**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное  
учреждение высшего образования**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ  
ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(СПбГЭУ)**

Факультетинформатики и прикладной математики

Кафедра прикладной математики и экономико-математических методов

**КУРСОВАЯ РАБОТА**

по дисциплине:

«**Системы компьютерной математики**»

Тема: «Имитационную модель рабочего дня таксопарка»

Направление 01.03.02 «Прикладная математика и информатика»

Направленность «Прикладная математика и информатика в экономике  
и управлении»

Обучающийся Крестьянинов Александр Александрович

Группа ПМ-2101 Подпись



Проверил Фридман Григорий Морицович

Должность д.т.н., профессор

Оценка Дата

Подпись

Санкт-Петербург

2022 г.

Содержание

[Введение 3](#_Toc120639337)

[Нью-йоркское такси 4](#_Toc120639338)

[Алгоритм распределения машин 6](#_Toc120639339)

[Визуализация 9](#_Toc120639340)

[Анализ 11](#_Toc120639341)

[Заключение 13](#_Toc120639342)

[Библиографический Список 14](#_Toc120639343)

[Приложение 15](#_Toc120639344)

[1.1. Объекты 15](#_Toc120639345)

[1.1.1. Объект заказа (*Order)* 15](#_Toc120639346)

[1.1.2. Объект машины (*Car*) 16](#_Toc120639347)

[1.1.3. Объект очереди (PriorityEntry) 16](#_Toc120639348)

[1.1.4. Объект отчета (*Report*) 16](#_Toc120639349)

[1.2. Глобальные переменные 17](#_Toc120639350)

[1.3. Функции 18](#_Toc120639351)

Введение

Цель курсовой работы – создание имитационную модель рабочего дня таксопарка в системе Wolfram Mathematica. В данной реализации предполагается обработка списков заказов за день, настраиваемое задаваемыми пользовательскими переменными имитирование работы таксопарка, визуализация результатов вычислений.

1. Нью-йоркское такси

Для создания модели было выбрано нью-йоркское желтое такси, насчитывающее около 13 000 машин и более 300 000 заказов ежедневно. Данные по заказам, полигоны районов и их названия взяты с официального сайта Нью-Йорка.

Область работы такси представляет собой 5 городских районов и один придаточный (EWR, аэропорт). Каждый из районов включает в себя подрайоны, которых в сумме 263. (см. Рисунок 1)

Изображение выглядит как карта

Автоматически созданное описание

Рисунок 1. Нью-Йорк

Соответственно каждому подрайону присвоен индекс от 1 до 263. (см. Рисунок 2)

Изображение выглядит как карта

Автоматически созданное описание

Рисунок 2. Пример распределения индексов в Бронкс

Данные о заказах разбиты по дням и представляют собой таблицы в формате CSV.

# Алгоритм распределения машин

Изначально мы распределяем машины в соответствии со спросом на них в начале дня (см. Рисунок 3) (см. Рисунок 4). Список всех объектов машин выглядит следующим образом:

Изображение выглядит как карта

Автоматически созданное описание

Рисунок . Распределение машин в начале дняИзображение выглядит как игрушка

Автоматически созданное описание

Рисунок . Манхеттен

Ключевое действие алгоритма – поиск машины, очевидно, что перебор всех машин – неоптимален. Для подбора подходящего такси, достаточно рассмотреть ближайшие районы. Представим все районы в виде очереди с приоритетом:

Стоит отметить, что в Wolfram Mathematica приоритетная очередь представляет собой кучу (см. Рисунок 5), поэтому взятие элемента происходит за Θ(1), заполнение за Θ(n log n).

Распределим машины по очередям, за приоритетность будет отвечать поле объекта машины *Time* умноженное на -1*,* т.е. доставаемая машина гарантированно будет лучшей (доедет быстрее всего до заказчика) среди всех такси в районе.

