

Т е м а т и к а т а з м і с т л е к ц і й

Змістовий модуль 1

Метрична теорія алгоритмів (аналіз алгоритмів)

1. Позначення O , o , Ω , ω , θ
(Література: [1, розд. 2.1–2.2], [2, розд. 3.1])
 - (а) Приклад: оцінка кількості дій алгоритму сортування вибором
2. Верхня та нижня оцінки складності задачі
(Література: [1, розд. 2.1, 10.1], [2, розд. 8.1])
 - (а) Означення
 - (б) Тривіальна нижня оцінка
 - (с) Доведення нижньої оцінки кількості порівнянь $\Omega(n \log n)$ для задачі сортування
 - (д) «Парадокс» сортування підрахунком та його пояснення
 - (е) Доведення нижньої оцінки шляхом зведення відомої задачі до досліджуваної (на прикладі побудови опуклої оболонки)

Змістовий модуль 2

Структури даних

1. Древа пошуку
(Література: [2, розд. 12], [1, розд. 1.4])
 - (а) Оцінка часу роботи всіх операцій на дереві пошуку
 - (б) Ідея розширення структур даних. Модифікація, що додає до операцій на дереві пошуку операцію «знайти за номером»
(Література: [2, розд. 14.1])
 - (с) Використання розширень дерев пошуку для балансування (на прикладі АВЛ-дерев)
(Література: [1, розд. 6.3])
2. STL-шаблони `set`, `multiset`, `map` та `multimap`
(Література: [4, розд. 6], [5])
3. Хеш-функції (hash)
(Література: [1, розд. 7.3], [2, розд. 11])
 - (а) Реалізація масивом списків або дерев
 - (б) Уникнення колізій шляхом подвійного хешування
4. АТД піраміда (heap)
(Література: [2, розд. 6])
 - (а) Означення та оцінка часу роботи всіх операцій
 - (б) Техніка реалізації у масиві
 - (с) STL-шаблон `priority_queue`
 - (д) Найпростіше застосування до сортування
 - (е) Застосування у алгоритмі Дейкстри
(Література: [2, розд. 24.3])
5. АТД для подання сукупності множин, що не перетинаються (disjoint sets)
(Література: [2, розд. 21])
 - (а) Реалізація реверсними деревами

- (b) Застосування у алгоритмі Краскала

Змістовий модуль 3

Методи побудови алгоритмів

1. Жадібні алгоритми
 - (a) Необхідність обґрунтувань жадібних алгоритмів. Задача про рюкзак
 - i. доведення правильності для неперервного випадку
 - ii. контрприклад до жадібного алгоритму для дискретного випадку
 - (b) Доведення коректності алгоритму Краскала
(Література: [1, розд. 9.1–9.2])
 - (c) Означення матроїда. Односторонність слідування «матроїд $\Rightarrow$ коректність жадібного алгоритму»
(Література: [2, розд. 16.4], [13, розд. 5])
Інші приклади доведення коректності жадібних алгоритмів
(Література: [2, розд. 16.1–16.3], [15, розд. 12.1])
2. Поступове зменшення розміру задачі
(Література: [1, розд. 5])
 - (a) Поняття інваріанту циклу
(Література: [19, розд. 1.1], [2, розд. 2.1])
 - i. Інваріанти алгоритмів сортування вибором та простими вставками
 - (b) Однопрохідні та k -прохідні алгоритми
3. Метод декомпозиції («поділяй і володарюй», «divide and conquer»)
(Література: [1, розд. 4], [2, розд. 2.3.1–2.3.2])
4. Динамічне програмування
(Література: [2, розд. 15], [15, розд. 13], [16])
 - (a) Найпростіші приклади
 - i. Стрибки по платформах
 - ii. Оптимальний шлях «праворуч-або-вниз»
 - (b) Умови застосовності та доцільності
 - (c) Ітераційна реалізація та рекурсія з запам'ятовуваннями (memoized recursion);
варіанти з використанням масивів та з використанням `map`)
 - (d) Псевдополіноміальні алгоритми
5. Перебірні методи
 - (a) Рекурсивна реалізація перебору
 - (b) Відтинання гарантовано недопустимих наборів (backtracking)
(Література: [1, розд. 11.1])
 - (c) Метод гілок та меж
(Література: [1, розд. 11.2])
 - i. Верхні та нижні оцінки
 - ii. Використання нижніх (для задач мінімізації) оцінок для відкидання варіантів
 - iii. Використання оцінок для вибору порядку розгляду варіантів

