**Градски транспорт**

*Тема 9*

Съдържание

1. Репрезентация2

2. Структури от данни2

3. Зареждане на данни3

3.1. Формат на текстовия файл4

**4. Намиране на оптимален път4**

5. Разширяване на пътната мрежа5

5.1. Удължаване на съществуваща линия5

5.2. Добавяне на нов курс към линия6

5.3. Добавяне на нова линия6

6. Редуциране на пътната мрежа7

6.1. Премахване на спирка7

6.2. Премахване на курс от линия7

**6.3. Премахване линия7**

1. **Репрезентация**

Проектът представлява пътна мрежа и дава възможност както за разширяването и намаляването ѝ, така и за намиране на оптимален път от една точка до друга. Също така се поддържат опции за показване на различни информации за пътната мрежа, запис на направени промени и отваряне на нова пътна мрежа. Пътната мрежа е представена чрез ориентиран, тегловен мултиграф. Спирките са върховете на графа, а курсовете, които автобусите изпълняват, са ребрата на графа.

1. **Структури от данни**

За съхранение на върховете на графа използвам ***unordered\_map*** с ключ – номер на спирка и стойност - структурата ***BusStop***, която от своя страна съдържа цялата информация за една спирка, а именно – име на спирката, номер, броя на автобусите минаващи през нея и самите автобуси. Последните ги съхранявам в ***unordered\_map<int, list<Bus>>***. Ключът е номер на някоя линия. Така по дадена линия се намират автобусите от тази линия, минаващи през някоя спирка, за ***О(1)*** време. Автобусите са сортирани по време на пристигане в спирката. Използвам ***unordered\_map*** за съхранението на върховете заради константното време за намиране на спирка по даден номер. Също така имам още един ***unordered\_map*** с ключ – низ и стойност номер на спирка. Така по дадено име на спирка също мога да достъпя съответната спирка за ***О(1)*** време, използвайки и двата ***unordered\_map***-а.

Ребро на графа е отсечката между 2 спирки през който минава един бус. Ребрата минаващи през 1 връх са точно автобусите минаващи през съответната му спирка. Така мога да отъждествя ребрата с автобусите които, както споменах по-горе, ги държа в ***unordered\_map<int, list<Bus>>***. В структурата Bus се съдържа информацията за номер на бус, време на тръгване/пристигане и посока на движение. През цялото време поддържам листа сортиран по време на тръгване на автобусите от някоя спирка.

За съхранението на транспортните линии използвам ***unordered\_map<int, TransportLine>.*** Структурата ***TransportLine*** съдържа номер на линията, брой спирки от които се състои и лист от номерата на тези спирки. Търсенето става по ключ – номер на някоя линия за ***О(1)*** време.

1. **Зареждане на данни**

Програмата прочита информацията за графа от текстови файл. При четенето от файла се следи за неправилно въведени символи *(пр. очаква се число, а се прочита буква)* и логически грешки (несвързаност, липса на информация, грешно въведени времена). Програмата съобщава и създава празния граф при намиране на грешка. Ако четенето от файла е успешно, чрез ***set\_busses\_times ()*** *(и неговите помощни функции)* едновременно се валидират и създават ребрата на графа (върховете директно се прочитат от файла). Тази функция взима последователно линиите и върви по спирките през които минава линията, свързвайки ги чрез информацията за минаващите автобуси на всяка спирка. При зареждане на програмата данните се прочитат от текстовия файл по подразбиране *„****Map.txt****“.* Съществуват опции за прочит от друг файл, запис във файла по подразбиране и запис в друг файл .

* 1. **Формат на текстовия файл**

В началото на файла първите 2 числа представляват съответно: брой спирки **(r)** и брой на линиите **(s)**. Следва **s** пъти номер на линията, брой спирки в линията, номерата на тези спирки. След това **r** пътипрочита номер на спирка, име, брой на бусовете минаващи през нея. За бусовете имаме съответно номер, време на пристигане и число и дестинация. Дестинацията е номер на спирка. Ако е ***-1***, то това означава, че това е последната спирка на този бус.

