Prof. Dr. Jasmin Blanchette
Elisabeth Lempa
Luca Maio

Ludwig-Maximilians-Universität München
Institut für Informatik
Besprechung 22.05.2025 bis 26.05.2025
Abgabe bis 06.02.2025, 10:00 Uhr

Lösungsvorschlag zur Übung 4 zur Vorlesung

Formale Sprachen und Komplexität

Wenn Sie Automaten angeben, tun Sie dies immer in Form eines Zustandsgraphen. Andere Formen der Darstellung (z.B. als Liste von Übergängen) werden nicht gewertet, da sie sehr viel aufwändiger zu korrigieren sind. Vergessen Sie nicht, im Zustandsgraph Start- und Endzustände zu markieren.

Reguläre Ausdrücke sind entsprechend Definition 4.7.1 im Vorlesungsskript anzugeben.

FSK4-1 *Pumping-Lemma für reguläre Sprachen*

Zeigen Sie mit dem Pumping-Lemma für reguläre Sprachen, dass die folgenden Sprachen nicht regulär sind.

a) $L_2 = L(G_2)$, wobei G_2 eine kontextfreie Grammatik ist mit

$$G_2 = (\{S, A, B\}, \{(,), [,]\}, P, S)$$
$$P = \{S \to (S), S \to [S], S \to A, S \to B, A \to (), A \to [], B \to S, B \to BB\}$$

L_2 ist die Sprache der zueinander passenden eckigen und runden Klammern, d.h. es sind z.B. $([])[] \in L_2$ und $()() \in L_2$, aber $([\notin L_2$ und $) \notin L_2$ (vgl. Aufgabe FSK1-3).

LÖSUNGSVORSCHLAG:
Beweis mit dem Pumping-Lemma.
Sei $n \in \mathbb{N}_{>0}$ die "Pumpingzahl" von L_2.
Wir wählen $z \in L_2$ als $z = [^n]^n$ mit $|z| \geq n$.
Sei $z = uvw$ eine beliebige Zerlegung von z, sodass $|uv| \leq n$, $|v| \geq 1$ und $uv^iw \in L_2$ für jedes $i \in \mathbb{N}$. Da $|uv| \leq n$ ist, ist $v = [^k$ für ein $k \in \mathbb{N}_{>0}$.
Wir wählen $i = 0$. Das Wort uv^iw hat weniger öffnende als schließende eckige Klammern und somit ist $uv^iw \notin L_2$. Widerspruch.

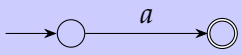
FSK4-2 *Reguläre Ausdrücke und Abschlusseigenschaften*

a) Betrachten Sie den regulären Ausdruck $\alpha = (a|b)^*(ab|ba)(a|b)^*$.

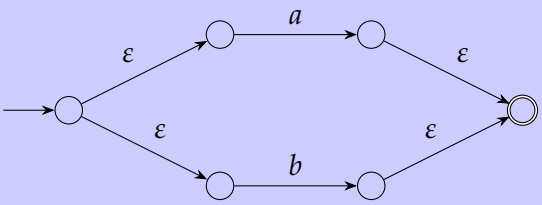
i) Geben Sie einen NFA ohne ε-Übergänge an, der $L(\alpha)$ erkennt. Sie können die Algorithmen aus der Vorlesung zur Konstruktion eines NFA aus einem regulären Ausdruck und zur Elimination von ε-Übergängen verwenden, müssen aber nicht.

LÖSUNGSVORSCHLAG:
Mit dem Algorithmus aus der Vorlesung ergeben sich folgende NFAs mit ε-Übergängen:

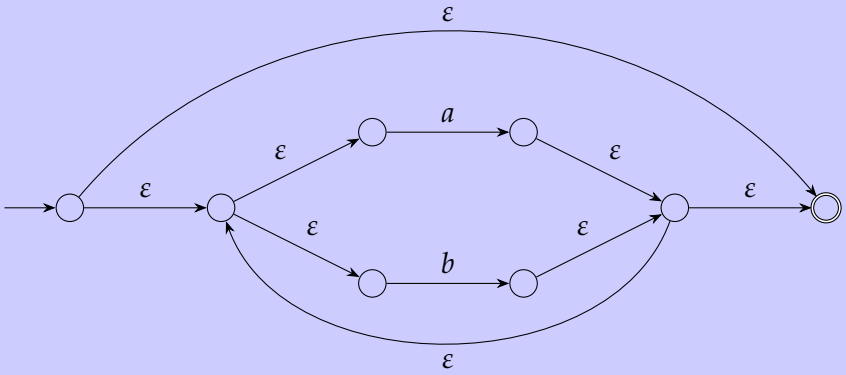
- M_a (und analog M_b):



