Theoretische Informatik I

Einheit 2.3

Reguläre Ausdrücke

1. Anwendungen
2. Syntax und Semantik
3. Vereinfachungsregeln
4. Beziehung zu endlichen Automaten

Eine algebraische Beschreibung für Sprachen

- **Automaten beschreiben Abarbeitung von Sprachen**
  - **Operationale Semantik**: Symbole führen zu Zustandsänderungen
  - Bestimmte Wörter bzw. Symbolketten werden durch Zustände akzeptiert
  - Für Automaten ist Sprache $\hat{=}$ Menge der akzeptierten Wörter

Eine algebraische Beschreibung für Sprachen

- **Automaten beschreiben Abarbeitung von Sprachen**
  – **Operationale Semantik**: Symbole führen zu Zustandsänderungen
  – Bestimmte Wörter bzw. Symbolketten werden durch Zustände akzeptiert
  – Für Automaten ist Sprache $\hat{=}$ Menge der akzeptierten Wörter

- **Wie beschreibt man Eigenschaften von Wörtern?**
  – **Deklarative Semantik**: äußere Form von Zeichenreihen einer Sprache
  z.B. *Wörter haben eine führende Null, dann beliebig viele Einsen*
  – Anwendungen brauchen präzise Beschreibungssprache für Wörter
  · Grundeinheiten von Programmiersprachen, Suchmuster für Browser, ...

Eine algebraische Beschreibung für Sprachen

- **Automaten beschreiben Abarbeitung von Sprachen**
  - **Operationale Semantik**: Symbole führen zu Zustandsänderungen
  - Bestimmte Wörter bzw. Symbolketten werden durch Zustände akzeptiert
  - Für Automaten ist Sprache $\hat{=}$ Menge der akzeptierten Wörter
- **Wie beschreibt man Eigenschaften von Wörtern?**
  - **Deklarative Semantik**: äußere Form von Zeichenreihen einer Sprache

    z.B. *Wörter haben eine führende Null, dann beliebig viele Einsen*
  - Anwendungen brauchen präzise Beschreibungssprache für Wörter
    - Grundeinheiten von Programmiersprachen, Suchmuster für Browser, ...
- **Reguläre Ausdrücke als formale Syntax**
  - Kurze, prägnante Beschreibung des Aufbaus der Wörter einer Sprache

    z.B. `01*`: "Zuerst eine Null, dann beliebig viele Einsen"

Anwendung: Textsuche

- **Suche nach Mustern in Texten**
  - Suche ob/wo/wie oft eine bestimmte Zeichenkette im Text erscheint
  - Textmuster kann Platzhalter enthalten

Anwendung: Textsuche

- **Suche nach Mustern in Texten**
  - Suche ob/wo/wie oft eine bestimmte Zeichenkette im Text erscheint
  - Textmuster kann Platzhalter enthalten
- **Beschreibe Textmuster durch reguläre Ausdrücke**
  - Zahl: Ziffernfolge dann möglicherweise Punkt und nichtleere Ziffernfolge
  - Formaler Ausdruck: $\texttt{(0+1+..+9)}^*\texttt{(}\epsilon\texttt{ + (.(0+1+..+9)(0+1+..+9)}^*\texttt{))}$

Anwendung: Textsuche

- **Suche nach Mustern in Texten**
  - Suche ob/wo/wie oft eine bestimmte Zeichenkette im Text erscheint
  - Textmuster kann Platzhalter enthalten
- **Beschreibe Textmuster durch reguläre Ausdrücke**
  - Zahl: Ziffernfolge dann möglicherweise Punkt und nichtleere Ziffernfolge
  - Formaler Ausdruck: $(0{+}1{+}..{+}9)^*(\epsilon + (.(0{+}1{+}..{+}9)(0{+}1{+}..{+}9)^*))$
- **Vielfältige Anwendungen**
  - Google Suche nach einfachen Texten
  - Erweiterte Google Suche nach Textmustern
  - Unix Kommando grep: suche nach Textmustern in Dateien
  - Programmiersprachen wie PERL und sf AWK
  - Textsuche und Textersetzung in Emacs
  - Lexikalische Analyse in Compilern

Reguläre Ausdrücke als Suchmuster für grep

- A regular expression is a pattern that describes a set of strings. Regular expressions are constructed by using various operators to combine smaller expressions.
- Fundamental building blocks are expressions that match a single character.
- A bracket expression is a list of characters enclosed by `[` and `]`. It matches any single character in that list. For example, `[0123456789]` matches any single digit.
- Within a bracket expression, a range expression consists of two characters separated by a hyphen. It matches any single character that sorts between the two characters. For example, in the default C locale, `[a-d]` is equivalent to `[abcd]`.
- Certain named classes of characters are predefined within bracket expressions. They are `[:alnum:]`, `[:alpha:]`, `[:cntrl:]`, `[:digit:]`, ...
- The period `.` matches any single character.
- The caret `^` and the dollar sign `$` are metacharacters that match the empty string ...
- A regular expression may be followed by one of several repetition operators:
  `?`: The preceding item is optional and matched at most once.
  `*`: The preceding item will be matched zero or more times.
  `+` The preceding item will be matched one or more times.
- Two regular expressions may be concatenated; the resulting regular expression matches any string concatenating two substrings that match the subexpressions.
- Two regular expressions may be joined by the infix operator `|`
  The resulting regular expression matches any string matching either subexpression.

Anwendung: Lexikalische Analyse

Wichtigster Grundbestandteil von Compilern

- **Reguläre Ausdrücke beschreiben Token**
  - Logische Grundeinheiten von Programmiersprachen
  - z.B. Schlüsselwörter, Bezeichner, Dezimalzahlen, ...

Anwendung: Lexikalische Analyse

Wichtigster Grundbestandteil von Compilern

- **Reguläre Ausdrücke beschreiben Token**
  - Logische Grundeinheiten von Programmiersprachen
  - z.B. Schlüsselwörter, Bezeichner, Dezimalzahlen, ...
- **"Lexer" transformieren reguläre Ausdrücke in Analyseprogramme**
  - Analyse kann die Token der Programmiersprache identifizieren

Anwendung: Lexikalische Analyse

Wichtigster Grundbestandteil von Compilern

- **Reguläre Ausdrücke beschreiben Token**
  - Logische Grundeinheiten von Programmiersprachen
  - z.B. Schlüsselwörter, Bezeichner, Dezimalzahlen, …
- **“Lexer” transformieren reguläre Ausdrücke in Analyseprogramme**
  - Analyse kann die Token der Programmiersprache identifizieren
  - Zugrundeliegende Technik: Umwandlung regulärer Ausdrücke in DEAs

Reguläre Ausdrücke präzisiert (Syntax)

- **Syntax:** **Terme über $\Sigma \cup \{\emptyset, \epsilon, +, \circ, {}^*, (,)\}$**
  Reguläre Ausdrücke sind induktiv wie folgt definiert

Reguläre Ausdrücke präzisiert (Syntax)

- **Syntax: Terme über $\Sigma \cup \{\emptyset, \epsilon, +, \circ, {}^*, (,)\}$**

  Reguläre Ausdrücke sind induktiv wie folgt definiert

  – $E = \mathtt{a}$ ist ein regulärer Ausdruck für jedes $a \in \Sigma$

  – $E = \emptyset$ und $F = \epsilon$ sind reguläre Ausdrücke

Reguläre Ausdrücke präzisiert (Syntax)

- **Syntax: Terme über $\Sigma \cup \{\emptyset, \epsilon, +, \circ, {}^*, (,)\}$**

  Reguläre Ausdrücke sind induktiv wie folgt definiert

  – $E = \texttt{a}$ ist ein regulärer Ausdruck für jedes $a \in \Sigma$

  – $E = \emptyset$ und $F = \epsilon$ sind reguläre Ausdrücke

  – Sind E und F reguläre Ausdrücke, dann sind auch

  $E \circ F$, E^*, $E + F$ und (E) sind reguläre Ausdrücke

Reguläre Ausdrücke präzisiert (Syntax)

- **Syntax: Terme über $\Sigma \cup \{\emptyset, \epsilon, +, \circ, ^*, (,)\}$**

  Reguläre Ausdrücke sind induktiv wie folgt definiert

  – $E = \mathtt{a}$ ist ein regulärer Ausdruck für jedes $a \in \Sigma$

  – $E = \emptyset$ und $F = \epsilon$ sind reguläre Ausdrücke

  – Sind E und F reguläre Ausdrücke, dann sind auch

  $\boldsymbol{E}\circ\boldsymbol{F}$, $\boldsymbol{E}^*$, $\boldsymbol{E}+\boldsymbol{F}$ und $(\boldsymbol{E})$ sind reguläre Ausdrücke

  Mehr Ausdrücke möglich, aber nicht erforderlich

Reguläre Ausdrücke präzisiert (Syntax)

- **Syntax: Terme über $\Sigma \cup \{\emptyset, \epsilon, +, \circ, {}^*, (,)\}$**

  Reguläre Ausdrücke sind induktiv wie folgt definiert

  – $E = \mathtt{a}$ ist ein regulärer Ausdruck für jedes $a \in \Sigma$

  – $E = \emptyset$ und $F = \epsilon$ sind reguläre Ausdrücke

  – Sind E und F reguläre Ausdrücke, dann sind auch
  $\boldsymbol{E \circ F}$, $\boldsymbol{E^*}$, $\boldsymbol{E + F}$ und $\boldsymbol{(E)}$ sind reguläre Ausdrücke

  Mehr Ausdrücke möglich, aber nicht erforderlich

- **Konventionen zur Vereinfachung**

  – $E \circ F$ wird üblicherweise als $\boldsymbol{EF}$ abgekürzt

Reguläre Ausdrücke präzisiert (Syntax)

- **Syntax:** **Terme über** $\Sigma \cup \{\emptyset, \epsilon, +, \circ, ^*, (,)\}$

