програмне забезпечення, зберігаються файли. Мережі, в своїй основі є дискретними структурами. Google Maps використовує дискретну математику для визначення найшвидших маршрутів та часу руху. Шифрування та дешифрування є частиною криптографії, яка є частиною дискретної математики. Наприклад, безпечні покупки в Інтернеті використовують криптографію з відкритим ключем. Розробка критеріїв паролів - це проблема підрахунку: чи достатньо великий простір паролів, щоб хакер не зміг зламати облікові записи, просто перебравши всі можливі варіанти? Наскільки довгими повинні бути паролі, щоб протистояти таким атакам?

Комп'ютерна графіка (наприклад, у відеоіграх) використовує лінійну алгебру для перетворення (переміщення, масштабування, зміни перспективи) об'єктів. Це справедливо як для таких додатків, як розробка ігор, так і для операційних систем. Компакт-диски зберігають багато даних, які кодуються за допомогою модифікованого коду Ріда-Соломона (двійковий код, а отже, дискретна математика) для автоматичного виправлення помилок передачі. Цифрова обробка зображень використовує дискретну математику для об'єднання зображень або застосування фільтрів. Теорія Рафа використовується в кібербезпеці для виявлення зламаних або злочинних серверів і в цілому для забезпечення мережевої безпеки. Лінійна алгебра - це дискретна математика, яка використовується для компресійного зондування (ефективного запису зображення/звуку) та медичної візуалізації. Стиснення даних, зменшення шуму в даних та автоматична рекомендація фільмів - все це використовує один і той самий інструмент лінійної алгебри.

Отже, дискретна математика активно використовується в сфері програмування, і без неї буде туго писати код. В залежності від області програмування – варіюється і потрібність різних математичних галузей. Але якщо брати курс на розробку ігор, роботу з базами даних, або програмування на мовах низького рівня – то без неї ну взагалі ніяк.

Список літературних джерел

1. *Програмування та математика! URL:* http://www.itschool.vn.ua/programming-math/
2. *Discrete Mathematics in the Real World URL:* http://www.mathily.org/dm-rw.html

УДК 004.01

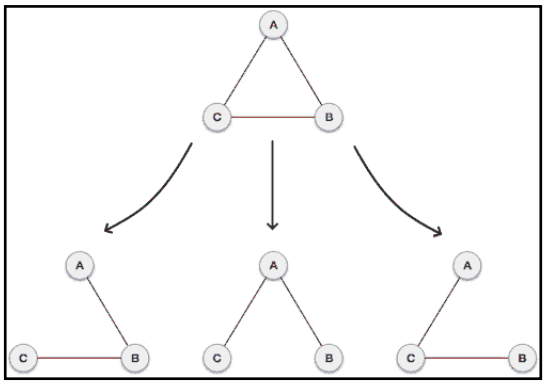
Зимич А. П., студент 1 курсу
спеціальності 122 «Комп'ютерні науки»
Ніколюк П. К., д-р фіз.-мат. наук,
Професор кафедри комп'ютерних наук

АЛГОРИТМИ ПОШУКУ МІНІМАЛЬНОГО ОСТОВНОГО ДЕРЕВА

Донецький національний університет імені Василя Стуса, м. Вінниця

Поняття остовного дерева

Остовне дерево — це підмножина деякого графа G, у якому всі вершини охоплені мінімально можливою кількістю ребер. Отже, остовне дерево не має циклів і його не можна від'єднати.



Малюнок 1. Остовні дерева(знизу) які можна отримати з графу G(зверху)

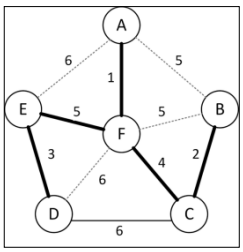
Один граф може мати більше одного остовного дерева. Нижче наведено кілька властивостей остовного дерева, пов'язаного з графом G:

- Зв'язний граф G може мати більше одного остовного дерева.
- Усі можливі остовні дерева графа G мають однакову кількість ребер і вершин.
- Остовне дерево не має жодного циклу (циклів).
- Видалення одного ребра з остовного дерева зробить граф роз'єднаним, тобто остовне дерево буде мінімально зв'язним.
- Додавання одного ребра до остовного дерева створить схему або петлю, тобто остовне дерево є максимально ациклічним.

Поняття мінімального остовного дерева (MST)

Для зважених графів існує поняття ваги остовного дерева, яке визначено як сума ваги всіх ребер, що входять до остовного дерева. З цього випливає поняття мінімального остовного дерева.

Мінімальне остовне дерево графа G - це його ациклічний зв'язний подграф, в який входять всі його вершини, що володіє мінімальною вагою ребер.



