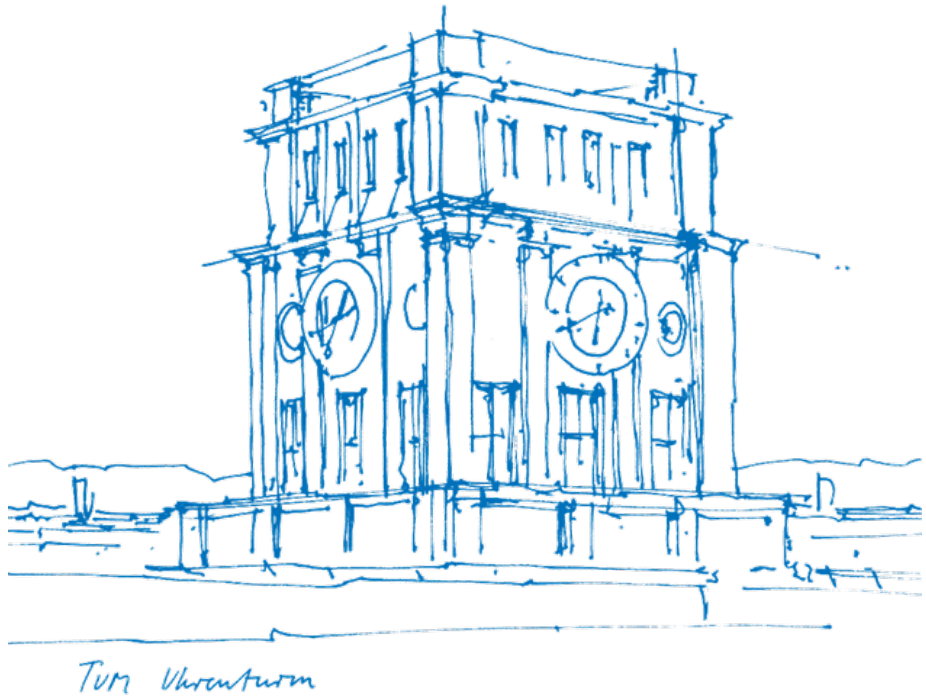
Einführung in die Theoretische Informatik

Tutorium – Woche 3

Esther Ney

Sommersemester 2025



Organisatorisches

Zulip

Für die diesjährige THEO-Vorlesung wurden **inoffizielle** Streams auf der ZULIP-Instanz der TUM INFORMATIK erstellt. Dieser wird von den meisten Tutor*innen, aber nicht von PROF. ALBERS gelesen; wir versuchen natürlich trotzdem, organisatorische Fragen bei Bedarf weiterzuleiten. Man kann mich dort auch in Direktnachrichten anschreiben.

Tutorin: ESTHER NEY

Tutorien: Dienstag 16-18 (Di-16-2) und Donnerstag 16-18 (Do-16-1)

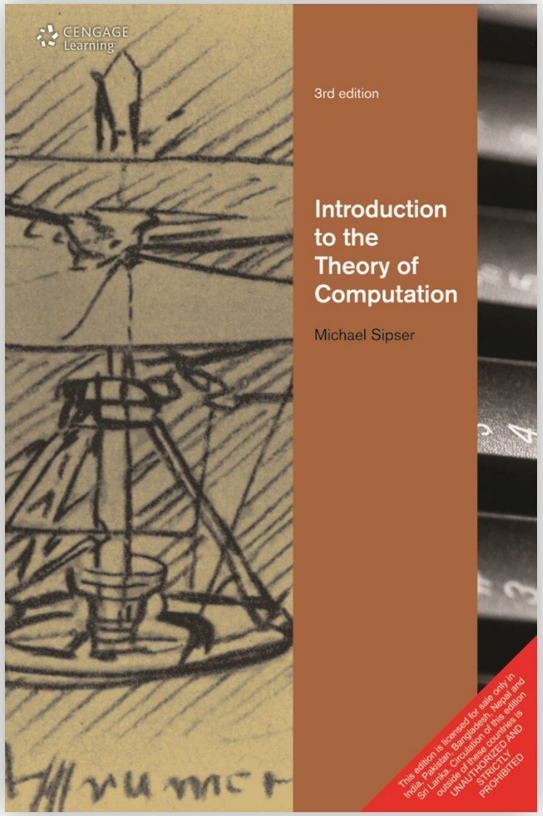
E-Mail: `esther.ney@tum.de`

Aufbau einer Tutorstunde: kurze Wiederholung wichtiger Definitionen und Lemmata, dann selbständiges Bearbeiten der Aufgaben

Folien: auf ZULIP oder `https://reflect.moe/uni/theo25/`

Literaturempfehlungen

Uwe Schöning – Theoretische Informatik kurz gefasst



Michael Sipser – Introduction to the Theory of Computation

Disclaimer

Hinweis

Diese Folien sind keine offizielle Musterlösung und dienen lediglich als zusätzliches Material für die Nachbereitung der Tutorien.

Tutorienlösungen können fehlerhaft sein.

