ГБОУ Гимназия 1505

Расчет пропускной способности перекрестка в среде программирования

Писал: Арсений Канеп 10а

Руководитель: Мокин Е.А.

Москва, 2017

Введение.

Возрастающий объем трафика на улично-дорожной сети города и транспортных пересечениях, зачастую, приводят к невозможности выполнения расчета пропускной способности ручными методами. Визуализация условий движения, в настоящее время, становится основным способом оценить условия и организацию движения. Автоматизировать расчеты пропускной способности и имитировать реальные условия движения возможно программными методами. Сейчас программные методы и высокий уровень развития ЭВТ и языков программирования позволяют создавать программные продукты, выполняющие расчеты и дающие оценку объемов трафика на улицах города, а так же предоставить инструментарий проектировщику для визуализации принятых решений. Сейчас многие фирмы представляют на рынке ПО свои продукты, выполняющие указанные функции моделирования транспортного движения. Среди таких можно отметить немецкий PTV Vissim, испанский aimsun. Есть и разработки отечественных фирм(менее известных). Данные продукты достигли высокого уровня реалистичности имитации дорожного движения. Они обладают проработанными математическими моделями, основанными на теории движения транспортных потоков. В своем исследовании я хочу постараться разобраться в теории движения транспортных потоков и сделать имитационную движения на транспортном перекрестке, используя язык программирования основанным на Java.

Глава 1.

1.1 Почему Java.

Я выбрал языком программирования Java, потому что это объектно-ориентированный язык. В моем случае это удобно. Я могу создать класс стандартной машины, и все объекты подогнать под эталонный размер машины. Так же Java является достаточно простым в изучении языком. Java частично интуитивно понятен, после прохождения простых языков, как Pascal и т.д. Так что я начинал не с нуля. В процессе написания программы, я понял, что такое же «приложение» можно сделать на многих языках.(см 2.2)

1.2 Суть исследования.

 Суть моего исследования – нахождение коэффициента загруженности дорожного перекрестка. Коэффициент загрузки – это отношение реальной пропускной способности перекрестка к теоретической, которая высчитывается, у меня в программе, в идеальном случае. Условия, при которых не будет заторов: коэффициент загрузки < 1.В наше время достаточно мало программ для столь неприятной работы, моя программа поможет сократить время, которое часто затрачивается «в пустую», даст возможность людям быстро высчитать коэффициент загрузки и оценить перекресток. Ежедневно эту процедуру выполняют. Кто-то покупает программы, кто-то вручную, а теперь кто-то сможет пользоваться моей программой и упростить себе жизнь. Просто представьте, что вы вместо того, что бы долго вычислять на листочке или , как делают более продвинутые, пользуются Exel, вы заходите в программу, которая все выполняет по нажатию пары кнопок и задание пары параметров.

Глава 2.

2.1 Основа программы.

Писал я свою программу в Intellij IDEA, потому что это среда разработки обладает удобным интерфейсом, можно переделать программу в консольный вид, что не потребует долгой установки. Это так же ускоряет работу, эту программу можно носить с собой и наглядно показывать на презентациях результаты исследований. Перед тем как выбрать Intellij IDEA, я просмотрел некоторые варианты, которые меня могли устроить, но в них я не мог понять, как сохранить программу в консольном виде.

 2.1.1 Описание программы.

 Моя программа основана на задание одного метода машины, и все параметры задаются в него. Благодаря такому методу, у меня вышло сократить код и сделать разные направления движения, так же можно доработать программы и добавить направлений. Перебирая и переписывая программу в разных вариантах, я пришел к такому виду, что позволяет легко изменять программу.

