ETH zürich



Theoretische Informatik

Olivér Kovács

1. Okt 2026

Hallo!

Meine Webseite: oliverkovacs.dev/ti

Ich bin jederzeit erreichbar:
REDACTED (WA)
okovacs@ethz.ch



Übersicht

1. Formalisierung
2. Formalismus einsetzen (Wir sind hier!)
3. Berechenbarkeit
4. Komplexitätstheorie

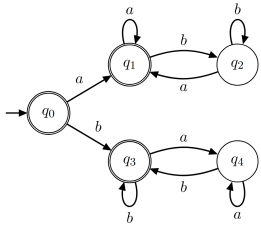
Serien

In Gruppen abgeben!

Mehr begründen!

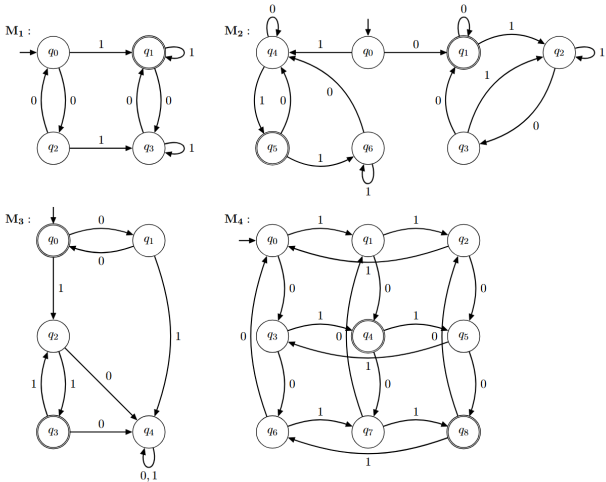
Das leere Wort (λ)

Quiz Frage 1

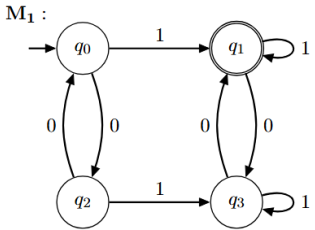


1. $L_1 = \{w \in \{a,b\}^* \mid w$ enthät das Teilwort ab gleich oft wie $ba\}$
2. $L_2 = \{w \in \{a,b\}^* \mid w = xv, x \in \{a,b\}, 2|v|_x \geq |v|\}$
3. $L_3 = \{w \in \{a,b\}^* \mid w = \lambda \vee (w = w_1 \dots w_{|w|} \wedge w_1 = w_{|w|})\}$
4. $L_4 = \{w \in \{a,b\}^* \mid w$ endet nicht in ab oder $ba\}$

Quiz Frage 2

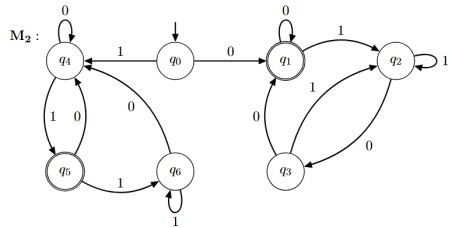

M1 :
q0
q1
q2
q3
M2 :
q4
q0
q1
q2
q5
q6
q3
M3 :
q0
q1
q2
q3
q4
0, 1
M4 :
q0
q1
q2
q3
q4
q5
q6
q7
q8

Quiz Frage 2



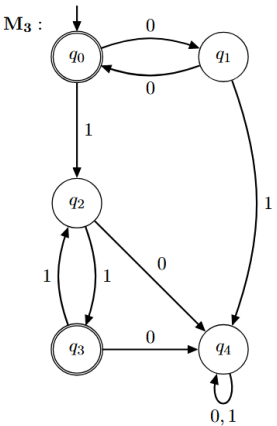
1. $L = \{w \in \{0,1\}^* \mid |w|_0 \cdot |w|_1 \equiv 1 \mod 3\}$
2. $L = \{0^{2k}1^{2l} \mid k, k \in \mathbb{N}\}$
3. $L = \{w \in \{0,1\}^* \mid (|w|_0)^{|w|_1} \equiv 0 \mod 2\}$
4. $L = \{xw \in \{0,1\}^* \mid x \in \{0,1\}$ und Nummer$(w) \equiv x \mod 4\}$

