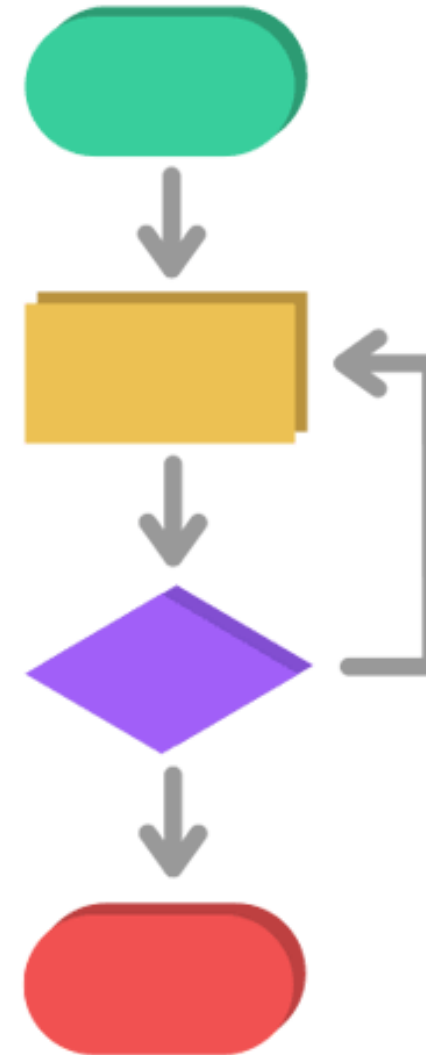


Algorithmen Tutorium 13

Beginn: 16:15



Organisation

- Nächste Woche: Fragen zur Klausur
 - Am Besten vorab (= bessere Vorbereitung)
- Umfrage zur Klausuranmeldung auf {Moodle, Alma}
- Tipp Blatt 13: Code

Dynamische Programmierung

- Idee: Teil- und Zwischenprobleme geschickt ausnutzen
- Problem: Man muss die Idee haben...
 - Vorbereitung: Gängige & Leichte Algorithmen wiederholen
 - Prüfung: Wenn keine Idee => Weglassen
- Bereits bekannt
 - Beispiele: Floyd Warshall, Dijkstra, Fibonacci(rekursiv), ...

