technische universität
dortmund

Plug&Play Kompression mit dem Framework

tudocomp

Dominik Köppl

Vortragsreihe der Regionalgruppe der
Gesellschaft für Informatik aus Dortmund
4. Juni 2018

tudocomp

- C++14
- Open-Source (Apache-License)
- Github

Verlustfreie Datenkompression

- Forschung seit 70'er Jahren
- wichtiger EDV-Bestandteil
  - Archivierung
  - komprimierte Dateisysteme
  - Datenübertragung

Probleme

- Kompressionsgüte
  - NP schwierig
- Geschwindigkeit
- RAM Verbrauch

für Kompression und Dekompression

Lösungen

- Heuristiken
- Oft Spezial-Kompressoren für bestimmte Einsatzgebiete notwendig
  - Gen-Sequenzen (FASTA)
  - XML

Kompressor / Kodierer

- Kompressor
  - Datenstrom
- Kodierer
  - Buchstaben
  - Zahlen
- Jeder Kodierer ist ein Kompressor

Arten verlustfreier Kompression

- Entropie-basierte Kompressoren
  - z.B. Huffman
- Wörterbuch-Kompressoren

	Generell	**Bilder**
LZ77	gzip, WinZip, WinRAR, 7zip	png
LZ78	compress	gif

Kompressionsbenchmarks

- Squash Compression Benchmark
- Large Text Compression Benchmark
- TurboBench: Compressor Benchmark
- lzbench
- ...

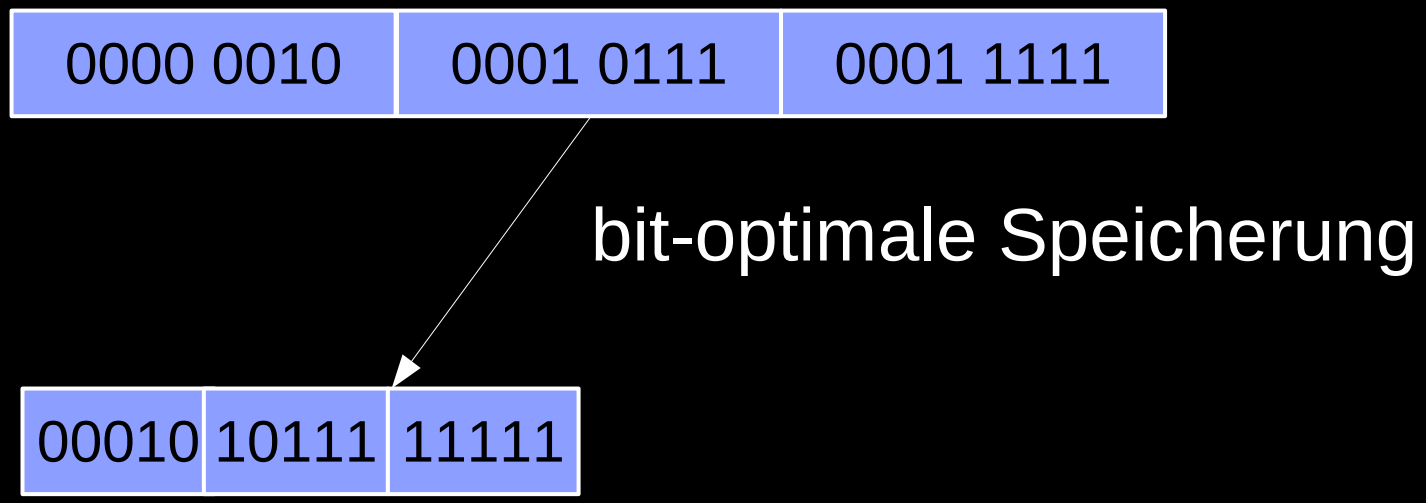
Alternativen

- OpenSource Kompressor modifizieren
  - Quellcode oft schlecht dokumentiert
  - Low-Level-Programmierung kryptisch
  - keine Referenzimplementierung vorhanden
- ExCom : Universität Prag
  - verwaist (~2013)
  - Plug&Play auf Byte-Ebene (**nicht bit-optimal**)

Bit-Optimale Integer

Problem

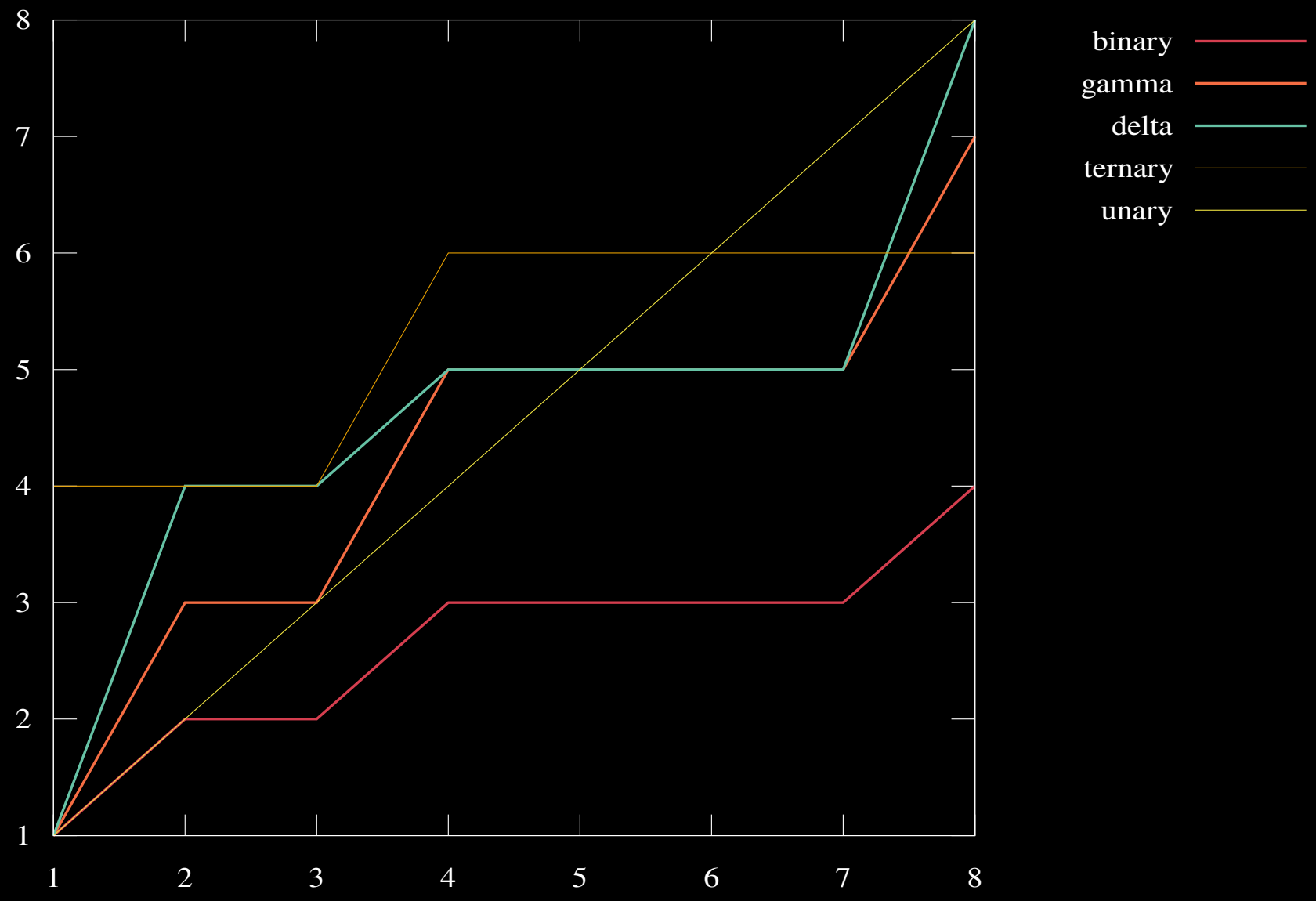
- Integer nur in fixen Byte-Größen verfügbar
- Idee: Frei wählbare Bitweite



