




Informatik II

Vorberechnung

G. Zachmann
 Clausthal University, Germany
zach@in.tu-clausthal.de




Definition

- **Preconditioning (Vorbereitung):**
 Falls nacheinander mehrere Probleme zu lösen sind, die einen gemeinsamen Anteil haben, d.h., falls wir nacheinander Eingaben der Form

$$\langle x_1; y \rangle \langle x_2; y \rangle \langle x_3; y \rangle \dots$$
 bekommen, so kann es sinnvoll sein, zunächst den gemeinsamen Anteil zu transformieren, also $\hat{y} := \text{Tr}(y)$, und dann die Probleme

$$\langle x_1; \hat{y} \rangle \langle x_2; \hat{y} \rangle \langle x_3; \hat{y} \rangle \dots$$
 zu lösen

G. Zachmann Informatik 2 - SS 06 Preprocessing 2

- Beispiele
 - Auswertung eines Polynoms an viele Stellen, Eingabe ist

$$\langle x_j; \underbrace{a_0, \dots, a_n}_{\text{„konstant“}} \rangle$$
 - Vorgänger-/Nachfolger-Beziehung in einem Baum testen, Eingabe ist

$$\langle v_1, v_2; \underbrace{\text{Baum}}_{\text{„konstant“}} \rangle$$
 - festen Text (z. B. Wörterbuch) nach einem Wort durchsuchen
 - Sortieren, Tries, ...
 - Optimale Suchbäume

G. Zachmann Informatik 2 - SS 06 Preprocessing 3

- **Precomputation (Vorbereitung):**

auch wenn die Eingabe nur einmal vorkommt –
aber dennoch aus 2 Teilen $\langle x; y \rangle$ besteht –
so kann es effizienter sein, zunächst einen Teil
zu transformieren, $\hat{y} := \text{Tr}(y)$, und dann die
Lösung aus $\langle x; \hat{y} \rangle$ zu berechnen

$$\begin{array}{c} \langle x; y \rangle \\ \downarrow Y_1 \\ \langle x; \hat{y} \rangle \end{array}$$

$$\searrow \text{Algo X}$$

$$\rightarrow \text{Algo Y}_2$$

$$z$$

 - Beispiel: dynamischen Text nach einem Wort durchsuchen
 - Achtung: im Umgangssprachgebrauch wird *"Precomputation"* oft für beide Arten verwendet
 - wesentliche Frage bei Precomputation / Preconditioning ist immer: Wie hoch ist der Aufwand? – bzw.: macht er evtl. den anschließenden Zeitgewinn kaputt?

G. Zachmann Informatik 2 - SS 06 Preprocessing 4



Beispiele bisher

- Closest Pairs
 - Vorsortieren der Punkte entlang x-Achse
- Huffman-Kodierung:
 - Vorsortieren der Zeichen gemäß Häufigkeit

G. Zachmann Informatik 2 - SS 06 Preprocessing 5



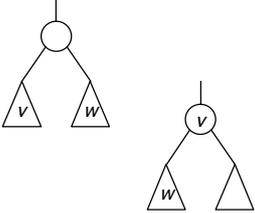
Vorgänger im Baum

- Problem:
 - gegeben ein Baum
 - wir sollen immer wieder für ein Paar von Knoten (v , w) entscheiden, ob v Vorgängerin, d.h. direkte/indirekte Mutter von w ist
- Naïve Lösung:
 - traversiere Baum von w aus bis zur Wurzel und vergleiche mit v
 - Aufwand:
 - $\Omega(n)$ im Worst-Case,
 - untere Schranke für worst-case = $O(\log n)$
- Behauptung: wir können auf dem Baum Preconditioning durchführen in Zeit $\Theta(n)$, so daß danach das Problem in Zeit $O(1)$ gelöst werden kann

G. Zachmann Informatik 2 - SS 06 Preprocessing 6

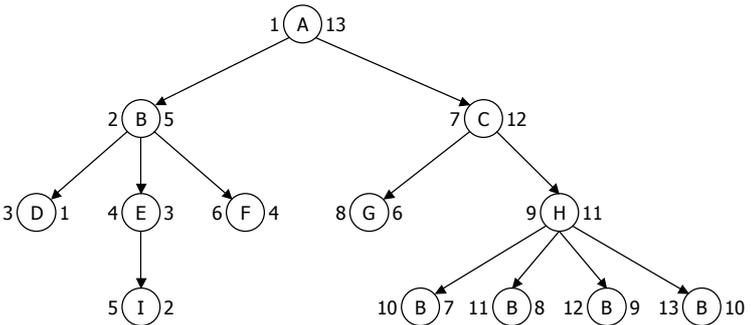
■ Methode:

1. Durchlaufe Baum in Preorder und nummeriere dabei die Knoten; bezeichne diese Nummer mit $N_{pre}(v)$
 - nummeriere also erst den Knoten selbst mit fortlaufender Nummer, dann nummeriere rekursiv linken, dann rechten Teilbaum
2. Nummeriere analog die Knoten in Postorder; Bezeichnung $N_{post}(v)$
3. Jetzt gilt für jedes Paar v, w von Knoten:
 - $N_{pre}(v) \leq N_{pre}(w) \Leftrightarrow$
 v Vorgänger von w \vee
 v links von w
 - $N_{post}(v) \geq N_{post}(w) \Leftrightarrow$
 v Vorgänger von w \vee
 v rechts von w
4. Zusammen:
 $N_{pre}(v) \leq N_{pre}(w) \wedge N_{post}(v) \geq N_{post}(w) \Leftrightarrow v$ Vorgänger von w



