UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR THEORETISCHE INFORMATIK

Veranstaltungskarte
CS3000 Algorithmendesign

ALGORITHMENDESIGN

Veranstaltungsziele
- Vertrautheit mit algorithmischen Entwurfsprinzipien
- neue komplexe Algorithmen durch Anwendung dieser Prinzipien entwickeln können
- Erfahrung beim algorithmischen Problemlösen

TILL TANTAU
WWW.TCS.UNI-LUEBECK.DE
12. OKTOBER 2016

Einführung, Organisatorisches

Offline-Probleme 1: Zahlen

Entwurfsmethode: Teile-und-Herrsche

November 2016

Entwurfsmethode: Dynamische Tabellen

Offline-Probleme 2: Texte

A&D: Indizieren mit Bäumen

A&D: Indizieren mit Arrays

A&D: Textkompression

Online-Probleme

Klassifikation: Kompetitive Rate

Dezember 2016

Analysemethode: Amortisierung

A&D: Union-Find

Weihnachtliche Herausforderungen

Januar 2017

Randomisierung

A&D: Suchen in hashbaren Daten

A&D: Perfektes Hashing

Entwurfsmethoden: Zufall

Schwere Probleme und ihre Lösung

Entwurfsmethode: Approximation

Februar 2017

Entwurfsmethode: Fixed-Parameter

Problemstellungen

1 Entwurfsmethode: Teile-und-Herrsche
- Die Schritte des Algorithmendesigns (Problem, Entwurf, Algorithmus und Datenstruktur, Analyse, Klassifikation) anhand von Zahl-Problemen nachvollziehen
- Den Karatsuba- und den Strassen-Algorithmus kennen
- Das Master-Theorem kennen und anwenden können

2 Entwurfsmethode: Dynamische Tabellen
- Beispiele von Problemstellungen kennen, die sich mit dynamischen Tabellen lösen lassen
- Konzept der dynamischen Tabelle in eigenen Anwendungen einsetzen können
- Auf dynamischen Tabellen aufbauende Algorithmen analysieren können

3 A&D: Indizieren mit Bäumen
- Die Datenstruktur des Tries kennen und implementieren können
- Die Datenstruktur des Suffix-Trees kennen
- Methoden zur Suche in Texten kennen, die auf diesen Datenstrukturen aufbauen

4 A&D: Indizieren mit Arrays
- Die Datenstruktur des Suffix-Arrays kennen
- Den dc3-Algorithmus verstehen

5 A&D: Textkompression
- Zeichenbasierte Kompressionsalgorithmen kennen
- Burrows-Wheeler-basierte Kompressionsalgorithmen kennen

6 Klassifikation: Kompetitive Rate
- Varianten von Scheduling- und Caching-Problemen kennen
- Greedy-Verfahren zu deren Lösung kennen
- Konzept des Online-Algorithmus verstehen
- Konzept der kompetitiven Rate verstehen
- Kompetitive Raten von Algorithmen bestimmen können

7 Analysemethode: Amortisierung
- Konzept der amortisierten Analyse verstehen
- Die Potential-Methode einsetzen können

8 A&D: Union-Find
- Den optimalen Algorithmus zur Verwaltung disjunkter Mengen erklären können.
- Die amortisierte Analyse des Algorithmus verstehen.

9 Weihnachtliche Herausforderungen
- Typische Programmieraufgaben lösen können

10 A&D: Suchen in hashbaren Daten
- Ideen und Methodik des Hashings auffrischen
- Hash-Tabellen mit veränderlicher Größe analysieren können

11 A&D: Perfektes Hashing
- Idee des perfekten Hashings und einfache perfekte Hash-Verfahren kennen
- Kuckucks-Hash-Tabellen verstehen und implementieren können

12 Entwurfsmethoden: Zufall
- Die Konzepte «zufällige Eingabe», «zufällige Ausgaben» und «zufällige Entscheidung» unterscheiden können
- Einschätzen können, wie verlässlich Zufallsalgorithmen sind
- Klassen von Zufallsalgorithmen erläutern können
- Einfache Beispiele von Zufallsalgorithmen implementieren können

13 Entwurfsmethode: Approximation
- Die Entwurfsmethode «Approximation» kennen und Approximationsalgorithmen entwerfen können
- Wichtige Approximationsalgorithmen für mehrere schwere Probleme kennen

14 Entwurfsmethode: Fixed-Parameter
- Konzept des Fixed-Parameter-Algorithmus verstehen
- Fixed-Parameter-Algorithmus für das Vertex-Cover-Problem kennen
- Konzept der Kernelisierung verstehen
- Kernelisierungs-Algorithmus für das Vertex-Cover-Problem kennen

Arten der Eingabe
- Datentypen
  - Zahlen
  - Arrays
  - Matrizen
  - Geometrische Objekte
  - Graphen
  - Datenbanken
  - Worte / Texte
- Temporale Eigenschaften
  - Online
  - Offline
- Ordnungseigenschaften
  - Linear geordnete Daten
  - Hashbare Daten
  - Geometrisch geordnete Daten

Konkrete Problemstellungen
- Arithmetische Probleme
  - Zahlmultiplikation
  - Matrixmultiplikation
  - Subset-Sum-Problem
  - Partition-Problem
  - Münzrückgabe-Problem
  - Rucksack-Problem
- Scheduling
  - Ohne / mit Deadlines
  - Ohne / mit Unterbrechungen
  - Eine / mehrere Maschinen
  - Caches
- Suchen in Daten
- Kompression
- Graphprobleme
  - Zusammenhang
  - Maximaler Fluss
  - Matching
  - Hamilton- und Eulerkreise
  - Vertex-Cover
  - Färbbarkeit
  - Independent-Set und Clique

