Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
дисциплины
«Искусственный интеллект в профессиональной деятельности»
Вариант 13

Выполнил:
Мотовилов Вадим Борисович
3 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Проверил:
Воронкин Р.А.
Ассистент департамента цифровых, робототехнических систем и электроники

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: исследование методов поиска в пространстве состояния

Цель: приобретение навыков работы по исследованию методов поиска в пространстве состояния

Порядок выполнения работы

Задание.

Необходимо воспользоваться сервисом Google Maps или аналогичными картографическими приложениями, чтобы выбрать на карте более 20 населённых пунктов, связанных между собой дорогами. Постройте граф, где узлы будут представлять населённые пункты, а рёбра — дороги, соединяющие их. Вес каждого ребра соответствует расстоянию между этими пунктами. Выбрать начальный и конечный пункты на графе. Определить минимальный маршрут между ними, который должен проходить через три промежуточных населённых пункта. Показать данный путь на построенном графе.

На картах Google Maps была выбрана страна Нигер на континенте Африка. В этой стране были определены точки населённых пунктов, которые представляют узлы, а рёбра (дороги) между ними их расстояния.

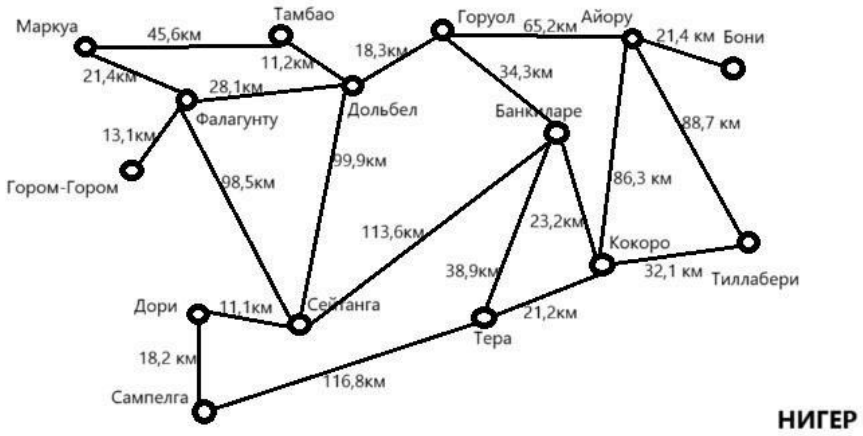

Рисунок 1. Карта населённых пунктов

Далее были определены точки: начальная – Тамбао и помеченная красным маркером, и конечная – Кокоро, помеченная фиолетовым маркером.

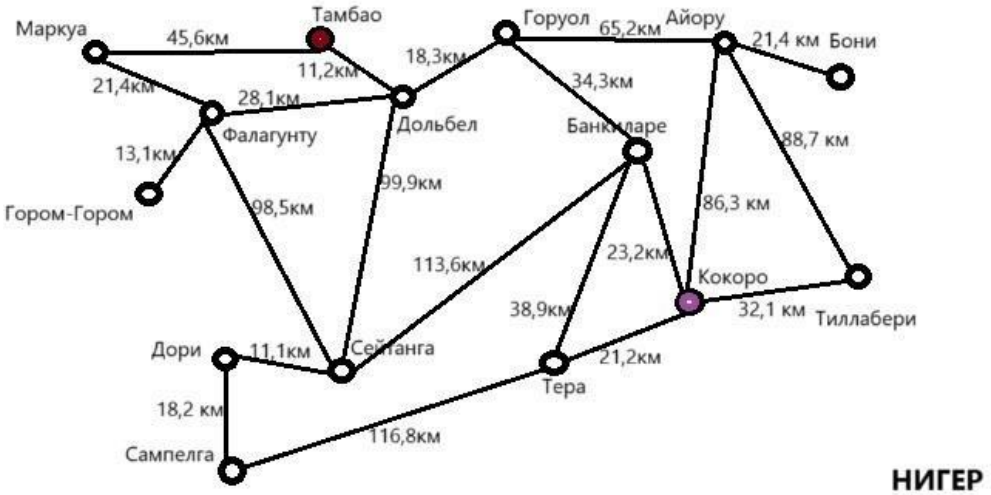

Рисунок 2. Помеченные узлы

После были проложены пути от начальной до конечной точки, которые являются возможными минимальными:

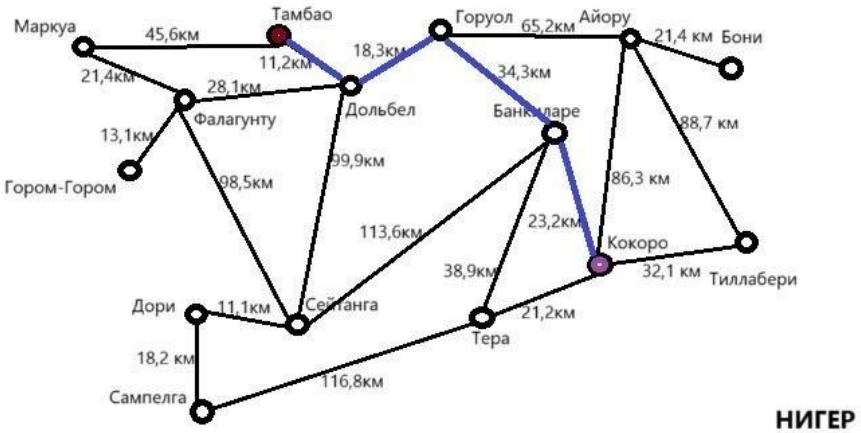

Рисунок 3. Первый маршрут

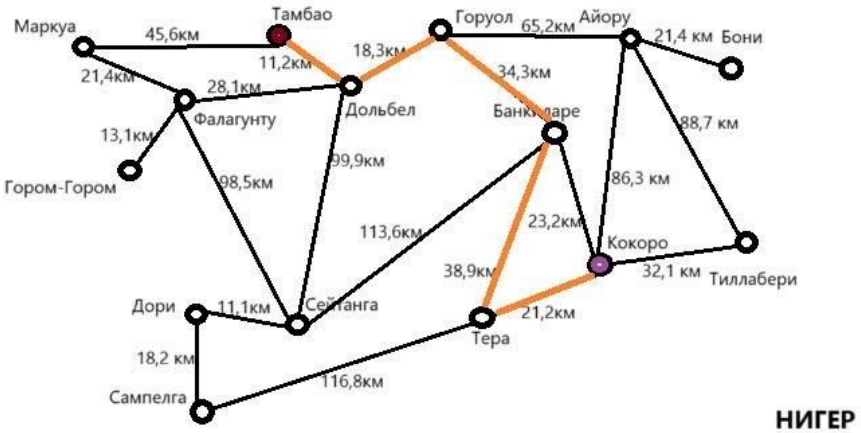

Рисунок 4. Второй маршрут

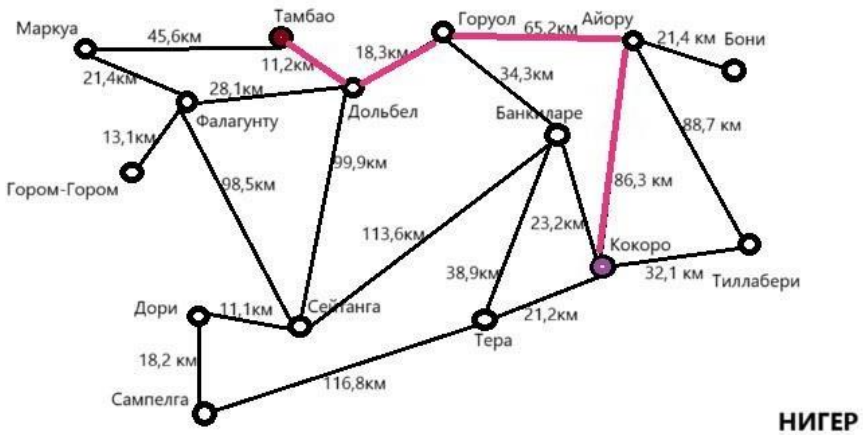

Рисунок 5. Третий маршрут

- Сумма рёбер первого маршрута составила 87 км;

- Сумма рёбер второго маршрута составила 123,9 км;

- Сумма рёбер третьего маршрута составила 181 км.

Задача Коммивояжора.

Для поиска оптимального маршрута может так же быть использовано дерево поиска, которое включает в себя листовые узлы, преемников, дочерние узлы.