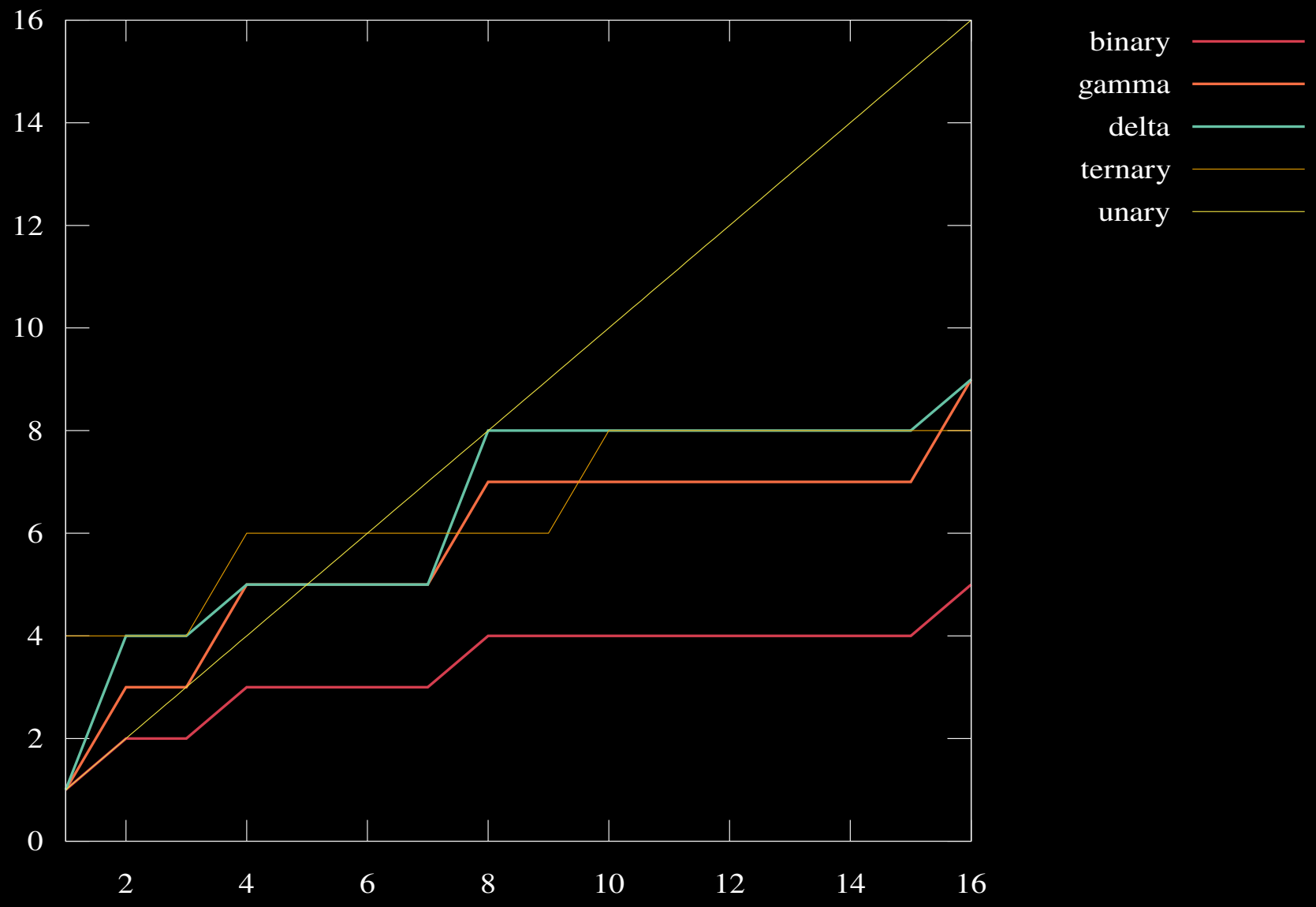
Komprimierte Integer

- Kodierungen
  - Elias γ
  - Elias δ
  - Rice
  - variable Byte
  - ternary

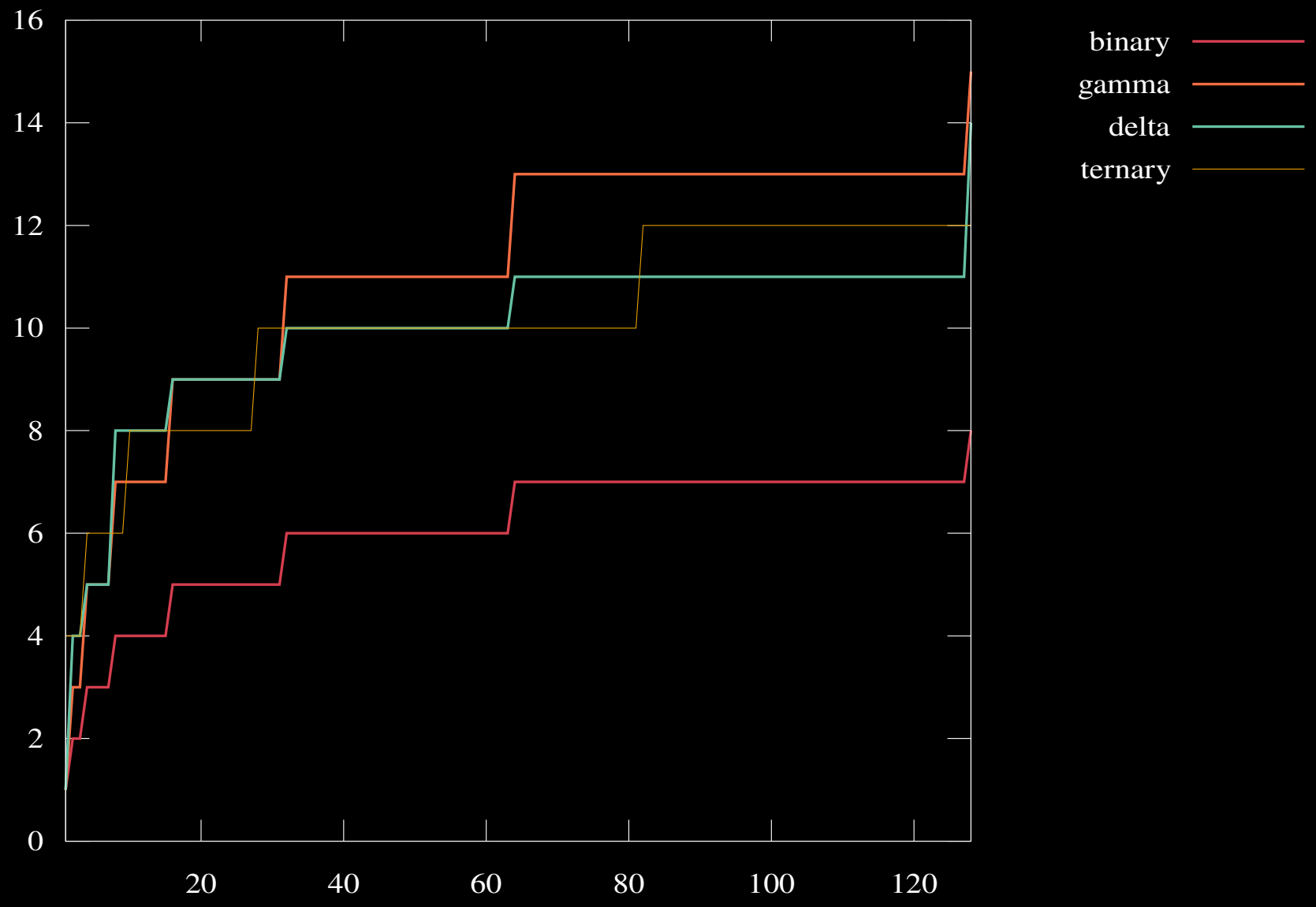
Integer-Kodierer



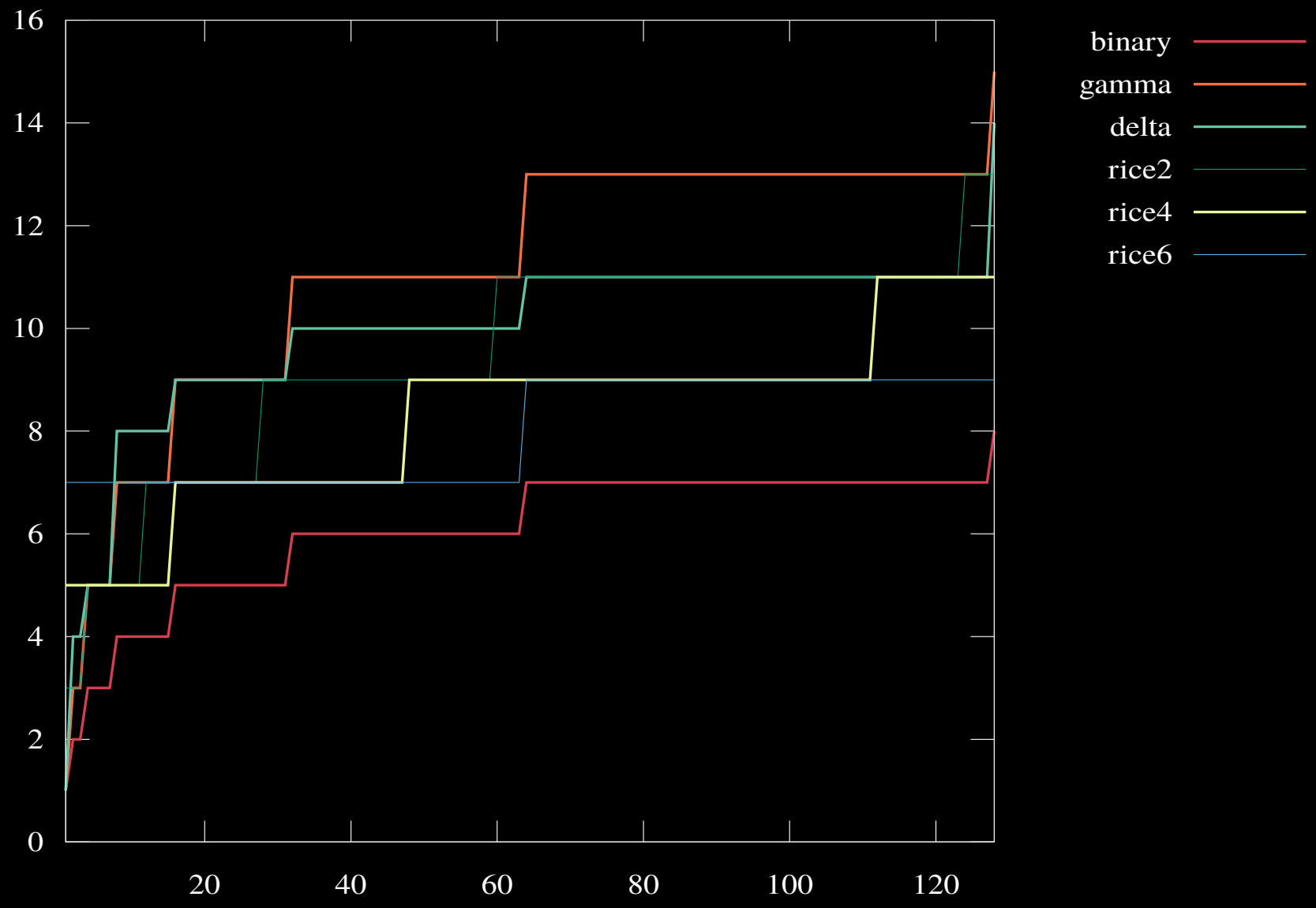
Integer-Kodierer



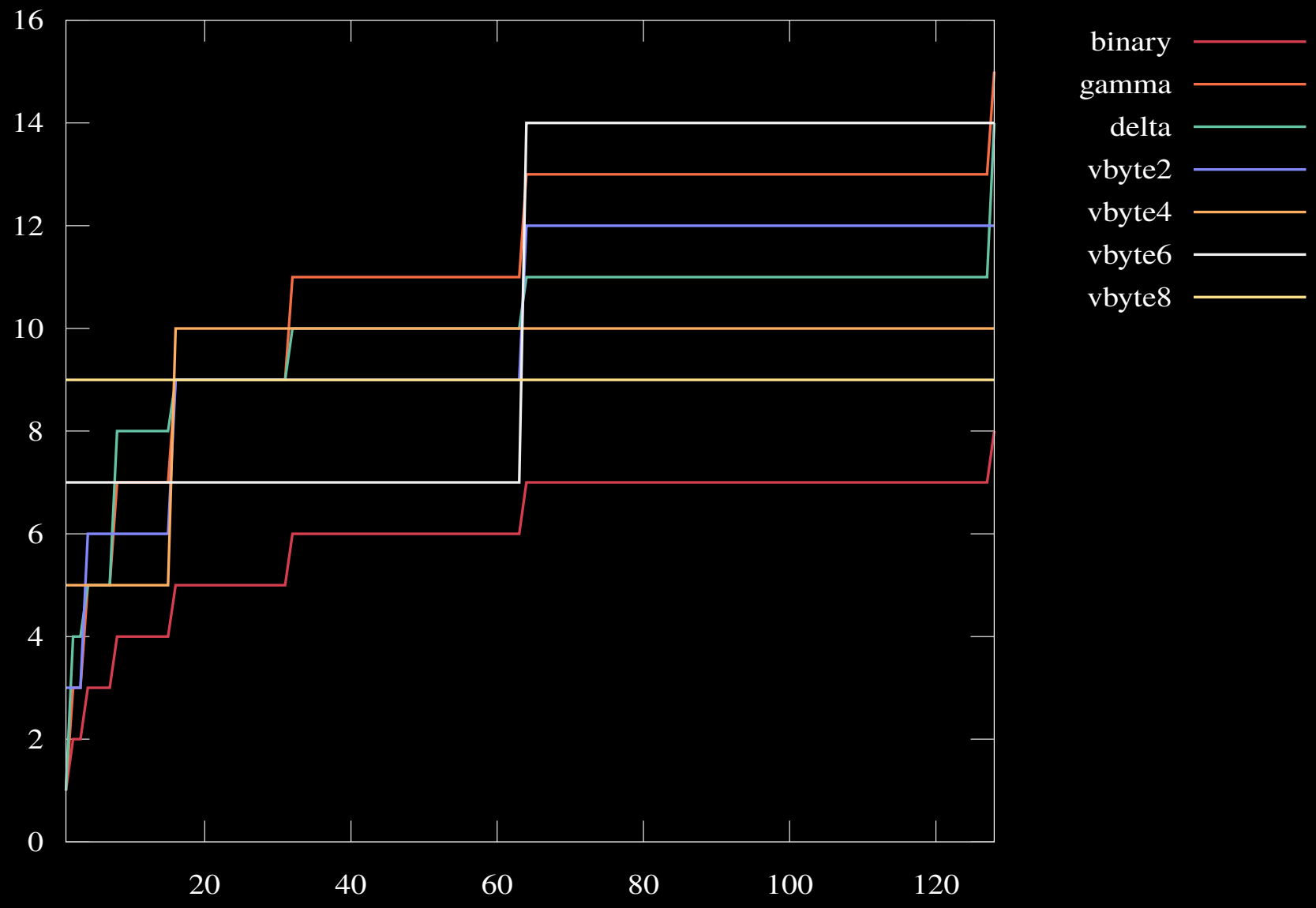
Integer-Kodierer



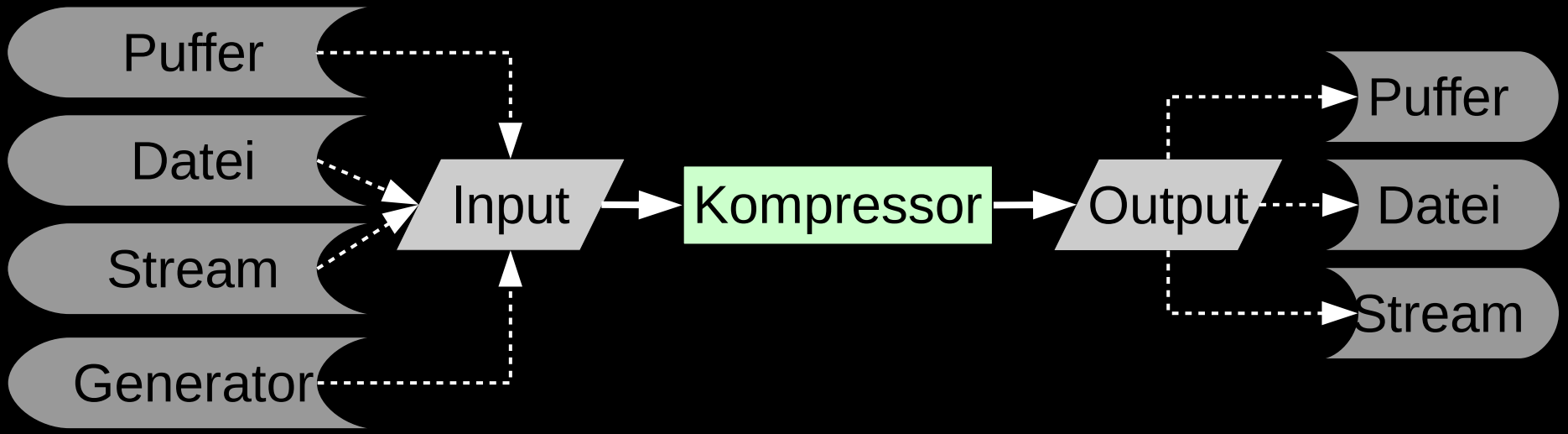
Rice-Kodierer



vByte-Kodierer



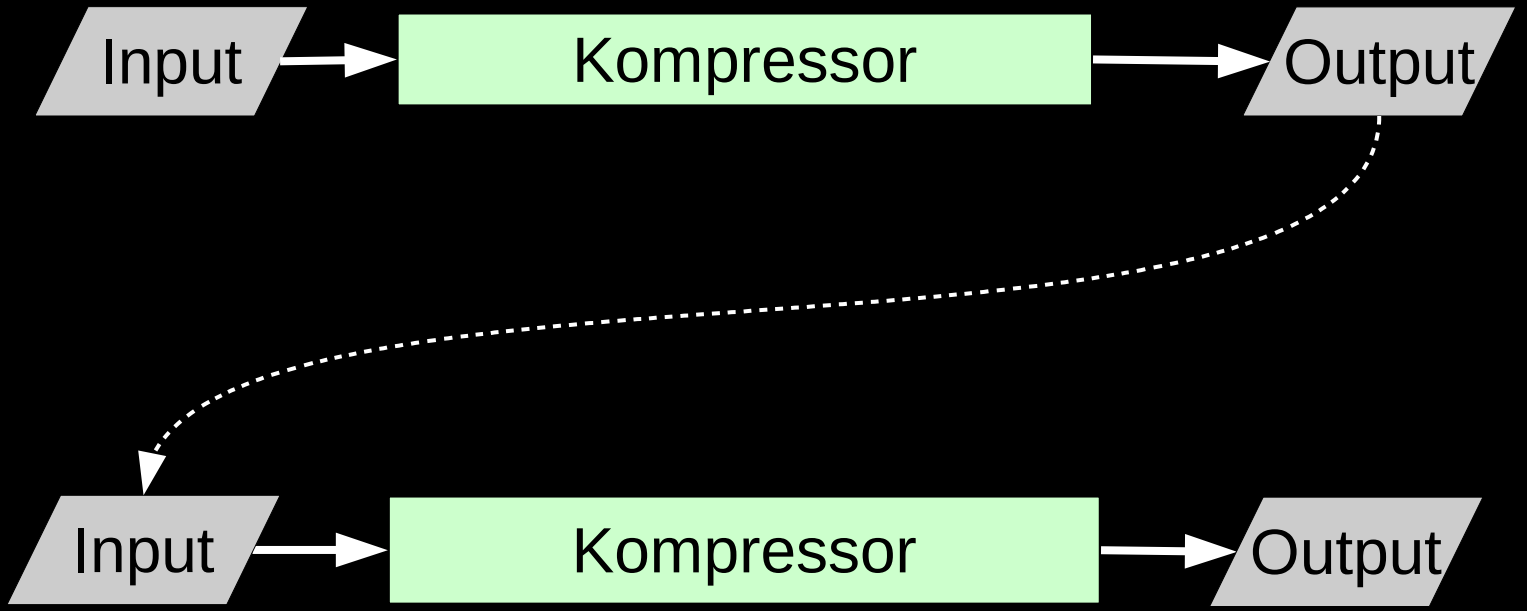
Ein/Ausgabe


Puffer
Datei
Stream
Generator
Input
Kompressor
Output
Puffer
Datei
Stream

Aufruf per Pipe

```
echo “Hallo” | ./tdc -a 'noop' --usestdout --usestdin
```

Verkettung



Verkettung

- Burrows-Wheeler-Transformation
- Run-Length-Encoding
- Move-To-Front-Encoding
- Huffman-Coder

```
./tdc -a 'bwt:rle:mtf:encode(huff)' -g 'fib(4)' --usestdout
```

viertes Fibonacci Wort

Ausgabe in der Konsole

Statistiken

Stastistiken mit Parameter `--stats`

- Aufruf in Phasen gegliedert
- JSON Format
- Visualiserung mit JavaScript

http://tudocomp.org

Komponenten

Kompressoren

- LZ77
- LZ78
- BWT
- etc.

