Wie kann das LCP-Array als Maß für die Komprimierbarkeit genutzt werden?

Überlegungen:

- Welche Texte sind gut komprimierbar?
  - ↳ All-a-Text: aaaaa...a $\xrightarrow{\text{LZ77}}$ (0, a)(1, n-1)
  - ↳ Texte mit vielen Wdh.: abc...xyz a a de a xy a
- Welche Texte lassen sich nicht gut komprimieren?
  - ↳ Texte mit wenigen Wdh. oder nur sehr kurzen Wdh.

abc ab d abc ab f... Abspeichern der Faktoren kann teuer werden

→ Was bedeutet das für das LCP-Array

- ↳ All-a-text: LCP: n-1 | n-2 | n-3 | ... | 0
- ↳ Texten mit vielen Wdh.: | 0 | 2 | 0 | \|a\| | \|a\| | \|a\| + 4 | ...
- ↳ Wenige Wdh. | 0 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | ...

Also etwas konkreter

↳ „Je höher die Summe der LCP-Werte ist, desto komprimierbarer sollte ein Text sein" (evtl + Normierung durch die Länge des Textes)

↳ Was ist z.B. mit dem Maximum

↳ a...a b...b

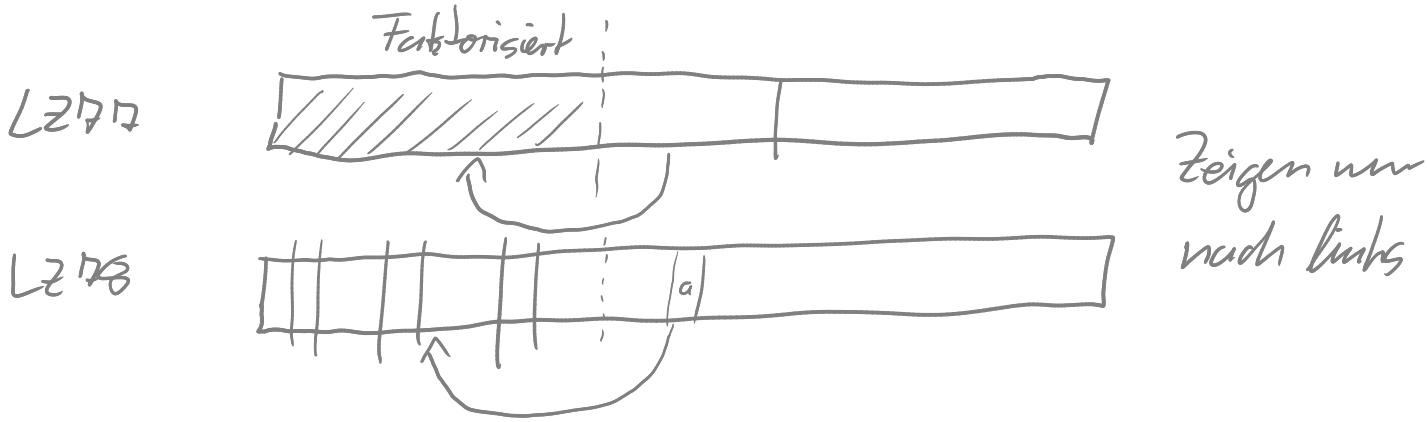
n/2 n/2

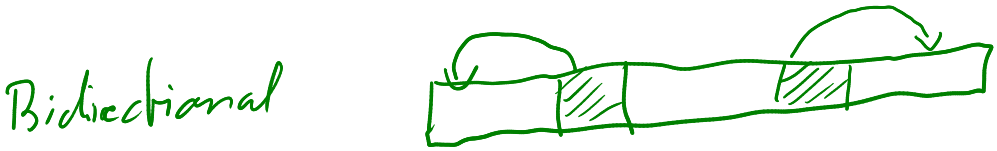
Das LCP-Array zum Komprimieren nutzen

Basierend auf „Bidirectional Text Compression in External Memory", Dinklage et al. ESA '19.

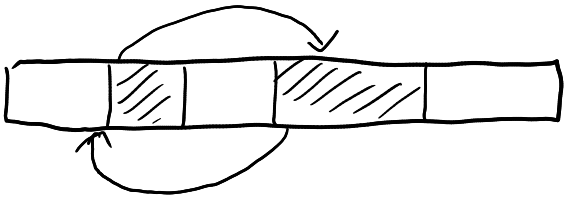
EM: ist uns egal (für diese Übung)

Bidirectional: Faktoren dürfen in beide Richtungen zeigen





Achtung: Kreise!