Рисунок 6. Дерево поиска с корневым узлом

На рисунке представлено дерево поиска оптимального пути. Оно содержит в себе:

- Корневой узел Тамбао

После расширения дерева оно будет иметь уже следующий вид:

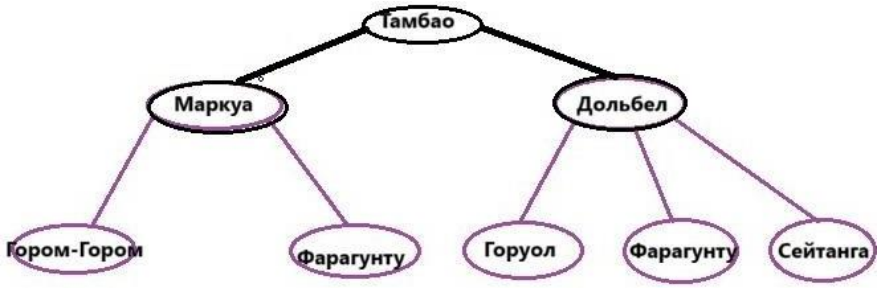

Рисунок 7. Дерево поиска с листовыми узлами

На рисунке 7 видно, что дерево приобрело новый вид после назначения листовых узлов. «Ветки» и листовые узлы окрасились в чёрный цвет, закрепив за собой узел.

Таким образом, поиск по дереву поиска продолжается до тех пор, пока не будет достигнута цель. В данном случае – это город Кокоро. Но не всегда дерево может прийти только к одной цели. Основная задача дерева – это проложить каждый представленный маршрут, даже если он заканчивается не на цели. Так и получается полное дерево путей.

Суть задачи Коммивояжёра является найти самые кратчайшие пути. Это представлено на рис. 8, где пути не продолжаются после того, как был найден кратчайший путь. В данном случае этих путей оказалось 8:

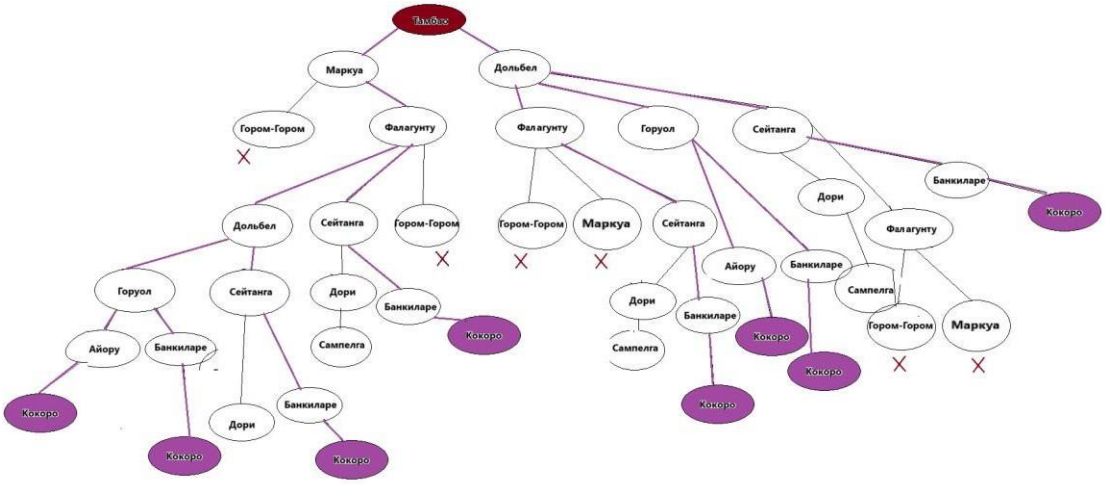

Рисунок 8. Задача Коммивояжёра

Пути были отмечены на карте:

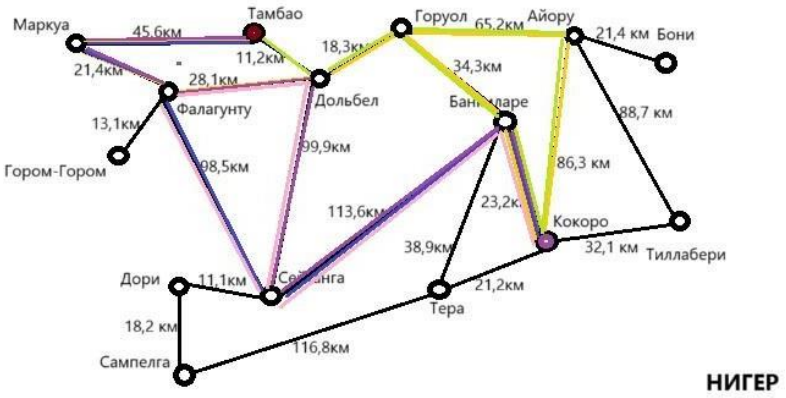

Рисунок 9. Отметка на карте возможных путей

Контрольные вопросы

1 Что представляет собой метод "слепого поиска" в искусственном интеллекте?