Branch & Bound

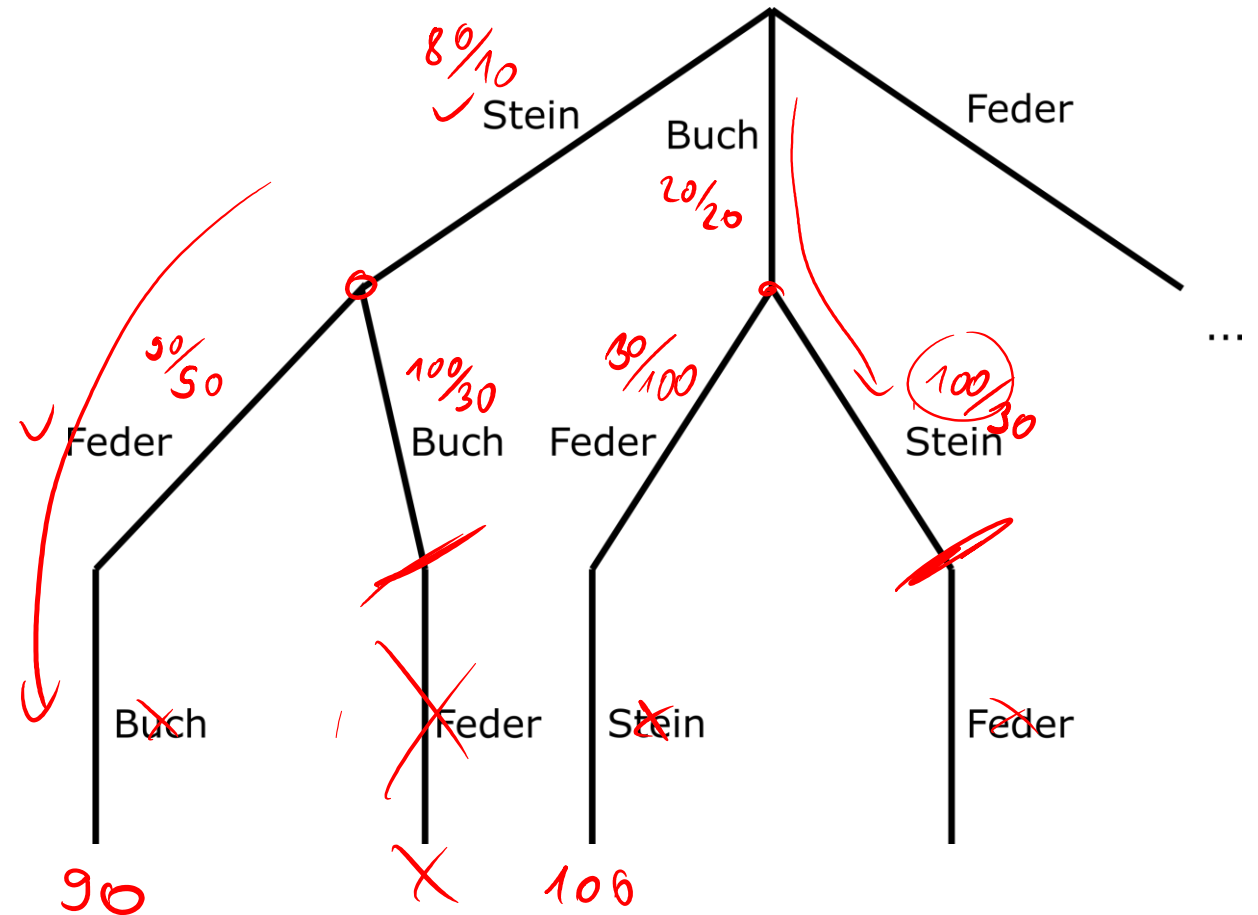
- Benötigt Bounding-Funktion für untere Schranke
 - Nutzt die bisher beste Lösung
- Baue Entscheidungsbaum auf
 - Z.B. durch Tiefen- oder Breitensuche
- Durchsuche den kompletten Baum
- Erkunde nur Pfade, die besser als die bisher beste Lösung sind (Bounding-Funktion)

Beispiel: Branch & Bound


- Knapsack Problem: **S**tein, **B**uch, **F**eder
 - Kapazität: 100
 - Gewichte{S,B,F}: {80, 20, 10}
 - ~~Kosten~~^{Preis}{S,B,F}: {10, 20, 80}
 - Bounding-Funktion:
 - Bisher bester Preis für eine „Füllung“
- Algorithmus nicht optimal☺

Preis für Füllung


= Bound :



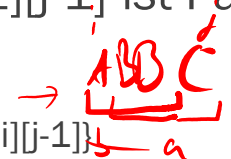

Longest Palindrome Subsequence

- Palindrom: Vorwärts und Rückwärts gleich (z.B. ABBA, Drehmal am Herd)
- Betrachten String s
 - $s[i]$: Buchstabe an Stelle i
 - $s[i][j]$: Buchstaben zwischen i und j
- Länge des Palindroms L für Teilfolgen:
 - $L[i][j]$: Längstes Palindrom für Teilfolge $s[i][j]$
- Wichtig: Teilfolgen, d.h. **MIT** Unterbrechungen
- Video: <https://www.youtube.com/watch?v=TLaGwTnd3HY>

Wann Palindrom?

