БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Кафедра дискретной математики и алгоритмики

КУЛИК СЕРГЕЙ ДМИТРИЕВИЧ

ПРИМЕНЕНИЕ ЭВРИСТИЧЕСКИХ АЛГОРИТМОВ ДЛЯ РЕШЕНИЯ ЗАДАЧ ОПТИМАЛЬНОЙ ДОСТАВКИ ТОВАРОВ С ПОМОЩЬЮ ДРОНОВ

Курсовая работа

студента 4 курса 3 группы

|  |  |
| --- | --- |
| “Допустить к защите“  Зав. кафедройДМА  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  “\_\_\_” \_\_\_\_\_\_\_\_\_\_\_ 2016 г. | **Руководитель**  Сатаневский Владислав Валерьевич, ассистент кафедры дискретной математики и алгоритмики |

Минск 2016

Содержание

[Введение 3](#_Toc450876067)

[Глава 1. Краткий обзор библиотеки testlib 4](#_Toc450876068)

[1.1 Общие положения 4](#_Toc450876069)

[1.2 Генераторы, валидаторы и чекеры 4](#_Toc450876070)

[1.3 Используемые нами средства библиотеки testlib 5](#_Toc450876071)

[Глава 2. Расширение библиотеки testlib 6](#_Toc450876072)

[2.1 Класс BaseGenerator 6](#_Toc450876073)

[2.2 Генерация последовательностей. Класс SequenceGenerator 6](#_Toc450876074)

[2.3 Генерация матриц. Класс MatrixGenerator 8](#_Toc450876075)

[2.4 Генерация строк. Класс StringGenerator. 9](#_Toc450876076)

[2.5 Вспомогательный класс Graph 11](#_Toc450876077)

[2.6 Генерация деревьев. Класс TreeGenerator 13](#_Toc450876078)

[2.7 Генерация графов. Класс GraphGenerator 16](#_Toc450876079)

[2.8 Замечания по реализации 17](#_Toc450876080)

[Глава 3. Демонстрация возможностей расширенной библиотеки 18](#_Toc450876081)

[3.1 Генерация тестов к задаче “Компьютерная сеть” 18](#_Toc450876082)

[3.2 Поиск неверных решений задачи “Открытки и конверты” 18](#_Toc450876083)

[3.3 Поиск неверных решений задачи “Лабиринт” 20](#_Toc450876084)

[Заключение 22](#_Toc450876085)

[Литература 23](#_Toc450876086)

## Введение

Интернет-магазины уже сильно изменили процесс покупки различных товаров. Но в то же время, между покупкой товара в интернет-магазине и его получением проходит, как правило, несколько дней, а иногда и месяцев, которые уходят на доставку.

Этот процесс может быть значительно улучшен с использованием дронов – летательных аппаратов, доставляющих товары покупателям. Используя дронов возможно также избежать проблем с дорожно-транспортной ситуацией.

В то же время, несмотря на то, что технологии разработки дронов стремительно развиваются, остается открытым вопрос их координации и управления. В данной курсовой работе мы разработаем эвристический подход к задаче организации оптимальной доставки товаров покупателям. Эта задача представлялась компанией Google на соревновании Hash Code как задача из реальной практики разработчиков и исследователей компании.

# Глава 1. Постановка задачи

## 1.1 Карта

В качестве карты, в задаче используется двумерная решетка с **R** строками и **C** столбцами. Решетка не является циклической и дронам не разрешается покидать пределы решетки. В любой момент времени, дрон может находится над любым узлом решетки и может быть задан парой своих координат **(r, c)**.

## 1.2 Товары

Есть несколько видов товаров, доступных для заказа. Для каждого товара известно, в каких хранилищах он содержится а также, в каком количестве. У каждого товара имеется вес, одинаковый для каждой единицы товара этого типа. Гарантируется, что этот вес не превышает максимальный возможный вес, который может нести дрон.

## 1.3 Хранилища

Все товары содержатся в нескольких хранилищах. Каждое хранилище расположено в заранее известном узле решетки и содержит заранее известное число товаров, свое для каждого типа товара. В нашей модели хранилища не получают новых товаров во время доставки.

**1.4 Заказы**

Каждый заказ содержит информацию о купленных товарах. Товары могут быть одного или нескольких типов. Заказ также может содержать несколько товаров одного типа.

Для каждого заказа также дан адрес – узел решетки, в которой надо доставить товары. Несколько заказов могут иметь совпадающий адрес, однако заказ не может иметь адрес, совпадающий с каким-либо хранилищем.

Заказ становится выполненым, когда все заказанные товары доставлены по адресу заказа. Товары из заказа можно доставлять по частям, в том числе и используя несколько различных дронов одновременно.

Гарантируется, что для каждого типа товара, количество товаров этого типа в хранилищах будет не меньше, чем общее количество товаров данного типа во всех заказах.

**1.5 Дроны**

Дроны используются для перемещения товаров из хранилищ к покупателям. Дрон всегда использует кратчайший путь для того, чтобы перемещаться от одного узла решетки к другому. Расстояние от ячейки **(ra, ca)** до ячейки **(rb, cb)** вычисляется как Евклидово расстояние, округленное вверх:

Таким образом, если Евклидово расстояние между узлами решетки будет равно 2.9, то расстояние в нашей задаче будет равно 3. Если же Евклидово расстояние будет равно 4, то расстояние в нашей задаче будет равно 4.

Несколько дронов могут находиться в любом узле решетки в любой момент времени – они никогда не столкнутся из-за того, что летают на различных высотах.

В начале симуляции, все дроны находятся в первом хранилище, имеющим уникальный идентификатор 0.

