

ALGORITHMEN UND DATENSTRUKTUREN

ÜBUNG 9: SUCHEN & ERSETZEN

Eric Kunze

`eric.kunze@tu-dresden.de`

TU Dresden, 29. November 2021

letzte Änderung:
26.11.2021, 14:04

KMP-Algorithmus

Aufgabe 1

- ▶ Mustersuche in (großen) Texten
- ▶ Ziel: Verschiebung des Musters um mehr als eine Position bei Nichtübereinstimmung.
- ▶ Methode: Ermittlung einer Verschiebetabelle `Tab[]` in **Phase 1**
- ▶ Bedeutung des Eintrags `Tab[i]=j`:
Bei Nichtübereinstimmung an Stelle i wird Position j des Musters an aktueller Vergleichsstelle angelegt.
- ▶ Suchprozess in **Phase 2**

j-algo: <http://j-algo.binaervarianz.de/>

KMP-ALGORITHMUS

Suche das Muster aaabaaaa im Text aaabaaabaaacaaabaaaa.

Position	0	1	2	3	4	5	6	7
Pattern	a	a	a	b	a	a	a	a
Tabelle	-1	-1	-1	2	-1	-1	-1	3

Erster Versuch:

```
  a a b a a a b a a c a a b a a a
    a a b a a a a
```

Tabelleneintrag an Position 7 ist 3, d.h. $\text{Tab}[7]=3$ — Lege Position 3 des Musters an aktueller Vergleichsposition an:

```
  a a b a a a b a a a c a a b a a a
    a a b a a a a
```

Gleicher Prozess noch einmal: Mismatch an Position 7 des Musters — verschiebe Muster auf Position 3.

KMP-ALGORITHMUS (FORTSETZUNG)

Suche das Muster aaabaaaa im Text aaabaaabaaacaaabaaaa.

Position	0	1	2	3	4	5	6	7
Pattern	a	a	a	b	a	a	a	a
Tabelle	-1	-1	-1	2	-1	-1	-1	3

Wir legen das Muster also wieder an Position 3 an:

```

a a a b a a a b a a a c a a a b a a a a
                a a a b a a a a
```

Wegen $\text{Tab}[3]=2$, lege Muster an Position 2 an:

```

a a a b a a a b a a a c a a a b a a a a
                a a a b a a a a
```

Wegen $\text{Tab}[2]=-1$, lege Muster an Position -1 an:

```

a a a b a a a b a a a c a a a b a a a a
                a a a b a a a a ☺
```

Zwei Phasen:

Zwei Phasen:

- ▶ **1. Phase:** Markieren der längsten Teilwörter im Pattern, die mit einem Präfix übereinstimmen
 - ▷ ein Zyklus beginnt an einer Patternposition i falls $i \neq 0$ und $\text{Pat}[0] = \text{Pat}[i]$
 - ▷ ein Zyklus endet an der kleinsten Patternposition $i+m$, sodass $\text{Pat}[m+1] \neq \text{Pat}[i+m+1]$

Zwei Phasen:

- ▶ **1. Phase:** Markieren der längsten Teilwörter im Pattern, die mit einem Präfix übereinstimmen
 - ▷ ein Zyklus beginnt an einer Patternposition i falls $i \neq 0$ und $\text{Pat}[0] = \text{Pat}[i]$
 - ▷ ein Zyklus endet an der kleinsten Patternposition $i+m$, sodass $\text{Pat}[m+1] \neq \text{Pat}[i+m+1]$
- ▶ **2. Phase:** Bestimmung der Tabelleneinträge
 - ▷ $\text{Tab}[0] = -1$
 - ▷ Tabelleneinträge nach einem Zyklus:
Länge des längsten dort endenden Zyklus
 - ▷ Tabelleneinträgen in einem Zyklus:
Tabelleneintrag der derzeitigen Position im längsten laufenden Zyklus
 - ▷ verbleibende Einträge: 0

Die Methode beruht auf der Gleichung

$$\text{Tab}[i] = \max \{-1\} \cup \left\{ m \left| \begin{array}{l} 0 \leq m \leq i-1 \\ b_0 \dots b_{m-i} = b_{i-m} \dots b_{i-1} \\ b_m \neq b_j \end{array} \right. \right\} \quad (*)$$

Daraus ergibt sich nach Initialisierung von $\text{Tab}[0] = -1$ für jeden folgenden Eintrag $\text{Tab}[i]$ folgendes Verfahren:

- ▶ *linker Finger*: wähle $m < i$ in absteigender Reihenfolge (also $i-1, i-2, \dots$), sodass $\text{Pat}[i] \neq \text{Pat}[m]$
- ▶ *Parallelverschiebung beider Finger bis zum linken Rand*: wenn $\text{Pat}[0 \dots m-1] = \text{Pat}[i-m \dots i-1]$, dann fülle $\text{Tab}[i] = m$.
- ▶ wenn keine passende Position m gefunden werden kann, dann fülle $\text{Tab}[i] = -1$.