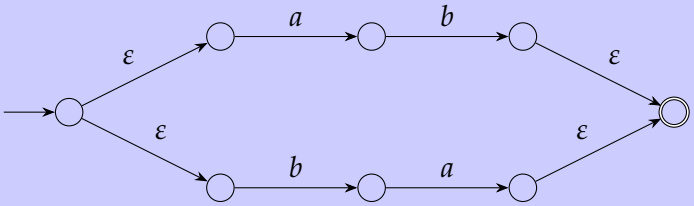
- $M_{a|b}$:



- $M_{(a|b)^*}$:



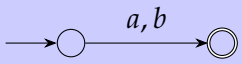
- $M_{ab|ba}$:



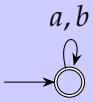
- $M_{(a|b)^*(ab|ba)(a|b)^*}$: $M_{(a|b)^*}$, $M_{ab|ba}$ und $M_{(a|b)^*}$, verbunden mit ε-Übergängen.

Sinnvollerweise eliminiert man bereits während der obigen Konstruktion ε-Übergänge und entfernt offensichtlich redundante Zustände. So erhält man kompaktere Automaten:

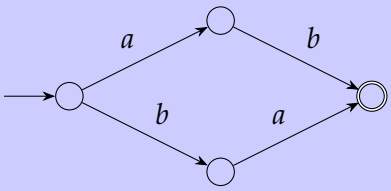
- $M_{a|b}$:



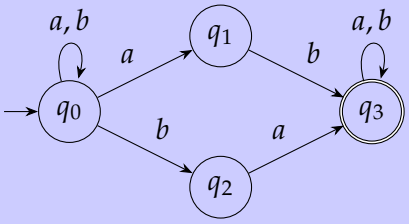
- $M_{(a|b)^*}$:



- $M_{ab|ba}$:



Insgesamt ergibt sich M_α:

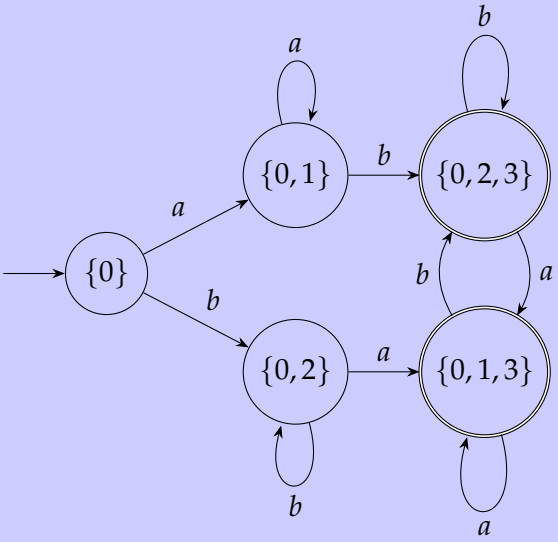


ii) Geben Sie einen DFA an, der $L(\alpha)$ erkennt. Sie können die Potenzmengenkonstruktion verwenden, müssen aber nicht.

LÖSUNGSVORSCHLAG:
Potenzmengenkonstruktion:

Start	$\longrightarrow$	Ziel
$\{0\}$	a	$\{0,1\}$
$\{0\}$	b	$\{0,2\}$
$\{0,1\}$	a	$\{0,1\}$
$\{0,1\}$	b	$\{0,2,3\}$
$\{0,2\}$	a	$\{0,1,3\}$
$\{0,2\}$	b	$\{0,2\}$
$\{0,1,3\}$	a	$\{0,1,3\}$
$\{0,1,3\}$	b	$\{0,2,3\}$
$\{0,2,3\}$	a	$\{0,1,3\}$
$\{0,2,3\}$	b	$\{0,2,3\}$

Der resultierende DFA M'_α:



Die zwei Endzustände könnte man zusammenfassen, aber es war kein Minimalautomat gefragt.

b) Geben Sie reguläre Ausdrücke an, die die folgenden Sprachen erkennen.

i) Die Sprache L_3 der Wörter über dem Alphabet $\Sigma_1 = \{a, b, c\}$, die mit a oder b anfangen und mindestens ein c enthalten.