Quiz Frage 2



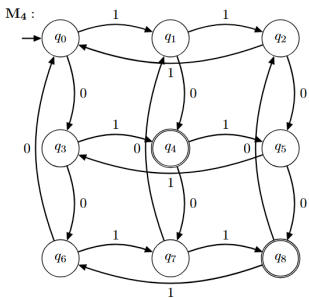
1. $L = \{w \in \{0,1\}^* \mid |w|_0 \cdot |w|_1 \equiv 1 \mod 3\}$
2. $L = \{0^{2k}1^{2l} \mid k, l \in \mathbb{N}\}$
3. $L = \{w \in \{0,1\}^* \mid (|w|_0)^{|w|_1} \equiv 0 \mod 2\}$
4. $L = \{xw \in \{0,1\}^* \mid x \in \{0,1\}$ und Nummer$(w) \equiv x \mod 4\}$

Quiz Frage 2



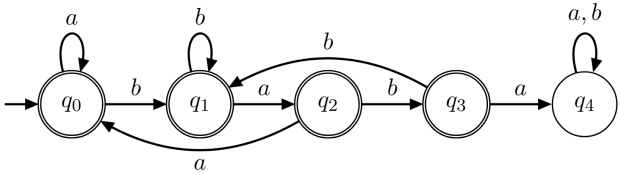
1. $L = \{w \in \{0,1\}^* \mid |w|_0 \cdot |w|_1 \equiv 1 \mod 3\}$
2. $L = \{0^{2k}1^{2l} \mid k, l \in \mathbb{N}\}$
3. $L = \{w \in \{0,1\}^* \mid (|w|_0)^{|w|_1} \equiv 0 \mod 2\}$
4. $L = \{xw \in \{0,1\}^* \mid x \in \{0,1\}$ und Nummer$(w) \equiv x \mod 4\}$

Quiz Frage 2



1. $L = \{w \in \{0,1\}^* \mid |w|_0 \cdot |w|_1 \equiv 1 \mod 3\}$
2. $L = \{0^{2k}1^{2l} \mid k, l \in \mathbb{N}\}$
3. $L = \{w \in \{0,1\}^* \mid (|w|_0)^{|w|_1} \equiv 0 \mod 2\}$
4. $L = \{xw \in \{0,1\}^* \mid x \in \{0,1\}$ und $\text{Nummer}(w) \equiv x \mod 4\}$

Quiz Frage 3



Erkennt die Sprache $L = \{w \in \{a, b\}* \mid w$ enthält $baba$ nicht als Teilwort $\}$.

Ändere die ausgehenden Übergänge eines Zustands so, dass der Automat stattdessen $L' = \{w \in \{a, b\}^* \mid w$ endet nicht auf $baba\}$ erkennt.

Regularität

Definition 3.1. [...] $\mathcal{L}_{\mathsf{EA}} = \{L(M) \mid M \text{ ist EA }\}$ ist die *Klasse der regulären Sprachen*. Jede Sprache $L \in \mathcal{L}_{\mathsf{EA}}$ wird *regulär* genannt.

Wie zeigt man Nichtregularität?

Lemma 3.3. Sei $A = (Q, \Sigma, \delta_A, q_0, F)$ ein EA. Seien $x, y \in \Sigma^*, x \neq y$, so dass

$$(q_0, x) \vdash_A^* (p, \lambda) \text{ und } (q_0, y) \vdash_A^* (p, \lambda)$$

für ein $p \in Q$ gilt. Dann existiert für jedes $z \in \Sigma^*$ ein $r \in Q$, so dass $xz, yz \in \mathsf{Kl}[r]$. Insbesondere gilt

$$xz \in L(A) \iff yz \in L(A).$$

Wie zeigt man Nichtregularität?

Korollar: Sei A ein EA. Seien x, y zwei Wörter für die ein Suffix z existiert, so dass gilt $xz \in L(A) \iff yz \notin L(A)$, also $|\{xz, yz\} \cap L(A)| = 1$.

Dann müssen x und y in zwei verschiedenen Zustände landen.

Wie zeigt man Nichtregularität?

Idee: Nehme an, L sei regulär. Dann gibt es ein EA A mit $L(A) = L$. Nehme an, A hat n Zustände.

Finde $n + 1$ solche Wörter $x_0, \ldots, x_n$, dass für alle Paare x_i, x_j ein Suffix $z_{i,j}$ existiert, so dass $|\{x_i z_{i,j}, x_j z_{i,j}\} \cap L| = 1$.

Dann müssen $x_0, \ldots, x_n$ in paarweise verschiedene Zustände landen. Das ist ein Wiederspruch.