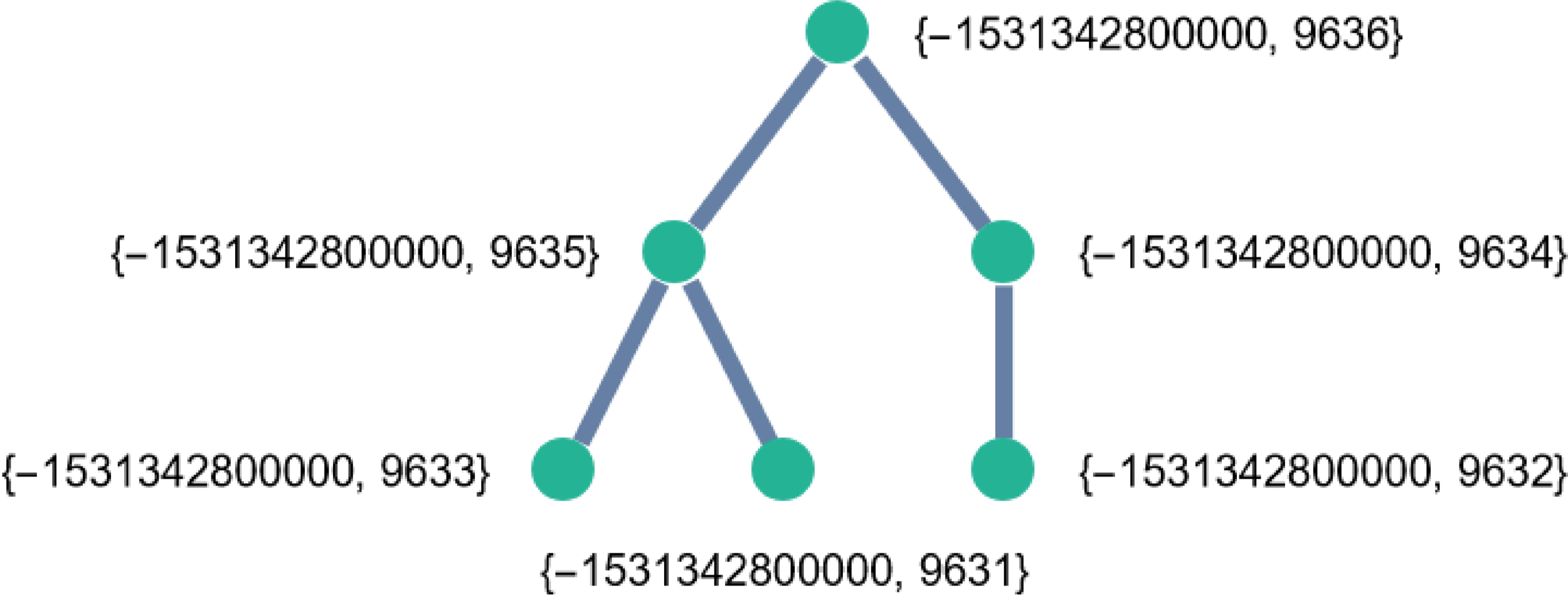


Рисунок 5. Визуализация одной из очередей

Количество рассматриваемых районов, определяется *sampleSize,* чем меньше эта константа, тем быстрее работа алгоритма, но вероятность нахождения машины для заказа уменьшается.

Из каждого рассматриваемого берется по машине, с помощью функции оценки выбирается лучшая машина, если она успеет доехать. Функция оценки учитывает:

* Время, нужное такси, для того чтобы доехать до заказчика
* Время готовности заказчика ожидать такси (константа *waitingTime*)
* Запас топлива (оставшееся количество километров)
* Время, если есть, чтобы доделать текущий заказ.

С помощью перечисленных параметров рассчитывает числовой коэффициент каждой машины, и выбирается машина, способная доехать до клиента за минимальное время.

В конце работы алгоритма формируется отчет (*Report*)

# Визуализация

Для наглядности, была выбрана визуализация в трёхмерном пространстве. Изначально для каждого района имеется собственный полигон. (см. Рисунок 6)