Kodierer

- **Integer**
- **Statistisch**
  - Huffman
  - Arithmetic

String-Generatoren

- Zufallsstrings
- Thue-Morse Sequenz
- Fibonacci-Wörter

Speichersparsame DS

- Bit-kompakte Integer-Arrays
- Bitweises I/O-Streaming

Benchmarking

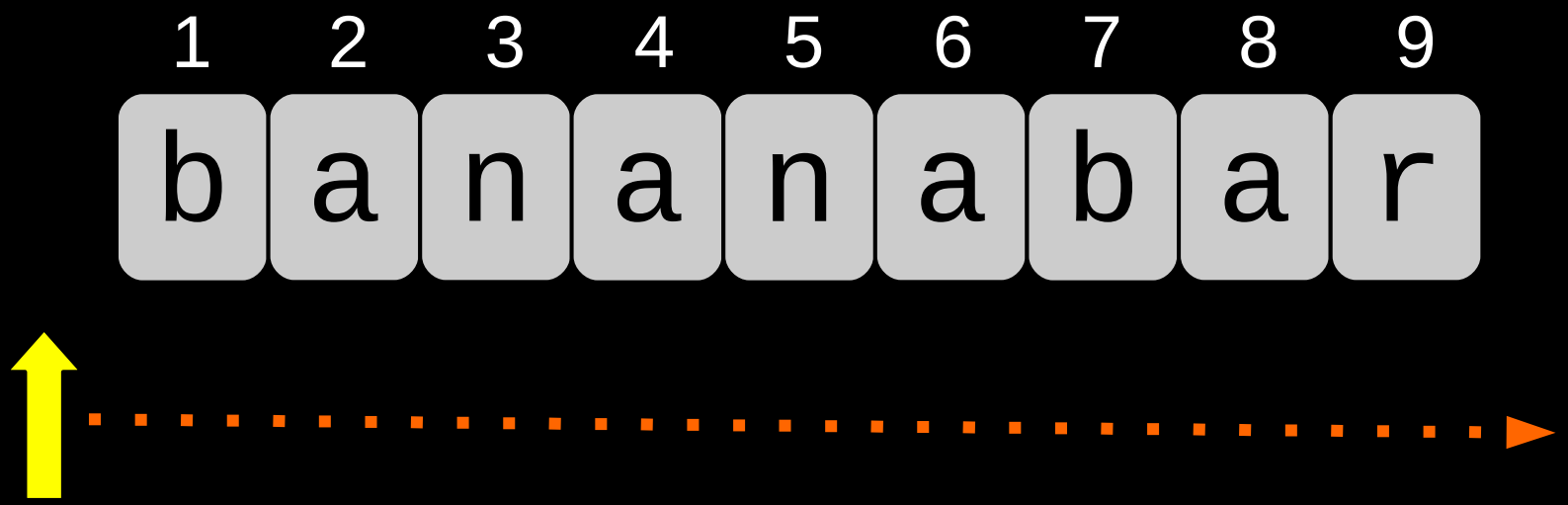
- Benchmark-Suite von 200MiB Texten
- DNA, natürlich sprachig, Quellcode, etc.
- repetitiv <-> schwer-komprimierbar
- Benchmark-Tool

Compressor	C Time	C Memory	C Rate	D Time	D Memory	chk
lcpcomp	103.1s	3.2GiB	2.8505%	36.6s	7.6GiB	OK
lz77	98.5s	2.9GiB	4.0530%	4.3s	230.6MiB	OK
bwt+mtf+rle	83.6s	1.7GiB	6.8688%	22.6s	1.4GiB	OK
huffman	2.7s	230.5MiB	28.1072%	5.9s	30.6MiB	OK
lzw	14.3s	480.9MiB	23.4411%	5.5s	452.6MiB	OK
lz78	13.6s	480.8MiB	29.1033%	10.3s	142.9MiB	OK
gzip -9	107.6s	6.6MiB	26.2159%	1.0s	6.6MiB	OK
bzip2 -9	13.8s	15.4MiB	25.2368%	5.6s	11.7MiB	OK
lzma -9	138.6s	691.7MiB	1.9047%	337.3ms	82.7MiB	OK

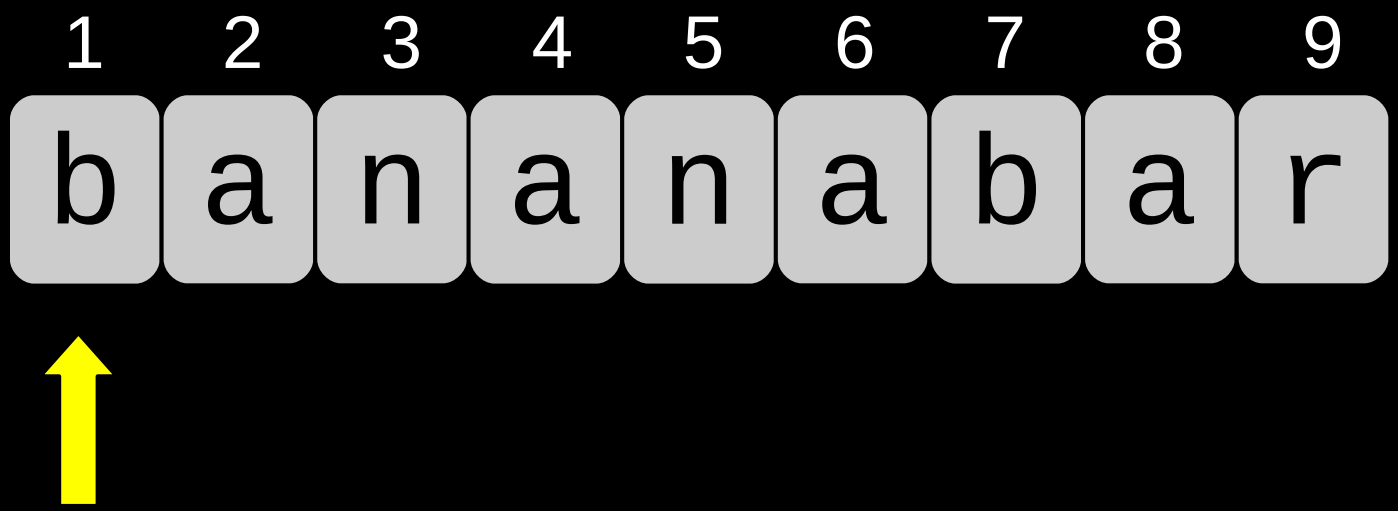
Einblick in zwei Kompressoren:

- LZ77
- LZ78

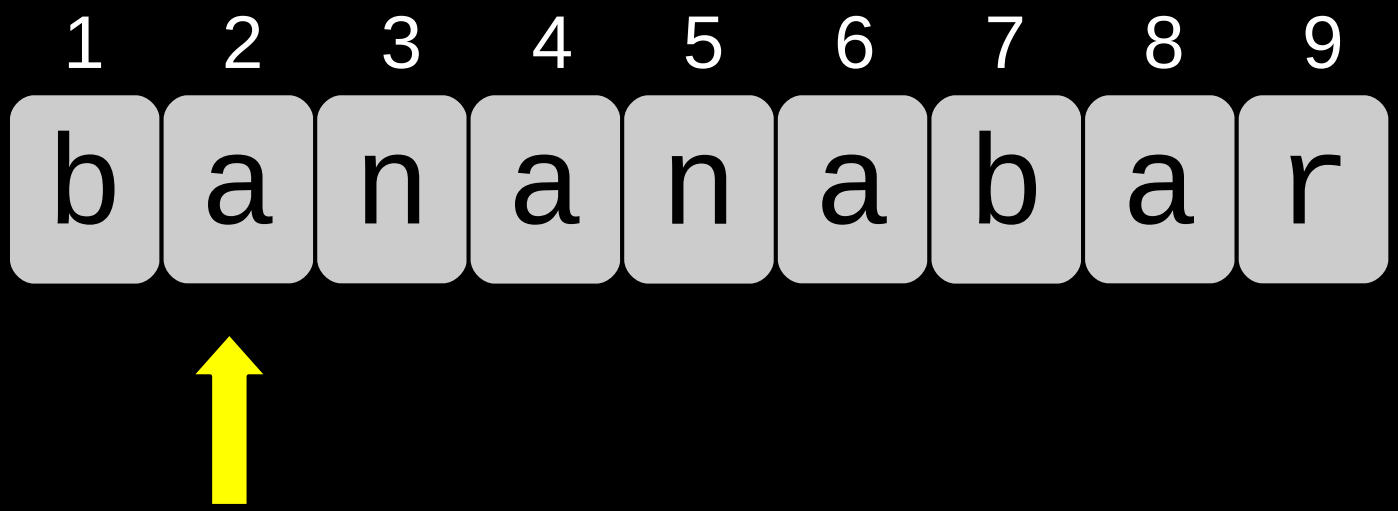
LZ77



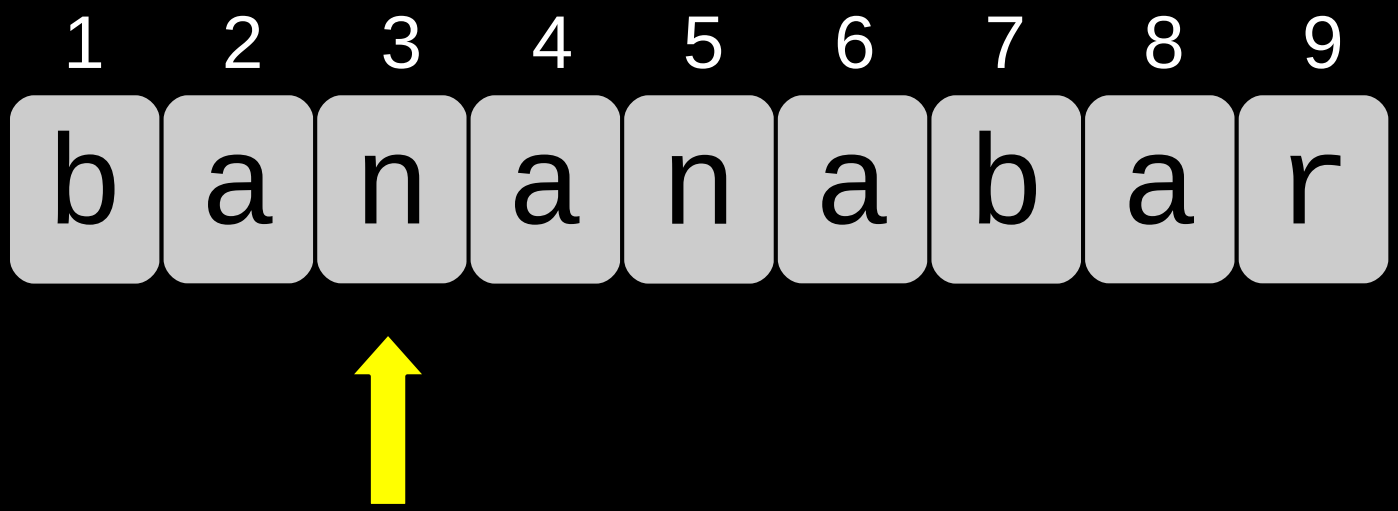
LZ77



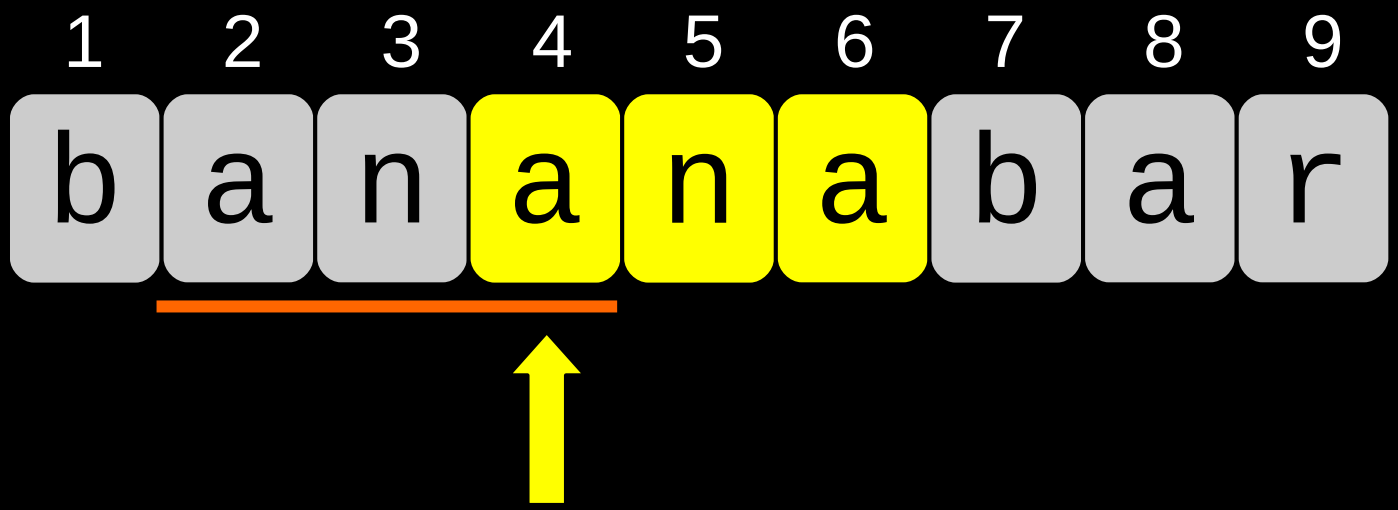
LZ77



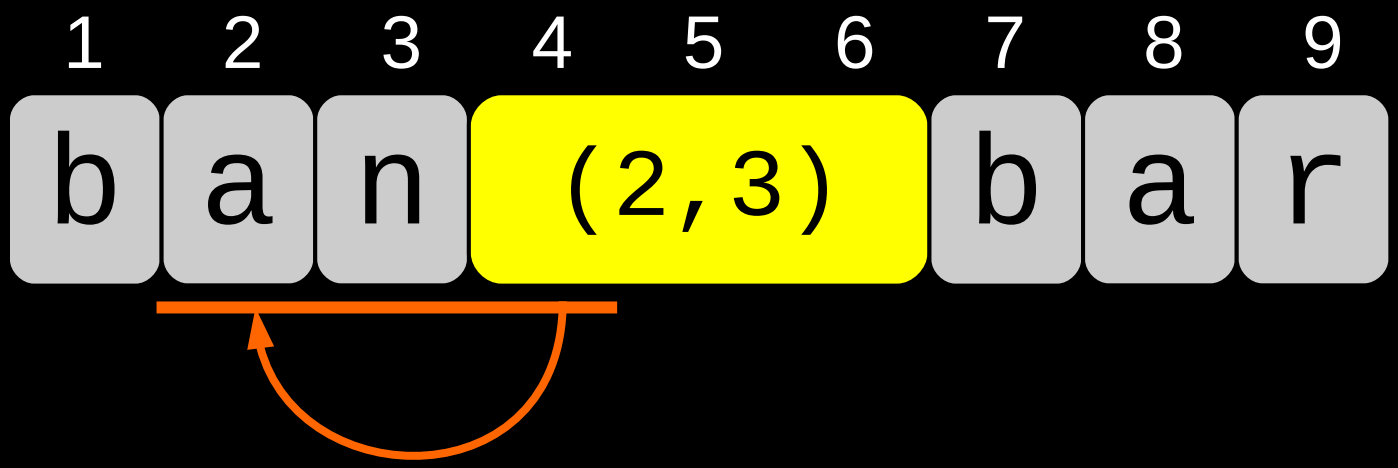
LZ77



LZ77

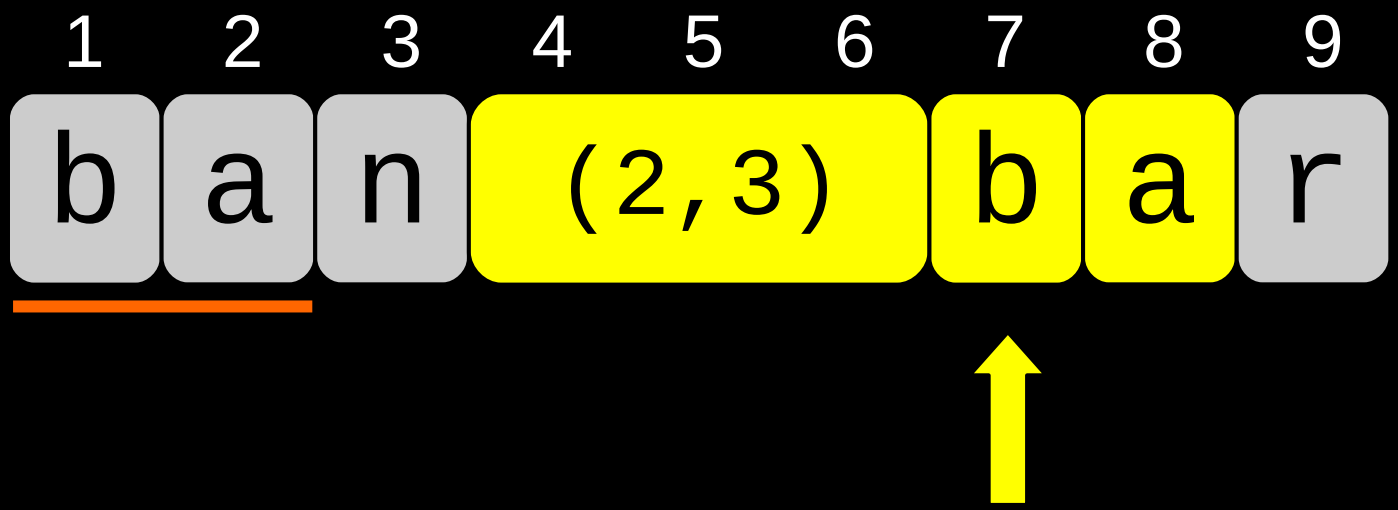


LZ77

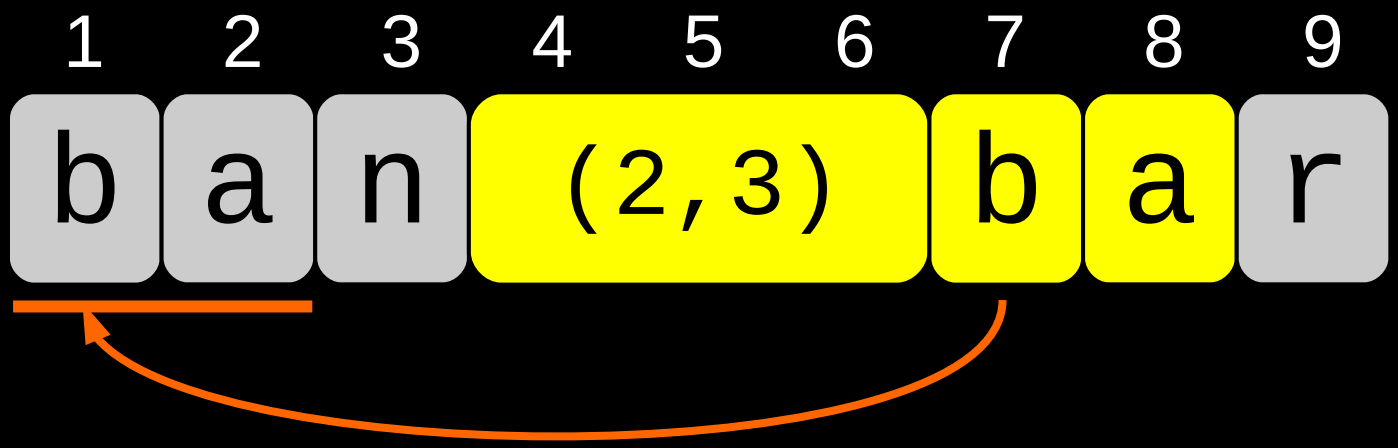


*Kopiere von Position **2** genau **3** Zeichen.*

LZ77

