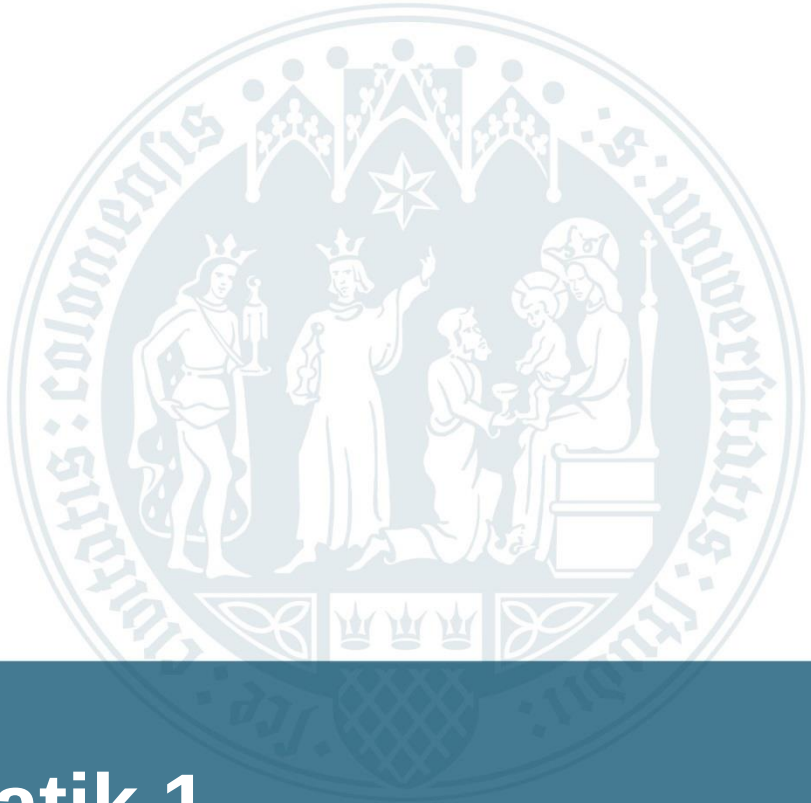
Universität zu Köln



Grundzüge der Informatik 1

Vorlesung 10 - flipped classroom

Dynamische Programmierung

```
Fib2(n)
1.  if n=1 then return 1
2.  if n=2 then return 1
3.  return Fib2(n-1) + Fib2(n-2)
```

Fibonacci Zahlen

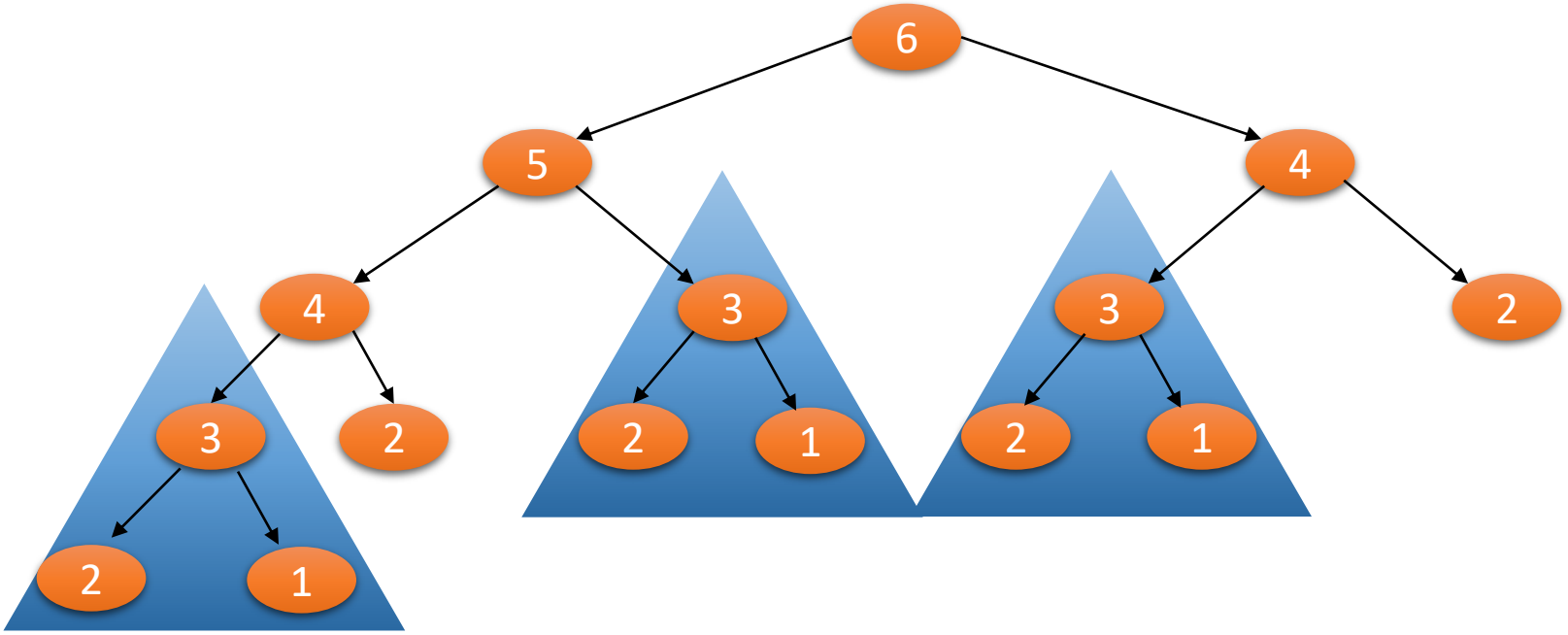
- Fib(1)=1
- Fib(2)=1
- Fib(n) = Fib(n-1) + Fib(n-2)

