Задачата за търговския пътник

Анализ и сравнение на алгоритми за решаване на проблема

1. Представяне на проблема

Търговски пътник трябва да посети списък от градове. Ако е известно разстоянието между всяка двойка градове, кой е най-късият път, който посещава всеки град точно по веднъж и се връща в града, от който е започнало пътуването?

Картата на градовете се представя чрез граф, където ребрата са пътищата, теглата им са разстоянието между двата края на реброто, а върховете са самите градове.

2. Алгоритми

2.1. Метод на пълното изчерпване (Brute force)

При използването на този метод всички елементи от множеството на върховете на графа се комбинират по всеки възможен начин, за да се намери търсеният резултат. Изходът е оптимален, но времето за изпълнение и използваната памет са много големи.

Най-често използвани са алгоритмите за търсене в дълбочина и в ширина.

2.2. Евристични методи

Евристиката се ползва в алгоритми, за които се очаква да са по-бързи, макар и не с най-оптималното решение. На локално ниво (при по-ограничен мащаб от данни) решението може да бъде максимално оптимално.

За всеки алгоритъм ще имаме стартова позиция.

2.2.1. Най-близък съсед (Nearest Neighbour)

1) Стартовата позиция е сегашната.
2) Ако няма повече върхове, през които не сме минали, добавяме теглото на реброто от последния посетен връх до стартовата позиция.
3) Намираме най-близкия съсед на сегашната точка (върха с най-малко тегло на реброто му със сегашната позиция).
4) Добавяме теглото към общото разстояние.
5) Отбелязваме най-близкия съсед за посетен.
6) Сегашната позиция е най-близкият съсед.
7) Връщаме се към стъпка 2).

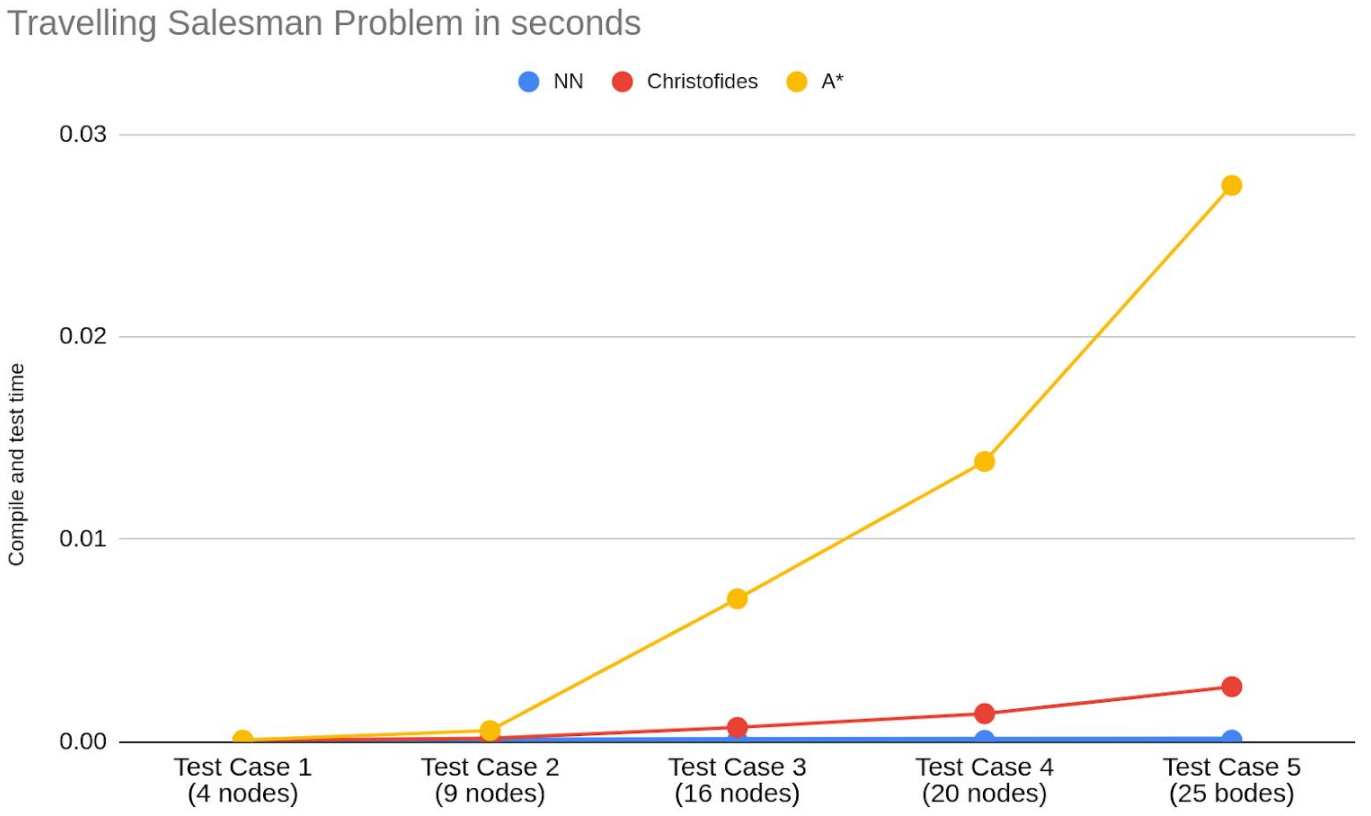
2.2.2. Алгоритъм на Кристофидис

1) Намираме минималното покриващо дърво на графа (алгоритъм на Прим).
2) Правим минимално съчетаване(MWM/perfect matching) на върховете с нечетна степен (върховете, които имат нечетен брой ребра).
3) Добавяме съчетаването към покриващото дърво
4) Взимаме Ойлеровия цикъл на получилия се граф.
5) Правим преки пътища, ако има такива.

2.2.3. Алгоритъмът A*

1) Слагане на стартовата позиция към опашка с приоритет.
2) Ако опашката е празна, търсенето се проваля. Изход.
3) Взимаме първия елемент на опашката и пресмятаме стойността на всяко негово дете. Стойността е дължината на досега изминатия път и евристиката. За евристика взимаме дължината на минималното покриващо дърво на непосетените градове.
4) Добавяме децата в опашката и слагаме сегашната позиция в пътя.
5) Ако сме минали през всички градове, добавяме стартовата позиция в края на пътя и пресмятаме неговата дължина. Изход.
6) Връщаме се към стъпка 2).

3. Сравнение



https://docs.google.com/spreadsheets/d/1QpDT-bQBLSQlDEwIDD5UPeLpJJWTQdNKx2sPDMmclgY/edit?usp=sharing - Линк към таблицата със стойности на време за изпълнение