Nebenbemerkung

You can't parse [X]HTML with regex. Because HTML can't be parsed by regex. Regex is not a tool that can be used to correctly parse HTML. As I have answered in HTML-and-regex questions here so many times before, the use of regex will not allow you to consume HTML. Regular expressions are a tool that is insufficiently sophisticated to understand the constructs employed by HTML. HTML is not a regular language and hence cannot be parsed by regular expressions. Regex queries are not equipped to break down HTML into its meaningful parts. so many times but it is not getting to me. Even enhanced irregular regular expressions as used by Perl are not up to the task of parsing HTML. You will never make me crack. HTML is a language of sufficient complexity that it cannot be parsed by regular expressions. Even Jon Skeet cannot parse HTML using regular expressions. Every time you attempt to parse HTML with regular expressions, the unholy child weeps the blood of virgins, and Russian hackers pwn your webapp. Parsing HTML with regex summons tainted souls into the realm of the living. HTML and regex go together like love, marriage, and ritual infanticide. The cannot hold it is too late. The force of regex and HTML together in the same conceptual space will destroy your mind like so much watery putty. If you parse HTML with regex you are giving in to Them and their blasphemous ways which doom us all to inhuman toil for the One whose Name cannot be expressed in the Basic Multilingual Plane, he comes. HTML-plus-regexp will liquify the nerves of the sentient whilst you observe, your psyche withering in the onslaught of horror. Regex-based HTML parsers are the cancer that is killing StackOverflow *it is too late it is too late we cannot be saved* the transgression of a child ensures regex will consume all living tissue (except for HTML which it cannot, as previously prophesied) *dear lord help us how can anyone survive this scourge* using regex to parse HTML has doomed humanity to an eternity of dread torture and security holes *using rege*x as a tool to process HTML establishes a brea*ch between this world* and the dread realm of corrupt entities (like SGML entities, but *more corrupt) a mere glimp*se of the world of reg**ex parsers for HTML will ins**tantly transport a *programmer's consciousness into* a *world* of ceaseless screaming, he comes, ~~the pestilent slith~~y regex-infection wil**l devour your HT**ML parser, application and existence for all time like Visual Basic only worse *he comes he comes do not f*ight h**e com~~es~~, hi**s unholy radiance *destroying all enlightenment, HTML tags* ***leaking from your eyes like liq****uid p*ain, the song of regular expres~~sion parsing~~ will ext*inguish the voices of mor***tal man from the sp***here I can see it can you see it it is beautiful* the f`inal snuf`fing of *the lie***s of Man ALL IS LOST ALL IS L**OST the *pony he come*s he comes he co~~mes he co~~**mes the ich**or permeates all MY FACE *MY FACE oh god no* ***NO NO*****OOO N**O stop *the ani*~~mals~~ `a/e` **not r[0]EZA_GO IS_T**ONY THE PONY, **HE COMES**

Nebenbemerkung

"HTML is not a regular language and hence cannot be parsed by regular expressions." (https://stackoverflow.com/questions/1732348/regex-match-open-tags-except-xhtml-self-contained-tags)

Aber warum?

Aufgaben

1. **Aufgabe 1.** Sei $L_1 = \{w \in \{a,b\}^* \mid |w| \geq 3 \text{ oder } |w|_a = |w|_b\}$. Konstruiere einen EA A mit $L(A) = L_1$.
2. **Aufgabe 2.** Sei $L_2 = \{a^n b^m \mid n, m \in \mathbb{N}, 2 \leq m \leq n-1, m \text{ ist Teiler von } n\}$. Beweise unter Zuhilfenahme von Lemma 3.3., dass L_2 nicht regulär ist.
3. **Aufgabe 3.** Ein EA A heisst minimal, falls kein kleinerer EA B mit $L(A) = L(B)$ existiert. Konstruiere einen minimalen EA A mit $L(A) = L_1$. Beweise unter Zuhilfenahme von Lemma 3.3., dass A minimal ist.
4. **Aufgabe (Bonus)** Beweise, dass HTML keine reguläre Sprache ist. :)

   Hint: Du kannst äquivalenterweise beweisen, dass die Sprache $L_{\text{Klammer}} = \{w \in \{\boxed{(}, \boxed{)}\}^* \mid w \text{ ist eine valide Verschachtelung von Klammern}\}$ *nicht regulär ist.*