LÖSUNGSVORSCHLAG:
$(a|b)(a|b|c)^*c(a|b|c)^*$ oder, äquivalent, $(a|b)(a|b)^*c(a|b|c)^*$.

c) Zeigen Sie mithilfe der Abschlusseigenschaften regulärer Sprachen, dass die Sprache $L_5 = \{a^i w d^{i+1} \mid i \in \mathbb{N}, w \in \{b, c\}^*\}$ über dem Alphabet $\Sigma_3 = \{a, b, c, d\}$ nicht regulär ist. Sie dürfen annehmen, dass die Sprache L_1 aus Aufgabe FSK4-Kb)

nicht regulär ist.

LÖSUNGSVORSCHLAG:
Widerspruchsbeweis: Nimm an, dass L_5 regulär ist.
Wir definieren die Sprache

$$\begin{aligned} L_5' &= \{a\} \cdot L_5 = \{a^{i+1} w d^{i+1} \mid i \in \mathbb{N},\, w \in \{b,c\}^*\} \\ &= \{a^i w d^i \mid i \in \mathbb{N}_{>0},\, w \in \{b,c\}^*\} \end{aligned}$$

L_5' ist regulär, denn $\{a\}$ ist eine reguläre Sprache und die regulären Sprachen sind unter Produkt abgeschlossen.
Nun definieren wir

$$\begin{aligned} L_5'' &= L_5' \cap L(aa^*bb^*cc^*dd^*) \\ &= L_5' \cap \{a^i b^j c^k d^l \mid i,j,k,l \in \mathbb{N}_{>0}\} \\ &= \{a^i b^j c^k d^i \mid i,j,k \in \mathbb{N}_{>0}\} \end{aligned}$$

L_5'' ist regulär, denn $L(aa^*bb^*cc^*dd^*)$ ist die Sprache eines regulären Ausdrucks (und damit regulär) und die regulären Sprachen sind unter Schnitt abgeschlossen.
Allerdings wissen wir aus Aufgabe FSK4-Kb), dass $L_5'' = L_1$ nicht regulär ist. Die Annahme, dass L_5 regulär sei, führt somit zu einem Widerspruch.

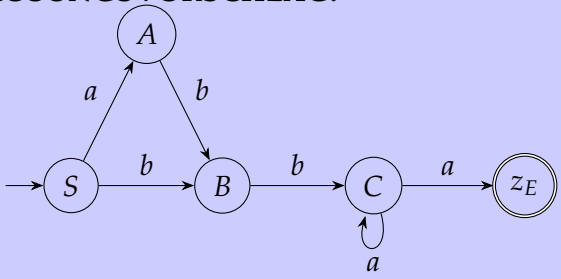
FSK4-3 ***Grammatik über Automaten zu Grammatik*** (0 Punkte)

Gegeben sei die reguläre Grammatik

$$G = (\{S, A, B, C\}, \{a, b\}, \{S \to aA \mid bB,\, A \to bB,\, B \to bC,\, C \to aC \mid a\}, S)$$

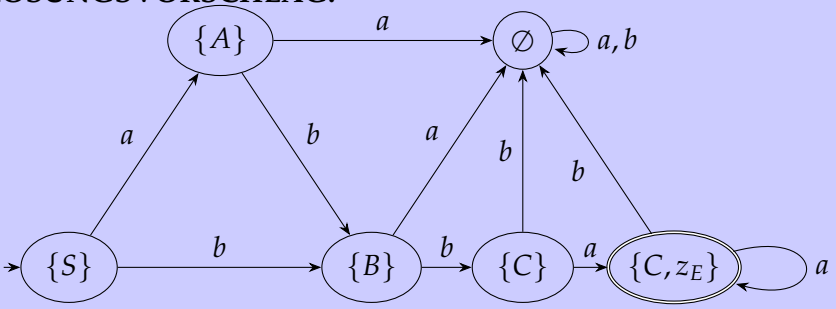
a) Erzeugen Sie gemäß der Konstruktion aus der Vorlesung aus G einen NFA A mit $L(G) = L(A)$.