G. Zachmann Informatik 2 - SS 06 Preprocessing 7

Beispiel



G. Zachmann Informatik 2 - SS 06 Preprocessing 8

Wiederholte Auswertung eines Polynoms

- Gegeben: Polynom $p(x) = a_n x^n + \dots + a_1 x + a_0$
- Gesucht: $p(x_i)$ für viele x_1, \dots, x_n
- Verboten ist: `p = a[0] + a[1]*x + a[2]*x*x + ...`
- noch schlimmer: `p = a[0] + a[1]*x + a[2]*pow(x,2.0) + a[3]*pow(x,3.0) ...`
- Einfache Methode: transformiere p in Horner-Schema

$$p(x) = (((\dots((a_n x + a_{n-1})x + a_{n-2}) \dots)x + a_1) + a_0$$

```
p = a[0]
for j in range( n-1,-1,-1 ):
    p = p*x + a[j]
```
- Reicht für mittellange Polynome völlig aus
- Aufwand: n Multiplikationen / n Additionen

G. Zachmann Informatik 2 - SS 06 Preprocessing 9

Auswertung an äquidistanten Stellen

- Definition: **Vorwärts-Differenz**

$$\Delta^1 p(x) := p(x+h) - p(x)$$

$$\Delta^i p(x) := \Delta^{i-1} p(x+h) - \Delta^{i-1} p(x)$$

$$\Delta^0 p(x) := p(x)$$
- Bemerkung: $\Delta^i p(x)$ ist ein Polynom vom Grad $n-i$
- Beweis:
$$\begin{aligned} \Delta^1 p(x) &= p(x+h) - p(x) \\ &= a_0 + \dots + a_n(x+h)^n - a_0 - \dots - a_n x^n \\ &= a_0 + \dots + a_n \sum_{i=0}^n \binom{n}{i} x^i h^{n-i} - a_0 - \dots - a_n x^n \\ &= a'_0 + \dots + a'_{n-1} x^{n-1} \end{aligned}$$
- Rest per Induktion

G. Zachmann Informatik 2 - SS 06 Preprocessing 10

- Korollar: $\Delta^n p(x) \equiv \text{const}$
 - **Achtung:** auf dieser Beobachtung basiert das ganze Verfahren!
- Bemerkung: $\Delta^i p(x)$ hängt ab von $p(x), p(x+h), \dots, p(x+ih)$
- Beispiel:

$$\begin{aligned} \Delta^2 p(x) &= \Delta^1 p(x+h) - \Delta^1 p(x) \\ &= (\Delta^0 p(x+h+h) - \Delta^0 p(x+h)) \\ &\quad - (\Delta^0 p(x+h) - \Delta^0 p(x)) \\ &= p(x+2h) - 2p(x+h) + p(x) \end{aligned}$$

Preprocessing 11

Das Verfahren

- **Vorwärtsdifferenzen-Pyramide:**
 - vereinfachende Schreibweise $\Delta^i p(x+jh) =: \Delta^i p_j$
 - die Pyramide:

$x+ih$	p	Δ^1	Δ^2	...	Δ^n
x	p_0	$\Delta^1 p_0$	$\Delta^2 p_0$		$\Delta^n p_0$
$x+h$	p_1	$\Delta^1 p_1$	$\Delta^2 p_1$		$\Delta^n p_1$
$x+2h$	p_2	$\Delta^1 p_2$	$\Delta^2 p_2$		$\Delta^n p_2$
\vdots	\vdots	\vdots	\vdots		\vdots
$x+nh$	p_n	$\Delta^1 p_{n-1}$	$\Delta^2 p_{n-2}$		$\Delta^n p_1$
$x+(n+1)h$	p_{n+1}	$\Delta^1 p_n$	$\Delta^2 p_{n-1}$		$\Delta^n p_0$

$$\Delta^{i+1} p_j = \Delta^i p_{j+1} - \Delta^i p_j$$

$$\Delta^i p_{j+1} = \Delta^i p_j + \Delta^{i+1} p_j$$

Preprocessing 12



- Algo zur Berechnung von vielen äquidistanten Werten von p :
 1. initialisieren die Pyramide mit den Punkten $x, \dots, x + nh$
 2. berechne den nächsten Punkt an der Stelle $x + (n+1)h$ durch Fortsetzen der Pyramide um eine Zeile am unteren Ende
- Aufwand: für jeden neuen Punkt braucht man nur n Additionen (keine Multiplikation)
- Bemerkungen:
 - man muß eigentlich nur die unterste Zeile der Pyramide speichern
 - Rundungsfehler akkumulieren sich → Pyramide ab und zu neu aufsetzen
- Durch Umformung kann man ein Polynom vom Grad n mit $\frac{n}{2}$ Multiplikationen an beliebigen Stellen auswerten (s. Knuth)

G. Zachmann Informatik 2 - SS 06 Preprocessing 13