**1.6 Команды**

Каждый дрон может получать и выполнять команды одного из следующих двух типов:

**Команда загрузки товаров**: загрузить заданное количество товаров заданного типа из заданного хранилища. Если дрон не находится в данном хранилище, он переместится туда за время, равное расстоянию между текущей позицией дрона и позицией хранилища. Запрошенное количество товаров заданного типа должно присутствовать в этом хранилище. Суммарный вес загруженных товаров после загрузки не должен превышать грузоподъемности дрона.

**Команда доставки товаров**: доставить заданное количество товаров заданного типа покупателю. Если дрон не находится в узле решетки, равном адресу заказа, он переместится туда за время, равное расстоянию между текущей позицией дрона и адресом заказа. На момент подачи команды, дрон должен содержать запрошенное число товаров запрошенного типа.

**1.7 Симуляция**

Процесс симуляции разделен на **T** моментов времени (тактов) одинаковой длины, пронумерованных целыми числами от **0** до **T-1** включительно. Дрон выполняет команды, данные ему в том порядке, в котором они перечислены по очереди. Первая команда выполняется в момент времени **0**. Время выполнения команды равняется **(d+1)** моментам времени, где **d** – расстояние, преодоленное дроном, а один момент времени тратится на выполнение самого действия – загрузку или выгрузку товаров. Суммарное время на выполнение всех команд, данных дрону не должно превышать номера последнего такта, то есть **(T-1)**.

**1.8 Метрика качества**

Значение метрики качества для доставки равняется сумме значений метрик качества для отдельных заказов.

Значение метрики качества для заказа находится в отрезке от 0 до 100 и вычисляется по следующей формуле:

В формуле выше **T** – общее время симуляции, **t** – время доставки последнего товара из заказа.

**1.9 Наборы входных данных**

Для соревнования компанией Google было предоставлено три набора входных данных. Приведем ограничения каждого из них:

1. Решетку составляют 400 строк и 600 столбцов. Симуляция продолжается в течении 112993 тактов. В наличии имеется 30 дронов. Имеется 400 типов товаров и 10 хранилищ, в которых располагаются эти товары. Дано 1250 заказов.
2. Решетку составляют 240 строк и 400 столбцов. Симуляция продолжается в течении 144612 тактов. В наличии имеется 20 дронов. Имеется 1000 типов товаров и одно хранилище, в котором располагаются эти товары. Дано 800 заказов.
3. Решетку составляют 300 строк и 500 столбцов. Симуляция продолжается в течении 145416 тактов. В наличии имеется 30 дронов. Имеется 2000 типов товаров и 16 хранилищ, в которых располагаются эти товары. Дано 1000 заказов.

# Глава 2. Обзор имеющихся решений и разработка алгоритма

## 2.1 Имеющиеся базовые подходы

Рассмотрим известные базовые подходы к решению данной задачи.

## 2.2 Генерация последовательностей. Класс SequenceGenerator

Класс

## 2.3 Генерация матриц. Класс MatrixGenerator

Класс

## 2.4 Генерация строк. Класс StringGenerator.

Класс

## 2.5 Вспомогательный класс Graph

поток.

## 2.6 Генерация деревьев. Класс TreeGenerator

Класс

## 2.7 Генерация графов. Класс GraphGenerator

Класс

# Глава 3. Анализ и сравнение результатов

## 3.1 Обзор верхних оценок на качество ответа

## К сожалению, вычислить точное максимально возможное значение метрики качества для наборов данных в такой задаче не представляется возможным из-за сложной природы задачи.

## Нам, однако, необходимо дать некоторые верхние оценки на эту величину для того, чтобы оценить, насколько качество найденных решений близко к оптимальному. В этом пункте мы дадим верхнюю оценку на величину оптимального значения целевой функции задачи.

## Утверждение 1. Максимальное значение метрики качества не превысит , где M – число заказов.

## Доказательство этого утверждения тривиально, ибо значение метрики качества для набора данных равняется сумме значений метрики качества по отдельным заказам, которых имеется M. В силу того, что значение метрики качества – действительное число из отрезка , верхней оценкой является .

## Отметим, что полученная нами верхняя оценка является довольно грубой. Попробуем ее уточнить.

## Заметим, что время на доставку товаров не может быть меньше, чем время, которое требуется на то, чтобы совершить перелет из стартовой точки всех дронов к точке заказа, то есть , где (xw, yw) – коордиты стартового хранилища, а (xi, yi) – коордиты адреса i-го заказа. Таким образом, подставляя это значение в качестве времени доставки мы получаем следующую верхнюю оценку на возможное значение метрики качества:

## Доказательство этой формулы непосредственно следует из рассуждений выше.

## 3.2 Поиск неверных решений задачи “Открытки и конверты”

## 3.3 Поиск неверных решений задачи “Лабиринт”

.

# Заключение

Нами был разработан эвристический алгоритм для управления дронами, обеспечивающий эффективную доставку товаров заказчикам. Сравнения нашего алгоритма с имеющимися подходами позволяет нам заключить, что полученный вариант является одним из наилучших известных вариантов решения данной задачи.

# Литература

* 1. [Электрон. ресурс – acm-test.bsu.by] – система тестирования Insight Runner.
  2. [Электрон. ресурс – <http://www.codechef.com>] – система онлайн-тестирования CodeChef.
  3. [Электрон. ресурс – <http://www.hackerrank.com>] – система онлайн-тестирования HackerRank.
  4. [Электрон. ресурс – <https://cms-dev.github.io/>] – система тестирования CMS, применяющаяся на международной олимпиаде по информатике.
  5. [Электрон. ресурс – <https://github.com/MikeMirzayanov/testlib>] – оригинальная версия библиотеки testlib
  6. [Электрон. ресурс – <https://github.com/zxqfd555/testlib>] – расширенная версия библиотеки testlib