Zusammenfassung zur Vorlesung Basismodul Computerlinguistik

Wiederholung

27.01.2022

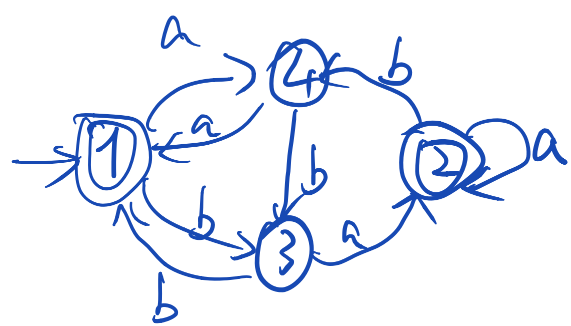
Minimierung eines totalen DEA

Aufgabe: Gegeben sei ein DEA mit totaler Übergangsfunktion: $\Sigma = \{a, b\}$, $Zustände = \{1, 2, 3, 4, 5, 6\}$, $F = \{1, 4, 5\}$, $s = 1$,
$\Delta = \{(1, a, 3), (1, b, 2), (2, a, 4), (2, b, 1), (3, a, 1), (3, b, 2), (4, a, 4), (4, b, 3),$ (5,a,3),(5,b,6),(6,a,4),(6,b,5)}.
Berechne für den DEA einen äquivalenten minimalen DEA.

	F			NF		
Zustände	1	4	5	2	3	6
	a:3 b:2	a:4 b:3	a:3 b:6	a:4 b:1	a:1 b:2	a:4 b:5
$\sim_0$	1	1	1	2	2	2
	122	112	122	211	212	211
$\sim_1$	1	2	1	3	4	3
	143	224	143	321	413	321
$\sim_2$	1	2	1	3	4	3

fertig.

Min DEA :

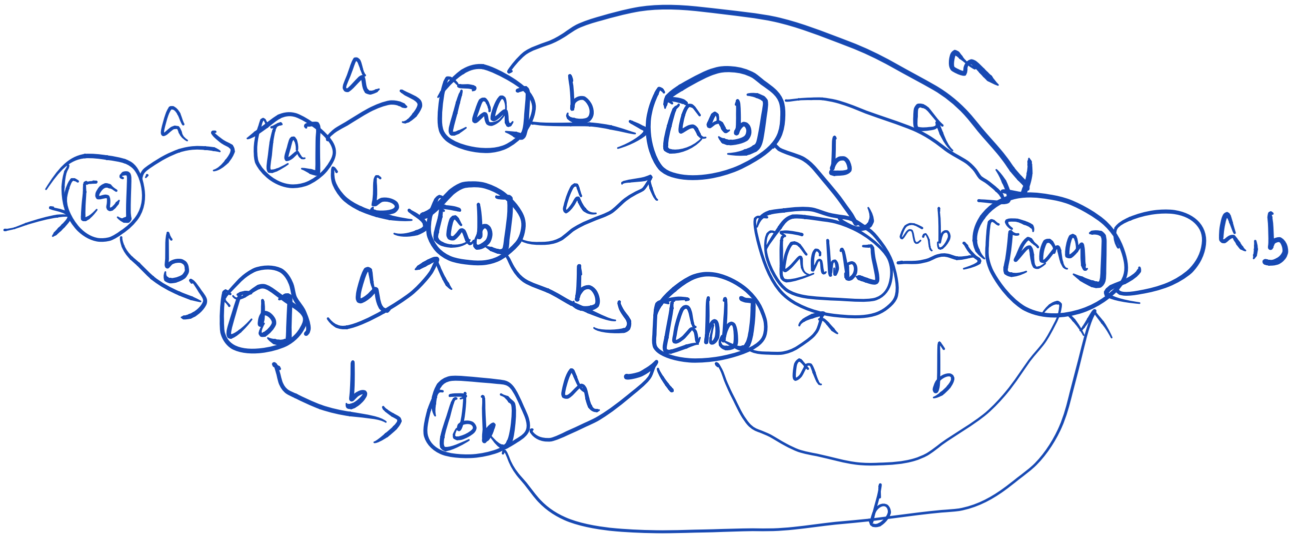


Nerode-Konstruktion

Aufgabe: Es sei Σ das Alphabet $\{a, b\}$. Die Menge L enthalte alle Wörter mit genau **2** a's und **2** b's. Berechne den minimalen DEA für L mittels Nerode-Äquivalenzklassen.

