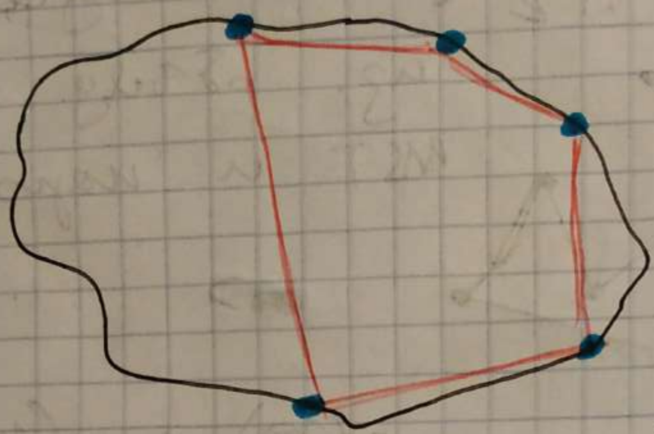
$$\Rightarrow w(S) \le w(MST) + w(M)$$

$$\begin{array}{ccc} & \wedge & \wedge \\ (\#\#) & w(OPT\ TSP) & \frac{1}{2} w(OPT\ TSP) \end{array}$$

Доказать!

Доказательство:

Допустим, что у нас в графе был какой-то оптимальный гамильт. цикл. Мы паросочет. строим для выбранных вершин. Первое что мы делаем мы сравняем наш OPT так цикл отн. выбр. вершин.



вес красного цикл, в силу нерав треугольник, не больше веса цикла. Но цикл можно разбить на два паросочетания (допустим вершины с чётными номерами и вершины с нечётными номерами).

Мы доказали что для обычного паросочет. его вес меньше $\leq \frac{1}{2} \cdot w(\text{TSPOPT}) \Rightarrow$ для мин. паросоч. $\hookrightarrow$

$\hookrightarrow w(M) \leq \frac{1}{2} w(\text{OPTTSP}) \Rightarrow$

$\Rightarrow$ ## - док. $w(S) \leq \frac{3}{2} w(\text{OPTTSP})$

Недостаток: долгое исполнение, но самый оптимальный алгоритм из трёх. Можно ещё сильнее оптимизировать выбрав оптимальный алгоритм для выборки паросочетаний.

Вывод: алгоритм Christofides наиболее подходящий из всех перечисленных алгоритмов