Die Idee von PLCP-Comp

PLCP[SA[i]] = LCP[i]

↳ „LCP-Array in Textreihenfolge"

↳ Das PLCP-Array wird beim „engineered linear time"-Algorithmen konstruiert

① Suche maximalen Wert im PLCP-Array
↳ Konstruiere aus diesem Wert einen Faktor
↳ Faktorisiere ab Textpos. i: $(\phi[i], PLCP[i])$

Text-Pos Länge

Mit wem möchte ich verglichen werden

$\phi[i] = SA[ISA[i]-1]$

Hier kommen Faktoren in beide Richtungen zeigen

Mit welchem Suffix kommt der LCP-Wert zustande

② PLCP-Array aktualisieren
a) Alles was überdeckt ist löschen
b) Alles was ein Präfix von dem Faktor ist (*)
muss reduziert werden (*)

③ Weiter bei ①, so lange sinnvolle Faktoren existieren

Die Implementierungen von LZ77 / LZ78

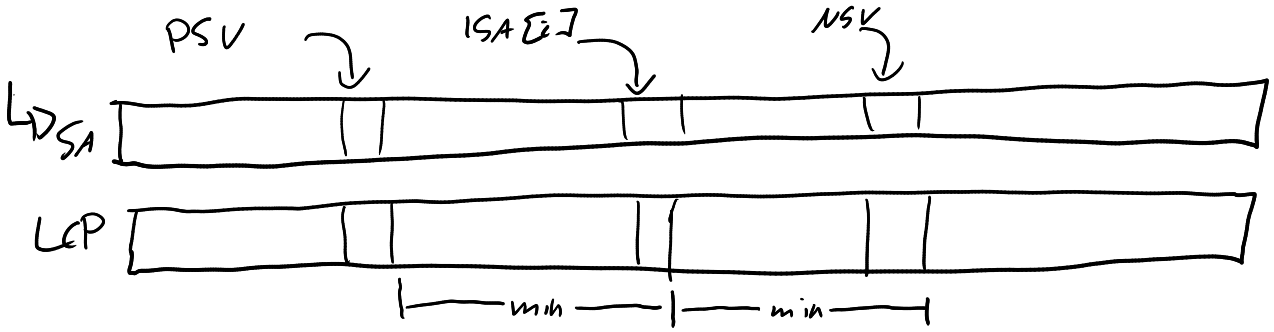
LZ77

Faktoren: Länge, Text Position (wenn Länge = 1, dann steht hier das Zeichen)

PSV:

(NSV analog)

Berechnung der Faktoren
↳ i ist Textpos. ab der der nächste Faktor beg.



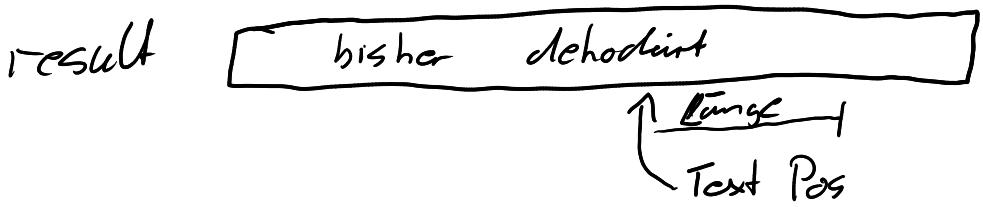
struct LZ77 mit Konstruktor erlaubt verwendung von emplace_back