  Reguläre Ausdrücke sind induktiv wie folgt definiert

  – $E = \mathtt{a}$ ist ein regulärer Ausdruck für jedes $a \in \Sigma$

  – $E = \emptyset$ und $F = \epsilon$ sind reguläre Ausdrücke

  – Sind E und F reguläre Ausdrücke, dann sind auch

  $\boldsymbol{E \circ F}$, $\boldsymbol{E^*}$, $\boldsymbol{E+F}$ und $\boldsymbol{(E)}$ sind reguläre Ausdrücke

  Mehr Ausdrücke möglich, aber nicht erforderlich

- **Konventionen zur Vereinfachung**

  – $E \circ F$ wird üblicherweise als $\boldsymbol{EF}$ abgekürzt

  – Definitorische Abkürzungen: $\boldsymbol{E^+} \equiv EE^*$, $\boldsymbol{[a_1 \ldots a_n]} \equiv a_1 + \ldots + a_n$

Reguläre Ausdrücke präzisiert (Syntax)

- **Syntax: Terme über $\Sigma \cup \{\emptyset, \epsilon, +, \circ, {}^*, (,)\}$**

  Reguläre Ausdrücke sind induktiv wie folgt definiert

  – $E = \mathtt{a}$ ist ein regulärer Ausdruck für jedes $a \in \Sigma$

  – $E = \emptyset$ und $F = \epsilon$ sind reguläre Ausdrücke

  – Sind E und F reguläre Ausdrücke, dann sind auch

  $\boldsymbol{E \circ F}$, $\boldsymbol{E^*}$, $\boldsymbol{E+F}$ und $\boldsymbol{(E)}$ sind reguläre Ausdrücke

  Mehr Ausdrücke möglich, aber nicht erforderlich

- **Konventionen zur Vereinfachung**

  – $E \circ F$ wird üblicherweise als $\boldsymbol{EF}$ abgekürzt

  – Definitorische Abkürzungen: $\boldsymbol{E^+} \equiv EE^*$, $\boldsymbol{[a_1 ... a_n]} \equiv a_1 + ... + a_n$

  – Prioritätsregelungen ermöglichen, überflüssige Klammern wegzulassen

  · $*$ (“Sternoperator”) bindet stärker als $\circ$, und dies stärker als $+$

  · Verkettung $\circ$ und Alternative $+$ sind assoziativ

Reguläre Ausdrücke präzisiert (Semantik)

- **Reguläre Ausdrücke beschreiben Sprachen über Σ**

Reguläre Ausdrücke präzisiert (Semantik)

- **Reguläre Ausdrücke beschreiben Sprachen über Σ**
- **Die Sprache $L(E)$ ist induktiv definiert**

Reguläre Ausdrücke präzisiert (Semantik)

- **Reguläre Ausdrücke beschreiben Sprachen über Σ**
- **Die Sprache $L(E)$ ist induktiv definiert**
  - Für für alle $a \in \Sigma$ ist $L(\mathtt{a}) = \{a\}$ (einelementige Sprache, die nur a enthält)

    $L(\emptyset)$ ist die leere Sprache (üblicherweise geschrieben als $\emptyset$ oder $\{\}$)

    $L(\epsilon) = \{\epsilon\}$ (einelementige Sprache, die nur das leere Wort enthält)

Reguläre Ausdrücke präzisiert (Semantik)

- **Reguläre Ausdrücke beschreiben Sprachen über Σ**
- **Die Sprache $L(E)$ ist induktiv definiert**
  - Für für alle $a \in \Sigma$ ist $L(\mathtt{a}) = \{a\}$ (einelementige Sprache, die nur a enthält)

    $L(\emptyset)$ ist die leere Sprache (üblicherweise geschrieben als $\emptyset$ oder $\{\}$)

    $L(\epsilon) = \{\epsilon\}$ (einelementige Sprache, die nur das leere Wort enthält)
  - $L(E \circ F) = L(E) \circ L(F) = \{v\,w \mid v \in L(E) \wedge w \in L(F)\}$

    $\circ$ steht für die Verkettung (der Wörter) zweier Sprachen

Reguläre Ausdrücke präzisiert (Semantik)

- **Reguläre Ausdrücke beschreiben Sprachen über Σ**
- **Die Sprache $L(E)$ ist induktiv definiert**
  - Für für alle $a\in\Sigma$ ist $L(\mathtt{a}) = \{a\}$ (einelementige Sprache, die nur a enthält)

    $L(\emptyset)$ ist die leere Sprache (üblicherweise geschrieben als $\emptyset$ oder $\{\}$)

    $L(\epsilon) = \{\epsilon\}$ (einelementige Sprache, die nur das leere Wort enthält)
  - $L(E\circ F) = L(E)\circ L(F) = \{v\,w \mid v\in L(E) \wedge w\in L(F)\}$

    $\circ$ steht für die Verkettung (der Wörter) zweier Sprachen
  - $L(E^*) = (L(E))^* = \{w_1w_2..w_n \mid n\in\mathbb{N} \wedge w_i\in L(E)\}$

    * steht für Verkettung beliebig vieler Wörter einer Sprache (Kleene'sche Hülle)

Reguläre Ausdrücke präzisiert (Semantik)

- **Reguläre Ausdrücke beschreiben Sprachen über Σ**
- **Die Sprache $L(E)$ ist induktiv definiert**
  - Für für alle $a \in \Sigma$ ist $L(\mathtt{a}) = \{a\}$ (einelementige Sprache, die nur a enthält)

    $L(\emptyset)$ ist die leere Sprache (üblicherweise geschrieben als $\emptyset$ oder $\{\}$)

    $L(\epsilon) = \{\epsilon\}$ (einelementige Sprache, die nur das leere Wort enthält)
  - $L(E \circ F) = L(E) \circ L(F) = \{v\,w \mid v \in L(E) \wedge w \in L(F)\}$

    $\circ$ steht für die Verkettung (der Wörter) zweier Sprachen
  - $L(E^*) = (L(E))^* = \{w_1 w_2..w_n \mid n \in \mathbb{N} \wedge w_i \in L(E)\}$

    * steht für Verkettung beliebig vieler Wörter einer Sprache (Kleene'sche Hülle)
  - $L(E+F) = L(E) \cup L(F) = \{w \in \Sigma^* \mid w \in L(E) \vee w \in L(F)\}$

    $+$ steht für die Vereinigung zweier Sprachen

Reguläre Ausdrücke präzisiert (Semantik)

- **Reguläre Ausdrücke beschreiben Sprachen über Σ**
- **Die Sprache $L(E)$ ist induktiv definiert**
  – Für für alle $a \in \Sigma$ ist $L(\mathtt{a}) = \{a\}$ (einelementige Sprache, die nur a enthält)
  $L(\emptyset)$ ist die leere Sprache (üblicherweise geschrieben als $\emptyset$ oder $\{\}$)
  $L(\epsilon) = \{\epsilon\}$ (einelementige Sprache, die nur das leere Wort enthält)
  – $L(E{\circ}F) = L(E){\circ}L(F) \quad = \{v\,w \mid v \in L(E) \wedge w \in L(F)\}$
  $\circ$ steht für die Verkettung (der Wörter) zweier Sprachen
  – $L(E^*) \quad = (L(E))^* \quad = \{w_1 w_2 .. w_n \mid n \in \mathbb{N} \wedge w_i \in L(E)\}$
  * steht für Verkettung beliebig vieler Wörter einer Sprache (Kleene'sche Hülle)
  – $L(E{+}F) = L(E) \cup L(F) \quad = \{w \in \Sigma^* \mid w \in L(E) \ \vee\ w \in L(F)\}$
  $+$ steht für die Vereinigung zweier Sprachen
  – $L(\mathtt{(}E\mathtt{)}) \ = L(E)$

Sprachen vs. Ausdrücke

- **Sprachen sind Mengen von Wörtern**
  – Abstraktes semantisches Konzept: Ungeordnete Kollektion von Wörtern

Sprachen vs. Ausdrücke

- **Sprachen sind Mengen von Wörtern**
  - Abstraktes semantisches Konzept: Ungeordnete Kollektion von Wörtern
  - Beschreibung von Mengen (auf Folie, Tafel, ...) benötigt textuelle Notation
  - Notation benutzt Kurzschreibweisen wie $\cup$, $\circ$, $*$ für Mengenoperationen

Sprachen vs. Ausdrücke

- **Sprachen sind Mengen von Wörtern**
  – Abstraktes semantisches Konzept: Ungeordnete Kollektion von Wörtern
  – Beschreibung von Mengen (auf Folie, Tafel, ...) benötigt textuelle Notation
  – Notation benutzt Kurzschreibweisen wie $\cup$, $\circ$, $*$ für Mengenoperationen
  ... aber ist selbst nur ein Hilfsmittel zur Kommunikation

Sprachen vs. Ausdrücke

- **Sprachen sind Mengen von Wörtern**
  - Abstraktes semantisches Konzept: Ungeordnete Kollektion von Wörtern
  - Beschreibung von Mengen (auf Folie, Tafel, ...) benötigt textuelle Notation
  - Notation benutzt Kurzschreibweisen wie $\cup$, $\circ$, $*$ für Mengenoperationen

    ... aber ist selbst nur ein Hilfsmittel zur Kommunikation
- **Reguläre Ausdrücke sind Terme**
  - Eine syntaktische Beschreibungsform, die ein Computer versteht

Sprachen vs. Ausdrücke

- **Sprachen sind Mengen von Wörtern**
  – Abstraktes semantisches Konzept: Ungeordnete Kollektion von Wörtern
  – Beschreibung von Mengen (auf Folie, Tafel, ...) benötigt textuelle Notation
  – Notation benutzt Kurzschreibweisen wie $\cup$, $\circ$, * für Mengenoperationen
  ... aber ist selbst nur ein Hilfsmittel zur Kommunikation

- **Reguläre Ausdrücke sind Terme**
  – Eine syntaktische Beschreibungsform, die ein Computer versteht
  – Reguläre Ausdrücke werden zur Beschreibung von Sprachen benutzt
  und sind ähnlich zur Standardnotation von Mengen

