Teil VI

Anwendungen, Teil 1: XML und deterministische reguläre Ausdrücke

XML anhand von Beispielen...

XML-Schema

- In vielen Anwendungen sollen nur bestimmte XML-Dokumente zugelassen werden.

XML-Schema

- In vielen Anwendungen sollen nur bestimmte XML-Dokumente zugelassen werden.
- Validierung: Entscheiden, ob ein XML-Dokument „korrekt“ ist.

XML-Schema

- In vielen Anwendungen sollen nur bestimmte XML-Dokumente zugelassen werden.
- Validierung: Entscheiden, ob ein XML-Dokument „korrekt“ ist.
- Wie spezifiziert man korrekte XML-Dokumente?

XML-Schema

- In vielen Anwendungen sollen nur bestimmte XML-Dokumente zugelassen werden.
- Validierung: Entscheiden, ob ein XML-Dokument „korrekt“ ist.
- Wie spezifiziert man korrekte XML-Dokumente?

Lösung

- XML-Schema: Gibt an, welche XML-Dokumente erlaubt sind.

XML-Schema

- In vielen Anwendungen sollen nur bestimmte XML-Dokumente zugelassen werden.
- Validierung: Entscheiden, ob ein XML-Dokument „korrekt“ ist.
- Wie spezifiziert man korrekte XML-Dokumente?

Lösung

- XML-Schema: Gibt an, welche XML-Dokumente erlaubt sind.
- verfasst in Schema-Sprache: DTD, XSD, Relax NG

XML-Schema

- In vielen Anwendungen sollen nur bestimmte XML-Dokumente zugelassen werden.
- Validierung: Entscheiden, ob ein XML-Dokument „korrekt“ ist.
- Wie spezifiziert man korrekte XML-Dokumente?

Lösung

- XML-Schema: Gibt an, welche XML-Dokumente erlaubt sind.
- verfasst in Schema-Sprache: DTD, XSD, Relax NG

Wir betrachten eine stark eingeschränkte Variante von DTDs.

XML-Bäume und DTDs

Definitionen

Sei Σ Alphabet (**Elementnamen**).
Sei DATA Buchstabe, $\mathrm{DATA} \notin \Sigma$.
Sei $\Sigma_D := \Sigma \cup \{\mathrm{DATA}\}$.

XML-Bäume und DTDs

Definitionen

Sei Σ Alphabet (**Elementnamen**).
Sei DATA Buchstabe, $\mathrm{DATA} \notin \Sigma$.
Sei $\Sigma_D := \Sigma \cup \{\mathrm{DATA}\}$.

XML-Baum

Gerichteter endlicher Baum,
Knotenbeschritung aus Σ_D.
Innere Knoten: Σ, kein DATA.

Verallgemeinerte DTD über Σ

(Σ, ρ, τ),

XML-Bäume und DTDs

Definitionen

Sei Σ Alphabet (**Elementnamen**).
Sei DATA Buchstabe, $\mathrm{DATA} \notin \Sigma$.
Sei $\Sigma_D := \Sigma \cup \{\mathrm{DATA}\}$.

XML-Baum

Gerichteter endlicher Baum,
Knotenbeschritung aus Σ_D.
Innere Knoten: Σ, kein DATA.

Verallgemeinerte DTD über Σ

(Σ, ρ, τ), **Wurzelelement** $\rho \in \Sigma$,

XML-Bäume und DTDs

Definitionen

Sei Σ Alphabet (**Elementnamen**).
Sei DATA Buchstabe, $\mathrm{DATA} \notin \Sigma$.
Sei $\Sigma_D := \Sigma \cup \{\mathrm{DATA}\}$.

XML-Baum

Gerichteter endlicher Baum,
Knotenbeschritung aus Σ_D.
Innere Knoten: Σ, kein DATA.