AUFGABE 1

Teil (a)

Pattern: aabaaacaab

Position	0	1	2	3	4	5	6	7	8	9
Pattern	a	a	b	a	a	a	c	a	a	b
Tabelle										

Teil (a) Pattern: aabaaacaab

Position	0	1	2	3	4	5	6	7	8	9
Pattern	a	a	b	a	a	a	c	a	a	b
Tabelle	-1	-1	1	-1	-1	2	2	-1	-1	1

AUFGABE 1 — TEIL (B)

Teil (b)

Position	0	1	2	3	4	5
Pattern	c	b				a
Tabelle	-1	0	-1	1	0	2

Teil (b)

Position	0	1	2	3	4	5
Pattern	c	b	c	c	b	a
Tabelle	-1	0	-1	1	0	2

- ▶ $\text{Pat}[0 \dots 1] = \text{Pat}[3 \dots 4]$ wegen $\text{Tab}[5] = 2$ (Zyklusmethode), d.h.
 $\text{Pat}[3] = \text{Pat}[0] = c$ und $\text{Pat}[4] = \text{Pat}[1] = b$
- ▶ wegen $\text{Tab}[3] = 1$ ist $\text{Pat}[2] = \text{Pat}[0] = c$ (Zyklusmethode)
- ▶ **oder:** wegen $\text{Tab}[3] = 1$ ist $\text{Pat}[1] \neq \text{Pat}[3]$ und
 $\text{Pat}[2] = \text{Pat}[0] = c$ (Parallelverschiebung in der
Zwei-Finger-Methode bzw. Gleichung (*))

Levenshtein-Distanz

Aufgabe 2

Kosten zur Überführung eines Wortes $w = w_1 \dots w_n$ in ein Wort $v = v_1 \dots v_k$; schreibe $d(w_1 \dots w_j, v_1 \dots v_i) = d(j, i)$.

LEVENSHTEIN-DISTANZ

Kosten zur Überführung eines Wortes $w = w_1 \dots w_n$ in ein Wort $v = v_1 \dots v_k$; schreibe $d(w_1 \dots w_j, v_1 \dots v_i) = d(j, i)$.

$$d(0, i) = i$$

$$d(j, 0) = j$$

$$d(j, i) = \min \{ d(j, i-1) + 1, d(j-1, i) + 1, d(j-1, i-1) + \delta_{j,i} \}$$

für alle $1 \leq j \leq n$ und alle $1 \leq i \leq k$ wobei

$$\delta_{j,i} = \begin{cases} 1 & \text{wenn } w_j \neq v_i \\ 0 & \text{sonst} \end{cases}$$

LEVENSHTEIN-DISTANZ

Kosten zur Überführung eines Wortes $w = w_1 \dots w_n$ in ein Wort $v = v_1 \dots v_k$; schreibe $d(w_1 \dots w_j, v_1 \dots v_i) = d(j, i)$.

$$d(0, i) = i$$

$$d(j, 0) = j$$

$$d(j, i) = \min \{ d(j, i-1) + 1, d(j-1, i) + 1, d(j-1, i-1) + \delta_{j,i} \}$$

für alle $1 \leq j \leq n$ und alle $1 \leq i \leq k$ wobei

$$\delta_{j,i} = \begin{cases} 1 & \text{wenn } w_j \neq v_i \\ 0 & \text{sonst} \end{cases}$$

Anschaulich: Überlagerung durch Pattern → Pfeile zeigen "Ursprung" des Minimums an

$w_j \neq v_i :$

+1	+1
+1	?

$w_j = v_i :$

+0	+1
+1	?

AUFGABE 2

Gegeben seien die Wörter $w = \text{espen}$ und $v = \text{beispiele}$.

- (a) Berechnen Sie die Levenshtein-Distanz $d(w, v)$. Geben Sie dazu die Berechnungsmatrix an. Tragen Sie alle Zelleneinträge zusammen mit den dazugehörigen Pfeilen ein.
- (b) Geben Sie die Levenshtein-Distanz $d(\text{espe}, \text{beispiel})$ an. Beachten Sie, dass `espe` und `beispiel` Präfixe von `espen` bzw. `beispiele` sind.
- (c) Geben Sie zwei Alignments zwischen `espen` und `beispiele` an, die zu den minimalen Kosten führen. Dabei sollen die Alignments die jeweils angewendeten Editieroperation enthalten.
- (d) Wieviele Alignments enthält die in Aufgabe (a) angegebene Berechnungsmatrix?