Nerode-Äquiv.	Suffixe (Rechtssprache)	Elemente
$[\varepsilon]$	L	$\{\varepsilon\}$
$[a]$	$\{abb, bab, bba\}$	$\{a\}$
$[b]$	$\{aab, aba, baa\}$	$\{b\}$
$[aa]$	$\{bb\}$	$\{aa\}$
$[ab]$	$\{ba, ab\}$	$\{ab, ba\}$
$[bb]$	$\{aa\}$	$\{bb\}$

[aab]	$\{b\}$	{aab, aba, baa}
[aaa]	$\{\ \}$	{aaab, bbb, ...}
[abb]	$\{a\}$	{abb, bab, bba}
[aabb]	$\{\varepsilon\}$	{aabb, abab, .

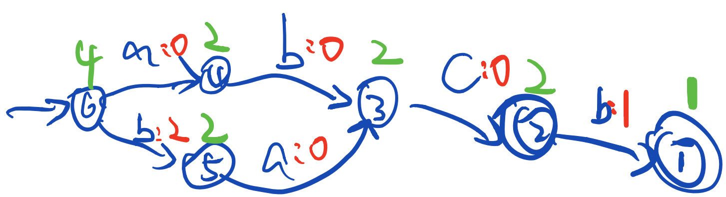


Lexikonautomat

Aufgabe: Gegeben sei ein Lexikon $\{bac, abc, bacb, abcb\}$.

1. Berechne den minimalen Lexikonautomaten für das Lexikon nach dem Daciuk-Algorithmus.
2. Notiere auf dem Bild für den Lexikonautomaten die Zahlen, die für die Berechnung von Perfekten Hashingwerten nötig sind.
3. Speichere den Lexikonautomaten als Tarjan-Tabelle.
4. Wie würde der Automat aussehen, wenn man das Wort *abcbe* hinzufügt, und dann das Wort *bac* löscht.

1. sortiert: abc, abcb, bac, bacb.



Lexikonautoma

2. Hashing.

3. Tarjan-Tabelle

$\{a:1,\ b:2,\ c:3\}$

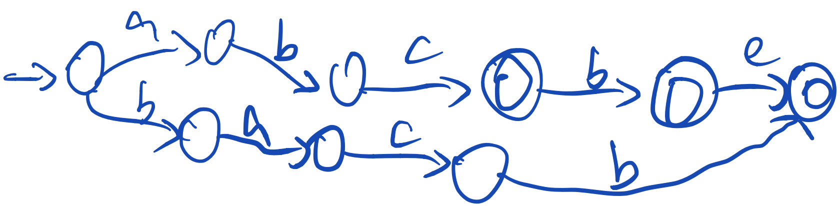
1	1	F
2	2	F
3	3	NF
4	b	1
5	4	NF
6	c	2
7	b	3
8	5	NF
9	a	3
10	6	NF
11	a	5
12	b	8
13		

4. Hinzufügen.

+abcbe

Löschen:

−bac



Universeller Levenshtein-Automat

Aufgabe: Betrachte den universellen Levenshtein-Automaten mit der Fehlerschranke $b = 3$. Zum aktuellen Zeitpunkt t werden die Zustände $I - 2^2, I + 1^2, I^3$ aktiviert. Es sei 01110001 der Bitvektor für t.

1. Welche Zustände werden zum nächsten Zeitpunkt $t + 1$ aktiviert?
2. Welche davon sind echte aktive Zustände mit Berücksichtigung der Normalisierung (d.h. Löschung aller redundanten Zustände)?

$b=3$ Höhe: $b+1=4$ Breite: $2b+1=7$.

aktive Zustände:

$I-2^{2}, I^{3}, I+1^{3}$

Berechnung von Bitvektoren

Aufgabe: Es sei v das Pattern-Wort *colour* und w das Wort *color*. Wir möchten einen Universellen Levenshtein-Automaten verwenden, um zu testen, ob der Levensheitenabstand zwischne v und w kleinergleich 2 ist.

1. Welche Schranke soll der Universelle Levenshtein-Automat haben?
2. Wie sieht die Folge der Automaten-Eingaben aus?

1. $b=2$

2. v: colour w: color.

① max. Länge ~~der~~ des Bitvektoren: $2b+2=6$.

② Anzahl des Füllers (\$): $b=2$.

\$\$colour color

③ Bitvektor

$\chi(c, \$\$colo) = 001000$

$\chi(o, \$colou) = 00101$

$\chi(l, colour) = 100000$

$\chi(o, olour) = 10100$

$\chi(r, lour) = 0001$