Т е м а т и к а т а з м і с т

л а б о р а т о р н и х з а н я т ь

Змістовий модуль 1

Метрична теорія алгоритмів (аналіз алгоритмів)

1. Приклади знаходження оцінок кількостей дій
 - (a) Алгоритм Флойда
 - (b) Пошук ушир при поданні графа матрицею суміжності
 - (c) Два алгоритми перетину множин (brute force і злиття)
 - (d) Два алгоритми видалення з масиву елементів (вилучення по одному, зсуваючи хвіст, та формування послідовності заново, пропускаючи непотрібні)
2. Груповий та амортизаційний аналіз
 - (a) Пошук ушир при поданні графа списками суміжності
 - (b) «Два вказівника» (у варіанті назустріч)
 - (c) Опукла оболонка обходом Грехема

Змістовий модуль 2

Структури даних

1. Древа пошуку — як додаткове бонусне завдання
 - (a) Реалізація дерев пошуку (вставка, пошук, вилучення) — як додаткове бонусне завдання
 - (b) Реалізація модифікації дерев пошуку, що ефективно реалізує операцію «знайти за номером» (вставка, пошук за значенням, пошук за номером, вилучення за значенням, вилучення за номером) — як додаткове бонусне завдання
 - (c) Реалізація АВЛ-дерев (вставка, пошук) — як додаткове бонусне завдання
2. STL-шаблони `set`, `multiset`, `map` та `multimap`
 - (a) Використання АТД `set` для вилучення з послідовності повторень
3. Подання графів
 - (a) Як `vector<list<int> >`
 - (b) Подання (мульти)графів як `vector<multiset<int> >`.
 - i. Застосування до побудови ейлерового шляху в неорієнтованому мультиграфі — як додаткове бонусне завдання
 - (c) Розгляд випадку, коли вершини задані не номерами, а назвами
 - i. Використання `map<T, int>` та `vector<T>` для перетворень між назвами та номерами при введенні/виведенні
 - ii. Безпосереднє подання графів як `map<T, set<T> >`. Реалізація алгоритму пошуку вшир для даного способу подання — як додаткове бонусне завдання
4. Хеш-функції (hash)
5. АТД піраміда (heap)
 - (a) STL-реалізація, застосування як черга з пріоритетами
 - (b) Власна реалізація, застосування як черга з пріоритетами
 - (c) Реалізація алгоритму Дейкстри з оцінкою $O(m \log n)$
 - (d) Застосування у задачі «Лінія горизонту» — як додаткове бонусне завдання
6. АТД для подання сукупності множин, що не перетинаються (disjoint sets)
 - (a) Реалізація реверсними деревами

