*МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ*

*КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ*

*ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО*

*НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕЛЕКТРИЧНОЇ ІНЖЕНЕРІЇ*

*ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ*

*КАФЕДРА АВТОМАТИЗАЦІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ*

*НАВЧАЛЬНА ДИСЦИПЛІНА*

*Алгоритми та методи обчислень*

*ЗВІТ*

*З ПРАКТИЧНИХ РОБІТ*

*Виконав:*

*студент групи КН-23-1*

*Гур’єв Д.П*

*Кременчук 2024*

*Практична робота № 8*

Тема. Жадібні алгоритми. Наближене розв’язання екстремальних задач

Мета: набути практичних навичок застосування деяких жадібних алгоритмів для розв’язання екстремальних задач.

***Практична частина 8-варіант***

1. Розв’язати задачу комівояжера для графа, заданого варіантом, використовуючи код, наведений вище.

2. Візуалізувати граф.

3. Обґрунтувати асимптотику для обох алгоритмів, неведену в табл. 1.4.

**Заданий зважений граф:** [(1,3,5), (1,4,8), (1,5,7), (2,3,4), (2,4,6), (2,5,8), (3,4,6), (3,5,5), (4,5,4)]

Ребро (1, 3) з вагою 5

Ребро (1, 4) з вагою 8

Ребро (1, 5) з вагою 7

Ребро (2, 3) з вагою 4

Ребро (2, 4) з вагою 6

Ребро (2, 5) з вагою 8

Ребро (3, 4) з вагою 6

Ребро (3, 5) з вагою 5

Ребро (4, 5) з вагою 4

1 -- 5-- 3

| \ |

| 8 4

| \ |

7 6 |

| \ |

2 --8-- 5

Алгоритм грубої сили є прийнятним для малих наборів даних або коли простота реалізації є ключовим фактором. Алгоритм розділяй і володарюй набагато ефективніший для великих наборів даних завдяки своїй кращій часовій складності, але складніший у реалізації.

Контрольні запитання

1. Що таке жадібний алгоритм?

2. Які головні принципи роботи жадібних алгоритмів?

3. Яка головна відмінність між жадібними алгоритмами та динамічним програмуванням?

4. Наведіть приклади задач, які можна розв’язати за допомогою жадібних алгоритмів.

5. Які можуть бути обмеження у використанні жадібних алгоритмів для розв'язання екстремальних задач?

6. Чому жадібні алгоритми часто використовуються для наближеного розв’язання екстремальних задач?

1. Що таке жадібний алгоритм?

Жадібний алгоритм (greedy algorithm) — це алгоритм, який робить локально оптимальний вибір на кожному кроці з надією, що ці локально оптимальні рішення приведуть до глобально оптимального розв'язку всієї задачі. Іншими словами, жадібний алгоритм вибирає на кожному кроці найкращий варіант з точки зору поточного стану задачі, не замислюючись про майбутні наслідки.

2. Які головні принципи роботи жадібних алгоритмів?

Головні принципи роботи жадібних алгоритмів включають:

Локальна оптимальність: На кожному кроці вибирається найкращий варіант, який видається найкращим в даний момент.

Жадібний вибір: Кожний вибір робиться без урахування попередніх виборів чи майбутніх наслідків, лише з урахуванням поточного стану.

Незмінність властивості: Прийняте рішення не впливає на оптимальність майбутніх рішень. Це означає, що після вибору, задача зменшується до подібної задачі меншого розміру.

3. Яка головна відмінність між жадібними алгоритмами та динамічним програмуванням?

Жадібні алгоритми:

Роблять локально оптимальний вибір на кожному кроці.

Не аналізують всі можливі комбінації виборів.

Простіше реалізуються.

Працюють швидше для деяких задач.

Динамічне програмування:

Розбивають задачу на підзадачі та зберігають результати для уникнення повторних обчислень.

Аналізують всі можливі комбінації виборів для знаходження оптимального рішення.

Потребують більше пам'яті.

Підходять для більш складних задач з оптимальним рішенням.

4. Наведіть приклади задач, які можна розв’язати за допомогою жадібних алгоритмів.

Деякі з задач, які можуть бути ефективно вирішені жадібними алгоритмами, включають:

Задача про здачу мінімальною кількістю монет: Вибір найкрупніших можливих монет на кожному кроці.

Задача про рюкзак (0/1 knapsack problem) з дробовими речами (Fractional knapsack problem): Вибір предметів з найвищим співвідношенням вартості до ваги.

Задача про покриття відрізками: Вибір мінімальної кількості відрізків, що покривають заданий інтервал.

Задача про вибір активностей (Activity selection problem): Вибір максимального набору непересічних активностей на основі їх часу завершення.

Побудова оптимальних шляхів (найкоротший шлях у графах): Наприклад, алгоритм Дейкстри для знаходження найкоротших шляхів у графі з не негативними вагами ребер.

5. Які можуть бути обмеження у використанні жадібних алгоритмів для розв'язання екстремальних задач?

Обмеження жадібних алгоритмів включають:

Не завжди гарантують глобальну оптимальність: Жадібний підхід не завжди приводить до глобально оптимального рішення, оскільки він не враховує всіх можливих комбінацій.

Вимоги до властивостей задачі: Жадібні алгоритми працюють ефективно лише для задач, які мають властивість оптимальної підструктури та жадібної вибірковості.

Потенційна неефективність: Для деяких задач жадібний підхід може призвести до неефективного рішення порівняно з іншими методами, такими як динамічне програмування чи повний перебір.

6. Чому жадібні алгоритми часто використовуються для наближеного розв’язання екстремальних задач?

Жадібні алгоритми часто використовуються для наближеного розв'язання екстремальних задач через наступні причини:

Простота реалізації: Жадібні алгоритми зазвичай простіші в реалізації порівняно з більш складними алгоритмами, такими як динамічне програмування.

Швидкість: Жадібні алгоритми мають меншу обчислювальну складність, що робить їх придатними для великих наборів даних або коли час виконання є критичним фактором.

Практична ефективність: У багатьох реальних задачах жадібні алгоритми забезпечують досить близькі до оптимальних рішення, навіть якщо теоретично вони не гарантують оптимальності.

Проміжні рішення: Жадібні алгоритми можуть бути використані як початковий етап для більш складних алгоритмів або для швидкого отримання приблизного рішення, яке потім може бути поліпшене іншими методами.