Sprachen vs. Ausdrücke

- **Sprachen sind Mengen von Wörtern**
  - Abstraktes semantisches Konzept: Ungeordnete Kollektion von Wörtern
  - Beschreibung von Mengen (auf Folie, Tafel, ...) benötigt textuelle Notation
  - Notation benutzt Kurzschreibweisen wie $\cup$, $\circ$, $*$ für Mengenoperationen

    ... aber ist selbst nur ein Hilfsmittel zur Kommunikation
- **Reguläre Ausdrücke sind Terme**
  - Eine syntaktische Beschreibungsform, die ein Computer versteht
  - Reguläre Ausdrücke werden zur Beschreibung von Sprachen benutzt

    und sind ähnlich zur Standardnotation von Mengen
- **Reguläre Ausdrücke sind selbst keine Sprachen**
  - Unterscheide Ausdruck E von Sprache des Ausdrucks $L(E)$

Sprachen vs. Ausdrücke

- **Sprachen sind Mengen von Wörtern**
  - Abstraktes semantisches Konzept: Ungeordnete Kollektion von Wörtern
  - Beschreibung von Mengen (auf Folie, Tafel, ...) benötigt textuelle Notation
  - Notation benutzt Kurzschreibweisen wie $\cup$, $\circ$, $*$ für Mengenoperationen

    ... aber ist selbst nur ein Hilfsmittel zur Kommunikation
- **Reguläre Ausdrücke sind Terme**
  - Eine syntaktische Beschreibungsform, die ein Computer versteht
  - Reguläre Ausdrücke werden zur Beschreibung von Sprachen benutzt

    und sind ähnlich zur Standardnotation von Mengen
- **Reguläre Ausdrücke sind selbst keine Sprachen**
  - Unterscheide Ausdruck E von Sprache des Ausdrucks $L(E)$
  - Man verzichtet auf den Unterschied wenn der Kontext eindeutig ist

Beispiele regulärer Ausdrücke

- $\mathtt{a}^*\mathtt{ba}^*$

Beispiele regulärer Ausdrücke

- $\mathtt{a}^*\mathtt{ba}^*$
  - steht für die Menge aller Wörter, die genau ein b enthalten

Beispiele regulärer Ausdrücke

- $\mathtt{a}^*\mathtt{ba}^*$
  – steht für die Menge aller Wörter, die genau ein b enthalten
- $\Sigma^*\mathtt{b}\Sigma^*$

Beispiele regulärer Ausdrücke

- $\texttt{a}^*\texttt{ba}^*$
  – steht für die Menge aller Wörter, die genau ein b enthalten
- $\Sigma^*\texttt{b}\Sigma^*$
  – steht für $\{w \in \Sigma^* \mid w \text{ enthält mindestens ein } b\}$

Beispiele regulärer Ausdrücke

- $\mathtt{a}^*\mathtt{ba}^*$
  – steht für die Menge aller Wörter, die genau ein b enthalten
- $\Sigma^*\mathtt{b}\Sigma^*$
  – steht für $\{w \in \Sigma^* \mid w \text{ enthält mindestens ein } b\}$
- $\mathtt{a}^*\mathtt{(b+}\epsilon\mathtt{)a}^*$

Beispiele regulärer Ausdrücke

- $\mathtt{a}^*\mathtt{ba}^*$
  – steht für die Menge aller Wörter, die genau ein b enthalten
- $\Sigma^*\mathtt{b}\Sigma^*$
  – steht für $\{w \in \Sigma^* \mid w \text{ enthält mindestens ein } b\}$
- $\mathtt{a}^*\mathtt{(b+}\epsilon\mathtt{)a}^*$
  – steht für $\{w \in \Sigma^* \mid w \text{ enthält maximal ein } b\}$

Beispiele regulärer Ausdrücke

- $\mathtt{a}^*\mathtt{ba}^*$
  - steht für die Menge aller Wörter, die genau ein b enthalten
- $\Sigma^*\mathtt{b}\Sigma^*$
  - steht für $\{w \in \Sigma^* \mid w \text{ enthält mindestens ein } b\}$
- $\mathtt{a}^*\mathtt{(b+}\epsilon\mathtt{)a}^*$
  - steht für $\{w \in \Sigma^* \mid w \text{ enthält maximal ein } b\}$
- $\mathtt{a}\emptyset$

Beispiele regulärer Ausdrücke

- $\texttt{a}^*\texttt{ba}^*$
  – steht für die Menge aller Wörter, die genau ein b enthalten
- $\Sigma^*\texttt{b}\Sigma^*$
  – steht für $\{w \in \Sigma^* \mid w \text{ enthält mindestens ein } b\}$
- $\texttt{a}^*\texttt{(b+}\epsilon\texttt{)a}^*$
  – steht für $\{w \in \Sigma^* \mid w \text{ enthält maximal ein } b\}$
- $\texttt{a}\emptyset$
  – steht für die leere Sprache, denn die Verkettung einer Sprache mit der leeren Sprache ist immer leer

Beispiele regulärer Ausdrücke

- $\mathtt{a}^*\mathtt{ba}^*$
  – steht für die Menge aller Wörter, die genau ein b enthalten
- $\Sigma^*\mathtt{b}\Sigma^*$
  – steht für $\{w \in \Sigma^* \mid w \text{ enthält mindestens ein } b\}$
- $\mathtt{a}^*\mathtt{(b+}\epsilon\mathtt{)a}^*$
  – steht für $\{w \in \Sigma^* \mid w \text{ enthält maximal ein } b\}$
- $\mathtt{a}\emptyset$
  – steht für die leere Sprache, denn die Verkettung einer Sprache mit der leeren Sprache ist immer leer
- $\emptyset^*$

Beispiele regulärer Ausdrücke

- $\mathtt{a}^*\mathtt{ba}^*$
  - steht für die Menge aller Wörter, die genau ein b enthalten
- $\Sigma^*\mathtt{b}\Sigma^*$
  - steht für $\{w \in \Sigma^* \mid w \text{ enthält mindestens ein } b\}$
- $\mathtt{a}^*\mathtt{(b+}\epsilon\mathtt{)a}^*$
  - steht für $\{w \in \Sigma^* \mid w \text{ enthält maximal ein } b\}$
- $\mathtt{a}\emptyset$
  - steht für die leere Sprache, denn die Verkettung einer Sprache mit der leeren Sprache ist immer leer
- $\emptyset^*$
  - steht für die Menge $\{\epsilon\}$, denn die beliebige Verkettung von Wörtern einer Menge enthält immer das leere Wort

Entwicklung regulärer Ausdrücke

Beschreibe Menge aller Wörter, in denen 0 und 1 abwechseln

1. Regulärer Ausdruck für die Sprache $\{01\}$

– `0` repräsentiert $\{0\}$, `1` repräsentiert $\{1\}$

– Also ist $L(\mathtt{01}) = L(\mathtt{0})\circ L(\mathtt{1}) = \{0\}\circ\{1\} = \{01\}$

Entwicklung regulärer Ausdrücke

Beschreibe Menge aller Wörter, in denen 0 und 1 abwechseln

1. **Regulärer Ausdruck für die Sprache $\{01\}$**
   - `0` repräsentiert $\{0\}$, `1` repräsentiert $\{1\}$
   - Also ist $L(\texttt{01}) \;=\; L(\texttt{0})\circ L(\texttt{1}) \;=\; \{0\}\circ\{1\} \;=\; \{01\}$
2. **Erzeuge $\{01, 0101, 010101, ..\}$ durch Sternbildung**
   - $L(\texttt{(01)}^*) = L(\texttt{01})^* = \{01\}^* = \{\epsilon, 01, 0101, 010101, \ldots\}$

Entwicklung regulärer Ausdrücke

Beschreibe Menge aller Wörter, in denen 0 und 1 abwechseln

1. Regulärer Ausdruck für die Sprache $\{01\}$

– `0` repräsentiert $\{0\}$, `1` repräsentiert $\{1\}$

– Also ist $L(\texttt{01}) = L(\texttt{0}) \circ L(\texttt{1}) = \{0\} \circ \{1\} = \{01\}$

2. Erzeuge $\{01, 0101, 010101, ..\}$ durch Sternbildung

– $L(\texttt{(01)}^*) = L(\texttt{01})^* = \{01\}^* = \{\epsilon, 01, 0101, 010101, \ldots\}$

3. Manche Wörter nicht erfaßt

– Start mit Eins statt Null: $\texttt{(10)}^*$

– Start und Ende mit Null: $\texttt{(01)}^*\texttt{0}$

– Start und Ende mit Eins: $\texttt{(10)}^*\texttt{1}$

Vollständiger Ausdruck: $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1}$

Entwicklung regulärer Ausdrücke

Beschreibe Menge aller Wörter, in denen 0 und 1 abwechseln

1. Regulärer Ausdruck für die Sprache $\{01\}$

– `0` repräsentiert $\{0\}$, `1` repräsentiert $\{1\}$

– Also ist $L(\texttt{01}) = L(\texttt{0}) \circ L(\texttt{1}) = \{0\} \circ \{1\} = \{01\}$

2. Erzeuge $\{01, 0101, 010101, ..\}$ durch Sternbildung

– $L(\texttt{(01)}^*) = L(\texttt{01})^* = \{01\}^* = \{\epsilon, 01, 0101, 010101, \ldots\}$

3. Manche Wörter nicht erfaßt

– Start mit Eins statt Null: $\texttt{(10)}^*$

– Start und Ende mit Null: $\texttt{(01)}^*\texttt{0}$

– Start und Ende mit Eins: $\texttt{(10)}^*\texttt{1}$

Vollständiger Ausdruck: $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1}$

4. Es geht auch kürzer

– Optional 1 am Anfang oder 0 am Ende: $\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}$

Bestimmung der Semantik von $(\epsilon\texttt{+}\texttt{1})(\texttt{01})^*(\epsilon\texttt{+}\texttt{0})$

$$L\big((\epsilon\texttt{+}\texttt{1})(\texttt{01})^*(\epsilon\texttt{+}\texttt{0})\big)$$

Bestimmung der Semantik von $\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}$

$$
\begin{aligned}
& L\big(\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}\big) \\
= & L\big(\texttt{(}\epsilon\texttt{+1)}\big) \circ L\big(\texttt{(01)}^*\big) \circ L\big(\texttt{(}\epsilon\texttt{+0)}\big)
\end{aligned}
$$