Im Zweifel gilt immer das, was in den offiziellen Vorlesungsfolien und in der Musterlösung steht.

Urheberrechtlicher Hinweis

Reguläre Ausdrücke

Definition 3.19 – Regulärer Ausdruck

Ein REGULÄRER AUSDRUCK (RE oder REGEX, engl. *regular expression*) ist rekursiv definiert:

$\varnothing$ ist ein RE (1)

$a \in (\Sigma \cup \{\epsilon\})$ ist ein RE (2)

Für REs α, β ist auch ein RE:

$\alpha\beta$ (3)

$\alpha \mid \beta$ (4)

α^* (5)

Die SPRACHE eines RES ist definiert als:

$L(\varnothing) \coloneqq \varnothing, L(\epsilon) \coloneqq \{\varepsilon\}, L(a) \coloneqq a$ (6)

$L(\alpha\beta) \coloneqq L(\alpha)L(\beta), L(\alpha \mid \beta) \coloneqq L(\alpha) \cup L(\beta), L(\alpha^*) \coloneqq L(\alpha)^*$ (7)

Gleichmächtigkeit DFA – NFA – RE – Typ 3-Grammatik (1)

Lemma 3.9 – Gleichmächtigkeit von Typ 3-Grammatiken und NFAs

Sei $G := (V, \Sigma, P, S)$ eine rechtslineare Grammatik, also von Typ 3. Dann ist $N := (Q, \Sigma, \delta, q_0, F)$ mit $Q := V \cup \{q_F\}$, $Y \in \delta(X, a) :\Leftrightarrow (X \to aY) \in P$, $q_F \in \delta(X, a) :\Leftrightarrow (X \to a) \in P$, $q_0 := S$ und $F := \{q_F\}$.

Falls $(S \to \varepsilon) \in P$, setzen wir stattdessen $F := \{q_F, S\}$.

Intuitiv: Bei der Ableitung eines Wortes in einer Typ 3-Grammatik taucht aufgrund der Rechtslinearität immer nur ein Nichtterminal auf; dieses können wir aber nichtdeterministisch zwischen verschiedenen Produktionen $X \to aY \mid bZ \mid \ldots$ wählen. Wir setzen als Endzustand, dass dieses Nichtterminal verschwindet.

Anmerkung: Man kann auch einen Automaten zu Typ 2-Grammatiken konstruieren, das passiert später in Woche 8-9, Satz 4.53 – PDA-Konstruktion.

Gleichmächtigkeit DFA – NFA – RE – Typ 3-Grammatik (2)

Lemma 3.10 – Gleichmächtigkeit von NFAs und DFAs

Sei $N := (Q, \Sigma, \delta, q_0, F)$ ein NFA. Dann ist $M := (\mathcal{P}(Q), \Sigma, \bar{\delta}, \{q_0\}, F_M)$ mit $F_M := \{S \subseteq Q \mid S \cap F \neq \varnothing\}$ ein DFA mit $L(N) = L(M)$.

Erinnerung: Nach Vorlesung ist $\bar{\delta} : \mathcal{P}(Q) \times \Sigma \to \mathcal{P}(Q)$.

Lemma 3.13 – Gleichmächtigkeit von DFAs und Typ 3-Grammatiken

Sei $M := (Q, \Sigma, \delta, q_0, F)$ ein DFA. Dann ist $G := (V, \Sigma, P, S)$ mit $V := Q$, $S := q_0$, $q_1 \to aq_2 \in P :\Leftrightarrow \delta(q_1, a) = q_2$, $q_1 \to a \in P :\Leftrightarrow \delta(q_1, a) \in F$, $q_0 \to \varepsilon \in P :\Leftrightarrow q_0 \in F$ eine Grammatik von Typ 3 mit $L(M) = L(G)$.

Intuitiv: Bei der Potenzmengenkonstruktion bauen wir einen DFA, der alle Zustände annimmt, die ein NFA nach Lesen eines Wortes annehmen kann. Bei der Grammatikkonstruktion sehen wir die Übergänge des DFA als Produktionen der Grammatik, mit Produktionen der Form $q \to a$, wenn $\delta(q, a) \in F$.

Die erste Konstruktion ist in $O(2^{|V|})$.

Gleichmächtigkeit DFA – NFA – RE – Typ 3-Grammatik (3)

Satz 3.23 – Gleichmächtigkeit von REs und ϵ-NFAs

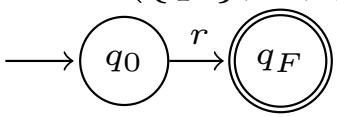
Diese Woche nur RE $\rightarrow$ ϵ-NFA. Die Rückrichtung gibts nächstes Blatt!

Sei r ein RE. Wir konstruieren einen ϵ-NFA N mit $L(r) = L(N)$.

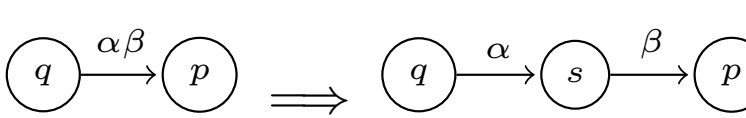
(1) Preprocessing: Ersetze alle Vorkommnisse von $\varnothing$ nach Folie 60, bis nur noch $\varnothing$ oder ein RE ohne $\varnothing$s übrig bleibt.

