

Prüfungsfragen DS (nach Wegener)

Soweit möglich findet sich nach der Frage eine Seitenangabe für das Skript vom WS 01/02.

1. Was ist der Unterschied zwischen der uniformen und der logarithmischen Zeitkomplexität? 14
2. Warum wird meist die worst case und nicht die average case Rechenzeit untersucht? 15
3. Bei Quicksort gibt es die selbe average case Rechenzeit bei Wahl des ersten Elements als Pivot-Element und bei Wahl eines zufälligen Elements. Wo liegt der Unterschied?
4. Definiere O , o , Ω , ω , Θ und warum ist ihre Verwendung sinnvoll? 18
5. Sortiere nach Größenordnung: $n^2 \log n$, $n^2 \log^2 n / (\log \log n)^{10}$, $(2^n)^{(1/10)}$, $2^{n/100}$, $n^2 + 3n^{3/2}$, $n^{15/8}$, $n^2/\log n$, $n^{\log n}$.
6. Beschreibe die Strategien lineare, binäre und geometrische Suche mit Vor- und Nachteilen. 22
7. Wie können spärlich besetzte Matrizen dargestellt werden und wie werden Operationen wie Zugriff, Addition und Multiplikation unterstützt? 23ff.
8. Vergleiche lineare Listen und Arrays bezüglich Suche nach Datum an Position n , Suche nach Datum x , Einfügen von Datum y hinter x , nachdem x gefunden wurde. 28f.
9. Gib Beispiele an, bei denen es sinnvoll ist, doppelte Verkettung oder Zeiger auf das letzte Element zu haben oder die Größe der Liste zu verwalten.
10. Algorithmus zum topologischen Sortieren. 30
11. Vergleiche Bitvektordarstellung und geordnete Listen zur Verwaltung von Mengen.
12. Beschreibe DFS für gerichtete Graphen. 43f.
13. Segmentbäume (wofür, wie, Rechenzeit) 46ff.
14. UNION-FIND mit Arrays und Listen (Darstellung der Struktur, FIND, UNION, Analyse) 48ff.
15. UNION-FIND mit Bäumen (Darstellung der Struktur, max. Baumhöhe bei n Daten, Pfadkomprimierung)
16. Welche Operationen werden zur Verifikation von Schaltkreisen benötigt? 52f.
17. OBDDs: Darstellung und Minimierungsregeln 54ff.
18. OBDDs: Synthese mit integrierter Reduktion / Minimierung
19. Wie sieht ein OBDD für die Addition (für einen Multiplexer) aus?
20. Was sind dynamische Dateien? 59
21. Vergleiche offenes und geschlossenes Hashing. 64f.
22. Beschreibe Strategien zur Kollisionsbehandlung. Unterscheiden sie sich in ihrem Verhalten überhaupt?
23. Was versteht man unter idealem Hashing? Wie sieht dabei der Ansatz zur Berechnung der idealen Suchzeit aus? 62
24. Was sind Suchbäume? (Search, Insert, Delete) 65ff.
25. Durchschnittliche Tiefe binärer Suchbäume mit n Daten, die in zufälliger Reihenfolge kommen (Ansatz und Ergebnis asymptotisch)?
26. 2-3-Bäume: Definition, minimale und maximale Tiefe, Search, Insert, Delete. (Erklärung wie auch sonst nicht nur am Beispiel, sondern auch strukturell, Beispiele aber möglich) 71ff.
27. Das gleiche für B-Bäume. 81ff.
28. Das gleiche für AVL-Bäume, Ansatz zur Berechnung der Maximaltiefe bei n Daten. Wann ist welcher Typ von Rotation richtig? 83ff.
29. Skiplisten: Struktur, durchschnittliche Höhe, Operationen, Analysetechnik und Anwendungen 91
30. Liste die bekannten Sortieralgorithmen mit Vor- und Nachteilen auf. 96ff.
31. Quicksort: Wie kommt das Zerlegungsdatum an die richtige Position? Strategien zur Wahl des Zerlegungsdatums. worst case, Ansatz zur average case Analyse und Ergebnis. 100

32. Heapsort: Darstellung des Heaps im Array, das Rahmenprogramm, Reheap-Strategien. Warum ist bottom-up besser? Warum ist heap creation effizienter als selection? Analyse der worst case Zeiten der verschiedenen Strategien. 104ff.
33. Was ist ein allgemeines Sortierverfahren? Untere Schranken für worst case und average case. 109ff.
34. Bucketsort: Verfahren und Analyse. 113f.
35. Beschreibe effiziente Algorithmen für das Auswahlproblem mit Analyseansatz. 115f.
36. Batcher sort: Wie geht es und warum klappt es? Sequentielle und parallele Rechenzeit. Beschreibe die folgenden Optimierungsstrategien allgemein: Greedy, dynamisch, divide & conquer, branch & bound. 117f.
37. Gib Beispiele für greedy Algorithmen an, die stets das Optimum berechnen (beliebig schlechte Ergebnisse berechnen können; nicht linear optimale, aber nie schlechte Ergebnisse berechnen). 120ff.
38. Algorithmus von Kruskal: wozu, wie, Datenstrukturen, Rechenzeit, immer optimal. 125
39. Dynamische Programmierung: Was sind Probleme vom Intervalltyp?
40. Berechnung statischer Suchbäume mit minimaler Rechenzeit. 129
41. Was sind pseudopolynomielle Rechenzeiten? Gib ein Beispiel an (Dynamische Programmierung für das Rucksackproblem). 130
42. Dynamische Programmierung für APSP (all pairs, shortest paths). 130f.
43. Algorithmus von Dijkstra: Strategie, Korrektheit, Analyse, Datenstruktur. 131ff.
44. Die Module von branch-&-bound-Algorithmen am Beispiel des Rucksackproblems. 134ff.
45. Divide & conquer: $R(n) = a * R(n/b) + cn$. Wie wirken sich a, b und c auf die Rechenzeit aus? 138f.
46. Welche divide-&-conquer-Algorithmen wurden behandelt?
47. Welche Algorithmen der dynamischen Programmierung wurden behandelt?
48. Beschreibe die wesentlichen Ideen des Strassen-Algorithmus. 141
49. FFT: Wozu und wie funktioniert es? 141ff.
50. Berechnung der nächsten Nachbarn in der Ebene. 146ff.
51. Was ist die Sweepline-Technik? 148ff.
52. Anwendung auf das Rechteckmaß-Problem: Wie wird der Segmentbaum verwaltet? 150
53. Beschreibe das - Pruning. (150ff.)
54. Was ist das Szenario der Black-Box-Optimierung? 153ff.
55. Wie arbeiten randomisierte Suchheuristiken allgemein?
56. Beschreibe randomisierte lokale Suche, simulated annealing und evolutionäre Algorithmen.

Weitere Fragen

57. Was besagt die Bellmansche Optimalitätsgleichung? 131
58. Was ist die primitive n-te Einheitswurzel? 142
59. Beschreibe das Maxsummenproblem.
60. Beschreibe das Maxmin-Problem.
61. Was ist das Traveling Salesman Problem? Welche Ansätze zur Lösung gibt es?