Bestimmung der Semantik von $\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}$

$$
\begin{aligned}
&L\big(\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}\big) \\
=\ &L\big(\texttt{(}\epsilon\texttt{+1)}\big) \circ L\big(\texttt{(01)}^*\big) \circ L\big(\texttt{(}\epsilon\texttt{+0)}\big) \\
=\ &L(\epsilon) \cup L(\texttt{1}) \circ L\big(\texttt{(01)}\big)^* \circ L(\epsilon) \cup L(\texttt{0})
\end{aligned}
$$

Bestimmung der Semantik von $(\epsilon+1)(01)^*(\epsilon+0)$

$$
\begin{aligned}
& L\big((\epsilon+1)(01)^*(\epsilon+0)\big)\\
= {} & L\big((\epsilon+1)\big) \circ L\big((01)^*\big) \circ L\big((\epsilon+0)\big)\\
= {} & L(\epsilon)\cup L(1) \circ L\big((01)\big)^* \circ L(\epsilon)\cup L(0)\\
= {} & \{\epsilon\}\cup\{1\} \circ (L(0)\circ L(1))^* \circ \{\epsilon\}\cup\{0\}
\end{aligned}
$$

Bestimmung der Semantik von $\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}$

$$
\begin{aligned}
& L(\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}) \\
= {} & L(\texttt{(}\epsilon\texttt{+1)}) \circ L(\texttt{(01)}^*) \circ L(\texttt{(}\epsilon\texttt{+0)}) \\
= {} & L(\epsilon) \cup L(\texttt{1}) \circ L(\texttt{(01)})^* \circ L(\epsilon) \cup L(\texttt{0}) \\
= {} & \{\epsilon\} \cup \{1\} \circ (L(\texttt{0}) \circ L(\texttt{1}))^* \circ \{\epsilon\} \cup \{0\} \\
= {} & \{\epsilon,1\} \circ \{01\}^* \circ \{\epsilon,0\}
\end{aligned}
$$

Bestimmung der Semantik von (ϵ+1)(01)*(ϵ+0)

$$
\begin{aligned}
& L(\mathtt{(}\epsilon\mathtt{+1)(01)^*(}\epsilon\mathtt{+0)}) \\
=\ & L(\mathtt{(}\epsilon\mathtt{+1)}) \circ L(\mathtt{(01)^*}) \circ L(\mathtt{(}\epsilon\mathtt{+0)}) \\
=\ & L(\epsilon) \cup L(\mathtt{1}) \circ L(\mathtt{(01)})^* \circ L(\epsilon) \cup L(\mathtt{0}) \\
=\ & \{\epsilon\} \cup \{1\} \circ (L(\mathtt{0}) \circ L(\mathtt{1}))^* \circ \{\epsilon\} \cup \{0\} \\
=\ & \{\epsilon,1\} \circ \{01\}^* \circ \{\epsilon,0\} \\
=\ & \{\epsilon,1\} \circ \{w \mid \exists n \in \mathbb{N}.\ w = \underbrace{01...01}_{n-mal}\} \circ \{\epsilon,0\}
\end{aligned}
$$

Bestimmung der Semantik von (ε+1)(01)*(ε+0)

$$
\begin{aligned}
& L(\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}) \\
= {} & L(\texttt{(}\epsilon\texttt{+1)}) \circ L(\texttt{(01)}^*) \circ L(\texttt{(}\epsilon\texttt{+0)}) \\
= {} & L(\epsilon) \cup L(\texttt{1}) \circ L(\texttt{(01)})^* \circ L(\epsilon) \cup L(\texttt{0}) \\
= {} & \{\epsilon\} \cup \{1\} \circ (L(\texttt{0}) \circ L(\texttt{1}))^* \circ \{\epsilon\} \cup \{0\} \\
= {} & \{\epsilon,1\} \circ \{01\}^* \circ \{\epsilon,0\} \\
= {} & \{\epsilon,1\} \circ \{w \mid \exists n \in \mathbb{N}.\ w = \underbrace{01\ldots01}_{n-mal}\} \circ \{\epsilon,0\} \\
= {} & \{w \mid \exists n \in \mathbb{N}.\ w = \underbrace{01\ldots01}_{n-mal} \ \vee w = 1\underbrace{01\ldots01}_{n-mal} \\
& \qquad \vee w = \underbrace{01\ldots01}_{n-mal}0 \vee w = 1\underbrace{01\ldots01}_{n-mal}0\,\}
\end{aligned}
$$

Bestimmung der Semantik von (ϵ+1)(01)*(ϵ+0)

$$
\begin{aligned}
& L(\texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}) \\
= {} & L(\texttt{(}\epsilon\texttt{+1)}) \circ L(\texttt{(01)}^*) \circ L(\texttt{(}\epsilon\texttt{+0)}) \\
= {} & L(\epsilon) \cup L(\texttt{1}) \circ L(\texttt{(01)})^* \circ L(\epsilon) \cup L(\texttt{0}) \\
= {} & \{\epsilon\} \cup \{1\} \circ (L(\texttt{0}) \circ L(\texttt{1}))^* \circ \{\epsilon\} \cup \{0\} \\
= {} & \{\epsilon,1\} \circ \{01\}^* \circ \{\epsilon,0\} \\
= {} & \{\epsilon,1\} \circ \{w \mid \exists n \in \mathbb{N}.\ w = \underbrace{01...01}_{n-mal}\} \circ \{\epsilon,0\} \\
= {} & \{w \mid \exists n \in \mathbb{N}.\ w = \underbrace{01...01}_{n-mal} \ \vee w = 1\underbrace{01...01}_{n-mal} \\
& \qquad \vee w = \underbrace{01...01}_{n-mal}0 \vee w = 1\underbrace{01...01}_{n-mal}0\,\} \\
= {} & \text{Die Menge aller Wörter, in denen 0 und 1 abwechseln}
\end{aligned}
$$

(Mühsamer Beweis durch Induktion)

"Rechenregeln" für reguläre Ausdrücke

Wie zeigt man $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1} \cong (\epsilon\texttt{+1})\texttt{(01)}^*(\epsilon\texttt{+0})$ **?**

- **Definiere Äquivalenz von Ausdrücken**
  - $\boldsymbol{E} \cong \boldsymbol{F}$, falls $L(E) = L(F)$

"Rechenregeln" für reguläre Ausdrücke

Wie zeigt man $\mathtt{(01)^*+(10)^*+(01)^*0+(10)^*1} \cong \mathtt{(\epsilon+1)(01)^*(\epsilon+0)}$ **?**

- **Definiere Äquivalenz von Ausdrücken**
  - $\boldsymbol{E} \cong \boldsymbol{F}$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke

“Rechenregeln” für reguläre Ausdrücke

Wie zeigt man $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1} \cong (\epsilon\texttt{+1})\texttt{(01)}^*(\epsilon\texttt{+0})$ **?**

- **Definiere Äquivalenz von Ausdrücken**
  - $\boldsymbol{E} \cong \boldsymbol{F}$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\emptyset+\boldsymbol{E} \cong \boldsymbol{E} \cong \boldsymbol{E}+\emptyset$:

"Rechenregeln" für reguläre Ausdrücke

Wie zeigt man $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1} \cong (\epsilon\texttt{+1})\texttt{(01)}^*(\epsilon\texttt{+0})$ **?**

- **Definiere Äquivalenz von Ausdrücken**
  - $\boldsymbol{E} \cong \boldsymbol{F}$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\emptyset+\boldsymbol{E} \cong \boldsymbol{E} \cong \boldsymbol{E}+\emptyset$: $\quad L(\emptyset+E) = L(\emptyset) \cup L(E) = \emptyset \cup L(E) = L(E)$

"Rechenregeln" für reguläre Ausdrücke

Wie zeigt man $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1} \cong (\epsilon\texttt{+1})\texttt{(01)}^*(\epsilon\texttt{+0})$?

- **Definiere Äquivalenz von Ausdrücken**
  - $\boldsymbol{E} \cong \boldsymbol{F}$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\boldsymbol{\emptyset + E \cong E \cong E + \emptyset}$: $\quad L(\emptyset+E) = L(\emptyset) \cup L(E) = \emptyset \cup L(E) = L(E)$
  - $\boldsymbol{\epsilon \circ E \cong E \cong E \circ \epsilon}$:

“Rechenregeln” für reguläre Ausdrücke

Wie zeigt man $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1} \cong (\epsilon\texttt{+1})\texttt{(01)}^*(\epsilon\texttt{+0})$ **?**

- **Definiere Äquivalenz von Ausdrücken**
  - $\boldsymbol{E} \cong \boldsymbol{F}$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\boldsymbol{\emptyset + E \cong E \cong E + \emptyset}$: $\quad L(\emptyset+E) = L(\emptyset) \cup L(E) = \emptyset \cup L(E) = L(E)$
  - $\boldsymbol{\epsilon \circ E \cong E \cong E \circ \epsilon}$: $\quad L(\epsilon \circ E) = L(\epsilon) \circ L(E) = \{\epsilon\} \circ L(E) = L(E)$

"Rechenregeln" für reguläre Ausdrücke

Wie zeigt man $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1} \cong (\epsilon\texttt{+1})\texttt{(01)}^*(\epsilon\texttt{+0})$?