Falls nur $\varnothing$ übrig blieb, bilde den leeren NFA $N := (\{q_0\}, \Sigma, (\lambda x.\varnothing), q_0, \varnothing)$.

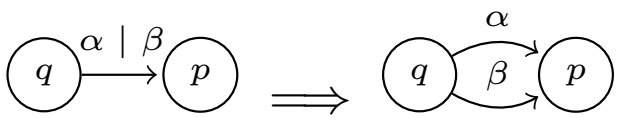
Ansonsten bilde den NFA-mit-RE-Übergängen:



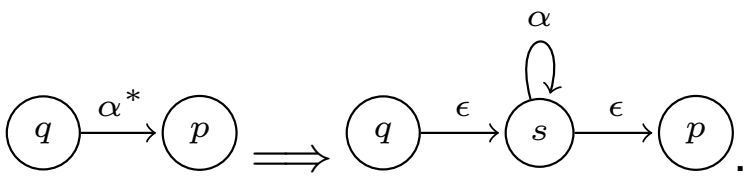
(2) Solange möglich, wende folgende Transformationsregeln an:



und



und



.

Übrig bleibt ein ϵ-NFA N mit $L(N) = L(r)$. N hat nur den Endzustand q_F.

Gleichmächtigkeit DFA – NFA – RE – Typ 3-Grammatik (4)

Lemma 3.16 – Gleichmächtigkeit von ϵ-NFAs und NFAs

Sei $N := (Q, \Sigma, \delta, q_0, F)$ ein ϵ-NFA. Dann ist $N' := (Q, \Sigma, \delta', q_0, F')$ mit $\delta'(q, a) := \cup_{i,j \geq 0} \delta(\{q\}, \epsilon^i a \epsilon^j)$ und $F' := F \cup [\![\texttt{if } \exists i. \delta(q_0, \epsilon^i) \cap F \neq \varnothing \texttt{ then } \{q_0\} \texttt{ else } \varnothing]\!]$ ein NFA mit $L(N) = L(N')$.

Intuition: Bei der Konstruktion RE $\rightarrow$ ϵ-NFA wandeln wir Zustände entlang der Rekursionsvorschrift für REs um, bis nur noch einzelne Symbole stehen bleiben. Bei der Konstruktion ϵ-NFA $\rightarrow$ NFA erlauben wir einen Übergang zum Buchstaben a, wenn wir dieses a mit ϵs "padden" können.

Bemerkung: Die Notation $[\![P]\!]$ ist auch als Iverson-Klammer bekannt, nach dem kanadischen Informatiker Kenneth E. Iverson.

Vorbereitung

Überprüfen Sie, dass Sie mit den folgenden Begriffen vertraut sind.

- Potenzmengenkonstruktion
- rechtslineare Grammatik
- ϵ-NFA
- regulärer Ausdruck

Aufgabe T 3.1

Mit $|w|_x$ bezeichnen wir die Anzahl der Vorkommen des Buchstabens $x \in \Sigma$ in $w \in \Sigma^*$.
Sei $\Sigma = \{a, b, c\}$ und $L = \left\{w \in \Sigma^* \mid \exists\, x \in \Sigma.\ |w|_x = 0\right\}$.

(a) Konstruieren Sie einen NFA N mit genau 4 Zuständen und $L(N) = L$.