LÖSUNGSVORSCHLAG:



b) Erzeugen Sie mit der Potenzmengenkonstruktion aus A einen DFA B mit $L(B) = L(A)$. Geben Sie nur den vom Startzustand erreichbaren Teil von A an.

LÖSUNGSVORSCHLAG:



c) Erzeugen Sie gemäß der Konstruktion aus der Vorlesung aus B eine Grammatik H mit $L(B) = L(H)$.

LÖSUNGSVORSCHLAG:
Neubenennung der Zustände (um sie klarer erkenntlich als Variablennamen schreiben zu können):
$V_S := \{S\}, V_A := \{A\}, V_B := \{B\}, V_C := \{C\}, V_{CE} := \{C, z_E\}, V_\emptyset = \emptyset$
$H = (\{V_S, V_A, V_B, V_C, V_{CE}, V_\emptyset\}, \{a, b\}, \{$
$V_S \to aV_A \mid bV_B,$
$V_A \to aV_\emptyset \mid bV_B,$
$V_B \to aV_\emptyset \mid bV_C,$
$V_C \to aV_{CE} \mid a \mid bV_\emptyset,$
$V_{CE} \to aV_{CE} \mid a \mid bV_\emptyset,$
$V_\emptyset \to aV_\emptyset \mid bV_\emptyset,$
$\}, V_S)$

d) Vergleichen Sie die Grammatiken G und H. Beschreiben Sie die Gemeinsamkeiten dieser Grammatiken, sowie ihre Unterschiede.

Überlegen Sie sich, wodurch diese Effekte zustande kommen.

LÖSUNGSVORSCHLAG:
Grund für diese Effekte ist, dass G direkt einem nichtdeterministischen Automaten entspricht und H direkt einem deterministischen Automaten. Darum übertragen sich die Eigenschaften der entsprechenden Automaten und Automatenmodelle auf die Grammatiken.
Gemeinsamkeiten:

- Sie akzeptieren die gleiche Sprache (aufgrund der Konstruktion).

- Es gibt sowohl in G als auch in H Variablen, die ein Terminalsymbol und sich selbst erzeugen.

Unterschiede:

- H enthält mehr Variablen und deutlich mehr Regeln als G (da die Automaten unterschiedlich groß sind).
- H enthält die Variable $V_\emptyset$, über die beliebig („unendlich“) lange abgeleitet werden kann, ohne dass je ein Wort produziert würde. Dadurch kann man für jedes Teilwort $w \in \Sigma^*$ eine Satzform wV_x (wobei V_x eine Variable ist) ableiten. Wendet man die Regeln von H „blind“ an, kommt man also leicht in eine Sackgasse. G hat dieses Problem nicht.

FSK4-4 ***DNA-Analyse mit NFA*** (0 Punkte)

Diese Aufgabe handelt von der Analyse von Desoxyribonukleinsäure (DNS/DNA) mithilfe von NFA. DNA ist eine Abfolge der Basen Adenin, Thymin, Guanin und Cytosin, typischerweise mit A, T, G und C abgekürzt. Dementsprechend ist das Alphabet aller Automaten in dieser Aufgabe $\Sigma = \{A, C, G, T\}$.

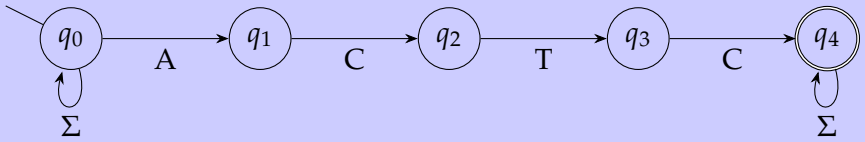
a) Um das Vorkommen einer Basensequenz zu finden, wird aus dieser Sequenz ein NFA erzeugt, der alle Wörter akzeptiert, in denen diese Sequenz als Teilwort vorkommt.

Geben Sie einen NFA B an, der genau diejenigen Wörter akzeptiert, in denen $ACTC$ als Teilwort vorkommt.

