Конспект по ДАА, лято 2013

1. Алгоритми – същност и неформално определение. Изчислителни задачи. Големина на входа на алгоритми. Сложност по време в най-лошия случай, средна и в най-добрия случай. Други аспекти на анализирането на алгоритмите – сложност по памет и коректност.

2. Асимптотичен анализ на сложността – важност, предимства и недостатъци. Нотации, използвани при асимптотичния анализ: $O, o, \Theta, \Omega, \omega$. Свойства на нотациите.

3. Сортиране – приложения, важност, примери за използване на сортирането като първа фаза от решението на други изчислителни проблеми. Стабилни и нестабилни сортиращи алгоритми. Важност на стабилността.

4. Елементарни сортиращи алгоритми: Selection Sort и Insertion Sort. Въвеждане на понятието инвариантa на цикъла. Анализ на коректността на двата алгоритъма чрез инварианта на цикъла. Анализ на сложността по им по време и по памет. Анализ на стабилността им.

5. Двоична пирамида. Двоичните пирамиди като попълнени двоични дървета и като масиви. Формули за индексите на родителя и на децата на даден връх, ако пирамидата е масив. Построяване на пирамида: наивен начин чрез HeapInsert.

6. Построяване на пирамида чрез функция Heapify. Анализ на сложността по време на двата начина за построяване на пирамида. Процедура Build Heap. Анализ на сложността: наивна имплементация ($\Theta(n.lg(n))$) и бърза имплементация ($\Theta(n)$). Пирамидална сортировка Heapsort.

7. Приоритетни опашки като абстрактен тип данни (АТД). Имплементация на приоритетни опашки чрез обикновени масиви и чрез двоични пирамиди – сравнителен анализ на тези две имплементации.

8. Рекурсивни алгоритми и рекурентни отношения. Методи за решаване на рекурентни отношения: развиване, дърво на рекурсията, индукция, Master Theorem, методът с характеристичното уравнение. Примери.

9. Алгоритмична схема Разделяй и владей: същност, приложение и ограничения върху възможностите за прилагането ѝ. Сортиращ алгоритъм Mergesort. Анализ на сложността по време и на стабилността на Mergesort. Анализ на коректността на Mergesort чрез инварианта на цикъла.

10. Сортиращ алгоритъм Quicksort . Анализ: сложност по време и памет, стабилност, анализ на средната сложност по време, коректност.

11. Подобрения на QuickSort: намаляне на използваната памет до $O(lg(n))$ чрез използване на опашкова рекурсия (tail recursion), по-ефективно избиране на разделител, техники за избягване на лошия случай.

12. Долни граници върху асимптотичната сложност по време на алгоритми. Дървета на решението (decision trees). Метод за доказване на долни граници, базиран на дървета на решението. Долна граница върху асимптотичната сложност по време на сортиращи алгоритми, базирани на директно сравнение.

13. Долна граница, доказана чрез анализ на дървото на решението, върху асимптотичната сложност на изчислителната задача Element Uniqueness. Доказване на асимптотични долни граници чрез редукции между масови задачи – пример с Closest Pair.

14. Сортиране, което не е базирано на директно сравнение. Сортиране в линейно време – алгоритми Counting Sort и Radix Sort. Анализ на коректността им и на сложността им по време.

15. Ориентирани и неориентирани графи от алгоритмична гледна точка. Примери за задачи, които се моделират чрез графи. Представяния на графи чрез матрици на съседства и чрез списъци на съседства. Сравнителен анализ на предимствата и недостатъците на тези представяния. Анализ на сложността по време на алгоритми върху графи – въвеждане на асимптотични нотации на функции на две променливи. Линейна сложност по време при графи.

16. Обхождане на графи. Основна схема за обхождането – маркиране на върховете с цветове. Коректност и ефикасност на основната схема. Обхождането в ширина и в дълбочина като различни имплементации на основната схема.