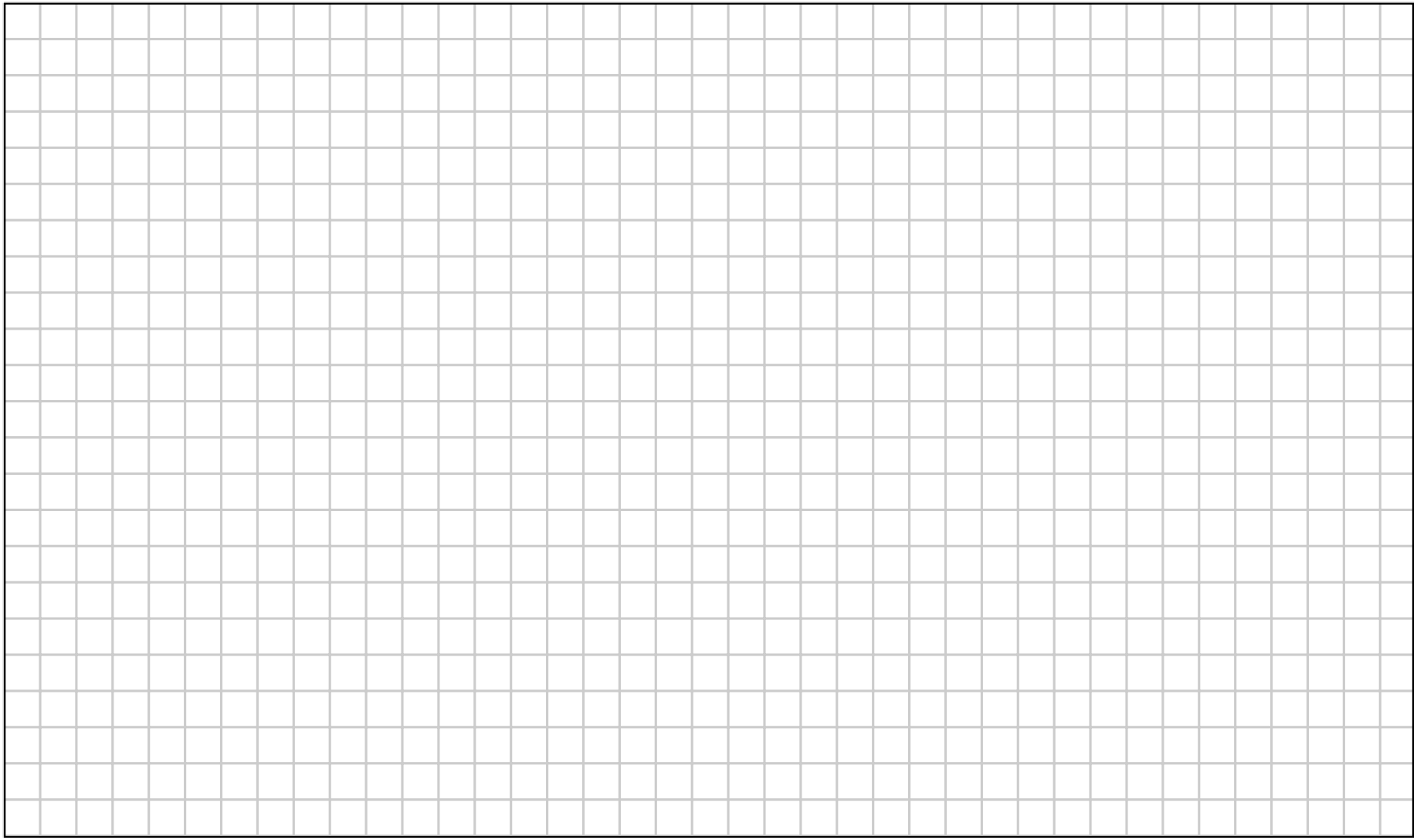
(b) Deterministieren Sie den NFA N aus (a) mittels der Potenzmengenkonstruktion, um einen DFA D mit $L(D) = L(N)$ zu erhalten.

(c) Geben Sie eine rechtslineare Grammatik G (mit SATZ 3.13) an, sodass $L(G) = L(D)$.

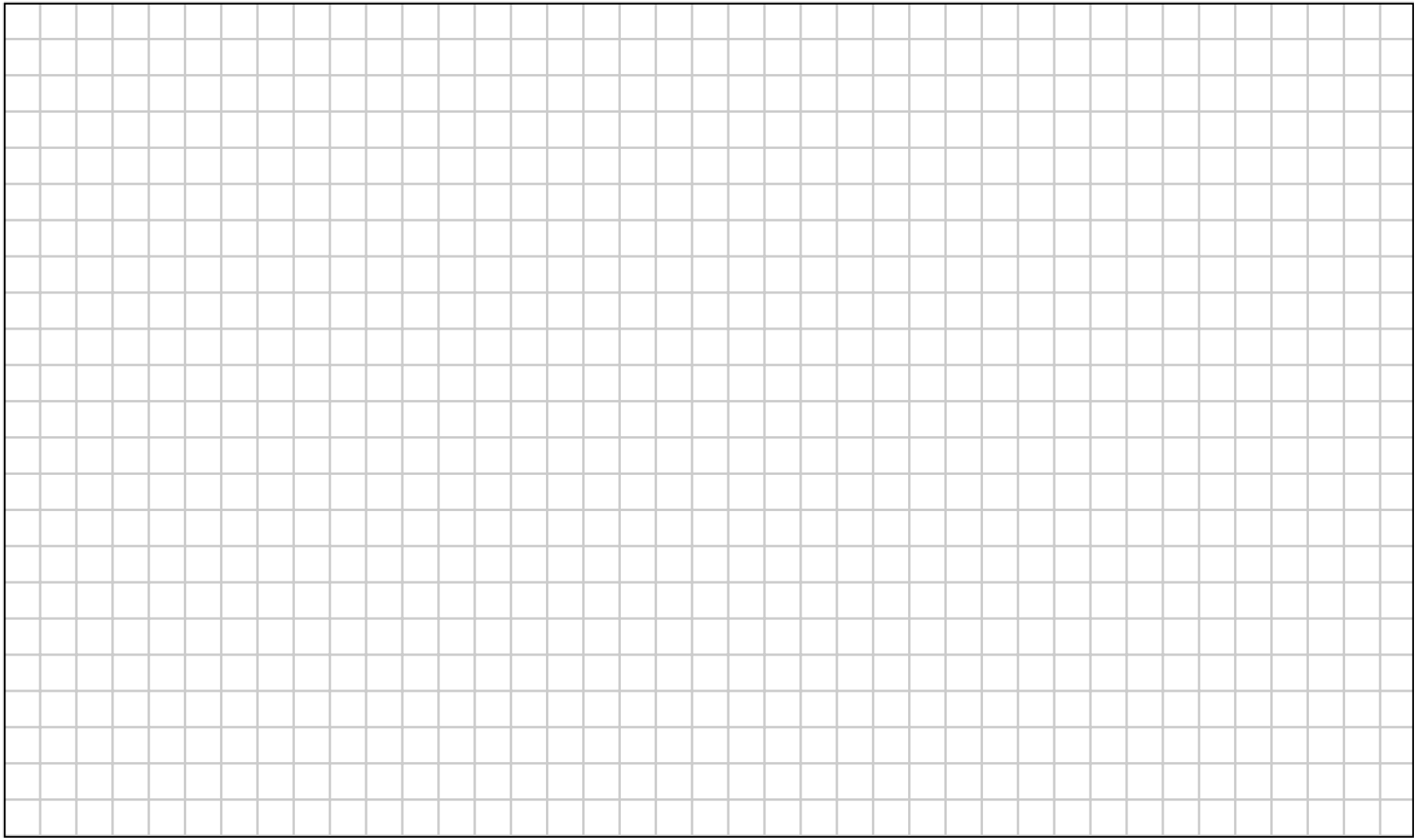
(d) Übersetzen Sie die Grammatik G (mit SATZ 3.9) in einen NFA N', sodass $L(N') = L(G)$.