AUFGABE 2

Teil (a)

$d(j, i)$	b	e	i	s	p	i	e	l	e
e									
s									
p									
e									
n									

Teil (a)

$d(\text{espen, beispiele}) = 5$

$d(j, i)$		b	e	i	s	p	i	e	l	e
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7	→ 8	→ 9
e	1	↓ ↘	1	↓ ↘	1	→ 2	→ 3	→ 4	→ 5	→ 6
s	2	↓ ↘	2	↓ ↘	2	→ 3	→ 4	→ 5	→ 6	→ 7
p	3	↓ ↘	3	↓ ↘	3	→ 2	→ 3	→ 4	→ 5	→ 6
e	4	↓ ↘	4	↓ ↘	4	↓ ↘	3	→ 4	→ 5	→ 6
n	5	↓ ↘	5	↓ ↘	4	→ 5	4	↓ ↘	4	→ 5

Teil (a) $d(\text{espen, beispiele}) = 5$

$d(j, i)$		b	e	i	s	p	i	e	l	e
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7	→ 8	→ 9
e	1	↘	↘				↘		↘	
s	2	↓	↓	↓	↓	→ 3	→ 4	→ 5	→ 6	→ 7
p	3	↓	↓	↓	↓	↓	2 → 3	→ 4	→ 5	→ 6
e	4	↓	↓	↓	↓	↓	↓	3 → 4	→ 5	→ 6
n	5	↓	↓	↓	↓	↓	↓	↓	4 → 5	→ 6

Teil (b) $d(\text{espe, beispiel}) = 4$

AUFGABE 2

Teil (c) Alignments mit minimaler Levenshtein-Distanz:

Teil (d)

Teil (c) Alignments mit minimaler Levenshtein-Distanz:

*	e	*	s	p	*	e	*	n
b	e	i	s	p	i	e	l	e
i		i			i		i	s

*	e	*	s	p	*	e	n	*
b	e	i	s	p	i	e	l	e
i		i			i		s	i

Teil (d) 2 Alignments = 2 Backtraces

Weitere Aufgaben aus der Aufgabensammlung *mit Lösungen*

AUFGABE 7.1.13 (AGS)

- (a) Bestimmen Sie die mit Hilfe des KMP-Algorithmus berechnete Verschiebetabelle für das Pattern `abbabbaa`.
- (b) Mit Hilfe des KMP-Algorithmus ist unten stehende Verschiebetabelle berechnet worden. Die mit einem „?“ markierten Einträge sind unbekannt. Vervollständigen Sie das aus den Symbolen `a`, `b` und `c` bestehende Pattern.

Position	0	1	2	3	4	5
Pattern	<i>b</i>					<i>c</i>
Tabelle	-1	?	?	0	?	3

AUFGABE 7.1.13 (AGS)

Teil (a)

Pattern: abbabbaa

Position	0	1	2	3	4	5	6	7
Pattern	a	b	b	a	b	b	a	a
Tabelle								

Teil (a) Pattern: abbabbaa

Position	0	1	2	3	4	5	6	7
Pattern	a	b	b	a	b	b	a	a
Tabelle	-1	0	0	-1	0	0	-1	4

Teil (b)

Position	0	1	2	3	4	5
Pattern	<i>b</i>					<i>c</i>
Tabelle	-1	?	?	0	?	3

Teil (a) Pattern: abbabbaa

Position	0	1	2	3	4	5	6	7
Pattern	a	b	b	a	b	b	a	a
Tabelle	-1	0	0	-1	0	0	-1	4

Teil (b)

Position	0	1	2	3	4	5
Pattern	<i>b</i>	<i>a</i>	<i>b</i>	<i>a</i>	<i>b</i>	<i>c</i>
Tabelle	-1	?	?	0	?	3

- ▶ $\text{Pat}[0 \dots 2] = \text{Pat}[2 \dots 4]$ wegen $\text{Tab}[5] = 3$ (Zyklenmethode), d.h. $\text{Pat}[2] = \text{Pat}[0] = \text{Pat}[4] = b$
- ▶ wegen $\text{Tab}[3] = 0$ ist $\text{Pat}[3] \neq \text{Pat}[0] = b$ und wegen $\text{Tab}[5] = 3$ ist $\text{Pat}[3] \neq \text{Pat}[5] = c$ (Zwei-Finger-Methode bzw. Gleichung (*))
 $\Rightarrow \text{Pat}[3] = \text{Pat}[1] = a$

AUFGABE 7.2.1 (AGS)

Gegeben seien die Wörter $w = \text{Dinstas}$ und $v = \text{Distanz}$.