Verallgemeinerte DTD über Σ

(Σ, ρ, τ), **Wurzelelement** $\rho \in \Sigma$, **Elementtypfunktion** τ:

von Σ zu regulären Ausdrücken über Σ_D

XML-Bäume und DTDs

Definitionen

Sei Σ Alphabet (**Elementnamen**).
Sei DATA Buchstabe, $\mathrm{DATA} \notin \Sigma$.
Sei $\Sigma_D := \Sigma \cup \{\mathrm{DATA}\}$.

XML-Baum

Gerichteter endlicher Baum,
Knotenbeschritung aus Σ_D.
Innere Knoten: Σ, kein DATA.

Verallgemeinerte DTD über Σ

(Σ, ρ, τ), **Wurzelelement** $\rho \in \Sigma$, **Elementtypfunktion** τ:

von Σ zu regulären Ausdrücken über Σ_D

XML-Baum T gültig unter DTD $D = (\Sigma, \rho, \tau)$:

- Wurzel von T ist mit ρ beschriftet.
- Kinder von Knoten a bilden Wort aus $\mathcal{L}(\tau(a))$.

XML-Bäume und DTDs

Definitionen

Sei Σ Alphabet (**Elementnamen**).
Sei DATA Buchstabe, $\mathrm{DATA} \notin \Sigma$.
Sei $\Sigma_D := \Sigma \cup \{\mathrm{DATA}\}$.

XML-Baum

Gerichteter endlicher Baum,
Knotenbeschritung aus Σ_D.
Innere Knoten: Σ, kein DATA.

Verallgemeinerte DTD über Σ

(Σ, ρ, τ), **Wurzelelement** $\rho \in \Sigma$, **Elementtypfunktion** τ:

von Σ zu regulären Ausdrücken über Σ_D

XML-Baum T gültig unter DTD $D = (\Sigma, \rho, \tau)$:

- Wurzel von T ist mit ρ beschriftet.
- Kinder von Knoten a bilden Wort aus $\mathcal{L}(\tau(a))$.

$T \models D$

XML-Bäume und DTDs

Definitionen

Sei Σ Alphabet (**Elementnamen**).
Sei DATA Buchstabe, $\mathrm{DATA} \notin \Sigma$.
Sei $\Sigma_D := \Sigma \cup \{\mathrm{DATA}\}$.

XML-Baum

Gerichteter endlicher Baum,
Knotenbeschritung aus Σ_D.
Innere Knoten: Σ, kein DATA.

Verallgemeinerte DTD über Σ

(Σ, ρ, τ), **Wurzelelement** $\rho \in \Sigma$, **Elementtypfunktion** τ:

von Σ zu regulären Ausdrücken über Σ_D

XML-Baum T gültig unter DTD $D = (\Sigma, \rho, \tau)$:

- Wurzel von T ist mit ρ beschriftet.
- Kinder von Knoten a bilden Wort aus $\mathcal{L}(\tau(a))$.

$T \models D$

$$\mathcal{L}(D) := \{T \text{ ist XML-Baum} \mid T \models D\}$$

Beispiele

- Beispiele für XML-Bäume und DTDs.

Beispiele

- Beispiele für XML-Bäume und DTDs.
- Beispiele für Nicht-Ausdrückbarkeit.

Beispiele

- Beispiele für XML-Bäume und DTDs.
- Beispiele für Nicht-Ausdrückbarkeit.

Beobachtung

- Durch reguläre Ausdrücke werden viele Entscheidungsprobleme für DTDs schwer.

Beispiele

- Beispiele für XML-Bäume und DTDs.
- Beispiele für Nicht-Ausdrückbarkeit.

Beobachtung

- Durch reguläre Ausdrücke werden viele Entscheidungsprobleme für DTDs schwer.
- Lösung: DTDs einschränken, indem die regulären Ausdrücke eingeschränkt werden.