(e) Vergleichen Sie die NFAs N' und N bezüglich Zustands- und Transitionszahl. Diskutieren Sie dann, inwiefern Sie Ihre Beobachtung verallgemeinern können.

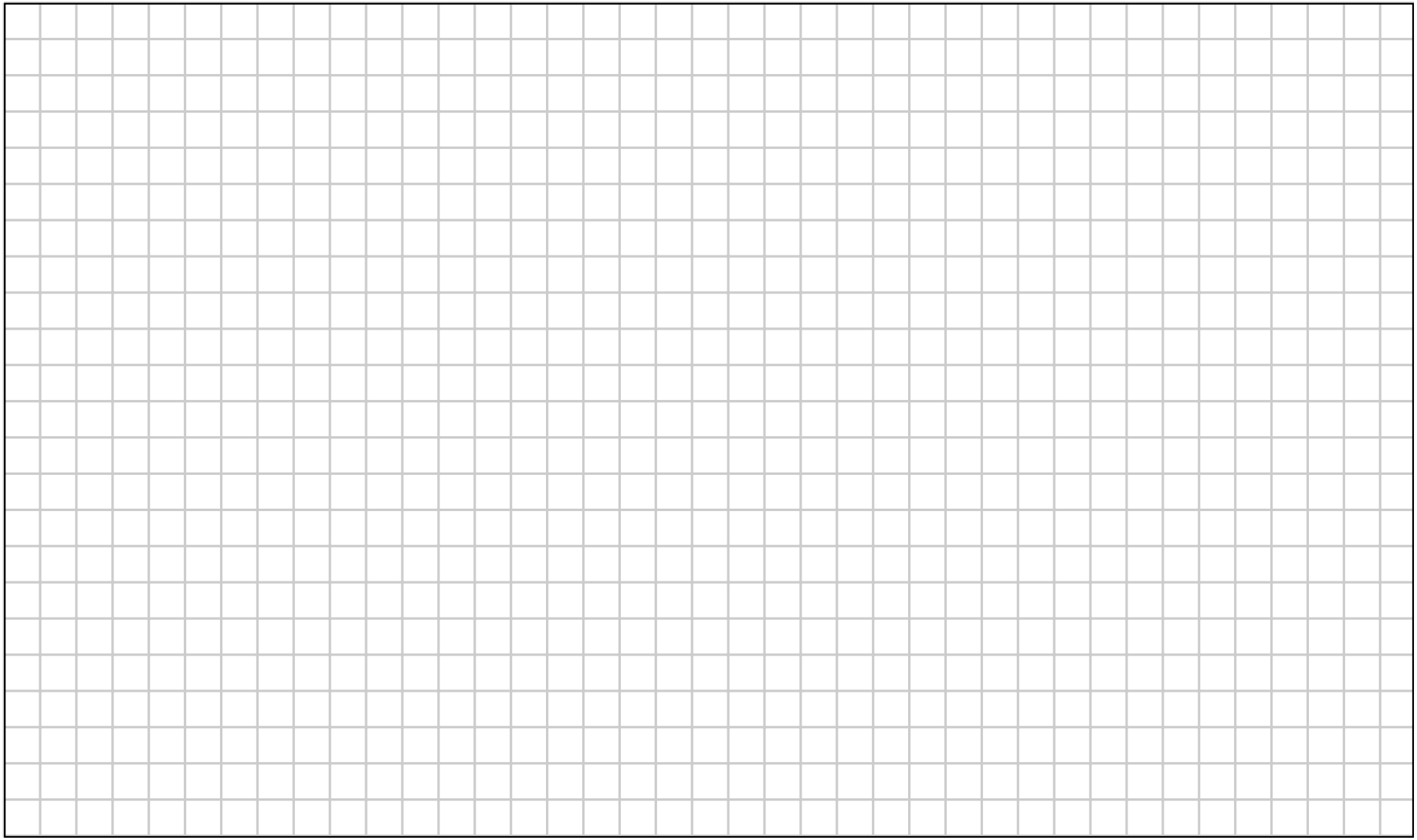
Aufgabe T 3.1



Aufgabe T 3.1



Aufgabe T 3.1



Aufgabe T3.2

Geben Sie für jede der folgenden Sprachen einen regulären Ausdruck an, der genau die Sprache beschreibt. Verwenden Sie für die ersten drei Aufgaben das Alphabet $\Sigma := \{a, b, c\}$ und für die nächsten beiden $\Sigma := \{0, 1\}$.

(a) Wörter gerader Länge, also: $L_A := \{w \in \Sigma^* \mid |w| \equiv 0 \text{ MOD } 2\}$.

(b) Wörter, die mit einem a beginnen und enden, sowie Wörter, die mit einem b beginnen und enden, also:
$L_B := \{w \in \Sigma^* \mid (w_0 = a \wedge w_{|w|-1} = a) \vee (w_0 = b \wedge w_{|w|-1} = b)\}$.

(c) Wörter, in denen kein a neben einem b steht, also:
$L_C := \{w \in \Sigma^* \mid \nexists x, y \in \Sigma^*.\, w = xaby \vee w = xbay\}$.

(d) Zahlen in Binärdarstellung (most-significant-bit-first), die durch 2 teilbar sind, also:
$L_D := \{w \in \Sigma^* \mid (w)_2 \equiv 0 \text{ MOD } 2\}$.

