

Федеральное государственное автономное образовательное
учреждение высшего образования
«Национальный исследовательский университет
ИТМО»

Системы искусственного интеллекта

Лабораторная работа 2

Вариант 8

Выполнил: Ву Минь Хиеу

Группа: Р33201

Санкт-Петербург

2022г.

1. Задание

Исследование алгоритмов решения задач методом поиска.

Описание предметной области. Имеется транспортная сеть, связывающая города СНГ. Сеть представлена в виде таблицы связей между городами. Связи являются двусторонними, т. е. допускают движение в обоих направлениях. Необходимо проложить маршрут из одной заданной точки в другую.

2. Таблица связей между городами

Город 1	Город 2	Расстояние, км
Вильнюс	Брест	531
Витебск	Брест	638
Витебск	Вильюс	360
Воронеж	Витебск	869
Воронеж	Волгоград	581
Волгоград	Витебск	1455
Витебск	Ниж.Новгород	911
Вильнюс	Даугавпилс	211
Калининград	Брест	699
Калиниград	Вильнюс	333
Каунас	Вильнюс	102
Киев	Вильнюс	734
Киев	Житомир	131
Житомир	Донецк	863

Житомир	Волгоград	1493
Кишинев	Киев	467
Кишинев	Донецк	812
С.Петербург	Витебск	602
С.Петербург	Калининград	739
С.Петербург	Рига	641
Москва	Казань	815
Москва	Ниж.Новгород	411
Москва	Минск	690
Москва	Донецк	1084
Москва	С.Петербург	664
Мурманск	С.Петербург	1412
Мурманск	Минск	2238
Орел	Витебск	522
Орел	Донецк	709
Орел	Москва	368
Одесса	Киев	487
Рига	Каунас	267
Таллинн	Рига	308
Харьков	Киев	471
Харьков	Симферополь	639
Ярославль	Воронеж	739
Ярославль	Минск	940
Уфа	Казань	525
Уфа	Самара	461

Вариант 8: | 8 | Вильнюс | Одесса |

Ссылка к репозитории : https://github.com/hieuminhvu/AI_System_ITMO/tree/master/Lab2

3. Неинформированный поиск

3.1) Поиск в ширину

Одесса
Киев
Вильнюс

3.2) Поиск в глубину

Одесса
Киев
Житомир
Волгоград
Витебск
Брест
Калининград
С.Петербург
Рига
Каунас
Вильнюс

3.3) Поиск с ограничением глубины

С Лимит = 2

Вильнюс
Киев
Одесса

С Лимит = 1

No path

3.4) Поиск с итеративным углублением

Вильнюс
Киев
Одесса

3.5) Двухнаправленный поиск

Точка пересечения: Вильнюс

BDS
Киев
Вильнюс
Киев
Одесса

3.6) Вывод:

Алгоритм поиска в ширину может дать оптимальные результаты, но требует большого объема памяти, поскольку он должен помнить все вершины. Хотя поиск в глубину не требует большой памяти, он может привести к ложным результатам (в моем случае результат все же называется оптимальным). Поиск с ограниченной глубиной требует дополнительного неполного условия, но может преодолеть слабость обычного поиска по глубине, заключающуюся в том, что трудно обнаружить тупики. Итеративный поиск в глубину является оптимальным решением при условии, что пространство поиска в глубину достаточно велико, а глубина неизвестна. Двусторонний поиск требует много памяти, но может сократить время поиска вдвое, что не является оптимальным

методом.

4. Информированный поиск. Воспользовавшись информацией о протяженности связей от текущего узла, выполнить:

4.1) Жадный поиск по первому наилучшему соответствию;

```
Одесса
Киев
Вильнюс
1221
```

Heuristics table

```
Житомир 618
Каунас 1323
Волгоград 5845
Рига 1590
С.Петербург 2183
Москва 2471
Вильнюс 1221
Даугавпилс 1432
Таллинн 1898
Калининград 1554
Киев 487
Одесса 0
Витебск 1581
Минск 3161
Воронеж 2450
Кишинев 954
Уфа 3811
Самара 4272
Казань 3286
Ярославль 3189
Орел 2103
Брест 1752
Харьков 958
Ниж.Новгород 3306
Мурманск 3595
Донецк 2812
Симферополь 1597
```

4.2) Затем, используя информацию о расстоянии до цели по прямой от каждого узла, выполнить поиск методом минимизации суммарной оценки A^* .

```
Одесса
Киев
Вильнюс
1221
```