Deterministische reguläre Ausdrücke

Sei Σ Alphabet, $n \in \mathbb{N}_{>0}$.
n-**Indizierung von** Σ: $\Sigma_{(n)} := \{a_i \mid 1 \leq i \leq n\}$.

Deterministische reguläre Ausdrücke

Sei Σ Alphabet, $n \in \mathbb{N}_{>0}$.
n-**Indizierung von** Σ: $\Sigma_{(n)} := \{a_i \mid 1 \leq i \leq n\}$.

Indizierung von α

Sei α regulärer Ausdruck über Σ.
Indizierung $\widetilde{\alpha}$: Für alle $a \in \Sigma$, alle i:
Ersetze i-tes Vorkommen von a in α durch a_i.

Deterministische reguläre Ausdrücke

Sei Σ Alphabet, $n \in \mathbb{N}_{>0}$.
n-**Indizierung von** Σ: $\Sigma_{(n)} := \{a_i \mid 1 \leq i \leq n\}$.

Indizierung von α

Sei α regulärer Ausdruck über Σ.
Indizierung $\widetilde{\alpha}$: Für alle $a \in \Sigma$, alle i:
Ersetze i-tes Vorkommen von a in α durch a_i.

Deterministische reguläre Ausdrücke

Sei Σ ein Alphabet. Ein regulärer Ausdruck α über Σ ist ein **deterministischer regulärer Ausdruck**, wenn:
Für alle Wörter $x, y, z \in \Sigma_{(n)}$ und alle $a_i, a_j \in \Sigma_{(n)}$ gilt:

$$\text{Aus } xa_iy \in \mathcal{L}(\widetilde{\alpha}) \text{ und } xa_jz \in \mathcal{L}(\widetilde{\alpha}) \text{ folgt } i = j.$$

Hierbei sei $n := \max\{|\alpha|_a \mid a \in \Sigma\}$.

Glushkov-Automaten

- Alternatives Konstruktionsverfahren, um regulären Ausdruck in NFA umzuwandeln.

Glushkov-Automaten

- Alternatives Konstruktionsverfahren, um regulären Ausdruck in NFA umzuwandeln.
- Verwendet keine ε-Übergänge.

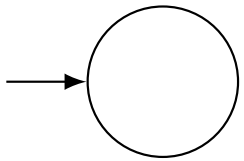
Glushkov-Automaten

- Alternatives Konstruktionsverfahren, um regulären Ausdruck in NFA umzuwandeln.
- Verwendet keine ε-Übergänge.
- Deterministische reguläre Ausdrücke werden zu DFAs.

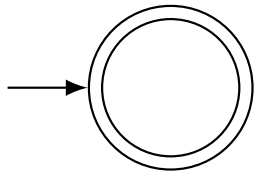
Glushkov-Automaten

- Alternatives Konstruktionsverfahren, um regulären Ausdruck in NFA umzuwandeln.
- Verwendet keine ε-Übergänge.
- Deterministische reguläre Ausdrücke werden zu DFAs.