Этот метод можно представить как попытку найти выход из затемненного лабиринта, ощупывая каждую стену, каждый угол, без заранее известного плана или карты. Он исследует пространство возможных решений методом проб и ошибок, не обладая информацией о том, насколько близко каждое принятое решение к финальной цели. Такой подход, хотя и элементарен, является основой для разработки более сложных алгоритмов, включая эвристический поиск.

2 Как отличается эвристический поиск от слепого поиска?

Эвристический поиск, в отличие от слепого, использует дополнительные знания или "эвристики" для направления процесса поиска, подобно тому, как путешественник использует компас в неизведанных землях.

3 Какую роль играет эвристика в процессе поиска?

Благодаря эвристическому алгоритму поиска может быть найден наиболее оптимальный путь к цели, основываясь на заранее заданных критериях и предположениях.

4 Приведите пример применения эвристического поиска в реальной задаче.

Эвристический поиск можно сравнить с методом применения заранее полученных знаний для решения, например, дискриминанта. Не зная

изначально что это за уравнение и методы его решения, это будет скорее слепой поиск.

5 Почему полное исследование всех возможных ходов в шахматах затруднительно для ИИ?

Существует большая вариативная разброска ходов в шахматах и их влияний на противника, из-за чего даже ИИ не может предвидеть всего.

6 Какие факторы ограничивают создание идеального шахматного ИИ?

Время и память.

7 В чем заключается основная задача искусственного интеллекта при выборе ходов в шахматах?

Поиск оптимального хода

8 Как алгоритмы ИИ балансируют между скоростью вычислений и нахождением оптимальных решений?

ИИ необходимо для начала находить эффективный, оптимальный путь решения поставленной задачи, благодаря чему создаётся баланс между скоростью вычислений и нахождением оптимальных решений. Т.е. он не работает лишь на одном и том же алгоритме, который может быть неэффективен в некоторых задачах, в отличие от других, что и делает ИИ сбалансированным.

9 Каковы основные элементы задачи поиска маршрута по карте?

Основными элементами задачи поиска маршрута по карте являются город А и город Б, между которыми располагаются другие точки городов и длины путей между ними.

10 Как можно оценить оптимальность решения задачи маршрутизации на карте Румынии?

Оптимальность решения в данном случае предполагает нахождение маршрутас минимальной стоимостью.

11 Что представляет собой исходное состояние дерева поиска в задаче маршрутизации по карте Румынии?

Исходным состоянием дерева поиска в задаче маршрутизации является

отправная точка (город Арад).

12 Какие узлы называются листовыми в контексте алгоритма поиска по дереву?

Листовой узел – возможное решение при построении и расширении дерева.

13 Что происходит на этапе расширения узла в дереве поиска?

Расширение подразумевает применение функции преемника, которая определяет все возможные действия, применимые к текущему состоянию.

14 Какие города можно посетить, совершив одно действие из Арада в примере задачи поиска по карте?

Сибиу, Тимишоара, Зеринд.

15 Как определяется целевое состояние в алгоритме поиска по дереву?

Целевым состоянием в алгоритме поиска по дереву является некая точка Б, то есть то, до какого момента будут разрастаться листовые узлы.

16 Какие основные шаги выполняет алгоритм поиска по дереву?

- Выбор исходного состояния

- Формирование листовых узлов

- Определение целевого состояния

17 Чем различаются состояния и узлы в дереве поиска?

Состояния и узлы - это не одно и то же: состояние - это конфигурация плиток в головоломке, а узел - это структура данных, включающая это состояние и дополнительную информацию.

18 Что такое функция преемника и как она используется в алгоритме поиска?

Функция расширения играет ключевую роль в алгоритме поиска по дереву. Она создаёт новые узлы, применяя действия к родительскому узлу и заполняя различные поля с использованием функции преемника, чтобы создать соответствующие состояния. Так образуется основная структура поиска: дерево инициализируется начальным состоянием, затем функция преемника многократно применяется к листьям дерева, проверяя, достигли ли мы цели.