- **Definiere Äquivalenz von Ausdrücken**
  - $E \cong F$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\emptyset{+}E \cong E \cong E{+}\emptyset$: $\quad L(\emptyset{+}E) = L(\emptyset) \cup L(E) = \emptyset \cup L(E) = L(E)$
  - $\epsilon \circ E \cong E \cong E \circ \epsilon$: $\quad L(\epsilon \circ E) = L(\epsilon) \circ L(E) = \{\epsilon\} \circ L(E) = L(E)$
  - $\emptyset \circ E \cong \emptyset \cong E \circ \emptyset$:

"Rechenregeln" für reguläre Ausdrücke

Wie zeigt man $\mathtt{(01)^* + (10)^* + (01)^*0 + (10)^*1} \cong \mathtt{(\epsilon+1)(01)^*(\epsilon+0)}$ **?**

- **Definiere Äquivalenz von Ausdrücken**
  - $E \cong F$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\emptyset{+}E \cong E \cong E{+}\emptyset$: $\quad L(\emptyset{+}E) = L(\emptyset){\cup}L(E) = \emptyset{\cup}L(E) = L(E)$
  - $\epsilon{\circ}E \cong E \cong E{\circ}\epsilon$: $\quad L(\epsilon{\circ}E) = L(\epsilon){\circ}L(E) = \{\epsilon\}{\circ}L(E) = L(E)$
  - $\emptyset{\circ}E \cong \emptyset \cong E{\circ}\emptyset$: $\quad L(\emptyset{\circ}E) = L(\emptyset){\circ}L(E) = \emptyset{\circ}L(E) = \emptyset = L(\emptyset)$

"Rechenregeln" für reguläre Ausdrücke

Wie zeigt man $\mathtt{(01)^*+(10)^*+(01)^*0+(10)^*1} \cong \mathtt{(\epsilon+1)(01)^*(\epsilon+0)}$ **?**

- **Definiere Äquivalenz von Ausdrücken**
  - $E \cong F$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\emptyset{+}E \cong E \cong E{+}\emptyset$: $\quad L(\emptyset{+}E) = L(\emptyset){\cup}L(E) = \emptyset{\cup}L(E) = L(E)$
  - $\epsilon{\circ}E \cong E \cong E{\circ}\epsilon$: $\quad L(\epsilon{\circ}E) = L(\epsilon){\circ}L(E) = \{\epsilon\}{\circ}L(E) = L(E)$
  - $\emptyset{\circ}E \cong \emptyset \cong E{\circ}\emptyset$: $\quad L(\emptyset{\circ}E) = L(\emptyset){\circ}L(E) = \emptyset{\circ}L(E) = \emptyset = L(\emptyset)$
- **Kommutativität von +**
  - $E{+}F \cong F{+}E$:

“Rechenregeln” für reguläre Ausdrücke

Wie zeigt man $\texttt{(01)}^*+\texttt{(10)}^*+\texttt{(01)}^*\texttt{0}+\texttt{(10)}^*\texttt{1} \cong \texttt{(}\epsilon\texttt{+1)(01)}^*\texttt{(}\epsilon\texttt{+0)}$?

- **Definiere Äquivalenz von Ausdrücken**
  - $\boldsymbol{E} \cong \boldsymbol{F}$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\boldsymbol{\emptyset+E \cong E \cong E+\emptyset}$: $\quad L(\emptyset+E) = L(\emptyset)\cup L(E) = \emptyset\cup L(E) = L(E)$
  - $\boldsymbol{\epsilon\circ E \cong E \cong E\circ\epsilon}$: $\quad L(\epsilon\circ E) = L(\epsilon)\circ L(E) = \{\epsilon\}\circ L(E) = L(E)$
  - $\boldsymbol{\emptyset\circ E \cong \emptyset \cong E\circ\emptyset}$: $\quad L(\emptyset\circ E) = L(\emptyset)\circ L(E) = \emptyset\circ L(E) = \emptyset = L(\emptyset)$
- **Kommutativität von +**
  - $\boldsymbol{E+F \cong F+E}$: $\quad L(E+F) = L(E)\cup L(F) = L(F)\cup L(E) = L(F+E)$

"Rechenregeln" für reguläre Ausdrücke

Wie zeigt man $\texttt{(01)}^* + \texttt{(10)}^* + \texttt{(01)}^*\texttt{0} + \texttt{(10)}^*\texttt{1} \cong (\epsilon\texttt{+1})\texttt{(01)}^*(\epsilon\texttt{+0})$?

- **Definiere Äquivalenz von Ausdrücken**
  - $\boldsymbol{E} \cong \boldsymbol{F}$, falls $L(E) = L(F)$
- **Beweise algebraische Gesetze regulärer Ausdrücke**
  - Liefert Hilfsmittel zur Vereinfachung regulärer Ausdrücke
- **Einheiten und Annihilatoren**
  - $\boldsymbol{\emptyset + E \cong E \cong E + \emptyset}$: $\quad L(\emptyset+E) = L(\emptyset) \cup L(E) = \emptyset \cup L(E) = L(E)$
  - $\boldsymbol{\epsilon \circ E \cong E \cong E \circ \epsilon}$: $\quad L(\epsilon \circ E) = L(\epsilon) \circ L(E) = \{\epsilon\} \circ L(E) = L(E)$
  - $\boldsymbol{\emptyset \circ E \cong \emptyset \cong E \circ \emptyset}$: $\quad L(\emptyset \circ E) = L(\emptyset) \circ L(E) = \emptyset \circ L(E) = \emptyset = L(\emptyset)$
- **Kommutativität von +**
  - $\boldsymbol{E+F \cong F+E}$: $\quad L(E+F) = L(E) \cup L(F) = L(F) \cup L(E) = L(F+E)$
  - Kommutativität von $\circ$ gilt nicht: $\quad = L(\texttt{01}) = \{01\} \neq \{10\} = L(\texttt{10})$

“Rechenregeln” für reguläre Ausdrücke II

- **Assoziativität von $\circ$ und $+$**
  $(E{\circ}F){\circ}G \cong E{\circ}(F{\circ}G)$:

“Rechenregeln” für reguläre Ausdrücke II

- **Assoziativität von $\circ$ und +**
  $\boldsymbol{(E\circ F)\circ G \cong E\circ(F\circ G)}$:
  - $L((E\circ F)\circ G) = L(E\circ F)\circ L(G) = L(E)\circ L(F)\circ L(G) = L(E)\circ L(F\circ G) = L(E\circ(F\circ G))$

"Rechenregeln" für reguläre Ausdrücke II

- **Assoziativität von ∘ und +**

  $(E{\circ}F){\circ}G \cong E{\circ}(F{\circ}G)$:

  – $L((E{\circ}F){\circ}G) = L(E{\circ}F){\circ}L(G) = L(E){\circ}L(F){\circ}L(G) = L(E){\circ}L(F{\circ}G) = L(E{\circ}(F{\circ}G))$

  $(E{+}F){+}G \cong E{+}(F{+}G)$:

  – $L((E{+}F){+}G) = L(E{+}F){\cup}L(G) = L(E){\cup}L(F){\cup}L(G) = \ldots = L(E{+}(F{+}G))$

"Rechenregeln" für reguläre Ausdrücke II

- **Assoziativität von $\circ$ und +**

  $(E\circ F)\circ G \cong E\circ(F\circ G)$:
  - $L((E\circ F)\circ G) = L(E\circ F)\circ L(G) = L(E)\circ L(F)\circ L(G) = L(E)\circ L(F\circ G) = L(E\circ(F\circ G))$

  $(E+F)+G \cong E+(F+G)$:
  - $L((E+F)+G) = L(E+F)\cup L(G) = L(E)\cup L(F)\cup L(G) = \ldots = L(E+(F+G))$

- **Distributivgesetze**
  - $(E+F)\circ G \cong E\circ G+F\circ G$:

$$\begin{aligned}
&L((E+F)\circ G) = (L(E)\cup L(F))\circ L(G)\\
&= \{w\in\Sigma^* \mid \exists u\in L(E)\cup L(F).\exists v\in L(G).w = uv\}\\
&= \{w\in\Sigma^* \mid \exists u\in L(E).\exists v\in L(G).w = uv \vee \exists u\in L(F).\exists v\in L(G).w = uv\}\\
&= \{w\in\Sigma^* \mid \exists u\in L(E).\exists v\in L(G).w = uv\} \cup \{w\in\Sigma^* \mid \exists u\in L(F).\exists v\in L(G).w = uv\}\\
&= L(E)\circ L(G)\cup L(F)\circ L(G) = L(E\circ G+F\circ G)
\end{aligned}$$

“Rechenregeln” für reguläre Ausdrücke II

- **Assoziativität von ∘ und +**

  $(E\circ F)\circ G \cong E\circ(F\circ G)$:

  – $L((E\circ F)\circ G) = L(E\circ F)\circ L(G) = L(E)\circ L(F)\circ L(G) = L(E)\circ L(F\circ G) = L(E\circ(F\circ G))$

  $(E+F)+G \cong E+(F+G)$:

  – $L((E+F)+G) = L(E+F)\cup L(G) = L(E)\cup L(F)\cup L(G) = \ldots = L(E+(F+G))$

- **Distributivgesetze**

  – $(E+F)\circ G \cong E\circ G+F\circ G$:

  $$\begin{aligned}
  &L((E+F)\circ G) = (L(E)\cup L(F))\circ L(G)\\
  &= \{w\in\Sigma^* \mid \exists u\in L(E)\cup L(F).\exists v\in L(G).w = uv\}\\
  &= \{w\in\Sigma^* \mid \exists u\in L(E).\exists v\in L(G).w = uv \vee \exists u\in L(F).\exists v\in L(G).w = uv\}\\
  &= \{w\in\Sigma^* \mid \exists u\in L(E).\exists v\in L(G).w = uv\} \cup \{w\in\Sigma^* \mid \exists u\in L(F).\exists v\in L(G).w = uv\}\\
  &= L(E)\circ L(G)\cup L(F)\circ L(G) = L(E\circ G+F\circ G)
  \end{aligned}$$

  – $G\circ(E+F) \cong G\circ E + G\circ F$

“Rechenregeln” für reguläre Ausdrücke II

- **Assoziativität von ∘ und +**
  $(E \circ F) \circ G \cong E \circ (F \circ G)$:
  - $L((E \circ F) \circ G) = L(E \circ F) \circ L(G) = L(E) \circ L(F) \circ L(G) = L(E) \circ L(F \circ G) = L(E \circ (F \circ G))$