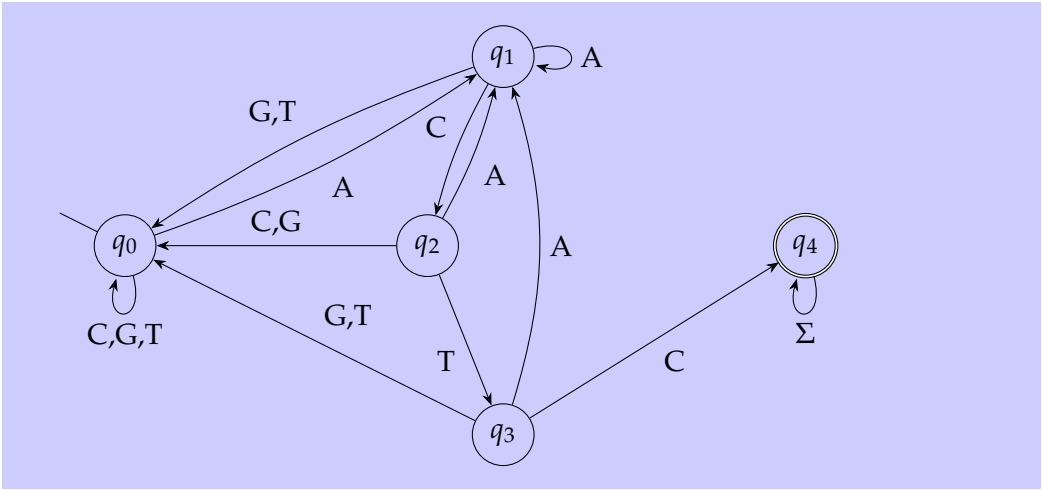
Hinweis: Sie können (müsssen aber nicht) dazu den regulären Ausdruck $(A|C|G|T)^*ACTC(A|C|G|T)^*$ verwenden, der genau diese Sprache akzeptiert.

Hinweis: Sie können auch einen DFA angeben, aber ein NFA ist übersichtlicher.

LÖSUNGSVORSCHLAG:



Ein dazu äquivalenter DFA ist



b) Beim Kopieren von DNA kann es vorkommen, dass Fehler auftreten. Zum Beispiel kann eine Base durch eine andere ersetzt werden; es kann eine Base ausgelassen werden; es kann eine zusätzliche Base eingefügt werden; und es können auch komplexere Fehler auftreten. Zur Vereinfachung behandeln wir hier nur den Fall, dass eine Base durch eine andere ersetzt wird.

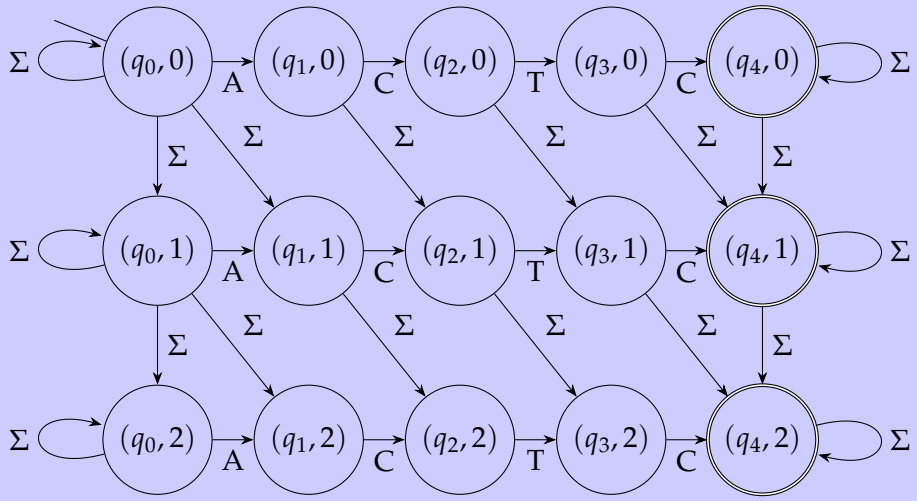
Aus einem NFA $D = (Z_D, \Sigma, \delta_D, S_D, E_D)$ kann ein NFA $F = (Z_F, \Sigma, \delta_F, S_F, E_F)$ erzeugt werden, der alle Wörter akzeptiert, die durch höchstens k fehlerhafte Ersetzungen aus D entsteht.

Dabei sind

- $Z_F = Z_D \times \{0, \ldots, k\}$
- $\delta_F((q,i),a) = \{(q',i) \mid q' \in \delta_D(q,a)\} \cup \{(q',i+1) \mid \exists b \in \Sigma.\, q' \in \delta_D(q,b) \wedge i+1 \leq k\}$
- $S_F = S_D \times \{0\} = \{(s,0) \mid s \in S_D\}$
- $E_F = E_D \times \{0, \ldots, k\}$

Berechnen Sie mit der obigen Konstruktion einen NFA H aus B, der Wörter mit bis zu 2 Fehlern akzeptiert.

