МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТРАНСПОРТНИЙ УНІВЕРСИТЕТ

НАДВІРНЯНСЬКИЙ КОЛЕДЖ НТУ

КУРСОВИЙ ПРОЕКТ

з дисципліни “Алгоритмізація та програмування”

на тему: «Знаходження максимального потоку

за методом Форда−Фалкерсона»

Студент групи ІТ-21

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Пицюк В.М.

(підпис, дата)

Керівник викладач

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Грига В.М.

(підпис, дата)

Оцінка\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (ініціали та прізвище)

\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (ініціали та прізвище)

\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (ініціали та прізвище)

НАДВІРНА

2017

**Зміст**

Вступ

1. Аналітичний розділ
   1. Загальні поняття
   2. Історія
   3. Математична модель
2. Опис програмної частини
   1. Опис алгоритму
   2. Опис меню користувача
3. Тестування програми
   1. Системні вимоги програми
   2. Запуск програми

Висновок

Перелік використаних джерел

**Вступ**

В наш час iнформацiйнi технологiї займають одне з найважливiших мiсць

у всiх сферах нашого життя. Комп’ютери застосовуються скрiзь: в навчаннi, в

менеджментi, в торгiвлi, на виробництвi та в iнших видах дiяльностi людини.

Але функцiонування будь-якого комп’ютера неможливе без необхiдних про-

грам, а отже i алгоритмiв, на основi яких пишуться програми. Таким чином,

рiзноманiтнi алгоритми щодня допомагають людинi у рiзних сферах дiяльно-

стi. I деякi з них вiдiграють дуже важливу роль в розвитку людства. Отже,

питання алгоритмiзацiї є дуже актуальними й потребують багато уваги для

подальшої розробки алгоритмiв та вдосконалення вже iснуючих. Поряд iз

цим, не може залишатися осторонь, також й процес програмування, як один

з фундаментальних роздiлiв iнформатики.

Однiєю за важливих задач, вирiшення якої допомагає оптимiзувати

транспортування вантажiв, побудову нафто-, водо- та газопроводiв, проекту-

вання електромереж є задача пошуку максимального потоку мережi. Для

вирiшення якої часто використовується метод Форда-Фалкерсона. Алгоритм

реалiзацiї якого, за допомогою методу обходу в глибину, ми i розглянемо

бiльш детально.

**Об’єктом** дослiдження є потоки в транспортних мережах.

**Предметом** дослiдження є процес знаходження максимального потоку у

транспортнiй мережi за допомогою алгоритму Форда-Фалкерсона.

**Метою** курсової роботи є реалiзацiя алгоритму Форда-Фалкерсона, за допомогою методу обходу графа в глибину, мовою програмування C++.

1. **Аналітичний розділ**

**Загальні поняття**

В теорії оптимізації та теорії графів, задача про максимальний потік полягає у знаходженні такого потоку за транспортною мережею, щоб сума потоків з витоку, або, що означає те ж саме, сума потоків до стоку була максимальна.

Задача про максимальний потік є окремим випадком більш складних задач, таких, як, наприклад, задача про циркуляцію.

**Алгоритм або метод Форда-Фалкерсона** знаходить максимальний потік у транспортній мережі. Метод Форда-Фалкерсона - метод, який базується на трьох концепціях: залишкові мережі, шляхи що збільшуються і розрізи. Ключову роль у методі Форда-Фалкерсона грають два поняття: залишкові мережі і доповнюють шляху. Дані концепції лежать в основі важливої теореми про максимальний потік і мінімальний розріз, яка визначає значення максимального нащадка за допомогою розрізів траспортної мережі.

Метод Форда-Фалкерсона є ітеративним. Спочатку величині потоку присвоюється значення 0: *f(u, v) = 0* при будь-яких u , *v* Є *V*. На кожній ітерації величина потоку збільшується за допомогою пошуку «шляху, що збільшується» (тобто деякого шляху від джерела s до стоку t, уздовж якого можна послати більший потік) і подальшого збільшення потоку. Цей процес повторюється до тих пір, поки вже неможливо буде відшукати збільшуючий шлях.

**Історія**

Після вступу США у Другу світову війну у 1941 році математик **Джордж Бернард Данціг** почав працювати у відділі статистичного управління Військово-повітряних сил США у Вашингтоні. З 1941 по 1946 роки він очолював підрозділ аналізу військових дій (Combat Analysis Branch), де працював над різноманітними математичними проблемами. Згодом з використанням роботи Данцига задача про максимальний потік була вперше розв'язана у ході підготовки повітряного мосту під час Берлінського повітряного мосту, що відбувався у 1948–1949 роках.

У 1951 році **Джордж Данциг** вперше сформулював задачу у загальному вигляді.