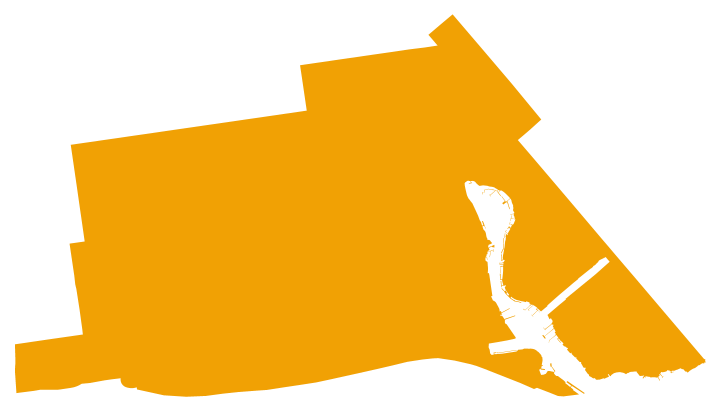


Рисунок 6. Полигон в двумерном пространстве

Для того чтобы отрисовать объемный район (полиэдр) нужно представить полигон в трехмерном пространстве, делается это добавлением третей координаты к каждой точке полигона. (см. Рисунок 7)

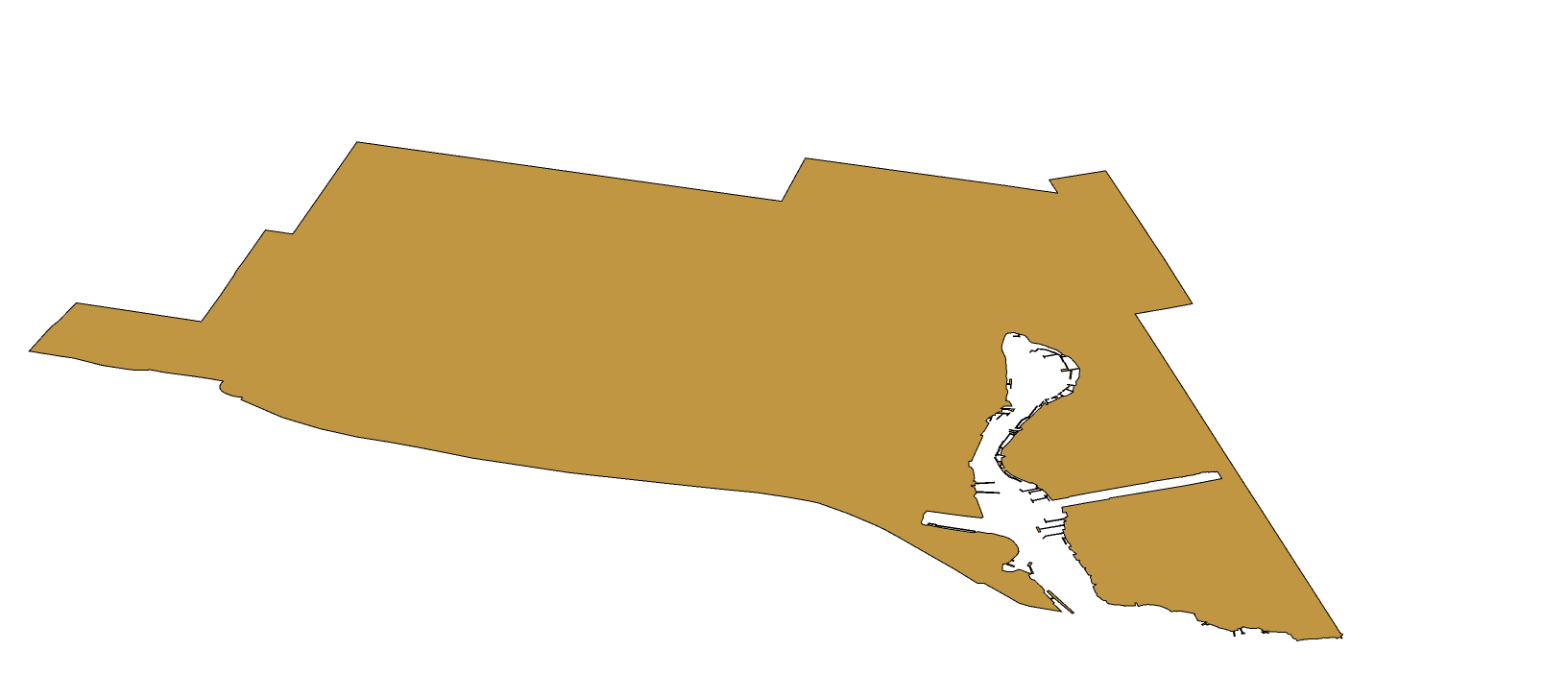


Рисунок 7. Полигон в трехмерном пространстве

Для создания полиэдра, нужно вычислить полигон каждой его грани, основания объемного многоугольника это изначальные полигоны с разным *z* координатами. Боковые грани представляют собой прямоугольники, составленными из последовательных пар точек верхней и нижней граней. (см. Рисунок 8)