- (a) Berechnen Sie die Levenshtein-Distanz $d(w, v)$ zwischen w und v . Geben Sie die Berechnungsmatrix vollständig an.
- (b) Geben Sie alle Alignments mit minimaler Levenshtein-Distanz zwischen w und v an.

AUFGABE 7.2.1 (AGS)

$d(j, i)$	D	i	s	t	a	n	z
D							
i							
n							
s							
t							
a							
s							

$d(j, i)$		D	i	s	t	a	n	z
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
D	↓ 1		0	→ 1	→ 2	→ 3	→ 4	→ 5
i	↓ 2	↓ 1		0	→ 1	→ 2	→ 3	→ 4
n	↓ 3	↓ 2	↓ 1	↓ 1	→ 2	→ 3	→ 3	→ 4
s	↓ 4	↓ 3	↓ 2	↓ 1	→ 2	→ 3	→ 4	→ 4
t	↓ 5	↓ 4	↓ 3	↓ 2	→ 1	→ 2	→ 3	→ 4
a	↓ 6	↓ 5	↓ 4	↓ 3	↓ 2	→ 1	→ 2	→ 3
s	↓ 7	↓ 6	↓ 5	↓ 4	↓ 3	↓ 2	↓ 2	→ 3

$$d(\text{Dinstas}, \text{Distanz}) = 3$$

$d(j, i)$		D	i	s	t	a	n	z
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
D	↓ 1		0	→ 1	→ 2	→ 3	→ 4	→ 5
i	↓ 2	↓ 1		0	→ 1	→ 2	→ 3	→ 4
n	↓ 3	↓ 2	↓ 1	↓ 1	→ 2	→ 3	↓ 3	→ 4
s	↓ 4	↓ 3	↓ 2	↓ 1	→ 2	→ 3	↓ 4	↓ 4
t	↓ 5	↓ 4	↓ 3	↓ 2		1	→ 2	→ 3
a	↓ 6	↓ 5	↓ 4	↓ 3	↓ 2		1	→ 2
s	↓ 7	↓ 6	↓ 5	↓ 4	↓ 3	↓ 2	↓ 2	→ 3

$$d(\text{Dinstas}, \text{Distanz}) = 3$$

$d(j, i)$		D	i	s	t	a	n	z
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
D	↓ 1		0	→ 1	→ 2	→ 3	→ 4	→ 5
i	↓ 2	↓ 1		0	→ 1	→ 2	→ 3	→ 4
n	↓ 3	↓ 2	↓ 1	↓ 1	→ 2	→ 3	↓ 3	→ 4
s	↓ 4	↓ 3	↓ 2	↓ 1	→ 2	→ 3	↓ 4	↓ 4
t	↓ 5	↓ 4	↓ 3	↓ 2		1	→ 2	→ 3
a	↓ 6	↓ 5	↓ 4	↓ 3	↓ 2		1	→ 2
s	↓ 7	↓ 6	↓ 5	↓ 4	↓ 3	↓ 2	↓ 2	↓ 3

$$d(\text{Dinstas}, \text{Distanz}) = 3$$

AUFGABE 7.2.1 (AGS)

Alignments mit minimaler Levenshtein-Distanz:

Alignments mit minimaler Levenshtein-Distanz:

D	i	n	s	t	a	*	s
D	i	*	s	t	a	n	z
		<i>d</i>				<i>i</i>	<i>s</i>

D	i	n	s	t	a	s	*
D	i	*	s	t	a	n	z
		<i>d</i>				<i>s</i>	<i>i</i>

AUFGABE 7.2.2 (AGS)

- (a) Berechnen Sie die Levenshtein-Distanz $d(\text{bürste}, \text{schürze})$. Geben Sie die Berechnungsmatrix vollständig an. Wieviele Backtraces enthält die Berechnungsmatrix?
- (b) Geben Sie zwei Alignments mit minimaler Levenshtein-Distanz zwischen den Wörtern `bürst` und `sch an`.

