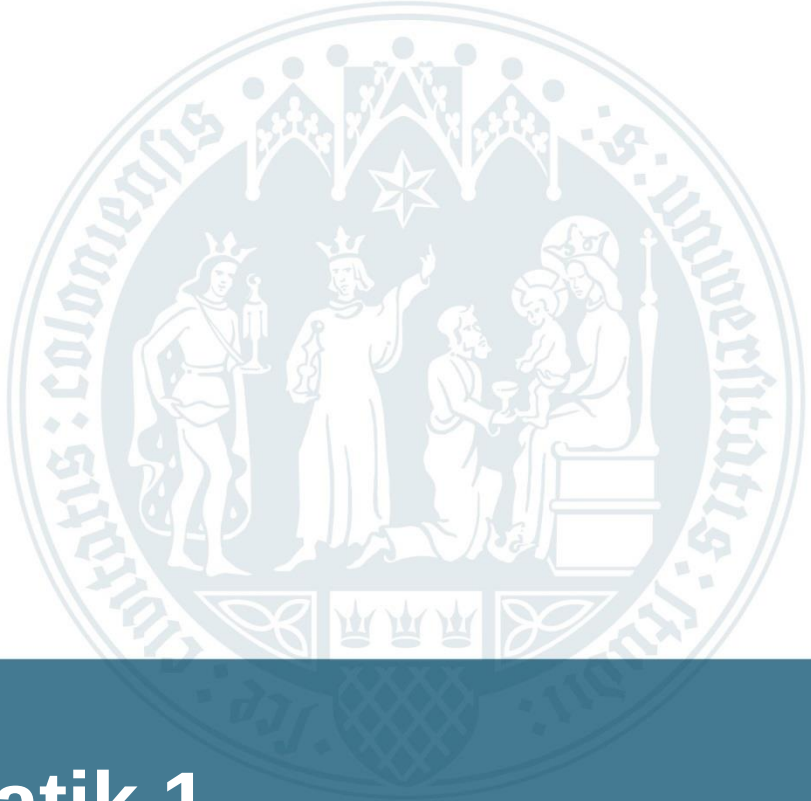
Universität zu Köln



Grundzüge der Informatik 1

Vorlesung 13 - flipped classroom

Dynamische Programmierung

Dynamische Programmierung für Optimierungsprobleme

1. Bestimme rekursive Struktur einer optimalen Lösung durch Zurückführen auf optimale Teillösungen

2. Entwerfe rekursive Methode zur Bestimmung des *Wertes* einer optimalen Lösung.

3. Transformiere rekursive Methode in eine iterative (bottom-up) Methode zur Bestimmung des Wertes einer optimalen Lösung.

4. Bestimmen aus dem Wert einer optimalen Lösung und in 3. ebenfalls berechneten Zusatzinformationen eine optimale Lösung.

Dynamische Programmierung

Aufgabe 1

- Eine Firma kann unterschiedliche Produkte aus zwei Ressourcen herstellen. Produkt i benötigt eine Menge r[i] von Ressource 1 und t[i] von Ressource 2 und hat einen Wert von w[i]. Die Einträge in r und t sind natürliche Zahlen.
- Angenommen, die Firma kann n unterschiedliche Produkttypen herstellen, deren Werte und Ressourcenbedarf in den Feldern r[1..n], t[1..n] und w[1..n] gegeben sind und sie verfügt über T Einheiten von Ressource 1 und S Einheiten von Ressource 2. Welche Produkte sollte die Firma herstellen, um ihren Gewinn zu maximieren?
- Finden Sie eine Rekursionsgleichung für den Erlös Opt(k,i,j), den die Firma mit der Produkttypen 1 bis k mit vorhandenen Ressourcen i und j erzielen kann.
- Entwickeln Sie einen Algorithmus für die Berechnung des optimalen Erlöses.

Gierige Algorithmen

Entwurfsprinzip „Gierige Algorithmen“

- Ziel: Lösung eines Optimierungsproblems
- Herangehensweise: Konstruiere Lösung Schritt für Schritt, indem immer ein einfaches „lokales“ Kriterium optimiert wird
- Vorteil: Typischerweise einfache, schnelle und leicht zu implementierende Algorithmen

Beobachtungen

- Gierige Algorithmen optimieren einfaches lokales Kriterium
- Dadurch werden nicht alle möglichen Lösungen betrachtet
- Dies macht die Algorithmen oft schnell
- Je nach Problem und Algorithmus kann die optimale Lösung übersehen werden

Gierige Algorithmen

Aufgabe 2

- Gegeben: Menge von reellwertigen Punkten $\{x_1,\ldots,x_n\}$ im 1D (abgespeichert in einem Feld A)
- Gesucht: Die minimale Anzahl Intervalle der Länge 1, die ausreicht, um die Punkte zu überdecken
- Was ist die Laufzeit Ihres Algorithmus?

Gierige Algorithmen

Aufgabe 3

- Beim SetCover Problem besteht die Eingabe aus n Teilmengen $S_1,\ldots,S_n$ der Menge {1,…,m}. Dabei soll jedes Element aus {1,…,m} in mindestens einer Teilmenge vorkommen.
- Eine gültige Lösung für das SetCover Problem ist eine Menge I von Indizes, so dass $\bigcup_{i\in I} S_i$={1,…,m} ist
- Gesucht ist eine kleinstmögliche gültige Lösung, d.h. eine gültige Lösung, die |I| minimiert
- Was ist eine optimale Lösung für die folgenden Teilmengen?
- {1,2,4,7}, {2,3,5}, {6}, {3,5,6,8}, {1,2,3}
- Entwickeln Sie einen gierigen Algorithmus für das SetCover Problem.
- Zeigen Sie entweder, dass Ihr Algorithmus das Problem immer exakt löst oder geben Sie ein Gegenbeispiel an.

Dynamische Programmierung

Aufgabe 4

- Die Editierdistanz zwischen zwei Zeichenketten S und T ist die minimale Anzahl an Operationen Insert, Delete und Replace, um die Zeichenkette S in die Zeichenkette T umzuwandeln
- Entwickeln Sie mit Hilfe dynamischer Programmierung einen Algorithmus zur Berechnung der Editierdistanz zwischen zwei Zeichenketten S und T der Längen n und m
- Erstellen Sie dazu zunächst eine Rekursion