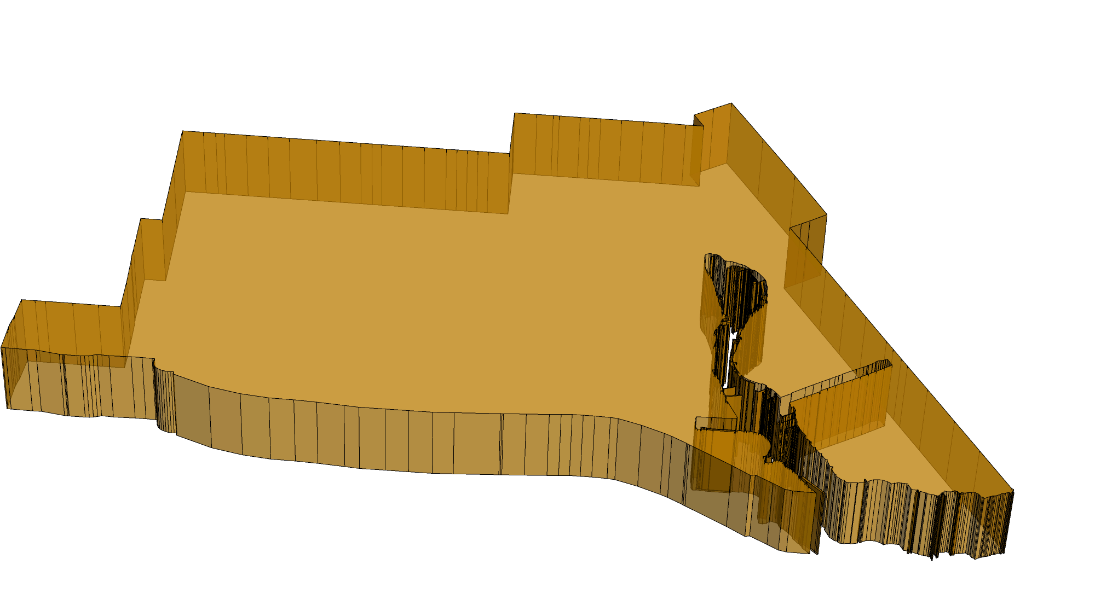


Рисунок 8. Полиэдр

# Анализ

Изображение выглядит как текст, темный

Автоматически созданное описание

Рисунок 9. Распределение заказов за день

Как видно из диаграммы заказов (см. Рисунок 9), самый загруженный район- Манхэттен, основное распределение машин будет происходить в нем. Два обособленных подрайона - аэропорты.

Запустим алгоритм с разным количеством машин.