(e) Zahlen in Binärdarstellung (most-significant-bit-first), die nicht durch 4 teilbar sind, also: $L_E := \{w \in \Sigma^* \mid (w)_2 \not\equiv 0 \text{ MOD } 4\}$.

(f) Sei $\Sigma := \{a, b\}$. Wörter, die gleich oft die Zeichenketten ab und ba enthalten, also: $L_F := \{w \in \Sigma^* \mid |w|_{ab} = |w|_{ba}\}$ (mit Notation wie aus AUFGABE T3.1).

Beispiel: Das Wort $abab$ enthält zweimal ab, aber nur einmal ba, also $abab \notin L_F$.

Aufgabe T 3.2

Geben Sie einen RE r mit $L(r) = L$ über dem Alphabet $\Sigma := \{a, b, c\}$ an.

(a) Wörter gerader Länge, also: $L_A := \{w \in \Sigma^* \mid |w| \equiv 0 \text{ MOD } 2\}$.

(b) Wörter, die mit einem a beginnen und enden, sowie Wörter, die mit einem b beginnen und enden, also:
$L_B := \{w \in \Sigma^* \mid (w_0 = a \wedge w_{|w|-1} = a) \vee (w_0 = b \wedge w_{|w|-1} = b)\}$.

(c) Wörter, in denen kein a neben einem b steht, also:
$L_C := \{w \in \Sigma^* \mid \nexists x, y \in \Sigma^*.\, w = xaby \vee w = xbay\}$.

Aufgabe T3.2

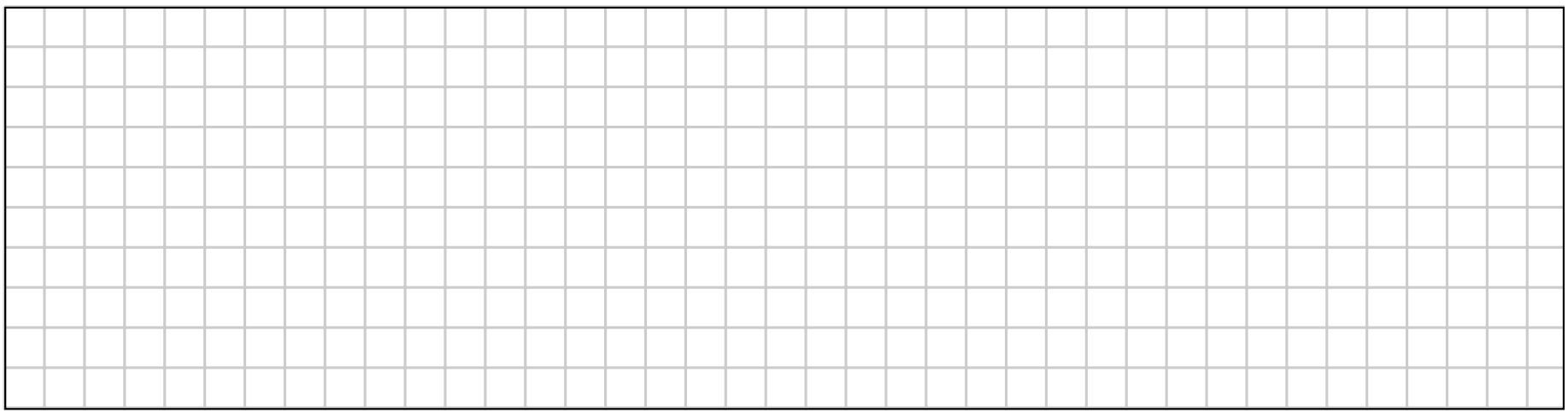
Geben Sie einen RE r mit $L(r) = L$ über dem Alphabet $\Sigma := \{0, 1\}$ bzw $\{a, b\}$ an.