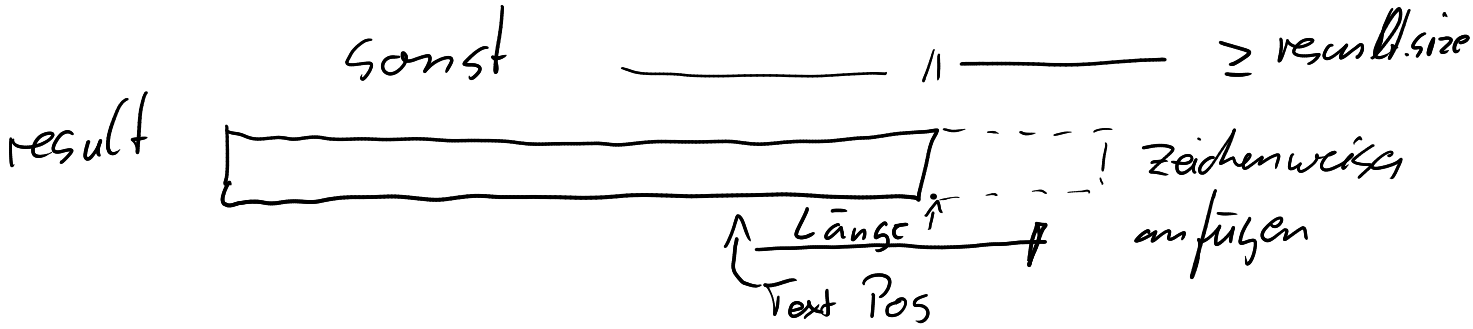
↳ Konstruiert an der richtigen Stelle im Speicher

↳ Kann eine Kopie sparen

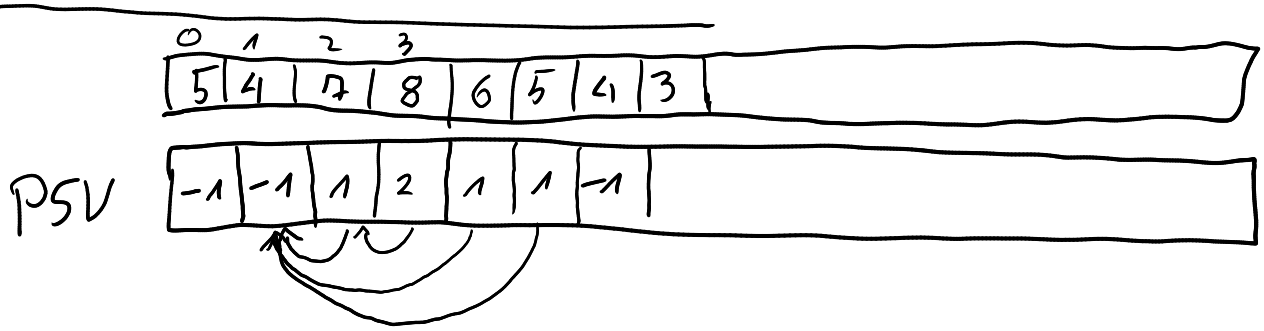
Beim Dekomprimieren

↳ Länge > 1 und factor.text_pos + factor.length < res.size





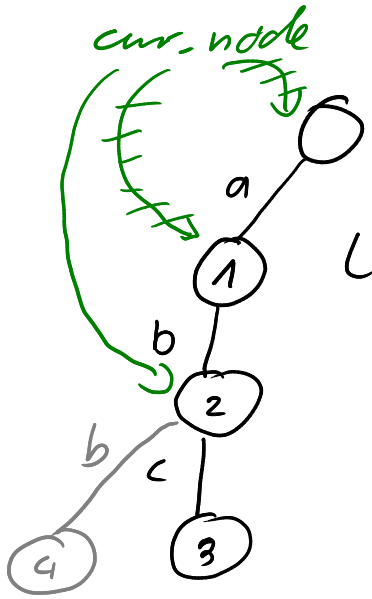
PSV in Linearzeit



LZ78

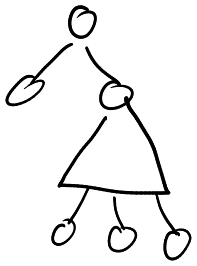
Faktoren: Referenz auf anderen Faktor (als Zahl mit 1 beg.), Zeichen

Ablauf:



a|ab|abc|abb

LZ78: (0,a)(1,b)(2,c)(2,b)



Dekomprimieren

↳ Erzeuge Dictionary für alle Faktoren

Eingabe: Faktor-Nummer

Ausgabe: „String, den der Faktor kodiert"

↳ Pfad von der Wurzel bis zum Knoten des Faktors

In der Implementierung geben wir Textpos. + Länge zurück

	0	1	2	3	4
Dict:	{}	(0,1)	(1,2)	(3,3)	(6,3) ..... ( ) ( )

	0	1	2	3	4	5	6	7	8
result:	a	a	b	a	b	c	a	b	b