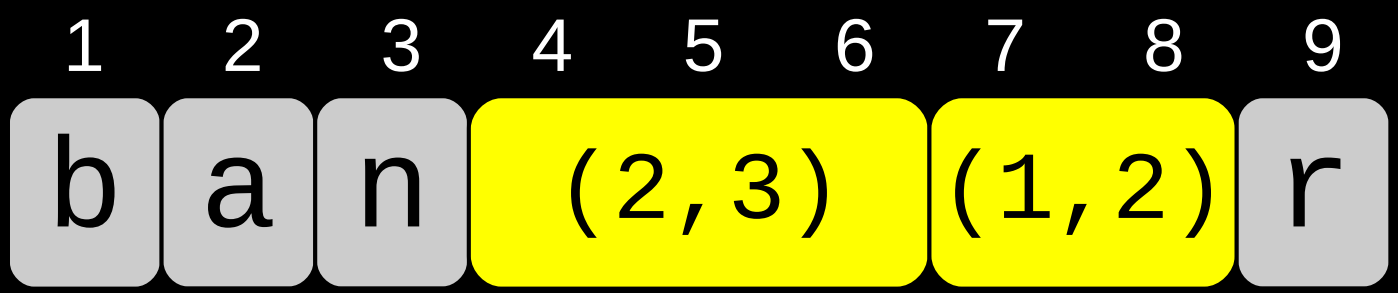


LZ77

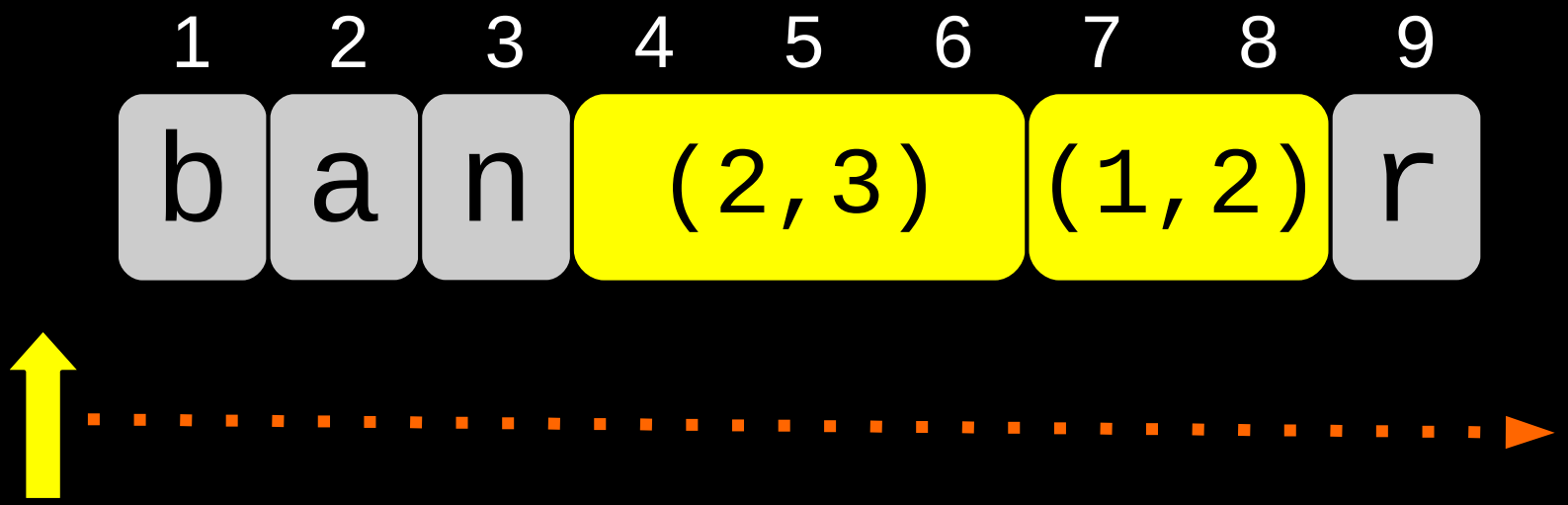


*Kopiere von Position **1** genau **2** Zeichen.*

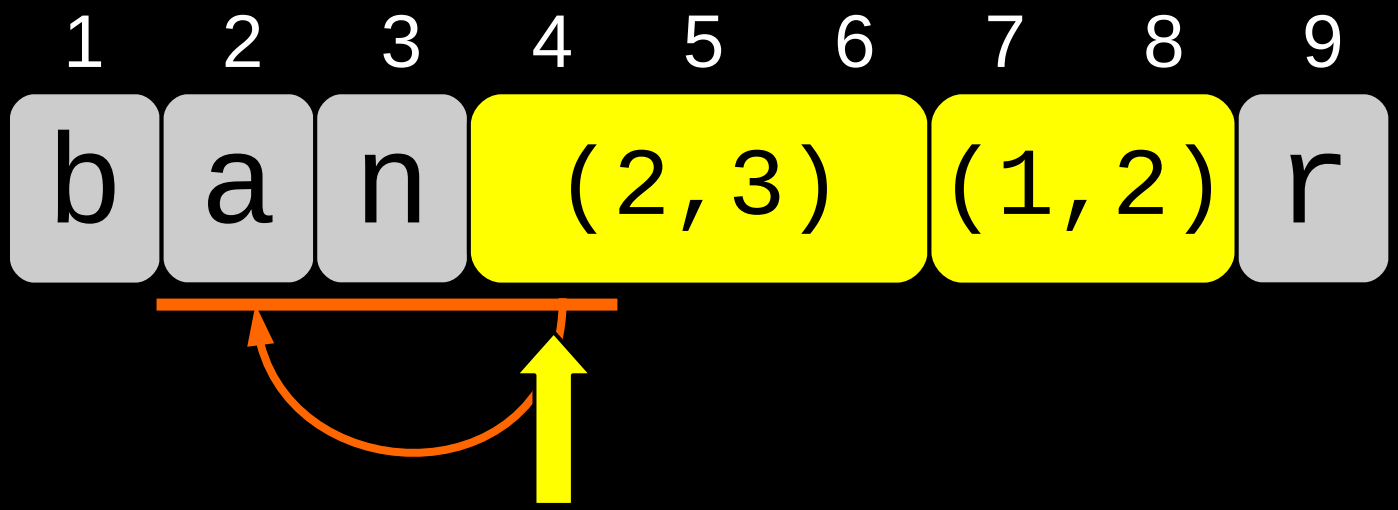
LZ77



LZ77 Dekompression

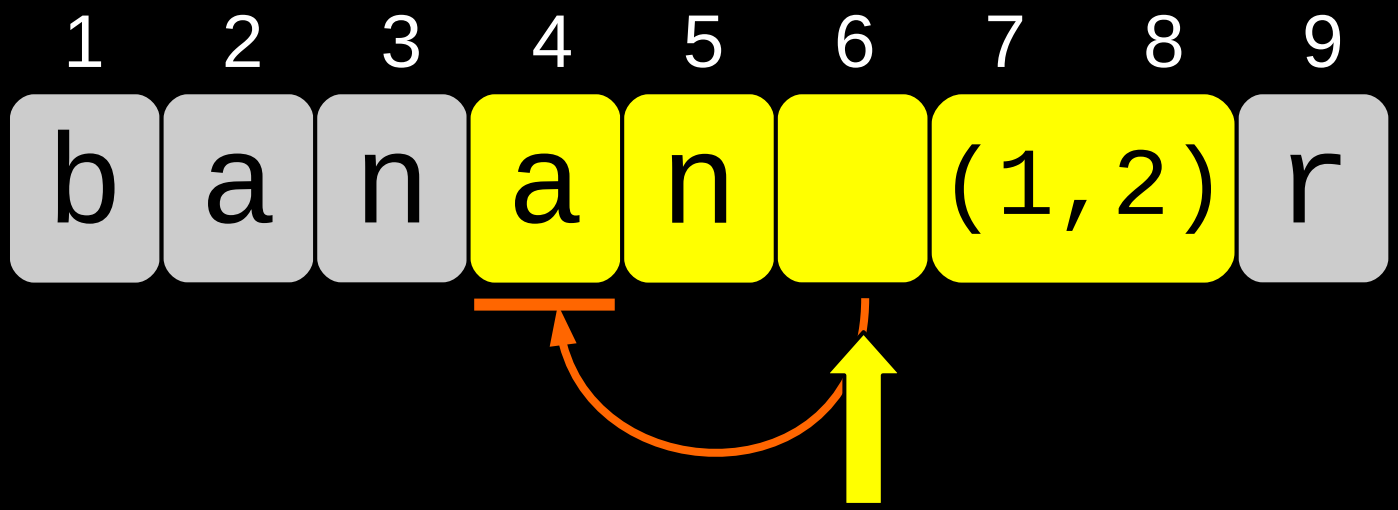


LZ77 Dekompression



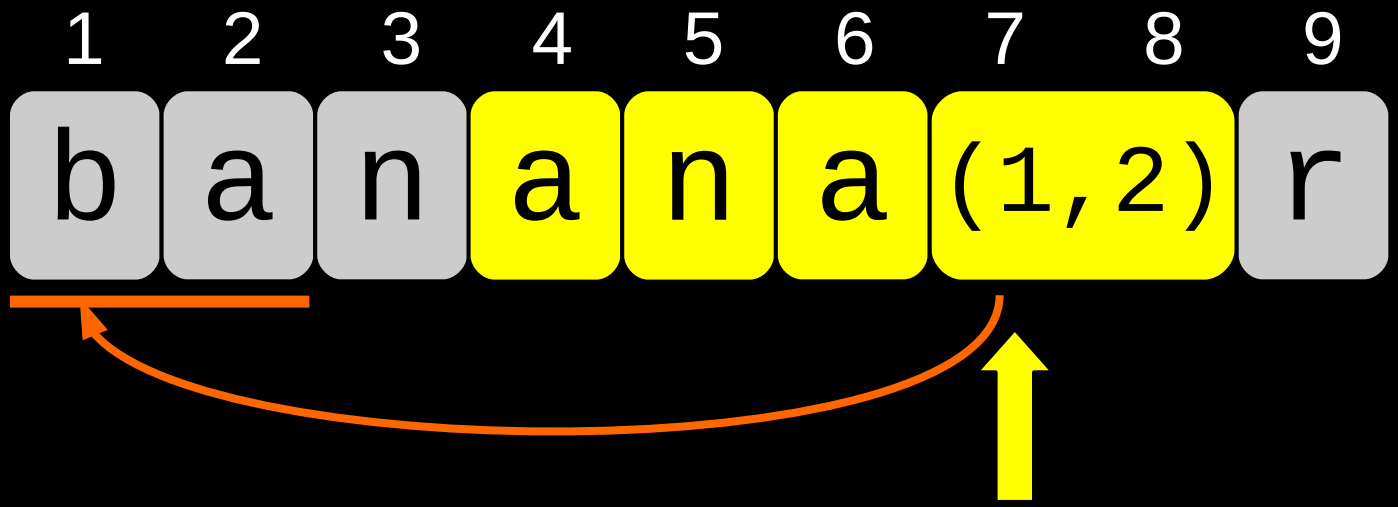
*Kopiere von Position **2** genau **3** Zeichen.*