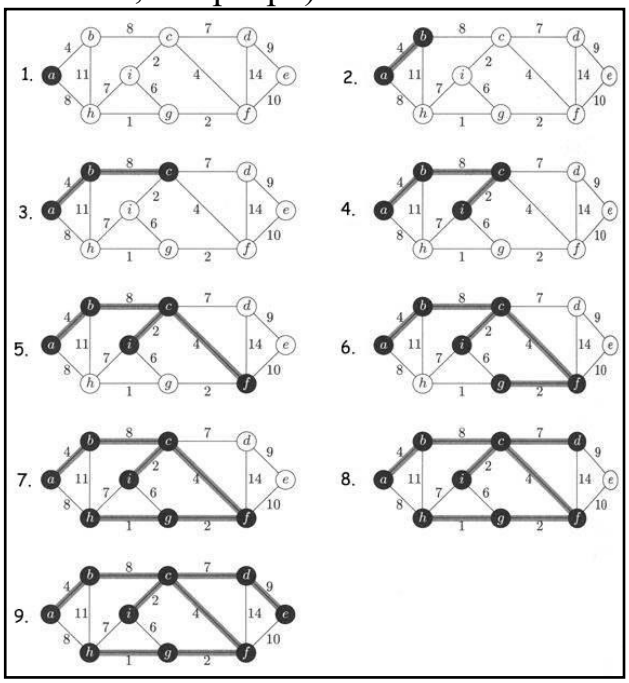
Малюнок 2. Приклад мінімального остовного дерева

Алгоритм Прима

Цей алгоритм названо на честь американського математика Роберта Пріма, який відкрив цей алгоритм у 1957 р. Втім, ще у 1930 р. цей алгоритм був відкритий чеським математиком Войтеком Ярником.

Шукане мінімальне остовне дерево будується поступово, додаванням до нього ребер по одному. Спочатку дерево складається з єдиної вершини (її

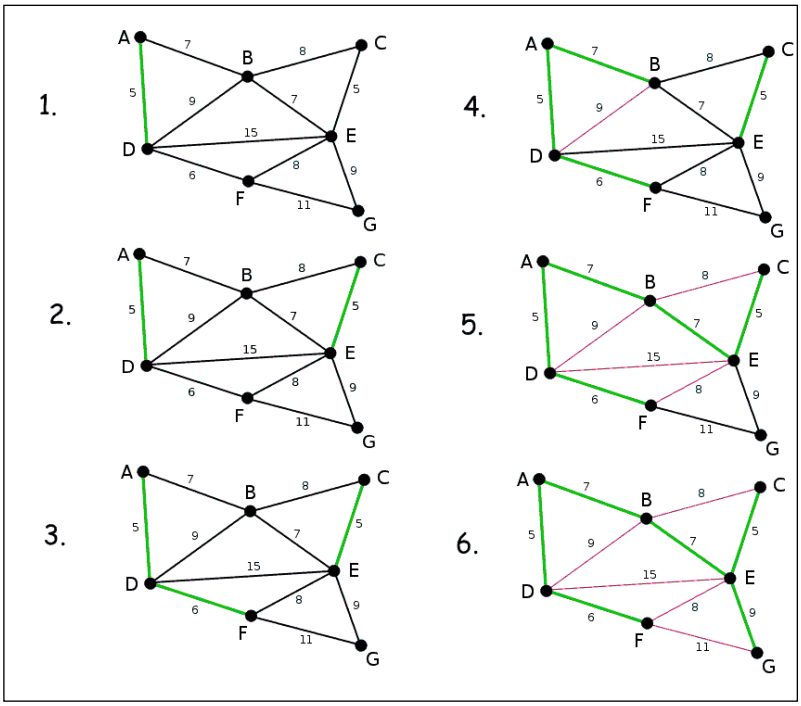
можна вибрати довільно). Потім вибирається ребро мінімальної ваги, що виходить з цієї вершини, і додається до дерева. Після цього воно містить вже дві вершини, і тепер шукається і додається ребро мінімальної ваги, що має один кінець в одній із двох обраних вершин, а інший - навпаки, у всіх інших, крім цих двох. Тобто, кожного разу шукається мінімальне за вагою ребро, один кінець якого - вже взята в дерево вершина, а інший кінець - ще не взята, і це ребро додається в дерево (якщо таких ребер кілька, можна взяти будь-яке). Цей процес повторюється до тих пір, поки остовне дерево не міститиме всі вершини (або, що те ж саме, n-1 ребро)



Малюнок 3. Приклад роботи алгоритма Прима

Алгоритм Краскала

На вході є порожній підграф, який і будемо добудовувати до потенційного мінімального оставного дерева. Спочатку ми робимо сортування ребер по зменшенню за їх вагами. Додаємо і-е ребро до нашого підграфа лише в тому випадку, якщо дане ребро з'єднує дві різні компоненти зв'язності, не утворюючі циклу. Алгоритм завершить свою роботу після того, як безліч вершин нашого підграфа збігатиметься з безліччю вершин вихідного графа.