**import** java.util.Scanner; // подключение библиотек

**import** java.util.concurrent.TimeUnit;

**public class** zadan {

 **public static int** time = 0, time2 = 0, timep = 0;// времена светофоров(1,2 направления и пешеходного перехода)

 **public static void** main(String args[]) **throws** InterruptedException {

 **int** r = 0, n = 0, check = 0, check2 = 0, flow1 = 0, flow2 = 0;

 **int** day = 14, mouth = 03, year = 2017;

 **double** mediumspeed = 0, p = 0;

 **final double** width = 15.0, speeding = 3.0, length = 4.5;

 **boolean** exit = **true**; // задание "рядовых" переменных

 System.out.println(**"Вас приветсвует программа определения колличества машин, проезжаемых в час через перекресток"**);

 System.out.println(day + **"."** + mouth + **"."** + year);

 System.out.println(**"Эту программу написал Арсений Канеп"**);

 Scanner m = **new** Scanner(System.in);

 **while** (exit)

 {*// задаваемые данные*

 System.out.println(**"Введите практический поток для 1ой линии"**);

 **if** (m.hasNextInt()) {

 flow1 = m.nextInt();

 }

 System.out.println(**"Введите практический поток для 2ой линии"**);

 **if** (m.hasNextInt()) {

 flow2 = m.nextInt();

 }

 System.out.println(**"Введите период времени зеленого света на светофоре первой линии"**);

 **if** (m.hasNextInt()) {

 time = m.nextInt();

 }

 System.out.println(**"Введите период времени зеленого света на светофоре второй линии"**);

 **if** (m.hasNextInt()) {

 time2 = m.nextInt();

 }

 System.out.println(**"Введите период времени зеленого света на светофоре пешеходов"**);

 **if** (m.hasNextInt()) {

 timep = m.nextInt();

 }

 p = 3600/(time + time2 + timep);//расчёт количества циклов в час

 System.out.println(**"Введите среднюю скорость машин"**);

 **if** (m.hasNextDouble()) {

 mediumspeed = m.nextDouble();

 }

 System.out.println(**"Введите 1-прямо\n** **2-с поворото\n** **3-прямо и с поворотом"**);

 **if** (m.hasNextInt()) {

 n = m.nextInt();

 }

 **switch** (n) {*//выбор направленний*

 **case** 1:*//прямо*

 check = way(time, width, speeding, length, mediumspeed);

 check2 = way(time2, width, speeding, length, mediumspeed);

 **if** (check % 2 != 0)// так как check и check2 машины, то не полная машина проехать не может. Поэтому я узнаю чётное или нечётное число машин проехало и если нечётное, то добавляюодну машину.

 check++;

 **if** (check2 % 2 != 0)

 check2++;

 **if** (check % 2 == 0 && check2 % 2 == 0) {

 System.out.println(**"Количество машин на первом потоке в одну сторону, которые проехали прямо: "** + (check) + **"\n** **в другую сторону: "** + check2);

 System.out.println(**"Количество машин на первом потоке в одну сторону, которые проехали за час прямо: "** + (check \* p) + **"\n** **в другую сторону: "** + (check2\*p));

 System.out.println(**"Коэффициент загруженности первой линии перекрестка прямо: "** + (flow1/(check\*p)) + **"\n** **в другую сторону: "** + (flow2/(check2\*p)));

 }

 **break**;

 **case** 2: *// поворот*

 check = way(time, width, speeding/2, length, mediumspeed / 2);

 check2 = way(time2, width, speeding/2, length, mediumspeed / 2);

 **if** (check % 2 != 0)

 check++;

 **if** (check2 % 2 != 0)

 check2++;

 **if** (check % 2 == 0 && check2 % 2 == 0) {

 System.out.println(**"Количество машин на первом потоке в одну сторону, которые проехали с поворотом: "** + (check) + **"\n** **в другую сторону: "** + check2);

 System.out.println(**"Количество машин на первом потоке в одну сторону, которые проехали за час с поворотом: "** + (check \* p) + **"\n** **в другую сторону: "** + (check2\*p));

 System.out.println(**"Коэффициент загруженности первой линии перекрестка с поворотом: "** + (flow1/(check\*p)) + **"\n** **в другую сторону: "** + (flow2/(check2\*p)));