LÖSUNGSVORSCHLAG:



c) Geben Sie an und begründen Sie, welche der folgenden Wörter von H akzeptiert werden. Prüfen Sie, ob Ihr Ergebnis korrekt ist, also ob die erkannten Wörter tatsächlich diejenigen sind, bei denen bis auf höchstens 2 Fehler das Wort $ACTC$ als Teilwort vorkommt.

$AAAACCCAAA$, $GAGGCGT$, $TAGCA$, $TCTCA$

LÖSUNGSVORSCHLAG:

- $AAAACCCAAA \in L(H)$ mit akzeptierendem Lauf $(q_0,0) \xrightarrow{A} (q_0,0) \xrightarrow{A} (q_0,0) \xrightarrow{A} (q_0,0) \xrightarrow{A} (q_1,0) \xrightarrow{C} (q_2,0) \xrightarrow{C} (q_3,1) \xrightarrow{C} (q_4,1) \xrightarrow{A} (q_4,1) \xrightarrow{A} (q_4,1) \xrightarrow{A} (q_4,1)$.

  Ist korrekt: $ACTC$ kommt mit 1 Fehler im unterstrichenen Teil vor: $AAA\underline{ACCC}AAA$

- $GAGGCGT \in L(H)$ mit akzeptierendem Lauf $(q_0,0) \xrightarrow{G} (q_0,0) \xrightarrow{A} (q_1,0) \xrightarrow{G} (q_2,1) \xrightarrow{G} (q_3,2) \xrightarrow{C} (q_4,2) \xrightarrow{G} (q_4,2) \xrightarrow{T} (q_4,2)$.

  Ist korrekt: $ACTC$ kommt mit 2 Fehlern im unterstrichenen Teil vor: $G\underline{AGGC}GT$

- $TAGCA \notin L(H)$. Das ist zu erkennen, indem für jede Wortposition die Menge der erreichbaren Zustände aufgeführt wird. (Dies entspricht auch dem Lauf auf dem zugehörigen DFA aus der Potenzmengenkonstruktion, der muss dafür aber nicht vollständig konstruiert werden.)

  $\{(q_0,0)\} \xrightarrow{T} \{(q_0,0),(q_0,1),(q_1,1)\} \xrightarrow{A}$
  $\{(q_0,0),(q_0,1),(q_0,2),(q_1,0),(q_1,1),(q_1,2),(q_2,2)\} \xrightarrow{G}$

$\{(q_0,0),(q_0,1),(q_0,2),(q_1,1),(q_1,2),(q_2,1),(q_2,2)\} \xrightarrow{C}$
$\{(q_0,0),(q_0,1),(q_0,2),(q_1,1),(q_1,2),(q_2,1),(q_2,2)\} \xrightarrow{A}$
$\{(q_0,0),(q_0,1),(q_0,2),(q_1,0),(q_1,1),(q_1,2),(q_2,2)\}$

Diese Mengen beinhalten je die mit diesem Wortanfang erreichbaren Zustände. Von einer Menge kommt man zur nächsten, indem man die Vereinigung aller $\delta_H(q,a)$ bildet. Zwei Beispiele aus dieser Berechnung:

$\{(q_0,0)\} \xrightarrow{T} \{(q_0,0),(q_0,1),(q_1,1)\}$, denn
$\delta_H((q_0,0),T) = \{(q_0,0),(q_0,1),(q_1,1)\}$

$\{(q_0,0),(q_0,1),(q_1,1)\} \xrightarrow{A}$
$\{(q_0,0),(q_0,1),(q_0,2),(q_1,0),(q_1,1),(q_1,2),(q_2,2)\}$, denn

$$\begin{aligned} & \delta_H((q_0,0),A) \cup \delta_H((q_0,1),A) \cup \delta_H((q_1,1),A) \\ = \; & \{(q_0,0),(q_0,1),(q_1,0),(q_1,1)\} \cup \{(q_0,1),(q_0,2),(q_1,1),(q_1,2)\} \cup \\ & \{(q_2,2)\} \\ = \; & \{(q_0,0),(q_0,1),(q_0,2),(q_1,0),(q_1,1),(q_1,2),(q_2,2)\} \end{aligned}$$

Nun ist aber $\{(q_0,0),(q_0,1),(q_0,2),(q_1,0),(q_1,1),(q_1,2),(q_2,2)\} \cap E_H = \varnothing$, also $TAGCA \notin L(H)$.