AUFGABE 7.2.2 (AGS) — TEIL (A)

$d(j, i)$	s	c	h	ü	r	z	e
b							
ü							
r							
s							
t							
e							

$d(j, i)$		s	c	h	ü	r	z	e
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
b	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘
	1	1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
ü	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘
	2	2	2	→ 3	3	→ 4	→ 5	→ 6
r	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘
	3	3	3	3	→ 4	3	→ 4	→ 5
s	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘
	4	3	→ 4	4	4	4	4	→ 5
t	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘
	5	4	4	→ 5	5	5	5	5
e	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘	↓ ↘
	6	5	5	5	→ 6	6	6	5

$$d(\text{bürste}, \text{schürze}) = 5$$

$d(j, i)$		s	c	h	ü	r	z	e
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
b	↓ ↘	1	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘
	1	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
ü	↓ ↘	2	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘
	2	→ 2	→ 2	→ 3	→ 3	→ 4	→ 5	→ 6
r	↓ ↘	3	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘
	3	→ 3	→ 3	→ 3	→ 4	→ 3	→ 4	→ 5
s	↓ ↘	4	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘
	4	→ 3	→ 4	→ 4	→ 4	→ 4	→ 4	→ 5
t	↓ ↘	5	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘
	5	→ 4	→ 4	→ 5	→ 5	→ 5	→ 5	→ 5
e	↓ ↘	6	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘	↘ ↘
	6	→ 5	→ 5	→ 5	→ 6	→ 6	→ 6	→ 5

$$d(\text{bürste}, \text{schürze}) = 5$$

$d(j, i)$		s	c	h	ü	r	z	e
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
b	↓ 1	↓ 1	↓ 2	↓ 3	↓ 4	↓ 5	↓ 6	↓ 7
ü	↓ 2	↓ 2	↓ 2	↓ 3	↓ 3	↓ 4	↓ 5	↓ 6
r	↓ 3	↓ 3	↓ 3	↓ 3	↓ 4	↓ 3	↓ 4	↓ 5
s	↓ 4	↓ 3	↓ 4	↓ 4	↓ 4	↓ 4	↓ 4	↓ 5
t	↓ 5	↓ 4	↓ 4	↓ 5	↓ 5	↓ 5	↓ 5	↓ 5
e	↓ 6	↓ 5	↓ 5	↓ 5	↓ 6	↓ 6	↓ 6	↓ 5

$$d(\text{bürste}, \text{schürze}) = 5$$

$d(j, i)$		s	c	h	ü	r	z	e
	0	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
b	↓	↓	↓	↓	↓	↓	↓	↓
	1	→ 1	→ 2	→ 3	→ 4	→ 5	→ 6	→ 7
ü	↓	↓	↓	↓	↓	↓	↓	↓
	2	→ 2	→ 2	→ 3	→ 3	→ 4	→ 5	→ 6
r	↓	↓	↓	↓	↓	↓	↓	↓
	3	→ 3	→ 3	→ 3	→ 4	→ 3	→ 4	→ 5
s	↓	↓	↓	↓	↓	↓	↓	↓
	4	→ 3	→ 4	→ 4	→ 4	→ 4	→ 4	→ 5
t	↓	↓	↓	↓	↓	↓	↓	↓
	5	→ 4	→ 4	→ 5	→ 5	→ 5	→ 5	→ 5
e	↓	↓	↓	↓	↓	↓	↓	↓
	6	→ 5	→ 5	→ 5	→ 6	→ 6	→ 6	→ 5

$d(\text{bürste}, \text{schürze}) = 5$ Anzahl der Backtraces = $3 * 2 = 6$

AUFGABE 7.2.2 (AGS) — TEIL (B)

Alignments mit minimaler Levenshtein-Distanz zwischen
den Wörtern `bürst` und `sch`

Alignments mit minimaler Levenshtein-Distanz zwischen den Wörtern *bürst* und *sch*

$d(j, i)$		s	c	h
	0	→ 1	→ 2	→ 3
b	↓ 1	↓ 1	↘ 2	↘ 3
ü	↓ 2	↓ 2	↘ 2	↘ 3
r	↓ 3	↓ 3	↘ 3	↘ 3
s	↓ 4	↓ 3	↘ 4	↘ 4
t	↓ 5	↓ 4	↘ 4	↘ 5

AUFGABE 7.2.2 (AGS) — TEIL (B)

Alignments mit minimaler Levenshtein-Distanz zwischen
den Wörtern `bürost` und `sch`

Alignments mit minimaler Levenshtein-Distanz zwischen
den Wörtern *bürst* und *sch*

b	ü	r	s	t
s	c	h	*	*
<i>s</i>	<i>s</i>	<i>s</i>	<i>d</i>	<i>d</i>

b	ü	r	s	t
*	*	s	c	h
<i>d</i>	<i>d</i>	<i>s</i>	<i>s</i>	<i>s</i>