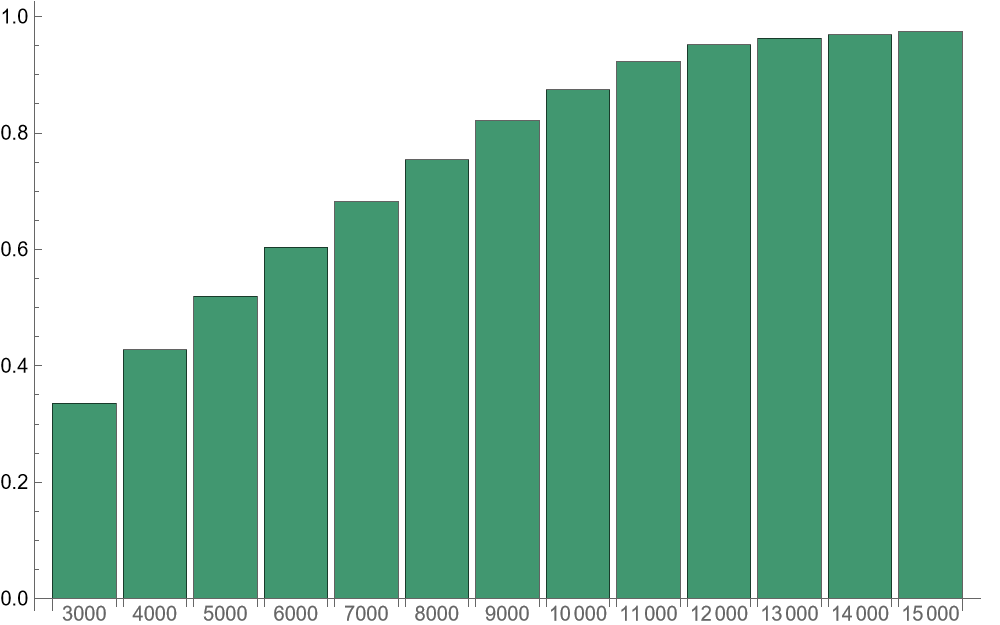


Рисунок 10 – Процент принятых заказов от количества машин

Обрати внимание на параметр отчета Accepted, увеличение числа машин прямо пропорционально росту процента принятых заказов (см. Рисунок 10).

Стоит обратить внимание на тот факт, что увеличение количества машин ведёт к дополнительным расходам. Исходя из открытых источников в среднем таксит в день зарабатывает 300$, получим чистый заработок. (см. Рисунок 11)

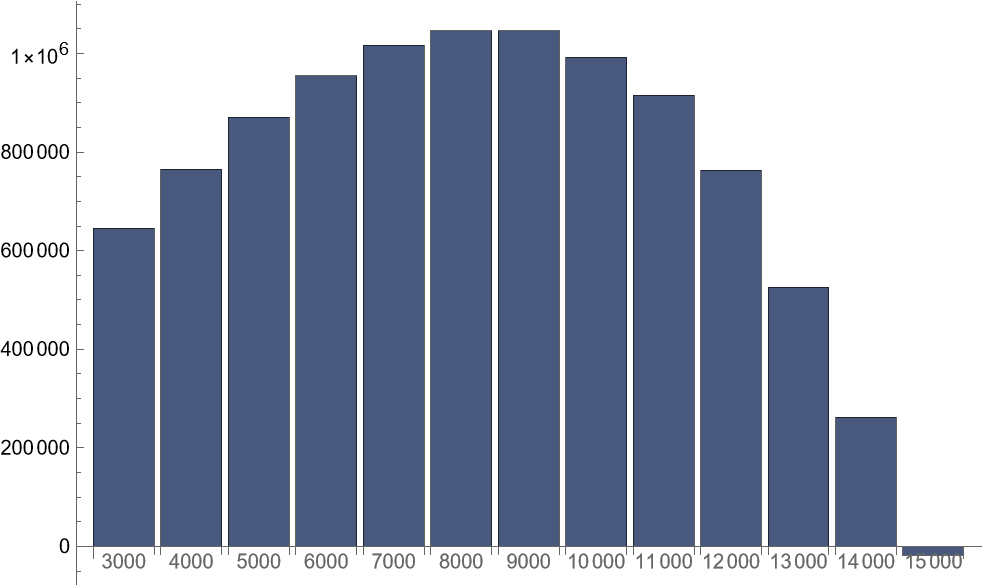


Рисунок 11 – Заработок от количества машин

Посмотрев на эти два графика, можно сделать вывод об оптимальном количестве машин, нужном для достаточного удовлетворения потребителя и максимизации прибыли. (см. Рисунок 12)