Arten der Problemstellungen
- Ziel der Berechnung
  - Entscheidungsprobleme
  - Funktionsprobleme
  - Optimierungsprobleme
- Kontinuität
  - Diskrete Problemstellungen
  - Numerische Problemstellungen

Kodierungen der Eingabe
- Zahlen
  - Unär
  - Binär
  - Gleitkomma
- Graphen
  - Adjazenzlisten
  - Adjazenzmatrizen
- Komprimierte Eingaben
- Redundanz in Eingaben

Entwurfsmethoden

Reduktion auf bekannte Probleme
- Reduktion auf kombinatorische Probleme
- Reduktion auf numerische Probleme
  - Darstellung als Lineares Programm
  - Darstellung als lineares Ganzzahlproblem

Randomisierung
- Pseudo-Zufallszahlen
- Zufallszahlen
- Hashing

Rekursive Ansätze
- Teile-und-Herrsche
- Backtracking
  - Branch-and-Bound

Iterative Ansätze
- Greedy-Verfahren
- Dynamische Tabellen
- Zunehmende Wege

Kompressionsmethoden
- Entropiebasierte Kodierung
- Wörtbuchtechniken
- Transformationen

Ansätze für Online-Probleme
- Häufigkeitsbasierte Strategien
- Dauerbasierte Strategien

Heuristische Ansätze
- Lokale Suche
- Evolutionäre Algorithmen
- Simulated-Annealing
- Approximative Verfahren

Fixed-Parameter-Verfahren
- Parameterwahl
- Suchbaum-Verfahren

Vorverarbeitung
- Kernelisierung
- Indizierung

Algorithmen

Hashing-Algorithmen
- Lineares Hashing
- Kuckucks-Hashing

Arithmetische Algorithmen
- Karatsuba-Algorithmus
- Strassen-Algorithmus
- Gauß-Algorithmus für Determinanten

Algorithmen auf Graphen
- Brute-Force
- Flussalgorithmen
  - Ford-Fulkerson
  - Matching und das Heiratsproblem
- Approximationsalgorithmen
  - für Vertex-Cover
  - für das Rucksack-Problem
- Fixed-Parameter-Algorithmen
  - für Vertex-Cover

Textverarbeitung
- Kompressionsverfahren
  - Huffman
  - Lempel-Ziv
  - Burrows-Wheeler-Transformation
- DC3-Algorithmus

Geometrische Algorithmen
- *k*-Means
- *k*-Server-Algorithmen

Scheduling-Algorithmen
- Offline-Algorithmen
  - Earliest-Finish-First
  - Start-Time-Ordering
  - Earliest-Deadline-First
- Online Caching-Algorithmen

Index-Strukturen
- Tries
  - Particia-Bäume
- Suffix-Indizes
  - Suffix-Trie
  - Suffix-Tree
  - Suffix-Array

Map-Strukturen
- Balanzierte Bäume
  - AVL-Bäume
  - 2-3-Bäume
  - Rot-Schwarz-Bäume
- Geometrische Bäume
  - Quad- und Oct-Bäume
  - Mehrdimensionale Suchbäume
- Hash-Tabellen
  - Offenes Hashing
  - Geschlossenes Hashing

Prioritäts-Warteschlangen
- Einfache Heaps
- Fibbonacci-Heaps
- Union-Find-Strukturen

Dynamische Tabellen

Datenstrukturen

Analysemethoden

Korrektheit
- Hoare-Kalkül

Rekursionsgleichungen
- Master-Theorem

Untere Schranken
- Härte mittels Reduktionen
- Konstruktion von Worst-Case-Eingaben
- Adversary-Argumente
- Exchange-Argumente

Obere Schranken
- Abschätzung der benötigten Ressourcen
  - Laufzeit
  - Speicherplatz
  - Tiefe
- Amortisierte Analyse
  - Potentialmethode
- Fehlerwahrscheinlichkeit

Eingabeverteilung
- Wahrscheinlichkeitsverteilungen
  - Worst-Case
  - Average-Case
  - Smoothed-Case
  - Best-Case
- Beschränkung von Eingabe-Parametern

Klassifikationen

Asymptotik von Funktionen
- O-Klassen und o-Klassen
- Θ-Klassen
- Ω-Klassen und ω-Klassen
- Ignorierte Faktoren
  - Konstante Faktoren
  - Logarithmische Faktoren
  - Polynomielle Faktoren

Klassifikation von Datenstrukturen
- Platzverbrauch von Datenstrukturen
  - $O(n)$
  - $O(n^2)$
  - $O(2^n)$
- Tiefe von Datenstrukturen
  - $O(\log n)$
  - $O(n)$

Klassifikation der Laufzeit
- Typische Zeitschranken
  - $O(1)$
  - $O(\log n)$
  - $O(n)$
  - $O(n\alpha(n))$
  - $O(n \log n)$
  - $O(n^{2-\delta})$
  - $O(n^2)$
  - $O(n^{2+\delta})$
  - $O(n^3)$
  - $O(n^c)$
  - $O(f(k)n^c)$
  - $O(2^n)$
- Kleine Zeitklassen
  - P
  - P-vollständig
  - ZPP
  - RP
  - coRP
  - BPP
  - FP (functional P)
  - FPT (fixed parameter tractable)
- Große Zeitklassen
  - NP
  - NP-vollständig
  - EXP
  - EXP-vollständig

Klassifikation des Platzverbrauchs
- Kleine Platzklassen
  - L
  - NL
  - NL-vollständig
  - FL
- Große Platzklassen
  - PSPACE
  - PSPACE-vollständig

Raten
- Kompetitive Rate
- Approximationsrate
- Kompressionsrate

Problemstellungen — Entwurfsmethoden — Algorithmen — Datenstrukturen — Analysemethoden — Klassifikationen