Малюнок 4. Приклад роботи алгоритма Красакла

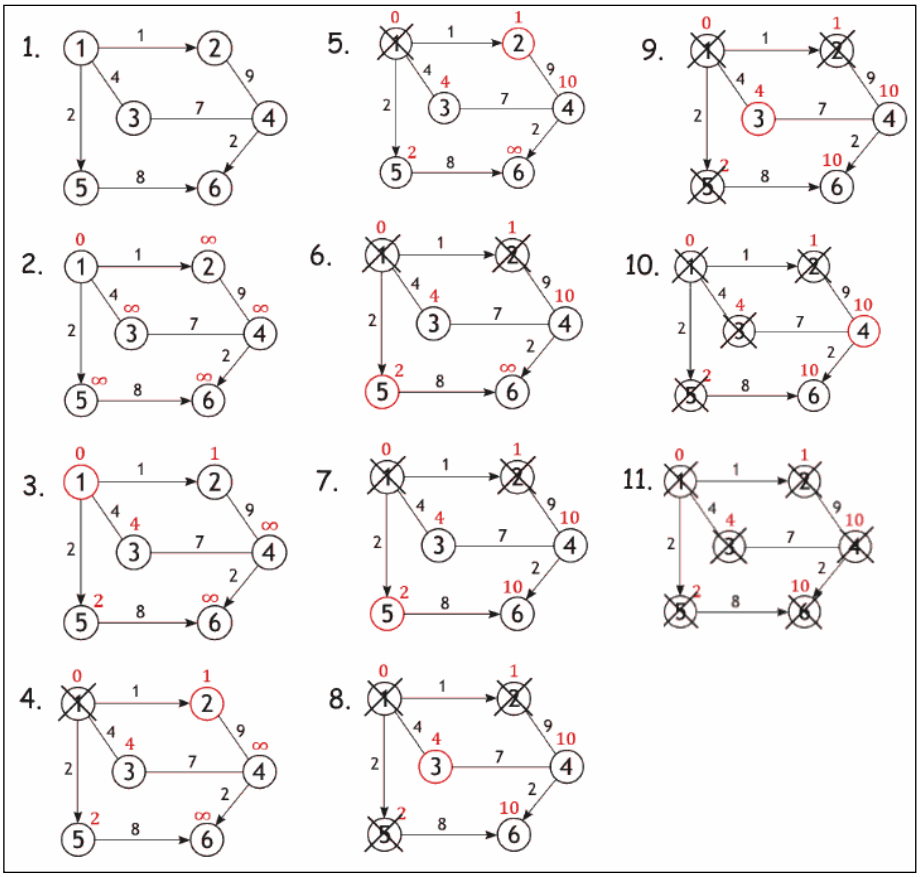
Алгоритм Дейкстри

У 1956 році у голландського програміста Едсгера В. Дейкстри виникло практичне завдання знайти найкоротший шлях з Роттердама в Гронінген. Дейкстра вчинив, як інформатик: він абстрагувався від проблеми, відкинувши такі дрібниці, як подорож із міста А до міста Б. Він сформулював більш загальну проблему пошуку за графом.

Алгоритм можна описати наступним чином:

1. Встановлюємо будь-який вузел як початковий.
2. Встановлюємо відстані між початковим вузлом та всіма іншими вузлами в ∞ (нескінченність), за винятком відстані самим з собою, яке приймаємо рівним 0.
3. Після цього ітеративно виконуємо наступні кроки:

    1) Вибираємо вузол з найменшим значенням як «поточний вузл» і відвідуємо всіх сусідів. При відвідуванні кожного сусіда оновлюємо їхню орієнтовну відстань від початкового вузла.
    2) Як тільки ми відвідаємо всіх сусідів поточного вузла та оновили їх відстані, помічаємо поточний вузол як visited (відвіданий). Позначка вузла як «відвіданого» означає, що знайдено його остаточну цінність.
    3) Повернемося до першого кроку і повторюємо доти, доки відвідаємо всі вузли графа.



Малюнок 5. Приклад роботи алгоритма Дейкстри

Головний недолік алгоритму в тому, що його не можна використовувати для пошуку найкоротшої відстані у графах із негативними ребрами. Стратегія алгоритму завжди вибирає негативне число як меншу відстань.

Список літературних джерел

1. *Data Structure & Algorithms - Spanning Tree URL:* https://www.tutorialspoint.com/data_structures_algorithms/spanning_tree.htm
2. *Алгоритм Краскала URL:* https://www.wikiwand.com/ru/Алгоритм_Краскала
3. https://kvodo.ru/dijkstra-algorithm.htm

УДК 004.01

Суліма В.К., студент 1 курсу
спеціальності 122 «Комп'ютерні науки»
Ніколюк П.К., професор
кафедри інформаційних технологій

ПАРАДОКС ДНІВ НАРОДЖЕННЯ ТА ЙОГО ЗАСТОСУВАННЯ

Донецький національний університет імені Василя Стуса, м. Вінниця

Парадокс днів народження – твердження, що у теорії ймовірностей, оцінює шанс того, що у випадково зібраній групі осіб, хоча б в одної пари