- 1 Buchstabe? Trivial ✓
- 2 Buchstabe?
 - $s[i] == s[i+1]$? *AA*
 - $s[i] != s[i+1]$? *AB*
- n Buchstaben? ($s[i+1][j-1]$ ist Palindrom)
 - $s[i] != s[j]$ *BACAB*

 - $s[i] == s[j]$

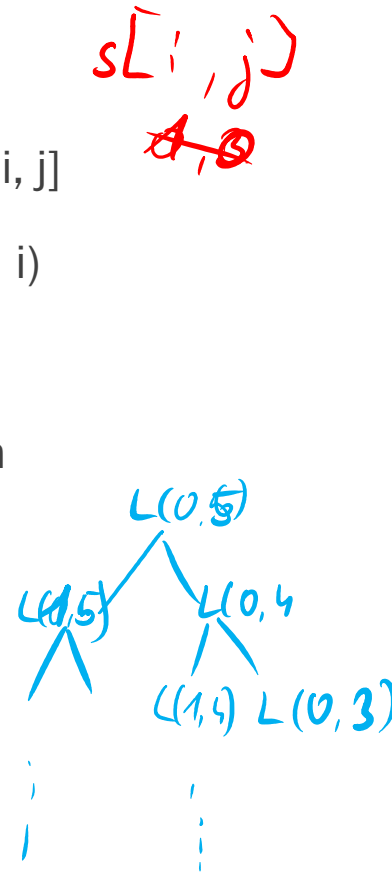
Palindrom Länge

- 1 Buchstabe? Trivial: 1 ✓
- 2 Buchstabe?
 - $L[i] == L[i+1]$? 2
 - $L[i] != L[i+1]$? 1
- n Buchstaben? ($s[i+1][j-1]$ ist Palindrom)
 - $s[i] != s[j]$?
 - $L[i][j] = \max\{L[i+1][j], L[i][j-1]\}$ 
 - $s[i] == s[j]$? 
 - $L[i][j] = \max\{L[i+1][j], L[i][j-1], L[i+1][j-1] + 2\}$ (bzw. nur $L[i+1][j-1] + 2$)

Algorithmus Tabellarisch

ABA
ACA $\rightarrow L=3$

- $n \times n$ Matrix L
- An Stelle $L[i][j]$ kommt LPS für String $s[i, j]$
- Unteres Linkes Dreieck ist leer (weil $j < i$)
- Beginnend mit allen $L[i][i]$
- Dann $L[i][j]$ mit $j = i + 1, \dots, n$ ausfüllen
- LPS steht oben rechts (in $L[0][n]$)
- (Backtracking für Palindrom)
- (Beachte Baumstruktur)



	<u>B</u>	<u>A</u>	<u>B</u>	<u>C</u>	<u>A</u>	<u>B</u>
i, j	0	1	2	3	4	5
0	1	1	1+2 =3	3	3	3+2 =5
1		1	1	1	3	3
2			1	1	1	3
3				1	1	1
4					1	1
5						1

1 a) ABA

000	}	$2^n \Rightarrow O(2^n), \Omega(2^n)$
010		
001		
111		
...		
111		

$O(n^2)$

b) for $k=0, \dots, n-1$ do in
 for $i=1, \dots, n-k$ do in
 if $k=0$ then
 $L[i, i] = 1$
 else if $k=1$ then
 if $S[i] = S[i+k]$ then
 $L[i, i+k] = 2$
 else
 $L[i, i+k] = 1$
 end

else:
 if $S[i] = S[i+k]$ then
 $L[i, i+k] = L[i+1, i+k-1] + 2$
 else:
 $L[i, i+k] = \max\{L[i+1, i+k], L[i, i+k-1]\}$
 end
 return $L[1, n]$

	D	A	D	B	B	C	A	A
i, j	0	1	2	3	4	5	6	7
0	1	1	$1+2=3$	3	3	3	4	4
1		1	1	1	2	2	4	4
2			1	1	2	2	2	2
3			0	1	2	2	2	2
4				0	1	1	1	2
5					0	1	1	2
6							1	2
7								1

A B B A