  $(E+F)+G \cong E+(F+G)$:
  - $L((E+F)+G) = L(E+F) \cup L(G) = L(E) \cup L(F) \cup L(G) = \ldots = L(E+(F+G))$
- **Distributivgesetze**
  - $(E+F) \circ G \cong E \circ G + F \circ G$:

    $$\begin{aligned} L((E+F) \circ G) &= (L(E) \cup L(F)) \circ L(G) \\ &= \{w \in \Sigma^* \mid \exists u \in L(E) \cup L(F). \exists v \in L(G). w = uv\} \\ &= \{w \in \Sigma^* \mid \exists u \in L(E). \exists v \in L(G). w = uv \vee \exists u \in L(F). \exists v \in L(G). w = uv\} \\ &= \{w \in \Sigma^* \mid \exists u \in L(E). \exists v \in L(G). w = uv\} \cup \{w \in \Sigma^* \mid \exists u \in L(F). \exists v \in L(G). w = uv\} \\ &= L(E) \circ L(G) \cup L(F) \circ L(G) = L(E \circ G + F \circ G) \end{aligned}$$
  - $G \circ (E+F) \cong G \circ E + G \circ F$
- **Idempotenz von +:** $E+E \cong E$

"Rechenregeln" für reguläre Ausdrücke II

- **Assoziativität von $\circ$ und +**

  $(E\circ F)\circ G \cong E\circ(F\circ G)$:

  – $L((E\circ F)\circ G) = L(E\circ F)\circ L(G) = L(E)\circ L(F)\circ L(G) = L(E)\circ L(F\circ G) = L(E\circ(F\circ G))$

  $(E+F)+G \cong E+(F+G)$:

  – $L((E+F)+G) = L(E+F)\cup L(G) = L(E)\cup L(F)\cup L(G) = \ldots = L(E+(F+G))$

- **Distributivgesetze**

  – $(E+F)\circ G \cong E\circ G+F\circ G$:

  $$\begin{aligned} & L((E+F)\circ G) = (L(E)\cup L(F))\circ L(G) \\ & = \{w\in\Sigma^* \mid \exists u\in L(E)\cup L(F).\exists v\in L(G).w=uv\} \\ & = \{w\in\Sigma^* \mid \exists u\in L(E).\exists v\in L(G).w=uv \vee \exists u\in L(F).\exists v\in L(G).w=uv\} \\ & = \{w\in\Sigma^* \mid \exists u\in L(E).\exists v\in L(G).w=uv\} \cup \{w\in\Sigma^* \mid \exists u\in L(F).\exists v\in L(G).w=uv\} \\ & = L(E)\circ L(G)\cup L(F)\circ L(G) = L(E\circ G+F\circ G) \end{aligned}$$

  – $G\circ(E+F) \cong G\circ E + G\circ F$

- **Idempotenz von +:** $E+E \cong E$

- **Hüllengesetze:** $\emptyset^* \cong \epsilon,\quad \epsilon^* \cong \epsilon,\quad (E^*)^* \cong E^*$

  $E^+ \cong E\circ E^* \cong E^*\circ E,\quad E^* \cong \epsilon + E^+$

Beweismethodik für weitere Äquivalenzen

- **Beispiel: Nachweis von $(E+F)^* \cong (E^*F^*)^*$**
  – Sei $w \in L((E+F)^*)$

Beweismethodik für weitere Äquivalenzen

- **Beispiel: Nachweis von $(E+F)^* \cong (E^*F^*)^*$**
  – Sei $w \in L((E+F)^*)$
  – Dann $w = w_1..w_k$ mit $w_i \in L(E)$ oder $w_i \in L(F)$ für alle i

Beweismethodik für weitere Äquivalenzen

- **Beispiel: Nachweis von $(E+F)^* \cong (E^*F^*)^*$**
  - Sei $w \in L((E+F)^*)$
  - Dann $w = w_1..w_k$ mit $w_i \in L(E)$ oder $w_i \in L(F)$ für alle i
  - Dann $w = w_1..w_k$ mit $w_i \in L(E^*F^*)$ für alle i (semantisches Argument)

Beweismethodik für weitere Äquivalenzen

- **Beispiel: Nachweis von $(E{+}F)^* \cong (E^*F^*)^*$**
  - Sei $w \in L((E{+}F)^*)$
  - Dann $w = w_1..w_k$ mit $w_i \in L(E)$ oder $w_i \in L(F)$ für alle i
  - Dann $w = w_1..w_k$ mit $w_i \in L(E^*F^*)$ für alle i (semantisches Argument)
  - Also $w \in L((E^*F^*)^*)$

Beweismethodik für weitere Äquivalenzen

- **Beispiel: Nachweis von $(E+F)^* \cong (E^*F^*)^*$**
  - Sei $w \in L((E+F)^*)$
  - Dann $w = w_1..w_k$ mit $w_i \in L(E)$ oder $w_i \in L(F)$ für alle i
  - Dann $w = w_1..w_k$ mit $w_i \in L(E^*F^*)$ für alle i (semantisches Argument)
  - Also $w \in L((E^*F^*)^*)$
- **Beweis verwendet keine Information über E und F**

Beweismethodik für weitere Äquivalenzen

- **Beispiel: Nachweis von $(E+F)^* \cong (E^*F^*)^*$**
  - Sei $w \in L((E+F)^*)$
  - Dann $w = w_1..w_k$ mit $w_i \in L(E)$ oder $w_i \in L(F)$ für alle i
  - Dann $w = w_1..w_k$ mit $w_i \in L(E^*F^*)$ für alle i (semantisches Argument)
  - Also $w \in L((E^*F^*)^*)$
- **Beweis verwendet keine Information über E und F**
  - Man könnte genauso gut $(a+b)^* \cong (a^*b^*)^*$ testen

    $(E+F)^* \cong (E^*F^*)^*$ gilt, weil $(a+b)^* \cong (a^*b^*)^*$ gilt

Beweismethodik für weitere Äquivalenzen

- **Beispiel: Nachweis von $(E+F)^* \cong (E^*F^*)^*$**
  - Sei $w \in L((E+F)^*)$
  - Dann $w = w_1..w_k$ mit $w_i \in L(E)$ oder $w_i \in L(F)$ für alle i
  - Dann $w = w_1..w_k$ mit $w_i \in L(E^*F^*)$ für alle i (semantisches Argument)
  - Also $w \in L((E^*F^*)^*)$
- **Beweis verwendet keine Information über E und F**
  - Man könnte genauso gut $(a+b)^* \cong (a^*b^*)^*$ testen
    $(E+F)^* \cong (E^*F^*)^*$ gilt, weil $(a+b)^* \cong (a^*b^*)^*$ gilt
- **Allgemeines Beweisverfahren**
  - E regulärer Ausdruck mit Metavariablen $E_1,..,E_m$ für Sprachen $L_1,..,L_m$
  - Ersetze im Beweis für $E \cong F$ alle Metavariablen durch Symbole $a \in \Sigma$
  - Teste Äquivalenz der konkreten Ausdrücke mit automatischem Prüfverfahren $\mapsto$ Einheit 2.5

Beweismethodik für weitere Äquivalenzen

- **Beispiel: Nachweis von $(E{+}F)^* \cong (E^*F^*)^*$**
  - Sei $w \in L((E{+}F)^*)$
  - Dann $w = w_1..w_k$ mit $w_i \in L(E)$ oder $w_i \in L(F)$ für alle i
  - Dann $w = w_1..w_k$ mit $w_i \in L(E^*F^*)$ für alle i (semantisches Argument)
  - Also $w \in L((E^*F^*)^*)$
- **Beweis verwendet keine Information über E und F**
  - Man könnte genauso gut $(a{+}b)^* \cong (a^*b^*)^*$ testen
    $(E{+}F)^* \cong (E^*F^*)^*$ gilt, weil $(a{+}b)^* \cong (a^*b^*)^*$ gilt
- **Allgemeines Beweisverfahren**
  - E regulärer Ausdruck mit Metavariablen $E_1,..,E_m$ für Sprachen $L_1,..,L_m$
  - Ersetze im Beweis für $E \cong F$ alle Metavariablen durch Symbole $a \in \Sigma$
  - Teste Äquivalenz der konkreten Ausdrücke mit automatischem Prüfverfahren $\mapsto$ Einheit 2.5