LZ77 Dekompression



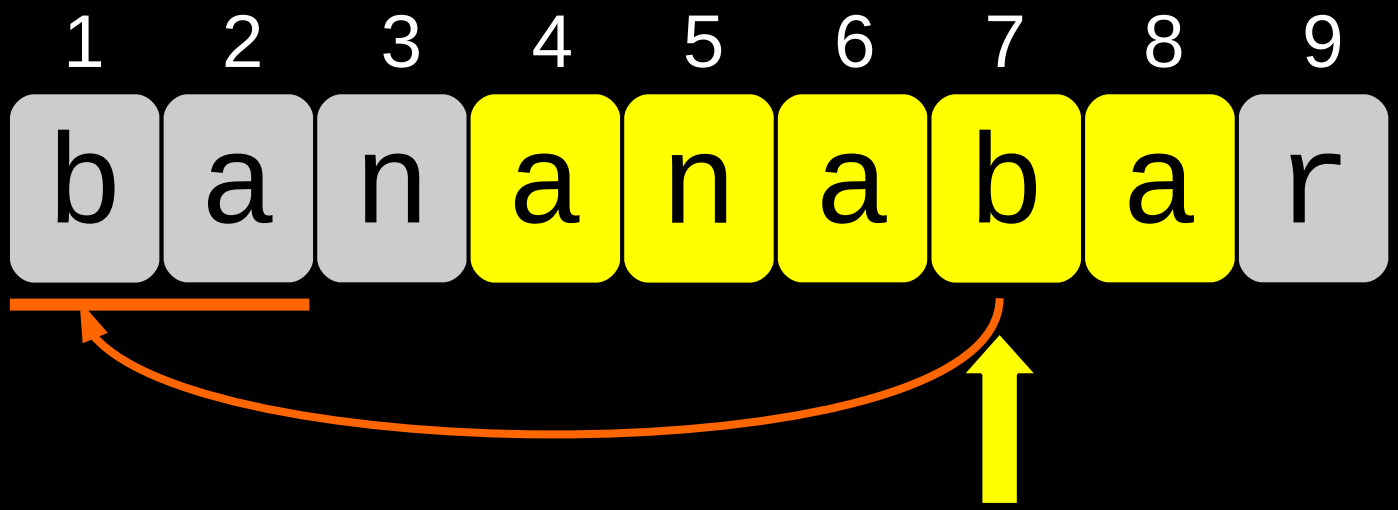
Fehlendes Zeichen bereits dekodiert

LZ77 Dekompression



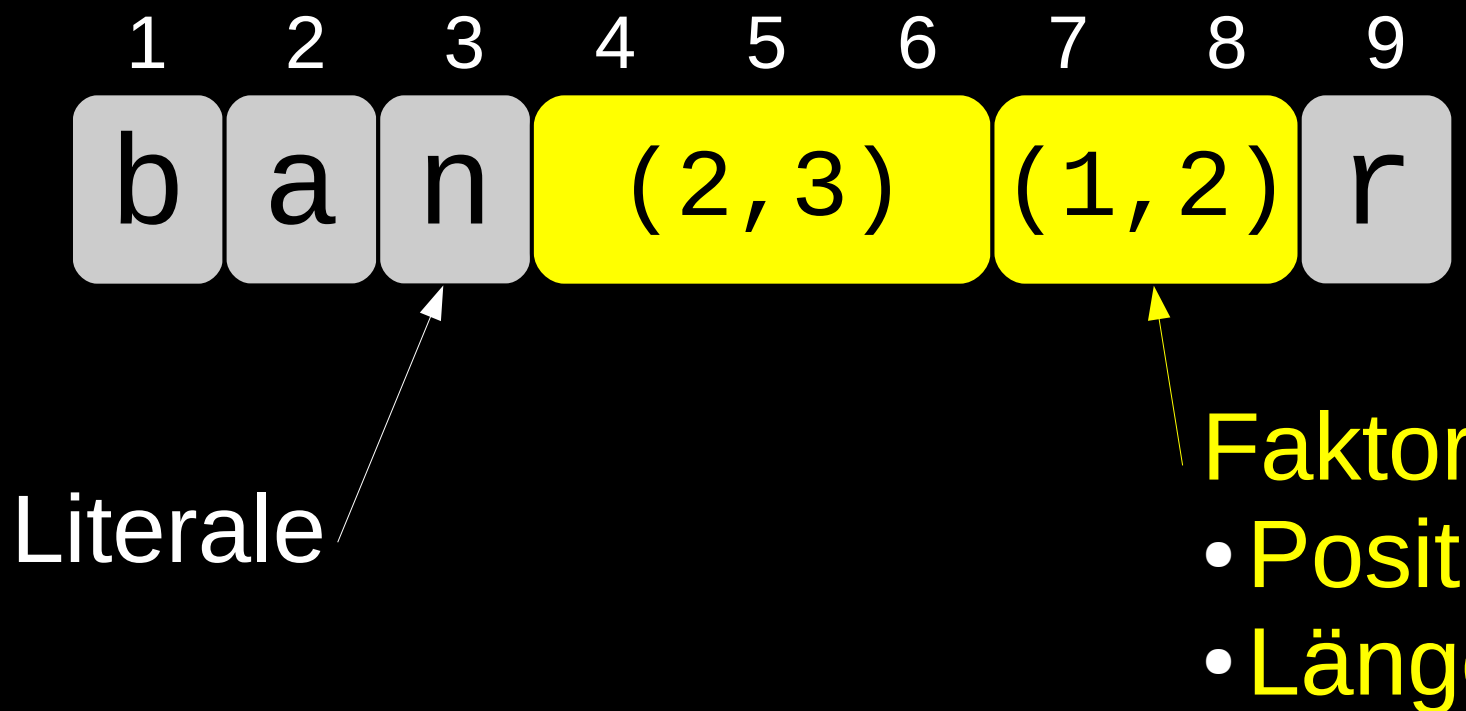
*Kopiere von Position **1** genau **2** Zeichen.*

LZ77 Dekompression

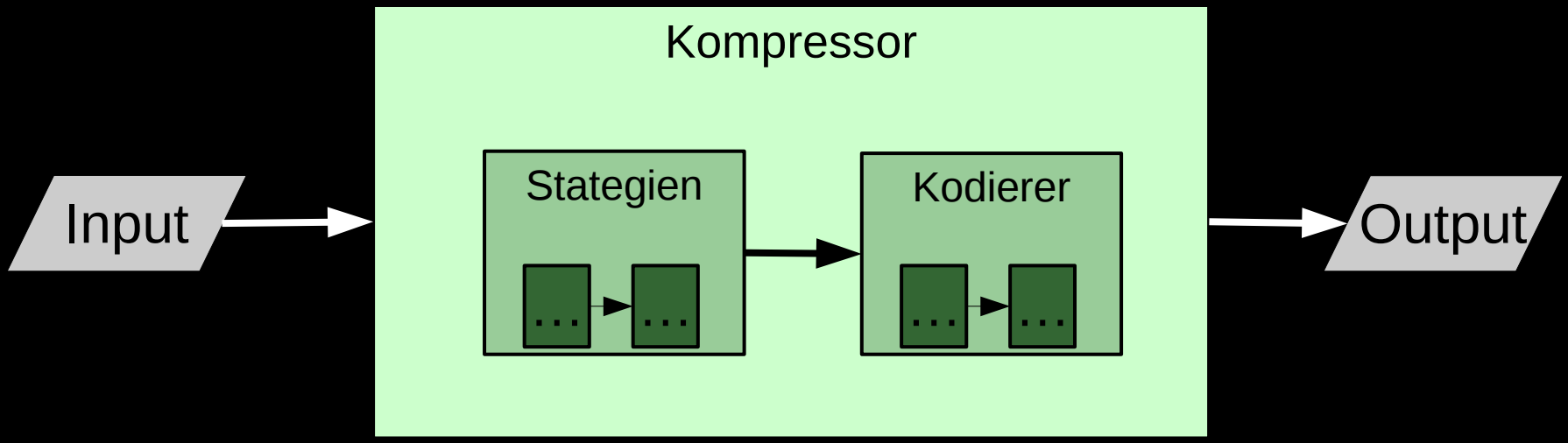


*Kopiere von Position **1** genau **2** Zeichen.*

LZ77 Kodierung

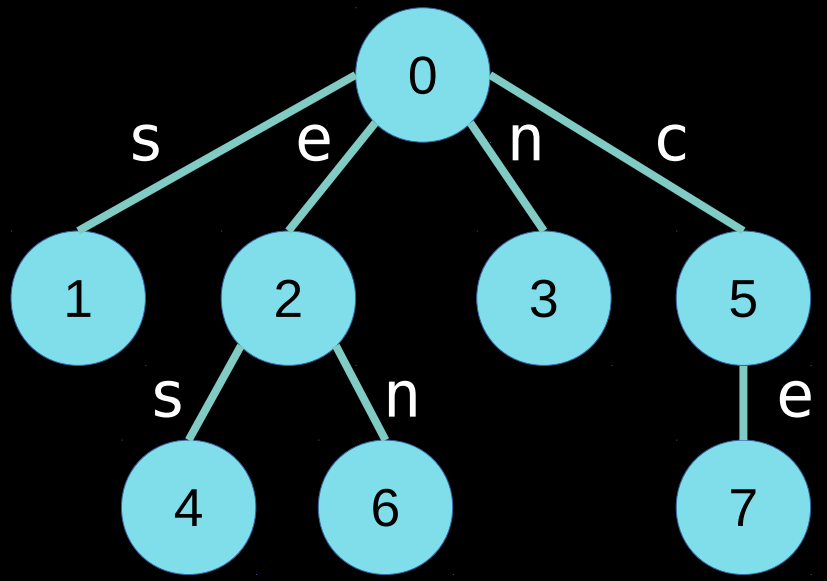


Modularität



LZ78

senescence

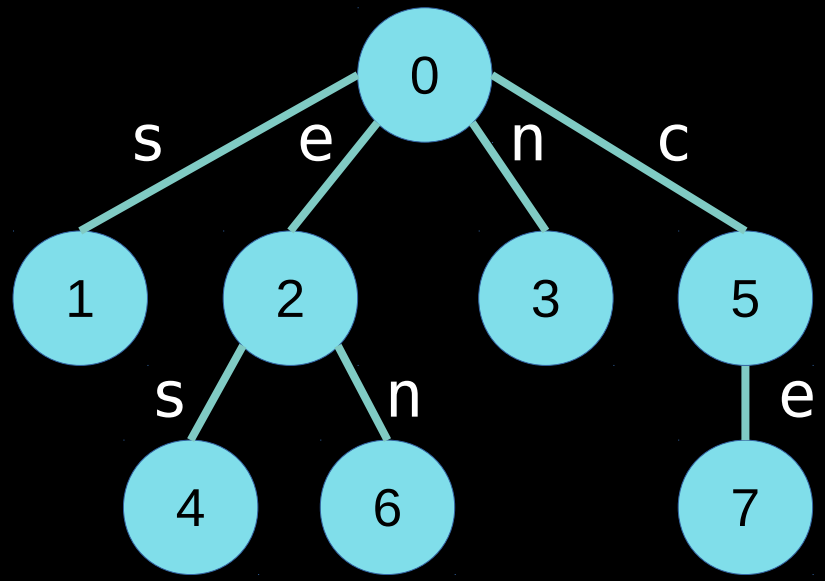


Output:

s,0 e,0 n,0 s,2 c,0 n,2 e,5

LZ78 Dekompression

senescence



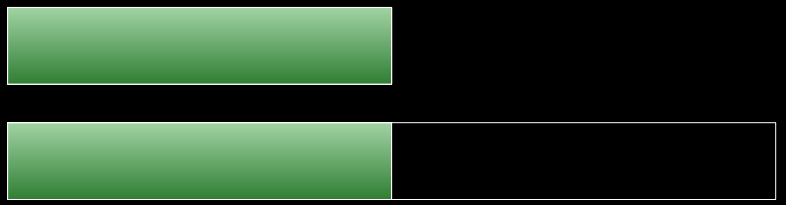
1	2	3	4	5	6	7
s,0	e,0	n,0	s,2	c,0	n,2	e,5