17. Обхождане в ширина (BFS) на графи. Дърво на обхождането в ширина. Видове ребра в обхождането в ширина. Обхождане в дълбочина (DFS) на графи. Дърво на обхождането в дълбочина. Видове ребра в обхождането в дълбочина при неориентираните и ориентираните графи.

18. Ориентирани ациклични графи (dags). Алгоритми за топологично сортиране на ориентирани ациклични графи. Анализ на сложността по време на тези алгоритми.

19. Силно свързани компоненти на ориентирани ациклични графи. Алгоритъм с линейна сложност по време за откриването на силно свързаните компоненти на ориентирани графи.

20. Срязващи върхове (cut vertices) и срязващи ребра (bridges) в неориентираните графи. Двусвързани компоненти в неориентирани графи. Алгоритми с линейна сложност по време за откриване на срязващите върхове на граф и на срязващите ребра на граф.

21. Минимални покриващи дървета на неориентирани графи. Примери за приложения на минимални покриващи дървета. Обща алгорит-

мична схема за алгоритми за откриване на минимални покриващи дървета. Greedy стратегии. Алгоритъм на Прим – псевдокод и анализ на сложността по време.

22. Алгоритъм на Крускал за откриване на минимално покриващо дърво. Сложност по време на наивната имплементация. Усъвършенстване на наивната имплементация – Union-Find операции върху множества, имплементирани чрез дървета. Евристики Union by rank и Path compression. Анализ на сложността по време на алгоритъма на Крускал при използването на двете евристики.

23. Най-къси пътища в ориентираните графи. Примери за приложения. Потенциални трудности при наличието на отрицателни тегла. Анализ на задачата и сравнението и със задачата за най-дълъг път в граф. Алгоритъм на Дийкстра за намиране на най-късите пътища от даден връх до всички останали върхове. Анализ на коректността и сложността по време на този алгоритъм.

24. Алгоритми за намиране на най-къси пътища между всички двойки върхове – алгоритъм, използващ алгоритъма на Дийкстра, и два алгоритъма, използващи идеята на динамичното програмиране: по броя на ребрата в пътя и по най-големия номер на връх в пътя (алгоритъм на Флойд-Уоршъл). Анализ на сложността по време.

25. Динамично програмиране – същност, предимства и приложения. Примери: числа на Фибоначи, оптимално верижно умножение на матрици, изчислителен проблем Independent Set при неориентираните графи, ограничен до дърветата. Сравнение между динамичното програмиране и схемата „Разделяй и владей“.

26. Практически решими изчислителни задачи – клас на сложност P. Практически нерешими (intractable) задачи. Примери върху неориентирани графи: VERTEX COVER, INDEPENDENT SET, DOMINATING SET. Практическата нерешимост като фундаментално, независещо от технологичния прогрес, ограничение – принцип на Landauer. Практически нерешими задачи с кратък сертификат – клас на сложност NP.

27. Недетерминирани алгоритми за решаване в полиномиално време на задачи с кратки сертификати. Въпросът дали P = NP като фундаментална нерешена задача на теоретичната компютърна наука. Класове на сложност co-P и co-NP – съпоставка. Примери за задачи от co-NP.

28. Полиномиални редукции между масови задачи. NP-пълнота. Изчислителна задача SATISFIABILITY. Доказване на NP-пълнотата на SATISFIABILITY – теорема на Кук.

29. Изчислителна задача 3-SAT. Доказване на NP-пълнотата на 3-SAT. Съпоставка на 3-SAT с 2-SAT. Доказване на NP-пълнотата на VERTEX COVER, CLIQUE, HAMILTON CYCLE и LONGEST PATH. Други NP-пълни масови задачи.

30. Апроксимацията като средство за „справяне“ с практическата нерешимост. Апроксимиращи алгоритми с точност до мултипликативна

константа. Примери с VERTEX COVER и задачата за търговския пътник в „планарен вариант“. Условна невъзможност за добро апроксимиране на някои масови задачи.