1. **Намиране на оптимален път**

За намирането на най-добрия път (от към време) използвам модифициран алг. на ***Дийкстра*** за най-къс път в граф, реализиран чрез минимална приоритетна опашка. Графът е възможно да съдържа отрицателни цикли, но това не пречи, т.к. ребрата образуващи отрицателни цикли не се разглеждат. Те биват *„елиминирани“* чрез проверката дали времето за достигане на дадена спирка е повече от времето на тръгване на някой автобус от тази спирка. На практика ако достигнем спирка ***А*** в ***10та*** минута, то не е възможно да хванем който и да е автобус минаващ през спирката преди ***10та*** минута. Това, че графът е мултиграф също не влияе на алгоритъма, т.к. се взема *„най-лекото“* ребро. Масивът на предшествениците от оригиналния алгоритъм е заменен с ***unordered\_map***, съдържащ както предишния връх от който идваме, така и времето в което пристигаме и номера на автобуса с който трябва да се движим. Ключът е номер на спирка. Също така ако има 2 различни пътя между 2 спирки, пристигащи в едно и също време, то алгоритъмът ще избере този, който минава през по-малко на брой спирки.

1. **Разширяване на пътната мрежа**
   1. **Удължаване на съществуваща линия**

Удължаването на линия се осъществява чрез функцията ***extend\_line,*** като е разрешено само добавяне на спирка отпред или отзад в поредицата. Също така не е разрешено повтарянето на спирка в дадена линия с изключение, че е възможно да свържат първата и последната спирка. Така линията ще стане циклична. Такава линия е невъзможно да бъде удължена. При опит за удължаване на линията с несъществуваща спирка, то системата автоматично създава спирката. От потребителя се очаква да въведе времето за започване на курса от новата спирка *(ако разширяваме отпред)* или времето за завършване на курса *(ако разширяваме отзад).* Възможностите за избор на време са строго ограничени *(повече в следващата подсекция)*.

* 1. **Добавяне на нов курс към линия**

Извършва се чрез функцията ***add\_new\_course.*** В началото потребителят въвежда време за начало на курса. То не трябва да съвпада с началото на друг курс от същата линия. Ако новият курс се явява ***р-ти*** по време, то автобусът пътуващ за следващата спирка от линията задължително пристига там не по-рано от ***р–1вия*** и не по-късно от ***р+1вия*** автобус от същата линия. Времето за пътуване между 2 спирки е поне ***1*** минута. Курс не може да започне след ***1439та*** минута (еквивалентно на ***23:59***). Също така курсът задължително завършва преди края на следващия ден (***2879***). Самото добавяне на курс между 2 спирки става чрез итериране през курсовете започващи от първата от двете спирки и добавянето на новия курс на ***р-та*** позиция в листа от курсовете на първата спирка за съответната линия.

* 1. **Добавяне на нова линия**

Добавянето на нова линия става чрез функцията ***add\_new\_line***. В началото потребителят въвежда номер на новата линия, брой спирки, които ще съдържа, и брой курсове. След това се въвеждат имената на спирките, като не се допуска вътрешен цикъл в линията *(възможен цикъл е само този при който началната спирка съвпада с последната)*. Ако потребителят въведе несъществуваща спирка, то системата автоматично създава нова спирка. След създаването на първото ребро *(между първите 2 спирки)* се използва ф-ята ***extend\_line*** до свързването на останалите спирки. Накрая се извиква ***add\_new\_course*** толкова пъти, колкото потребителят е заявил.

1. **Редуциране на пътната мрежа**
   1. **Премахване на спирка**

Спирки се премахват чрез ***remove\_bus\_stop().*** Изтриване на спирка става чрез премахването на всички курсове започващи или сочещи към тази спирка. Ако спирката не е крайна за някоя линия, то курсът сочещ към спирката се пренасочва към следващата в линията, като времето за достигането ѝ се запазва. Всяка линия, съдържаща вече премахнатата спирка, се намалява, като ако остава само една спирка, то линията бива премахната.

* 1. **Премахване на курс от линия**

Премахването на курс от линия става чрез ***remove\_course***. Системата показва съществуващите курсове за тази линия и дава на потребителя да избере кой курс да премахне. Ако линията съдържа само един курс, то цялата линия бива изтрита. В противен случай изтриването на 1 курс става чрез последователното вземане на всички спирки от линията и премахването на избрания курс, минаващ през текущата спирка.

* 1. **Премахване на линия**

Премахването на линия става чрез ***remove\_line***. Последователно се вземат всички спирки от линията и се изтриват всички курсове от тази линия, минаващи през текущата спирка. След това самата линия бива извадена от ***lines***.