 }

 **break**;

 **case** 3:*// оба*

 check = way(time, width, (speeding+speeding/2)/2, length, (mediumspeed + mediumspeed / 2) / 2);

 check2 = way(time2, width, (speeding+speeding/2)/2, length, (mediumspeed + mediumspeed / 2) / 2);// сколько прошло машин на поворот, если считать, что каждая 2 машина поворачивает

 **if** (check % 2 != 0)

 check++;

 **if** (check2 % 2 != 0)

 check2++;

 **if** (check % 2 == 0 && check2 % 2 == 0) {

 System.out.println(**"Количество машин на первом потоке в одну сторону, которые проехали прямо и с поворотом: "** + (check) + **"\n** **в другую сторону: "** + check2);

 System.out.println(**"Количество машин на первом потоке в одну сторону, которые проехали за час прямо и с поворотом: "** + (check \* p) + **"\n** **в другую сторону: "** + (check2\*p));

 System.out.println(**"Коэффициент загруженности первой линии перекрестка прямо и с поворотом: "** + (flow1/(check\*p)) + **"\n** **в другую сторону: "** + (flow2/(check2\*p)));

 }

 **break**;

 }

 System.out.println(**"Примечание: первый поток одно направление в одну сторону, второй поток второе направление перекрестка в одну сторону"**);

 TimeUnit.SECONDS.sleep(3);

 System.out.println(**"0 - exit;"**);

 **if** (m.hasNextInt()) {

 r = m.nextInt();

 }

 **if** (r == 0)

 exit = **false**;

 }

 }

 **private static int** way(**int** time, **double** width, **double** speeding, **double** length, **double** mediumspeed) {

 **int** check = 0;

 **double** speeds[];// обозначение типа массива

 **double** roads[];

 **double** track[];

 speeds = **new double**[time];

 roads = **new double**[time];*//сколько должна пройти*

 track = **new double**[time];*//сколько прощла*

 roads[1] = width;*//ширина*

 **for** (**int** i = 1; i < time; i++) {

 speeds[i] = speeding;*// начальная скорость машины*

 track[i] = track[i] + speeds[i];*// сколько прошла машина*

 **if** (i > 1) {

 roads[i] = roads[i - 1] + length;*// сколько должна пройти машины, что бы преодалеть перекресток*

 **for** (**int** q = 0; q < i; q++) {

 **if** (speeds[q] < mediumspeed) {*// машина набирает скорость до того, как ее скорость станет равна средней скорости и ищет сколько прошла машина*

 speeds[q] = speeds[q] + speeding;

 track[q] = track[q] + speeds[q];

 } **else**

 track[q] = track[q] + speeds[q];

 }

 }

 }

 **for** (**int** q = 1; q < time; q++) {

 **if** (track[q] - roads[q] >= 0)*//прошла ли машина перекресток*

 check++;

 }

 **return** check;

 }

}

 2.1.2 Вызов.

**** Вызов программы осуществляется запуском Launcher.bat. В процессе работы программа взаимодействует с оператором через консоль, выполняющий следующие функции: Описать назначение и функции элементов управления интерфейса. На рисунке показано, как выводятся данные из программы, как выглядит программа в консоли.

2.2 Иследование.

 Мое иследование заключается в том, что я основываясь на реальных данных, которые я собирал, стоя на Балаклавском проспекте, ввёл и получилось более точный ответ. Данные я собирал, стоя на перекрестке в течение 15 минут, приходя каждый час, что бы данные были более точными. Умножив данные на 4, считая каждую машину. По моим данным при 250 машин проезжающих по одной полосе с поворотом, 0,4 - коэффициент загрузки.

Заключение.

Итак, основываясь на исследование, я могу сказать, что моя программа полезна. Благодаря ей я и многие другие люди смогут этой программой упростить себе жизнь.

Список литературы.

1. <https://javatalks.ru> - форум по java
2. <https://habrahabr.ru/post/43293/> - форум по java
3. [http://codingbat.com/java](https://vk.com/away.php?to=http%253A%252F%252Fcodingbat.com%252Fjava&cc_key=) - форум по java