  Korrektheitsbeweis: Induktion über Struktur regulärer Ausdrücke

Umwandlung regulärer Ausdrücke in Automaten

Sprachen regulärer Ausdrücke sind endlich erkennbar

Umwandlung regulärer Ausdrücke in Automaten

Sprachen regulärer Ausdrücke sind endlich erkennbar

Für jeden regulären Ausdruck E gibt es einen ϵ-NEA A mit

– A hat genau einen akzeptierenden Zustand q_f

– Der Startzustand von A ist in keinem $\delta_A(q, a)$ enthalten

– Für alle $a \in \Sigma$ ist $\delta_A(q_f, a) = \emptyset$

– $L(E) = L(A)$

Umwandlung regulärer Ausdrücke in Automaten

Sprachen regulärer Ausdrücke sind endlich erkennbar

Für jeden regulären Ausdruck E gibt es einen ϵ-NEA A mit

– A hat genau einen akzeptierenden Zustand q_f

– Der Startzustand von A ist in keinem $\delta_A(q,a)$ enthalten

– Für alle $a \in \Sigma$ ist $\delta_A(q_f,a) = \emptyset$

– $L(E) = L(A)$

Beweis durch strukturelle Induktion über Aufbau regulärer Ausdrücke

Umwandlung regulärer Ausdrücke in Automaten

Sprachen regulärer Ausdrücke sind endlich erkennbar

Für jeden regulären Ausdruck E gibt es einen ϵ-NEA A mit

– A hat genau einen akzeptierenden Zustand q_f

– Der Startzustand von A ist in keinem $\delta_A(q,a)$ enthalten

– Für alle $a \in \Sigma$ ist $\delta_A(q_f,a) = \emptyset$

– $L(E) = L(A)$

Beweis durch strukturelle Induktion über Aufbau regulärer Ausdrücke

- **Induktionsanfänge**

Umwandlung regulärer Ausdrücke in Automaten

Sprachen regulärer Ausdrücke sind endlich erkennbar

Für jeden regulären Ausdruck E gibt es einen ϵ-NEA A mit

– A hat genau einen akzeptierenden Zustand q_f

– Der Startzustand von A ist in keinem $\delta_A(q, a)$ enthalten

– Für alle $a \in \Sigma$ ist $\delta_A(q_f, a) = \emptyset$

– $L(E) = L(A)$

Beweis durch strukturelle Induktion über Aufbau regulärer Ausdrücke

- **Induktionsanfänge**

  – Für $E = \epsilon$ wähle $A =$

Umwandlung regulärer Ausdrücke in Automaten

Sprachen regulärer Ausdrücke sind endlich erkennbar

Für jeden regulären Ausdruck E gibt es einen ϵ-NEA A mit

– A hat genau einen akzeptierenden Zustand q_f

– Der Startzustand von A ist in keinem $\delta_A(q, a)$ enthalten

– Für alle $a \in \Sigma$ ist $\delta_A(q_f, a) = \emptyset$

– $L(E) = L(A)$

Beweis durch strukturelle Induktion über Aufbau regulärer Ausdrücke

- **Induktionsanfänge**

  – Für $E = \epsilon$ wähle $A =$ 

  – Für $E = \emptyset$ wähle $A =$ 

Umwandlung regulärer Ausdrücke in Automaten

Sprachen regulärer Ausdrücke sind endlich erkennbar

Für jeden regulären Ausdruck E gibt es einen ϵ-NEA A mit

– A hat genau einen akzeptierenden Zustand q_f

– Der Startzustand von A ist in keinem $\delta_A(q,a)$ enthalten

– Für alle $a \in \Sigma$ ist $\delta_A(q_f,a) = \emptyset$

– $L(E) = L(A)$

Beweis durch strukturelle Induktion über Aufbau regulärer Ausdrücke

- **Induktionsanfänge**

  – Für $E = \epsilon$ wähle $A =$

  – Für $E = \emptyset$ wähle $A =$

  – Für $E = \mathtt{a}$ wähle $A =$

Umwandlung regulärer Ausdrücke in Automaten

Sprachen regulärer Ausdrücke sind endlich erkennbar

Für jeden regulären Ausdruck E gibt es einen ϵ-NEA A mit

– A hat genau einen akzeptierenden Zustand q_f

– Der Startzustand von A ist in keinem $\delta_A(q,a)$ enthalten

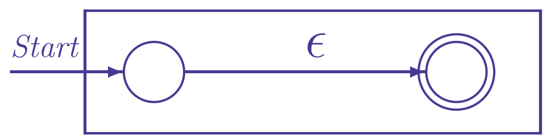
– Für alle $a \in \Sigma$ ist $\delta_A(q_f,a) = \emptyset$

– $L(E) = L(A)$

Beweis durch strukturelle Induktion über Aufbau regulärer Ausdrücke

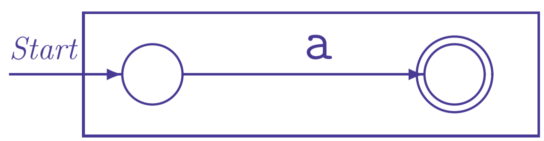
- **Induktionsanfänge**

– Für $E = \epsilon$ wähle $A =$



– Für $E = \emptyset$ wähle $A =$

– Für $E = \mathtt{a}$ wähle $A =$



– **Korrektheit offensichtlich**, da jeweils maximal ein Zustandsübergang

Umwandlung regulärer Ausdrücke in Automaten

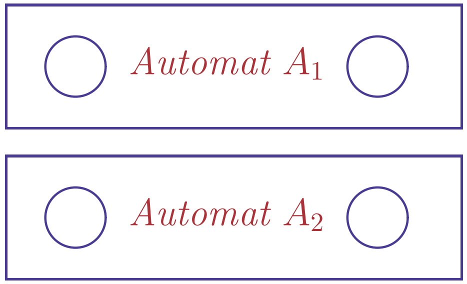
- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**

Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien $\boldsymbol{A_1}$ und $\boldsymbol{A_2}$ $\boldsymbol{\epsilon}$-NEAs für $\boldsymbol{E_1}$ und $\boldsymbol{E_2}$**

- **Induktionsschritt**

  – Für $E = E_1 + E_2$ wähle $A =$

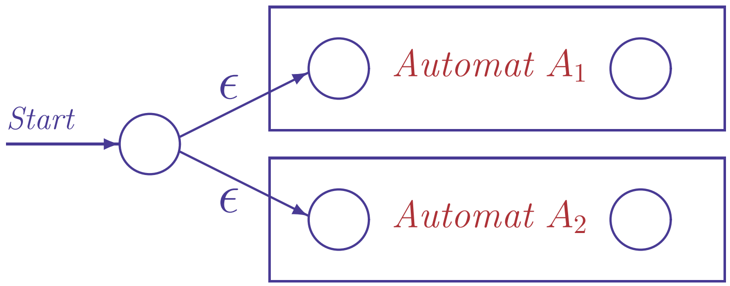


Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**

- **Induktionsschritt**

  – Für $E = E_1$+E_2 wähle $A =$

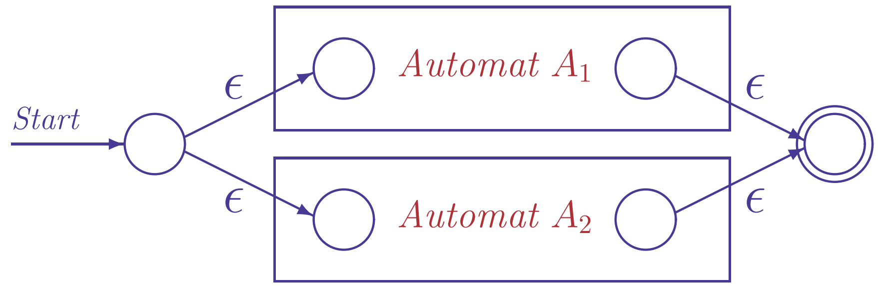


Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**

- **Induktionsschritt**

  – Für $E = E_1 + E_2$ wähle $A =$

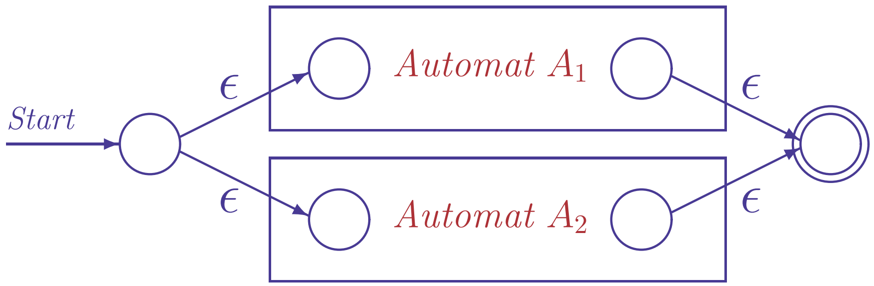


Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**

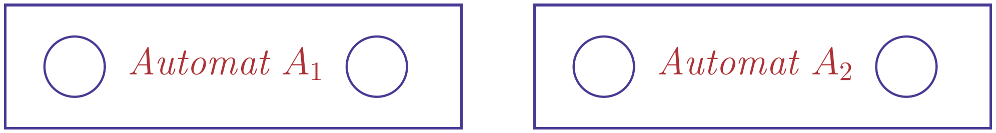
- **Induktionsschritt**

  – Für $E = E_1 + E_2$ wähle $A =$



  – Für $E = E_1 \circ E_2$ wähle

$A =$

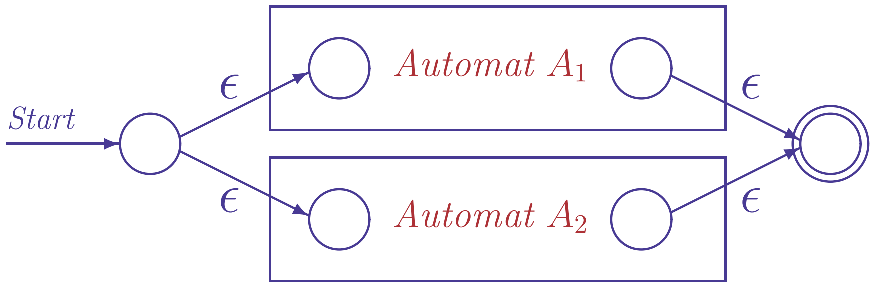


Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien $\boldsymbol{A_1}$ und $\boldsymbol{A_2}$ $\boldsymbol{\epsilon}$-NEAs für $\boldsymbol{E_1}$ und $\boldsymbol{E_2}$**