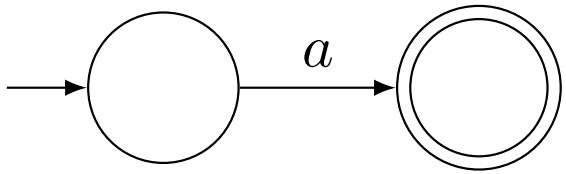
$A_{\mathcal{G}}(\emptyset)$



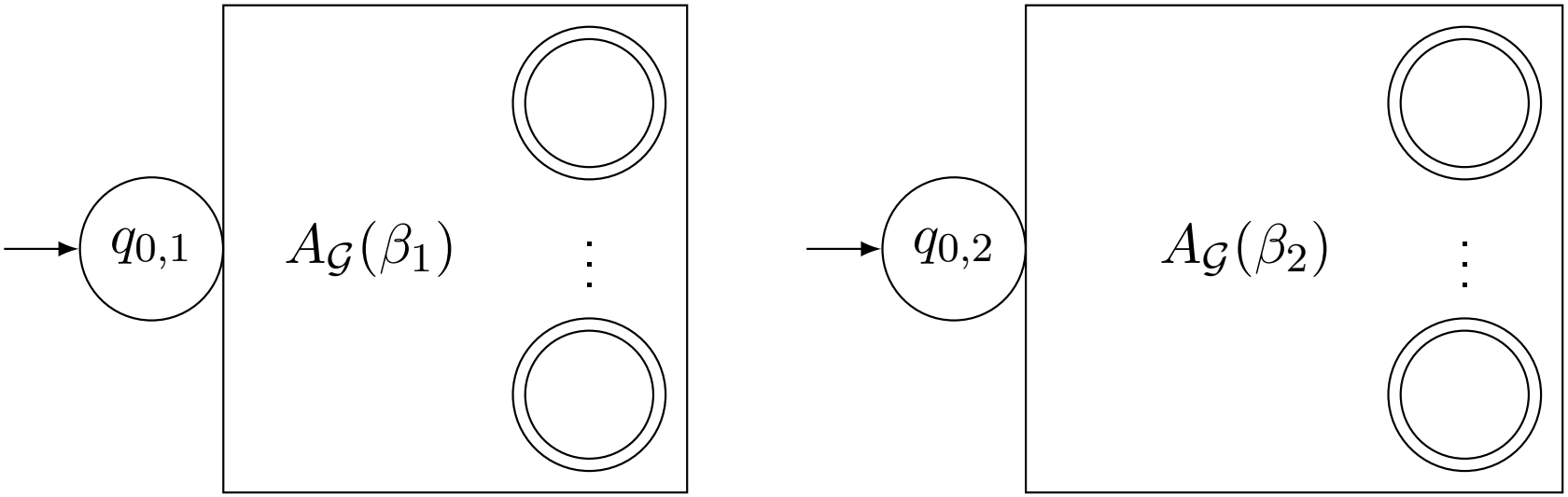
$A_{\mathcal{G}}(\varepsilon)$



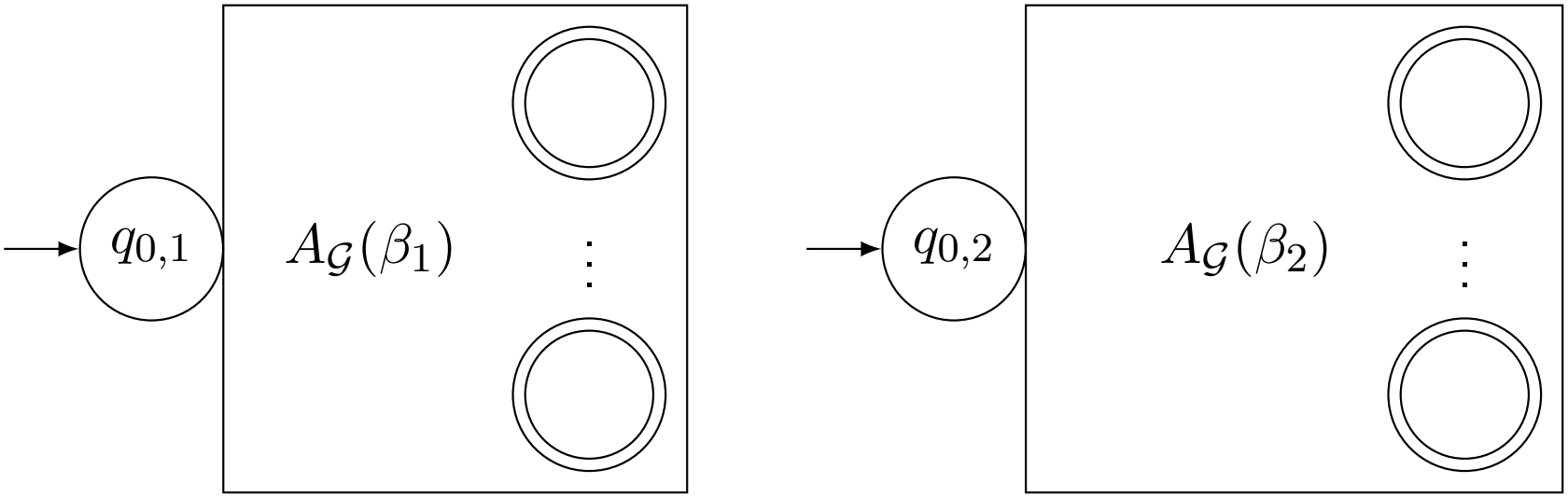