- (b) Застосування у алгоритмі Краскала

Змістовий модуль 3

Методи побудови алгоритмів

1. Жадібні алгоритми
 - (a) Розклад для задачі про час друку + час доставки
 - (b) Задача про трикутник максимальної площі
 - i. Неправильність ідеї «взяти 3 найдовші відрізки»
 - ii. Неправильність ідеї «взяти 3 найдовші відрізки, з яких можна утворити трикутник»
 - iii. Доведення правильності ідеї «перебирати лише послідовні в порядку спадання трійки відрізків»
 - (c) Задача про розмін монет
 - i. Контрприклад до жадібного алгоритму при довільних номіналах
 - ii. Доведення при номіналах $1, a, a^2, a^3, \dots, a^k$.
 - iii. Доведення при номіналах українських монет — як додаткове бонусне завдання
 - iv. Доведення варіанту жадібного алгоритму для номіналів $1, 10, 25, 100, 1000, 2500, 10000, 100000, 250000$ — як додаткове бонусне завдання
 - v. Доведення при номіналах радянських монет — як додаткове бонусне завдання
2. Поступове зменшення розміру задачі. Поняття інваріанту.
 - (a) Інваріант у рекурсивній реалізації індійського алгоритму піднесення до степеню
3. Метод декомпозиції («поділяй і володарюй», «divide and conquer»)
 - (a) Сортування злиттям
 - (b) Сортування Хоара (quicksort)
 - (c) Алгоритм Карацуби множення «довгих» чисел
4. Динамічне програмування
 - (a) Однопараметрове динамічне програмування
 - i. Задача про з'єднання цвяхів
 - ii. Розклад роботи конвеєра — як додаткове бонусне завдання
 - iii. Оптимальна купівля квитків
 - iv. Розбиття абзаца на рядки
 - v. Максимізація сумарного об'єму вкладених коробок — як додаткове бонусне завдання
 - (b) Двопараметрове різнотипне динамічне програмування
 - i. Оптимальне розбиття алфавіту
 - ii. Оптимальне розбиття роману на томи — як додаткове бонусне завдання
 - iii. Мінімізація кількості монет (псевдополіноміальний алгоритм)
 - iv. Задача про рюкзак (псевдополіноміальний алгоритм) — як додаткове бонусне завдання
 - (c) Двопараметрове діапазонне динамічне програмування — як додаткове бонусне завдання
 - i. Оптимальна послідовність множення матриць — як додаткове бонусне завдання
 - ii. Оптимальне двійкове дерево пошуку — як додаткове бонусне завдання
5. Перебірні методи
 - (a) Рекурсивна реалізація перебору

- i. Генерація розміщень
- ii. Генерація перестановок
- iii. Генерація k -елементних підмножин — як додаткове бонусне завдання
- iv. Генерація усіх розбиттів на суму — як додаткове бонусне завдання
- (b) Відтинання гарантовано недопустимих наборів (backtracking)
 - i. Перевірка існування гамільтонового циклу
 - ii. Евклідова задача комівояжера
 - iii. Генерація підмножин додатніх чисел з потрібною сумою — як додаткове бонусне завдання
 - iv. Перевірка ізоморфності графів — як додаткове бонусне завдання
- (c) Верхні та нижні оцінки. Метод гілок та меж
 - i. Евклідова задача комівояжера
 - ii. Задача про рюкзак — як додаткове бонусне завдання
 - iii. Розподіл даних між рівнями пам'яті — як додаткове бонусне завдання
 - iv. Мінімізація кількості монет — як додаткове бонусне завдання
 - v. Триетапна задача фарбувальника — як додаткове бонусне завдання
 - vi. Остовне дерево найменшого діаметру — як додаткове бонусне завдання
 - vii. Узагальнення задачі ОДМВ на випадок орграфів — як додаткове бонусне завдання

С а м о с т і й н а р о б о т а с т у д е н т і в

Частина самостійної роботи студентів припадає на виконання завдань лабораторних робіт та іншу домашню підготовку.

Крім того, для студентів, які добре опанували мінімально необхідний рівень і бажають продовжувати вивчення даної дисципліни на поглибленому рівні, один з напрямів самостійної роботи полягатиме у відшукуванні (в літературі, на Internet-сайтах, присвячених алгоритмічному програмуванню та олімпіадним задачам, тощо) задач, для науково-обґрунтованого розв'язання яких потрібні засоби даного курсу. Тематика такого пошуку та аналізу може включати:

1. Знаходження нижніх та верхніх оцінок складності цих задач
2. Аналіз (доведення або спростування) NP-повноти цих задач
3. Аналіз (доведення або спростування) застосовності жадібних алгоритмів до цих задач

Враховуючи, що тематика даної дисципліни має дуже значний перетин з тематикою алгоритмічних олімпіад (зокрема, тих, які підтримує Association for Computing Machinery), справедливо вважати більшість видів діяльності, пов'язаної з тренуваннями олімпіадного напрямку, самостійною роботою з даної дисципліни.