Dynamische Programmierung

Warum ist die Laufzeit so schlecht?

- Betrachte Rekursionsbaum von Fib2(6)

Bei der Berechnung von Fib2(6) wird Fib2(3) dreimal aufgerufen!



Dynamische Programmierung

Zwischenspeichern von Rechenergebnisse

- Idee: Wir speichern die Ergebnisse, die wir bereits kennen

Dynamische Programmierung

Beobachtung

- Die Tabelle wird bottom-up ausgefüllt

Vereinfachter Code

```
Fib1(n)
1.  F = new array[1…n]
2.  F[1] = 1
3.  F[2] = 1
4.  for i=3 to n do
5.     F[i] = F[i-1] + F[i-2]
6.  return F[n]
```

Dynamische Programmierung

Dynamische Programmierung

- Beschreibe optimale Lösung einer gegebenen Instanz durch optimale Lösungen „kleinerer“ Instanzen (hier kleinere Fibonacci-Zahlen)
- Beschreibe Rekursionsabbruch
- Löse die Rekursion „bottom-up“ durch schrittweises Ausfüllen einer Tabelle der benötigten Teillösungen

Wann verbessert der Ansatz die Laufzeit?

- Die Anzahl unterschiedlicher Funktionsaufrufe (Größe der Tabelle) ist klein
- Bei einer „normalen Ausführung“ des rekursiven Algorithmus ist mit vielen Mehrfachausführungen zu rechnen

Dynamische Programmierung

Aufgabe 1

- Die Funktion n!! Ist definiert als
  - n (n-2) (n-4) … 2, wenn n gerade
  - n (n-2) (n-4) … 1, wenn n ungerade
- Geben Sie eine Rekursion für n!! an

Dynamische Programmierung

Aufgabe 2

- Beschreiben Sie einen rekursiven Algorithmus für n!!

Dynamische Programmierung

Aufgabe 3

- Lösen Sie das Problem mit Hilfe dynamischer Programmierung
- Was ist die Laufzeit Ihrer Algorithmen?

Dynamische Programmierung

Aufgabe 4

- Geben Sie einen Beweis für die Laufzeit

Dynamische Programmierung

Aufgabe 5

- Beim Spiel „Jump“ gibt es ein Feld A[1..n], das jeweils einen Punktwert enthält (der auch negativ sein kann)
- Die Spielfigur startet auf Position 1 und kann sich entweder einen oder 2 Schritte nach rechts bewegen
- Für jedes Feld, auf dem die Figur steht, erhält sie seine Punkte
- Am Ende muss die Figur auf Position n stehen
- Entwickeln Sie eine Rekursion für die optimale Anzahl an erreichbaren Punkten

Dynamische Programmierung

Aufgabe 6

- Wandeln Sie die Rekursion in einen Algorithmus um

Dynamische Programmierung

Aufgabe 7

- Skizzieren Sie die Anzahl rekursiver Aufrufe für den Fall n=5

Dynamische Programmierung

Aufgabe 8

- Lösen Sie das Problem mit Hilfe von dynamischer Programmierung
- Was ist die Laufzeit Ihres Algorithmus?