$A_{\mathcal{G}}(a)$



Konkatenation

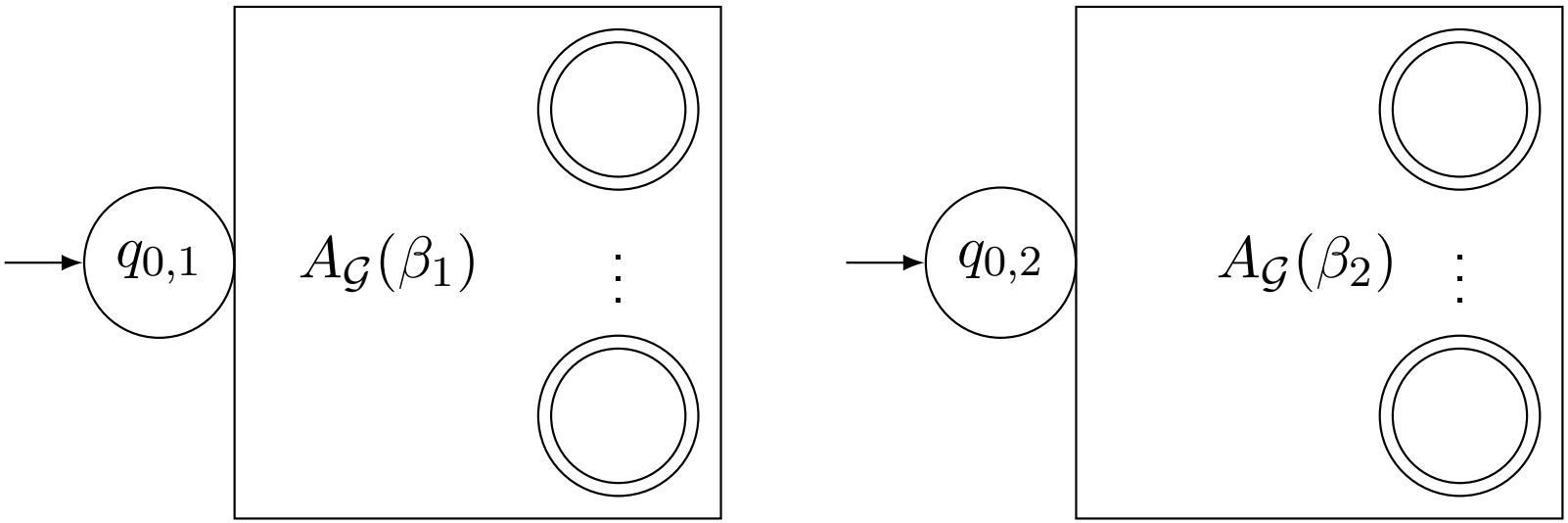


Konkatenation

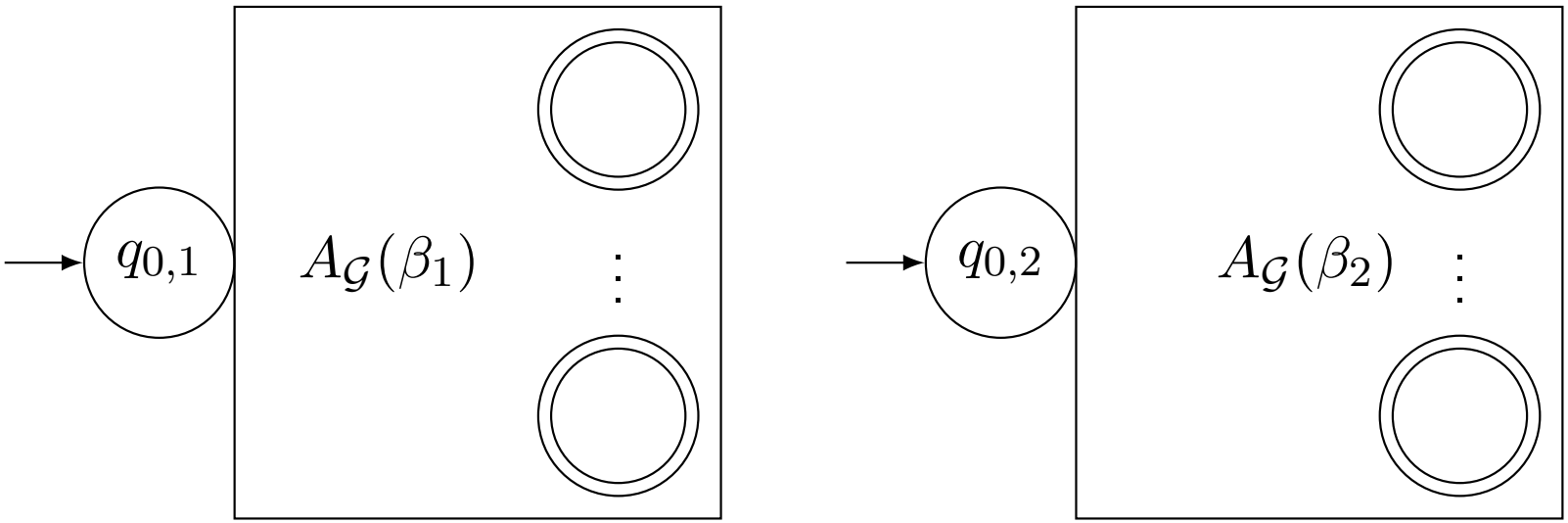


- Jeder akzeptierende Zustand von $A_{\mathcal{G}}(\beta_1)$ erhält die gleichen Ausgänge wie $q_{0,2}$,
- $q_{0,2}$ wird gelöscht.
- Zustände von $A_{\mathcal{G}}(\beta_1)$ sind nur akzeptierend, wenn $q_{0,2}$ akzeptierend.