Планується використовувати наступні форми самостійної роботи, пов'язані з тренуваннями олімпіадного напрямку:

1. Розв'язування задач, зарані відібраних і класифікованих викладачем за алгоритмом або структурою даних, які там потрібно застосувати.
2. Розв'язування задач невідомої зарані тематики.

3. Проведення студентами самостійної класифікації задач (з сайтів, де набір задач зберігається у некласифікованому вигляді), та обґрунтування цієї класифікації.

Використання олімпіадних ресурсів та систем автоматичної перевірки програм для самостійної роботи студентів поки що не набуло широкого поширення, але є публікації досить авторитетних педагогів, що обґрунтовують доцільність такого підходу. Серед них: Пасіхов Юрій Якович, заслужений учитель України, керівник центру дистанційної підтримки традиційних форм навчання при ФМГ № 17 м. Вінниці; Матюхін Віктор Александрович, голова наукового комітету Всеросійської олімпіади школярів з інформатики, вчитель Московської гімназії на Юго-Западе № 1543, та інші.

Р е к о м е н д о в а н а л і т е р а т у р а

1 Список літератури

Основна література

1. Левитин А., «Алгоритмы. Введение в разработку и анализ», М.–СПБ–К., Вильямс, 2006.
2. Кормен Т., Лейзерсон Ч., Ривест Р, Штайн К. «Алгоритмы: построение и анализ» (второе издание) М.–СПБ–К., Вильямс, 2005.
3. Вирт Н., «Алгоритмы и структуры данных», СПб, Невский диалект, 2005.
4. Джосьютис Н., «С++. Стандартная библиотека. Для профессионалов» СПб, Питер, 2004.
5. Королёв Д., «Введение в STL» (статья).
6. <http://informatics.mccme.ru>

Додаткова література

(Список впорядкований за алфавітом.)

7. Ахо А., Хопкрофт Дж., Ульман Дж. «Построение и анализ вычислительных алгоритмов» М., Мир, 1979.
8. Ахо А., Хопкрофт Дж., Ульман Дж. «Структуры данных и алгоритмы» М.–СПБ–К., Вильямс, 2001.
9. Бондарев В. М., Рублинецкий В. И., Качко Е. Г. «Основы программирования» Хрк., Фолио, Р.-н-Д., Феникс, 1998.
10. Гудман С., Хидетниеми С. «Введение в разработку и анализ алгоритмов» М., Мир, 1981.
11. Гэри М., Джонсон Д., «Вычислительные машины и труднорешаемые задачи» М., Мир, 1982.
12. Ласло М., «Вычислительная геометрия и компьютерная графика на С++» М., Випом, 1997.
13. Липски В. «Комбинаторика для программистов» М., Мир, 1988.
14. Мозговой М. «Занимательное программирование. Самоучитель» СПб, Питер, 2004.
15. Порублёв И. Н., Ставровский А. Б., «Алгоритмы и программы. Решение олимпиадных задач» М.–СПБ–К., Диалектика, 2007.
16. И. Н. Порублёв, Р. В. Черненко, К. В. Регент, И. В. Радченко, Е. Ю. Полищук. Конспект Ильи Порублёва и команды ЧНУ. Динамическое программирование и другие методы решения оптимизационных задач. // Сборник "Зимняя школа по программированию Харьков, ХНУРЭ, 2010, стр. 173-221

17. Рейнгольд Э., Нивергельт Ю., Део Н. «Комбинаторные алгоритмы. Теория и практика» М., Мир, 1980.
18. Седжвик Р., «Фундаментальные алгоритмы на C++» М.-СПб.-К., DiaSoft, 2002.
19. Шень А., «Программирование. Теоремы и задачи» М., МЦНМО, 2007.
20. <http://acm.dvpion.ru>
21. <http://acm.lviv.ua>
22. <http://acm.timus.ru>
23. <http://algolist.manual.ru>
24. <http://neerc.ifmo.ru>
25. <http://uva.onlinejudge.org>
26. <http://www.olymp.vinnica.ua>
27. <http://www.topcoder.com/tc>