У 1955 році **Лестер Форд** і **Делберт Фалкерсон** вперше побудували алгоритм, призначений для вирішення цього завдання. Їх алгоритм отримав назву **алгоритм Форда — Фалкерсона**.

Надалі рішення задачі багато разів поліпшувалося.

У 2010 році дослідники **Джонатан Келнер** (Jonathan Kelner) і **Олександр Мондри** (Aleksander Mądry) з МТІ разом зі своїми колегами **Даніелєм Спілманом** з Єльського університету і **Шень-Хуа Тенем** з університету Південної Каліфорнії продемонстрували чергове покращення алгоритму, вперше за 10 років.

**Матиматична модель**

Нехай *G(V, E)* граф, і для кожного ребра з u в v, нехай *c(u, v)* буде ємність і *f(u , v)* буде потік. Ми хочемо знайти максимальний потік від джерела s до раковини t. Після кожного кроку в алгоритмі виходить наступне:

* **Обмеженість потенціалу:** Потік уздовж краю не може перевищувати свій потенціал
* **Косі симетрії:** Чистий потік від u до v повинен бути протилежністю чистого припливу від *v* до *u*
* **Збереження потоку:** Тобто, якщо u не s або t. Чистий потік до вузла дорівнює нулю, для джерела, який "виробляє" потік, і раковину, яка "поглинає" потік
* **Значення (F):** Тобто, потік виходячи з s повинен бути рівним потоку, що надходить у *t*

Це означає, що потік через мережу є легальним потоком після кожного раунду в алгоритмі. Визначимо залишкову мережу *Gf(V, Ef)*, щоб бути в мережі з потужністю *cf(u, v)=c(u, v)−f(u, v)*, і без потоку. Зверніть увагу, що може статися, що потік від v до u дозволений в залишковій мережі, хоча заборонений в вихідній мережі: якщо *f(u, v)>0* та *c(v, u)=0*, тоді *cf(v, u) = c(v, u)−f(v, u)=f(u, v)>0*.

1. **Опис програмної частини**

**Опис алгоритму**

Iдея алгоритму полягає в наступному. Ми вибираємо такий шлях вiд

джерела до стоку, щоб для кожного ребра залишкова пропускна здатнiсть

була строго бiльше нуля. При цьому ребра на даному шляху можуть прохо-

дитися як у прямому, так i в зворотному напрямку. Вибираємо мiнiмальне

значення серед залишкових пропускних спроможностей ребер даного шляху.

Збiльшуємо потiк на кожному з ребер даного шляху на обране мiнiмаль-

не значення. Далi шукаємо наступний аналогiчний шлях. Робота алгоритму

продовжується до тих пiр, поки вдається знаходити данi шляхи. Вiдразу вiд-

значимо, що даний алгоритм вiдноситься до класу недетермiнованих, тобто

кожен наступний крок алгоритму визначено неоднозначно. I час роботи (кiль-

кiсть крокiв) алгоритму залежить вiд того, як будуть вибиратися кроки.

Алгоритм Форда-фалкерсона:

1. Прирiвнюємо до нуля всi потоки. *∀e(vi, vj)∈E f(e) = 0* . Залишкова мережа спочатку збiгається з вихiдною мережею;
2. У залишкової мережi знаходимо будь-який шлях з джерела s у стiк t . Дуги якого задовольняють умовi *f(vi, vj) <= (vi, vj)* . Якщо такого шляху немає, то потiк у мережi максимальний;
3. Пускаємо через знайдений шлях (вiн називається збiльшувальним шля- хом) максимально можливий потiк;
4. На знайденому шляху в залишковiй мережi шукаємо ребро з мiнiмаль- ною пропускною здатнiстю *Cmin*;
5. Для кожного ребра на знайденому шляху збiльшуємо потiк на *Cmin* , а в протилежному йому — зменшуємо на *Cmin*;
6. Модифiкуємо залишкову мережу. Для всiх ребер на знайденому шляху, а також для протилежних їм ребер, обчислюємо нову пропускну здатнiсть. Якщо нова пропускна здатнiсть не дорiвнює нулю, додаємо ребро до залишкової мережi, а якщо дорiвнює нулю, стираємо його;
7. Повертаємося на крок 2.

Важливо те, що алгоритм не конкретизує, який саме шлях ми шукаємо на кроцi 2 або як ми це робимо. З цiєї причини алгоритм гарантовано сходиться тiльки для цiлих пропускних спроможностей, але навiть для них при великих значеннях пропускних спроможностей вiн може працювати дуже довго або зовсiм не привести до оптимального рiшення.

**Опис меню користувача**

// TODO

1. **Тестування програми**

**Системні вимоги**

Будь-яка машина, на яку є компілятор C++, для коректної роботи на компіляторі Visual Studio і ОС Windows треба розкоментувати деякі строки (вказано в коментарях).

**Запуск програми**



Скріншот виконання програми

**Кот**