Vereinigung

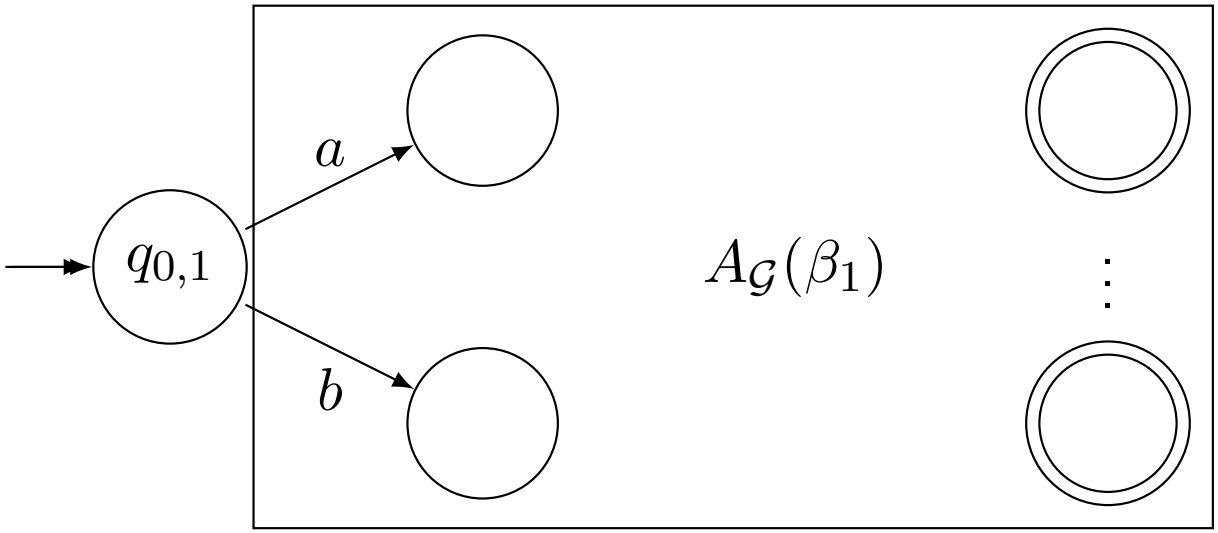


Vereinigung

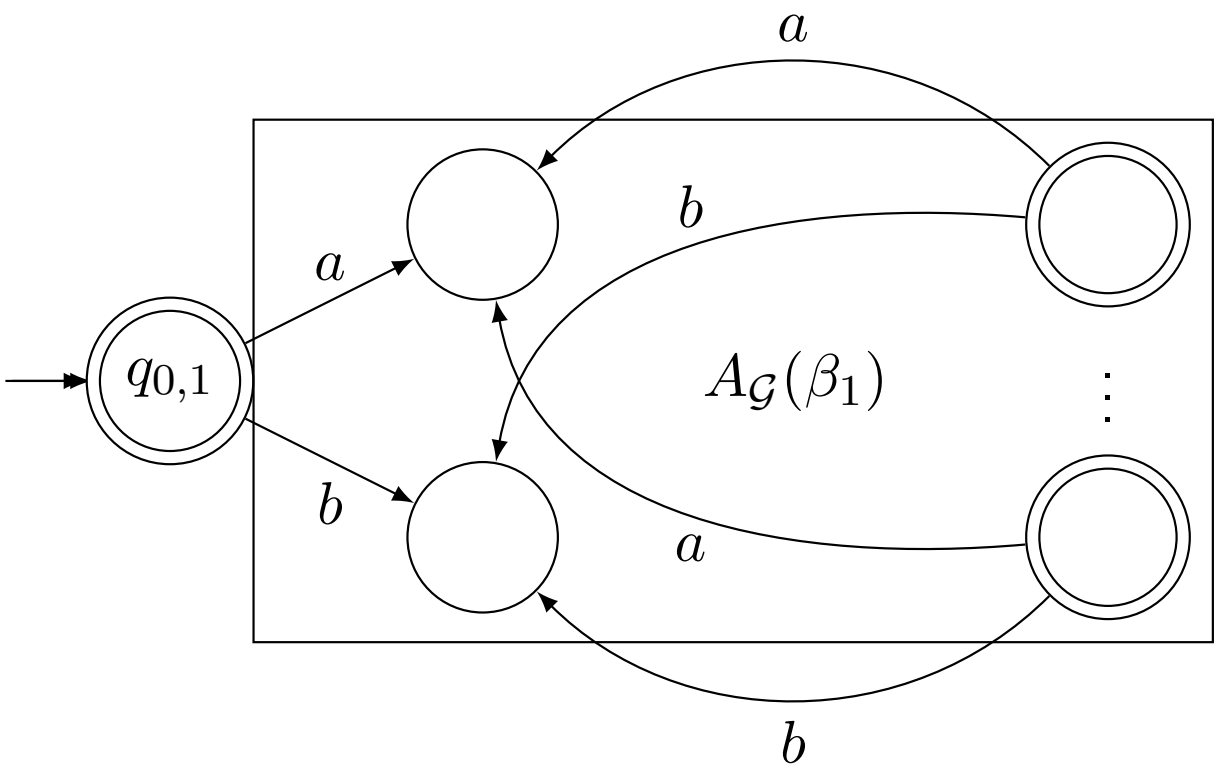


- $q_{0,1}$ und $q_{0,2}$ werden verschmolzen.