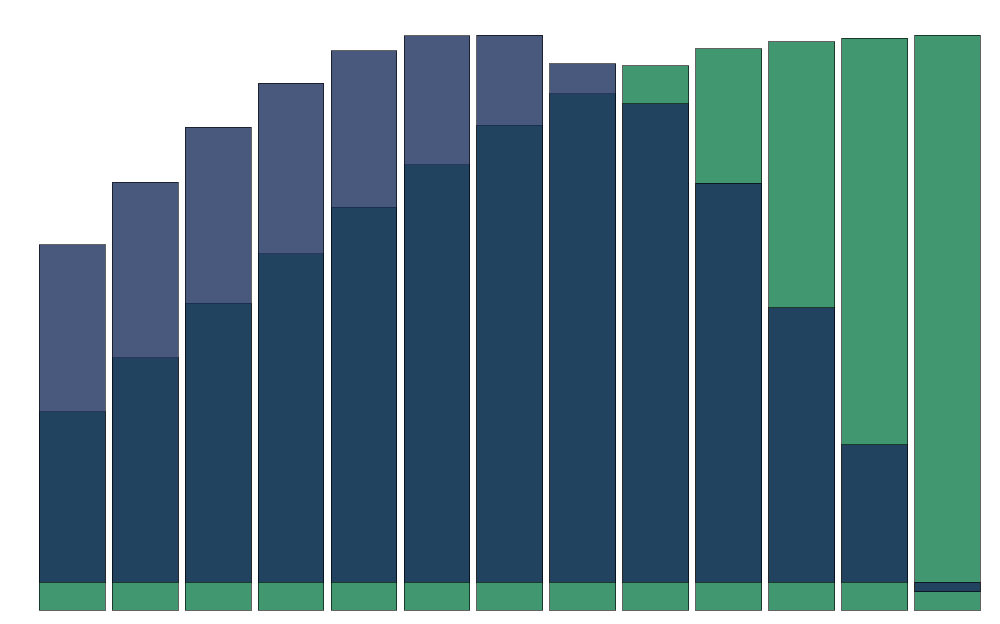


Рисунок 12 – Сравнение графиков

Для выбранного дня оптимальное количество машин – 10 000.

# Заключение

В ходе написания работы были получены навыки работы с большими объемами данных, их обработки и форматирования, также освоение главных функций графики в Wolfram Mathematica.

Задачи объемы курсовой работы были выполнены полностью, также стоит отметить, что получившийся алгоритм можно модифицировать для данных из других городов.

Библиографический Список

1. Wolfram Documentation [Электронный ресурс] / Documentation center.

URL: https://reference.wolfram.com/language/ (дата обращения – 18.05.22).

1. TLC Trip Record Data [Электронный ресурс]

URL: <https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page/> (дата обращения – 20.05.22)

# Приложение

## Объекты

### Объект заказа (*Order)*

1. ***VendorID*** - код, указывающий поставщика TPEP, предоставившего запись.
2. ***tpepPickupDatetime*** - дата и время включения счетчика.
3. ***tpepDropoffDatetime*** - дата и время отключения счетчика.
4. ***PassengerCount*** - количество пассажиров в автомобиле
5. ***TripDistance*** - пройденное расстояние поездки в милях, указанное таксометром
6. ***RatecodeID*** - окончательный код тарифа, действующий в конце поездки.
7. ***StoreAndFwdFlag*** –этот флаг указывает, хранилась ли запись о поездке в транспортном средстве. Память перед отправкой поставщику, также известная как «хранить и пересылать, потому что у автомобиля не было подключения к серверу.
8. ***PULocationID*** - зона такси TLC, в которой был задействован таксометр
9. ***DOLocationID*** - зона такси TLC, в которой был отключен таксометр
10. ***PaymentType***- числовой код, обозначающий, как пассажир оплатил поездку.
11. ***FareAmount*** - стоимость проезда по времени и расстоянию, рассчитанная счетчиком.
12. ***Extra*** - различные дополнения и доплаты. В настоящее время сюда входят только 0,50 и 1 доллар в час пик и ночные сборы.
13. ***MTAtax*** - налог MTA в размере 0,50 доллара США, который автоматически срабатывает на основе счетчика скорость использования.
14. ***TipAmount*** - сумма чаевых
15. ***TollsAmount*** - общая сумма всех сборов, уплаченных за поездку.
16. ***ImprovementSurcharge*** - доплата за улучшение в размере 0,30 доллара США оценивается в поездках при спуске флага. Доплата за благоустройство начала взиматься в 2015 году.
17. ***TotalAmount*** - общая сумма, начисленная пассажирам, включает чаевые.

### Объект машины (*Car*)

1. ***Zone*** – индекс района в котором машина находится
2. ***Time*** – время окончание последнего принятого заказа
3. ***Gas*** – расстояния (в милях), которое может пройти машина с текущим количеством бензина
4. ***Amount*** – общая стоимость всех принятых заказов
5. ***Count*** – количество принятых заказов

### Объект очереди (PriorityEntry)

1. ***Priority*** – число означающее приоритетность объекта в очереди
2. ***CarId*** – id объекта машины

### Объект отчета (*Report*)

* ***AlgTime*** – абсолютное время работы алгоритма
* ***OrderToCar*** – список, сформированный заменой заказа на машину, выполняющую заказ
* ***Accepted*** – процент, принятых заказов
* ***TotalDayAmount*** – заработок таксопарка за день