(d) Zahlen in Binärdarstellung (most-significant-bit-first), die durch 2 teilbar sind, also: $L_D := \{w \in \Sigma^* \mid (w)_2 \equiv 0 \text{ MOD } 2\}$.

(e) Zahlen in Binärdarstellung (most-significant-bit-first), die nicht durch 4 teilbar sind, also: $L_E := \{w \in \Sigma^* \mid (w)_2 \not\equiv 0 \text{ MOD } 4\}$.

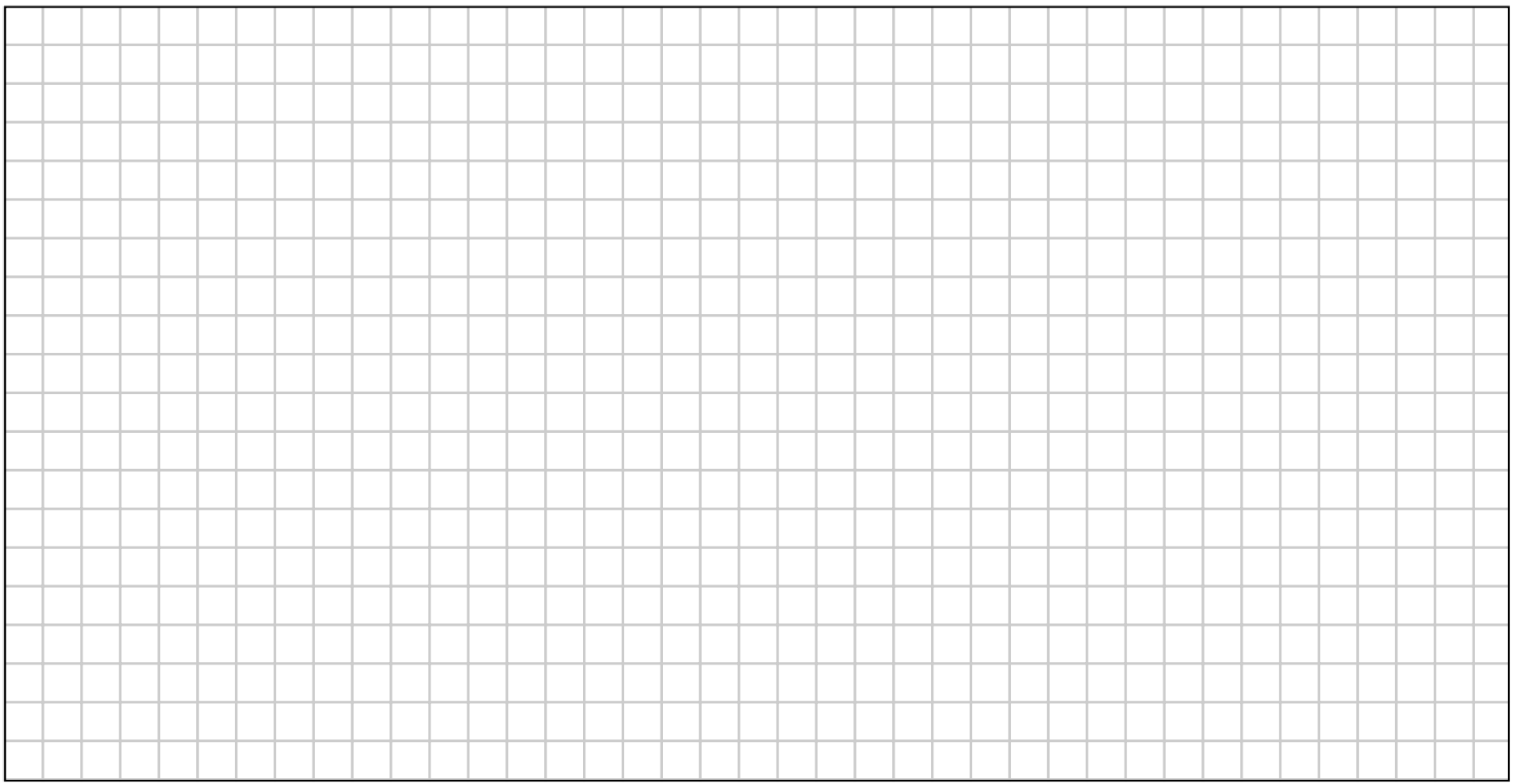
(f) Sei $\Sigma := \{a, b\}$. Wörter, die gleich oft die Zeichenketten ab und ba enthalten, also: $L_F := \{w \in \Sigma^* \mid |w|_{ab} = |w|_{ba}\}$ (mit Notation wie aus AUFGABE T3.1).