Trie Repräsentationen

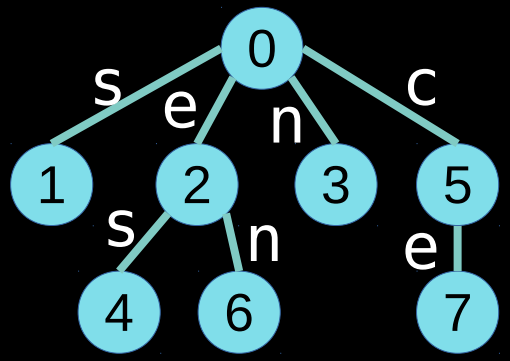
- binary trie [folklore]
- ternary trie [Bentley, Sedgewick'97]

dynamische Arrays

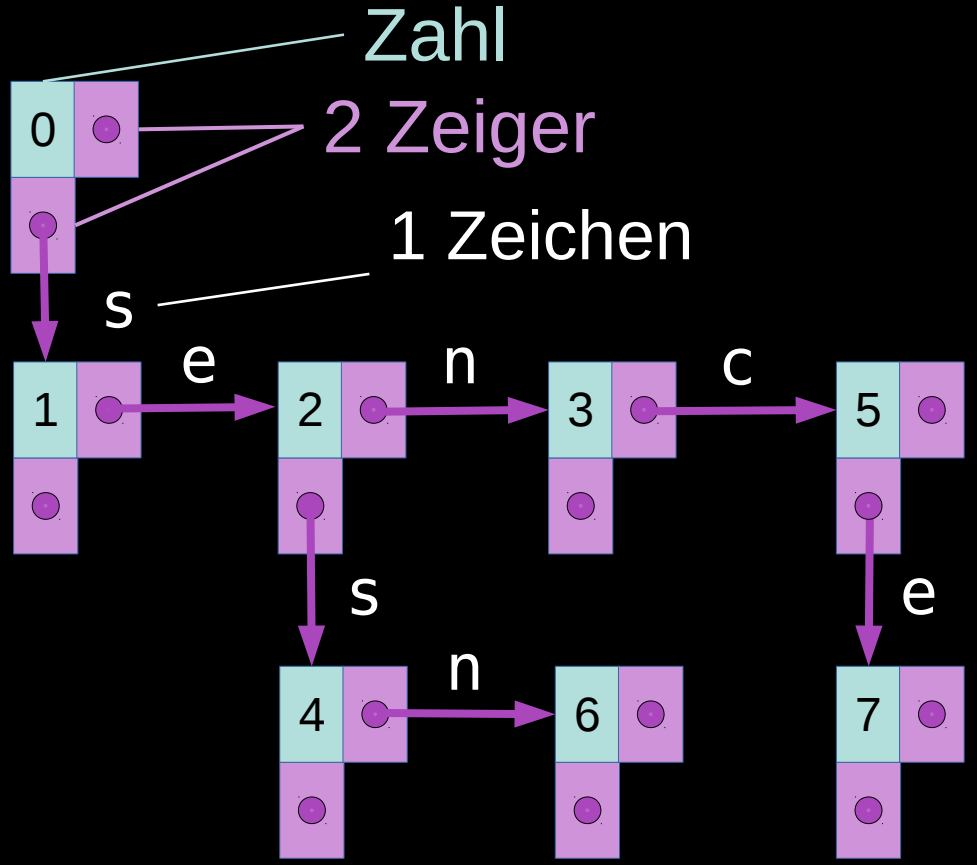
Platzverdopplung



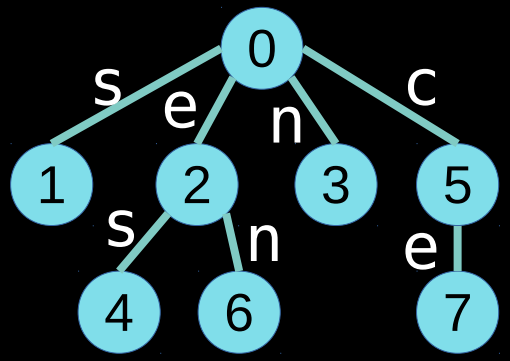
binary trie



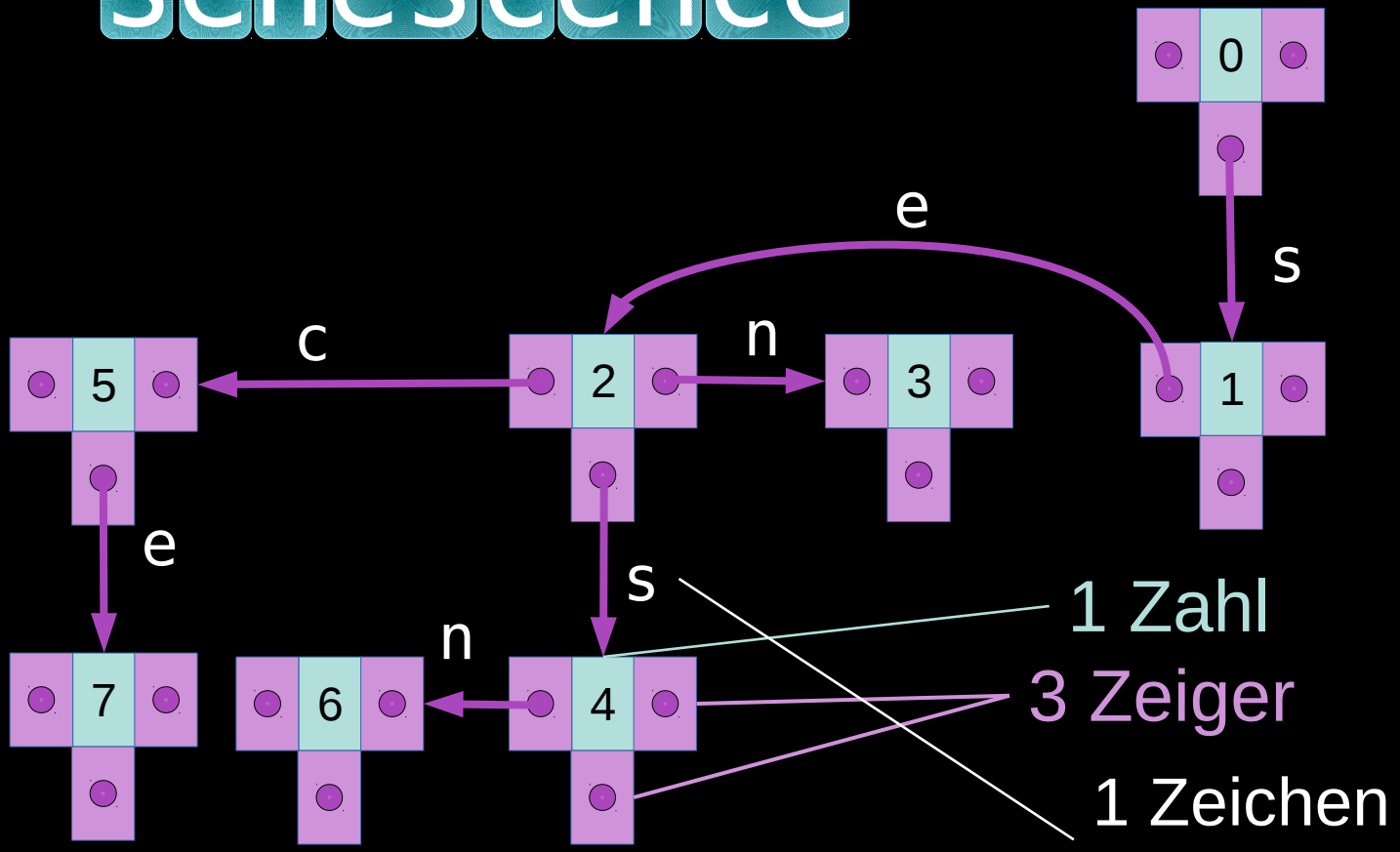
senescence



ternary trie



senescence



Evaluation

Datensatz `pcr_cere`

- 200 MiB
- Alphabetgröße: 6
- hoch repetitiv

Compressor	C Time	C Memory	C Rate	D Time	D Memory	chk
lz77(bit)	208.4s	2.9GiB	3.6867%	6.9s	437.2MiB	OK
lz77(opt)	174.0s	2.9GiB	2.6401%	7.8s	437.2MiB	OK
huff	4.6s	453.1MiB	28.1072%	10.0s	53.1MiB	OK
lz78(binary)	29.0s	263.9MiB	29.1033%	19.9s	165.4MiB	OK
lz78(ternary)	25.7s	324.2MiB	29.1033%	16.2s	165.4MiB	OK
gzip -9	175.9s	6.4MiB	26.2159%	1.6s	6.4MiB	OK
bzip2 -9	21.9s	15.2MiB	25.2368%	12.3s	11.5MiB	OK
lzma -9	225.2s	689.5MiB	1.9047%	379.4ms	80.5MiB	OK

Zusammenfassung

tudocomp

- Modulares C++14 Framework
- Werkzeuge
  - Benchmarks
  - Memory-Tracking
  - Visualisierung
- Bibliothek für Kompression
  - Text-Datenstrukturen (SA, LCP)
  - Bit-Vektoren
  - Bitweises I/O
- Klassische Kompressoren (als Baseline)
- Beliebte Kodierer

Ausblick

- schnellere Kompilation
- speichersparsameres LZ78
- Externspeicher-Kompression von 256 GiB Datensätzen