Dieses Ergebnis ist auch korrekt, denn alle Teilwörter der Länge 4 unterscheiden sich in mehr als 2 Zeichen von $ACTC$: $TAGC$ (Unterschied: 3) und $AGCA$ (Unterschied 3).

- $TCTCA \in L(H)$ mit akzeptierendem Lauf $(q_0,0) \xrightarrow{T} (q_1,1) \xrightarrow{C} (q_2,1) \xrightarrow{T} (q_3,1) \xrightarrow{C} (q_4,1) \xrightarrow{A} (q_4,1)$.

  Ist korrekt: $ACTC$ kommt mit 1 Fehler im unterstrichenen Teil vor: $\underline{TCTC}A$

d) Beweisen Sie, dass die Konstruktion aus b) korrekt ist, also tatsächlich für jeden NFA D und jedes k einen NFA F liefert, der maximal k Fehler zulässt.

Hinweis: Sie können auch als Vorüberlegung dies erst für den NFA H zeigen.

LÖSUNGSVORSCHLAG:
Die Konstruktion ist korrekt, wenn folgende beiden Eigenschaften gelten, die wir getrennt zeigen:

- Wenn $w \in L(D)$ ist und u sich von w an maximal k Stellen unterscheidet, dann ist $u \in L(F)$.

- Wenn $w \in L(F)$ ist, dann gibt es ein $u \in L(D)$ sodass sich u von w an maximal k Stellen unterscheidet.

Gelten diese beiden Eigenschaften, dann akzeptiert F genau die Fehlersprache. Jetzt zeigen wir diese Eigenschaften:

- Da $w \in L(D)$, gibt es einen Lauf ϱ von D auf w.

  Wir definieren eine Folge, die angibt, wie viele Fehler wir bis zu einem bestimmten Punkt im Wort gesehen haben:

  $$f_i = \begin{cases} 0 & \text{wenn } i = 0 \\ f_{i-1} & \text{wenn } u[i] = w[i] \\ f_{i-1} + 1 & \text{wenn } u[i] \neq w[i] \end{cases}$$

  Es ist $f_{|w|} \leq k$, da sich u und w an maximal k Stellen unterscheiden.

  Nun ist $(\varrho_0, f_0)(\varrho_1, f_1) \dots (\varrho_{|w|}, f_{|w|})$ ein akzeptierender Lauf von F auf u, da die Übergänge alle möglich sind. An jeder Stelle passt wahlweise der Übergang, da ϱ ein Lauf von D auf w ist und sich die Fehlerzahl f nicht ändert, oder die Fehlerzahl erhöht sich um 1 und der Buchstabe an dieser Position ist irrelevant. Zudem ist $(\varrho_{|w|}, f_{|w|})$ ein Endzustand in F.

- Da $w \in L(F)$, gibt es einen Lauf $(\varrho_0, x_0)(\varrho_1, x_1) \dots (\varrho_{|w|}, x_{|w|})$ von F auf w.

  Wähle nun ein Wort u mit $|u| = |w|$ und für alle $0 < i \leq |w|$:

  $$u[i] = \begin{cases} w[i] & \text{wenn } x_{i-1} = x_i \\ \text{beliebig aus } \{a \mid \varrho_i \in \delta_D(\varrho_{i-1}, a)\} & \text{sonst} \end{cases}$$

  Dabei kann die Menge $\{a \mid \varrho_i \in \delta_D(\varrho_{i-1}, a)\}$ nicht leer sein, da es sonst in F keinen Übergang von (ϱ_{i-1}, x_{i-1}) nach (ϱ_i, x_i) gäbe.

  Da $(\varrho_{|w|}, x_{|w|}) \in E_F$ und damit $\varrho_{|w|} \in E_D$, ist $\varrho_0 \dots \varrho_{|w|}$ ein akzeptierender Lauf von D auf u, also $u \in L(D)$. Zudem unterscheidet sich u von w an maximal k Stellen, weil $x_{|w|} \leq k$ ist.