Beispiel: Das Wort $abab$ enthält zweimal ab, aber nur einmal ba, also $abab \notin L_F$.

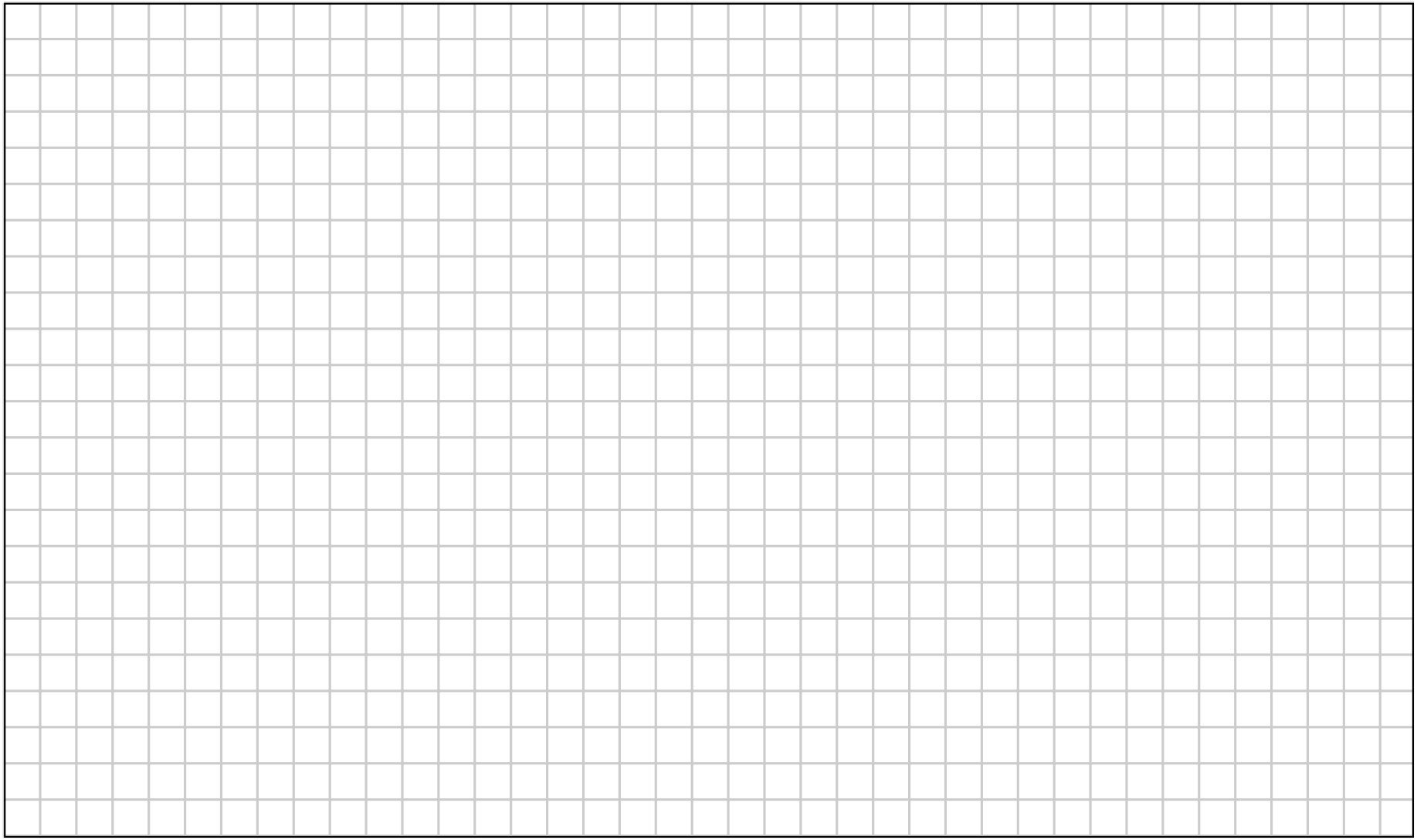


Aufgabe T 3.3

Wandeln Sie den regulären Ausdruck $a \mid (b \mid \varnothing)(a(b\varnothing)^*)^*$ in einen ϵ-NFA um. Folgen Sie hierfür strikt dem Vorgehen aus der Vorlesung.



Aufgabe T 3.3



Altklausuraufgabe

Endterm 2023 – Aufgabe 2b), 3 Punkte

Betrachten Sie den regulären Ausdruck $s := (0 \mid 1(10)^*(0|11)(01^*01|01^*00)^*1)^*$ über dem Alphabet $\Sigma := \{0, 1\}$. Wie viele Zustände hat der resultierende ϵ-NFA, wenn Sie den Algorithmus aus der Vorlesung (SATZ 3.23) ausführen würden? Begründen Sie Ihre Antwort!

Hinweis: Sie müssen den Algorithmus nicht ausführen.

Weitere 12 von 100 Punkten gibt es übrigens für das stumpfe Anwenden dieses Algorithmus.

Altklausuraufgabe