Kleene-Stern



Kleene-Stern



Eigenschaften von Glushkov-Automaten

Lemma

Sei α ein regulärer Ausdruck, sei $A_{\mathcal{G}}(\alpha)$ der Glushkov-Automat von α.

Eigenschaften von Glushkov-Automaten

Lemma

Sei α ein regulärer Ausdruck, sei $A_{\mathcal{G}}(\alpha)$ der Glushkov-Automat von α.

1. Der Startzustand von $A_{\mathcal{G}}(\alpha)$ hat keine eingehenden Kanten.

Eigenschaften von Glushkov-Automaten

Lemma

Sei α ein regulärer Ausdruck, sei $A_{\mathcal{G}}(\alpha)$ der Glushkov-Automat von α.

1. Der Startzustand von $A_{\mathcal{G}}(\alpha)$ hat keine eingehenden Kanten.
2. Wenn α (für $n \in \mathbb{N}$) n Vorkommen von Terminalsymbolen enthält, dann hat $A_{\mathcal{G}}(\alpha)$ insgesamt $n + 1$ Zustände.

Eigenschaften von Glushkov-Automaten

Lemma

Sei α ein regulärer Ausdruck, sei $A_{\mathcal{G}}(\alpha)$ der Glushkov-Automat von α.

1. Der Startzustand von $A_{\mathcal{G}}(\alpha)$ hat keine eingehenden Kanten.
2. Wenn α (für $n \in \mathbb{N}$) n Vorkommen von Terminalsymbolen enthält, dann hat $A_{\mathcal{G}}(\alpha)$ insgesamt $n + 1$ Zustände.
3. Für jeden Zustand in $A_{\mathcal{G}}(\alpha)$ sind alle eingehenden Kanten mit dem gleichen Terminal beschriftet.

Eigenschaften von Glushkov-Automaten

Lemma

Sei α ein regulärer Ausdruck, sei $A_{\mathcal{G}}(\alpha)$ der Glushkov-Automat von α.

1. Der Startzustand von $A_{\mathcal{G}}(\alpha)$ hat keine eingehenden Kanten.
2. Wenn α (für $n \in \mathbb{N}$) n Vorkommen von Terminalsymbolen enthält, dann hat $A_{\mathcal{G}}(\alpha)$ insgesamt $n + 1$ Zustände.
3. Für jeden Zustand in $A_{\mathcal{G}}(\alpha)$ sind alle eingehenden Kanten mit dem gleichen Terminal beschriftet.
4. Kein Zustand von $A_{\mathcal{G}}(\alpha)$ ist eine Sackgasse.

Eigenschaften von Glushkov-Automaten

Lemma

Sei α ein regulärer Ausdruck, sei $A_{\mathcal{G}}(\alpha)$ der Glushkov-Automat von α.

1. Der Startzustand von $A_{\mathcal{G}}(\alpha)$ hat keine eingehenden Kanten.
2. Wenn α (für $n \in \mathbb{N}$) n Vorkommen von Terminalsymbolen enthält, dann hat $A_{\mathcal{G}}(\alpha)$ insgesamt $n + 1$ Zustände.
3. Für jeden Zustand in $A_{\mathcal{G}}(\alpha)$ sind alle eingehenden Kanten mit dem gleichen Terminal beschriftet.
4. Kein Zustand von $A_{\mathcal{G}}(\alpha)$ ist eine Sackgasse.

Lemma

Ein regulärer Ausdruck α ist genau dann ein deterministischer regulärer Ausdruck, wenn $A_{\mathcal{G}}(\alpha)$ als DFA interpretiert werden kann.