## Глобальные переменные

1. ***year, month, day*** – определяют дату, за которую берутся заказы

2. ***stime*** – индекс времени подачи машины клиенту в объекте заказа

3. ***ftime*** – индекс времени окончания поездки в объекте заказа

4. ***distance*** – индекс расстояния, пройденного машиной в объекте заказа

5. ***price*** – индекс стоимости поездки в объекте заказа

6. ***szone*** – индекс идентификатора района, в который была вызвана машина в объекте заказа

7. ***fzone*** – индекс идентификатора района, в котором машина завершила поездку в объекте заказа

8. ***carsCount*** – количество машин, выделяемых на день

9. ***fz*** – индекс идентификатора района, в котором машина завершила поездку в объекте машины

10. ***ft*** – индекс времени окончания поездки в объекте машины

11. ***gas*** – индекс расстояния (в милях), которое может пройти машина с текущим количеством бензина, в объекте машины

12. ***amount*** – индекс суммы всех принятых машиной заказов в объекте машины

13. ***count*** – индекс количества принятых заказов в объекте машины

14. ***waitingTime*** – максимальное время ожидания машины заказчиком

15. ***sampleSize*** – максимальное количество просматриваемых близлежащих районов для выбора оптимальной машины

16. ***dictionary*** – список полных названий районов

17. ***polygons*** – список объектов **Polygon**

18. ***polygonsCoordinates*** – список координат полигонов в трехмерном пространстве

19. ***сenters*** – список координат центров полигонов в трехмерном пространстве

## Функции

1. ***NormTime*** – получает на вход время и переводит его из миллисекунд в абсолютное время валидное для **DateObject**.

2. ***MronTime*** – получает на вход абсолютное время и переводит его из абсолютного в миллисекунды

3. ***PositiveRound*** – получает на вход число, и для отрицательных чисел возвращает значение 0, для остальных – возвращает их без изменения.

4. ***OO*** – получает на вход однозначное или двузначное число; для однозначных возвращает это же число строкой с добавленным слева 0, для двузначных чисел возвращает их в виде строки без изменения

5. ***GetMatrix*** – получает на вход список объектов (каждый объект состоит из индекса объекта в матрице и его значения), возвращает матрицу средних значений, по полученным индексам

6. ***GetMatrixSet*** – получает на вход список заказов за день, возвращает пару из матрицы средних расстояний между районами, распределенных по часам, и матрицы среднего времени поездки между районами, распределенного по часам

7. ***GetNearestZones*** – получает на вход матрицу среднего времени поездки между районами по часам, возвращает список упорядоченных по расстоянию списков индексов близлежащих районов

8. ***GetNumberHour*** – получает на вход объект заказа, возвращает номер часа в сутках от 0 до 23

9. ***AddZ*** – переводит список координат из двухмерного пространства в трехмерное

10. ***PolygonZoneBoarders*** – создает список недостающих граней полиэдрона для района

11. ***PolyhedronZone*** – создает из списка полигонов список полиэдров района

12. ***Formatting*** – форматирует объект машины для вывода в таблицу

13. ***FillCar*** – заправляет объект машины

14. ***GetOrders*** – получает на вход дату, за которую берутся заказы, и возвращает список заказов за этот день

15. ***GetCars*** – возвращает список объектов машин, распределённых по раинам в зависимости от спроса

16. ***GoOneDay*** – симулирует один день таксопарка, получает на вход список заказов и количество машин, возвращает объект отчета

17. ***GetDayDemandMap*** – возвращает диаграмму распределения заказов по раинам

18. ***GetCarWayMap*** – возвращает карту пути, пройденного машиной, на вход получает список районов

19. ***GetTimeLine*** – возвращает временной график одного заказа

20. ***GetOrderData*** – возвращает таблицу данных по выполненному заказу

21. ***GetDayCarReportByOrder***– возвращает графический отчет по машине за день

22. ***GetCarOrders*** – возвращает все заказы выполненные машиной за день

23. ***GetCarReport*** – возвращает графический отчет по состоянию машины после каждого заказа

24. ***GetCarWay*** – возвращает список индексов районов в которых была машина