19 Какое влияние на поиск оказывают такие параметры, как b

(разветвление), d (глубина решения) и m (максимальная глубина)?

b (максимальный коэффициент разветвления) - показывает, сколько дочерних узлов может иметь один узел.

d (глубина наименее дорогого решения) - определяет, насколько далеко нужно спуститься по дереву для нахождения оптимального решения.

m (максимальная глубина дерева) - показывает, насколько глубоко можно в принципе спуститься по дереву. В некоторых случаях это значение может быть бесконечным.

20 Как алгоритмы поиска по дереву оцениваются по критериям полноты, временной и пространственной сложности, а также оптимальности?

Полнота означает, что алгоритм находит решение, если оно существует. Отсутствие решения возможно только в случае, когда задача невыполнима.

Временная сложность измеряется количеством сгенерированных узлов, а не временем в секундах или циклах ЦПУ. Она пропорциональна общему количеству узлов.

Пространственная сложность относится к максимальному количеству узлов, которые нужно хранить в памяти в любой момент времени. В некоторых алгоритмах она совпадает с временной сложностью.

Оптимальность - это способность алгоритма всегда находить наилучшее решение с минимальной стоимостью, если таковое существует. Она не связана напрямую с временной или пространственной сложностью, которые измеряют количество узлов. Оптимальность не гарантирует быстродействие или экономию памяти, а лишь обеспечивает нахождение решения с наименьшей стоимостью.

21 Какую роль выполняет класс Problem в приведенном коде?

Абстрактный класс для формальной задачи. Новый домен специализирует этот класс, переопределяя «actions» и «results», и, возможно, другие методы.

22 Какие методы необходимо переопределить при наследовании класса Problem ?

Необходимо переопределить методы actions, result, is_goal, action_coost, h.

23 Что делает метод is_goal в классе Problem?

Проверяет, достигнуто ли целевое состояние.

24 Для чего используется метод action_cost в классе Problem ?

Этот метод и h предоставляют стандартные реализации для стоимости действия и эвристической функции соответственно.

25 Какую задачу выполняет класс Node в алгоритмах поиска?

Является узлом в дереве поиска.

26 Какие параметры принимает конструктор класса Node?

Принимает текущее состояние state, ссылку на родительский узел parent, действие, которое привело к этому узлу action и стоимость пути path_cost.

27 Что представляет собой специальный узел failure?

Необходим для обозначения неудачи в поиске.

28 Для чего используется функция expand в коде?

Расширяет узел и генерирует дочерние узлы.

29 Какая последовательность действий генерируется с помощью функции path_actions?

Возвращает последовательность действий, которые привели к этому узлу.

30 Чем отличается функция path_states от функции path_actions?

Функция возвращает последовательность состояний, ведущих к path_states данному узлу, в отличие от path_actions, которые возвращают последовательность действий.

31 Какой тип данных используется для реализации FIFOQueue?

FIFOQueue - это очередь, где первый добавленный элемент будет первым извлеченным. Она реализуется с помощью deque из модуля collections. deque - это обобщенная версия стека и очереди, которая поддерживает добавление и удаление элементов с обоих концов.

32 Чем отличается очередь FIFOQueue от LIFOQueue ?

FIFOQueue – очередь первым пришел, первым ушел.

LIFOQueue – очередь последним пришел, первым ушел.

33 Как работает метод add в классе PriorityQueue?

add: Метод для добавления элемента в очередь. Каждый элемент добавляется в кучу с его приоритетом, определенным функцией key.

34 В каких ситуациях применяются очереди с приоритетом?

Очереди с приоритетом могут применяться, например, в:

Системы моделирования, в которых ключи могут соответствовать моментам возникновения событий для обработки их в хронологическом порядке.

Системы планирования заданий в вычислительных системах, где ключи могут соответствовать приоритетам, указывающим, какой из пользователей должен быть обслужен первым.

Численные расчёты, в которых ключами могут быть погрешности вычислений, а приоритеты показывают, что максимальная погрешность должна быть обработана первой.

35 Как функция heappop помогает в реализации очереди с приоритетом?

Функция heappop() в модуле heapq в Python используется для удаления и возврата наименьшего элемента данных из кучи.

Вывод: в ходе выполнения лабораторной работы были приобретены навыки по исследованию методов поиска в пространстве состояния.