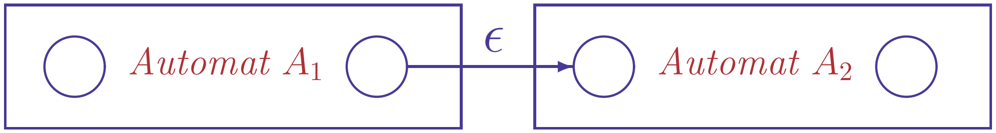
- **Induktionsschritt**

  – Für $E = E_1{+}E_2$ wähle $A =$



  – Für $E = E_1 \circ E_2$ wähle

$A =$

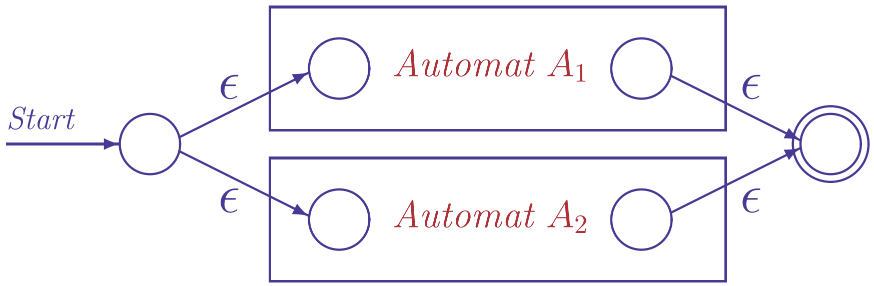


Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**

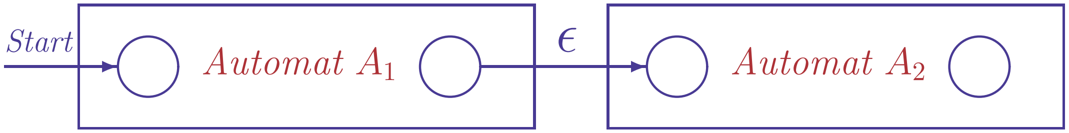
- **Induktionsschritt**

  – Für $E = E_1 + E_2$ wähle $A =$



  – Für $E = E_1 \circ E_2$ wähle

$A =$

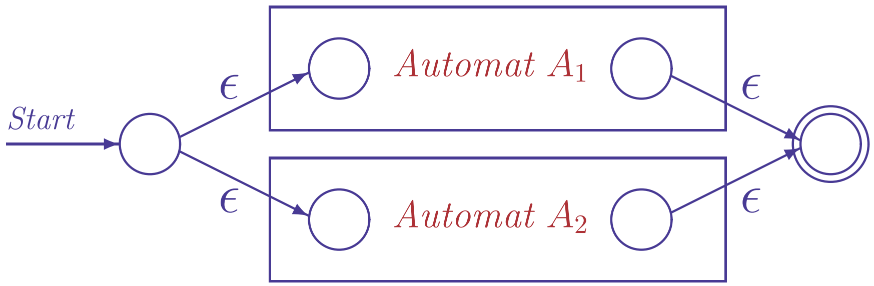


Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**

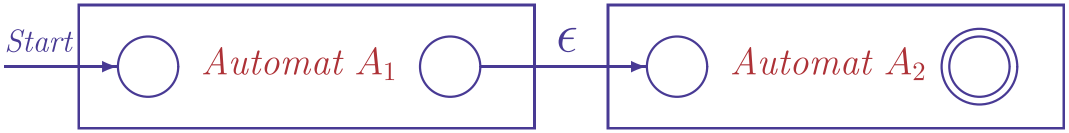
- **Induktionsschritt**

  – Für $E = E_1{+}E_2$ wähle $A =$



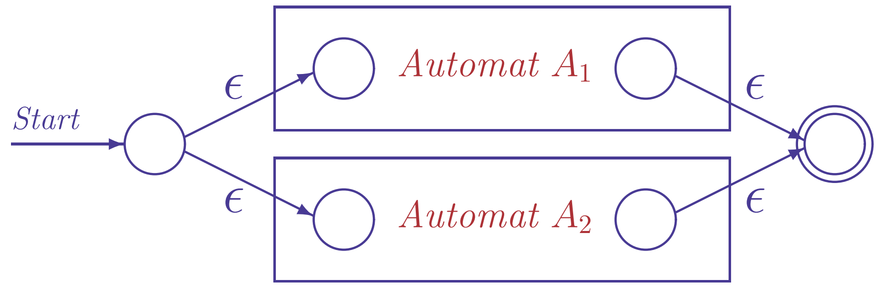
  – Für $E = E_1{\circ}E_2$ wähle

$A =$



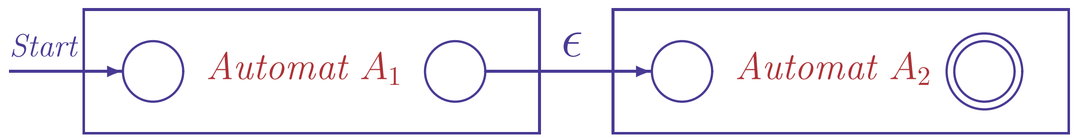
Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**
- **Induktionsschritt**
  - Für $E = E_1 + E_2$ wähle $A =$

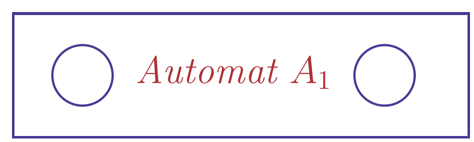


  - Für $E = E_1 \circ E_2$ wähle

$A =$



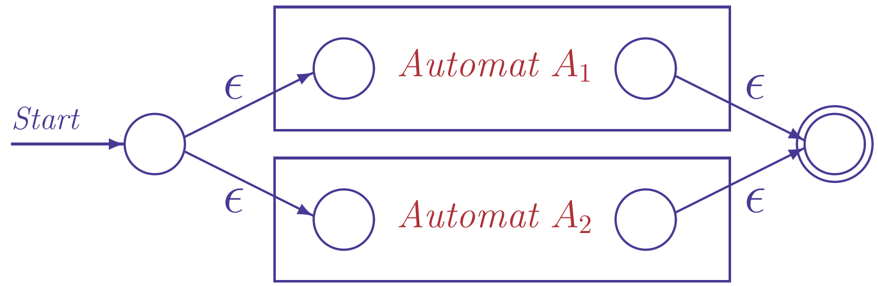
  - Für $E = E_1^*$ wähle $A =$



Umwandlung regulärer Ausdrücke in Automaten

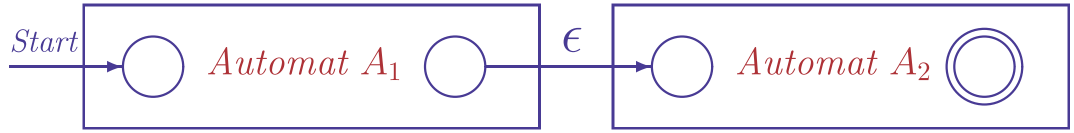
- **Induktionsannahme: seien $\boldsymbol{A_1}$ und $\boldsymbol{A_2}$ $\boldsymbol{\epsilon}$-NEAs für $\boldsymbol{E_1}$ und $\boldsymbol{E_2}$**

- **Induktionsschritt**
  - Für $E = E_1 + E_2$ wähle $A =$

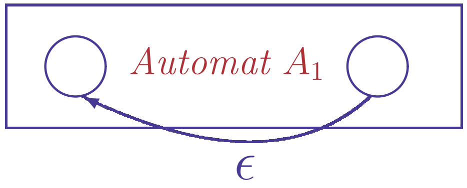


  - Für $E = E_1 \circ E_2$ wähle

$A =$

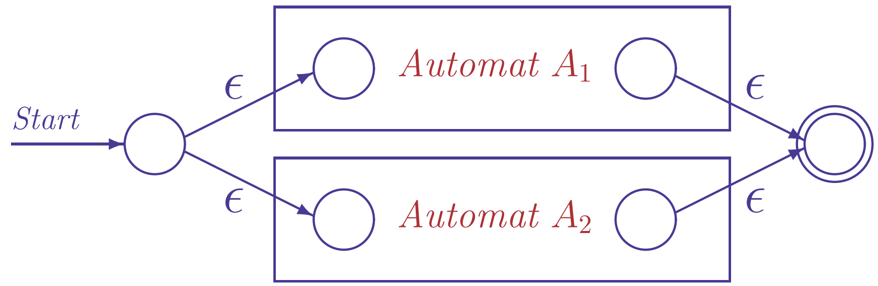


  - Für $E = E_1^*$ wähle $A =$



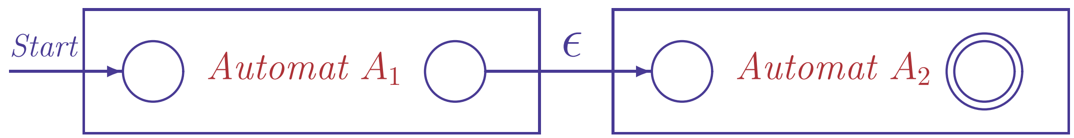
Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**
- **Induktionsschritt**
  - Für $E = E_1 + E_2$ wähle $A =$

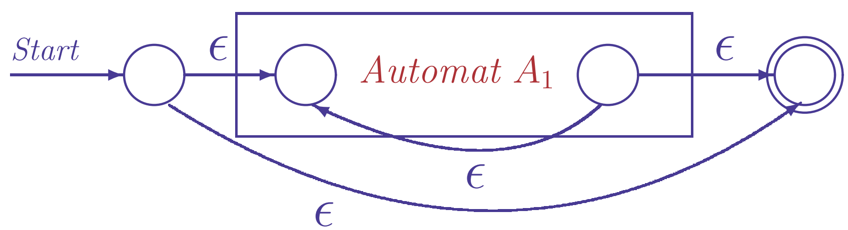


  - Für $E = E_1 \circ E_2$ wähle

$A =$

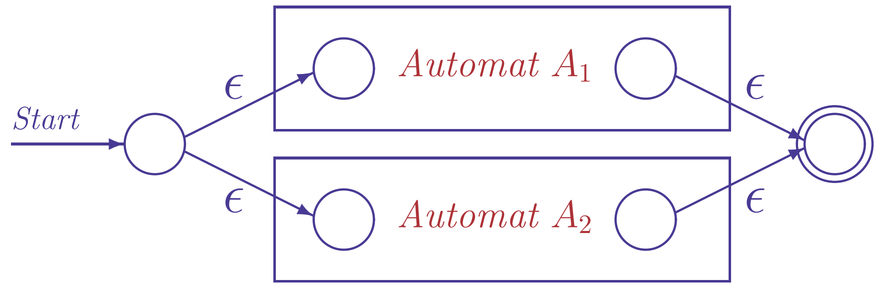


  - Für $E = E_1^*$ wähle $A =$



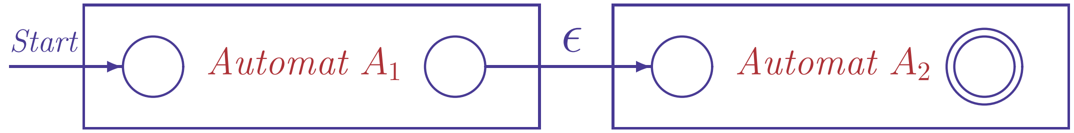
Umwandlung regulärer Ausdrücke in Automaten

- **Induktionsannahme: seien A_1 und A_2 ϵ-NEAs für E_1 und E_2**
- **Induktionsschritt**
  - Für $E = E_1{+}E_2$ wähle $A =$

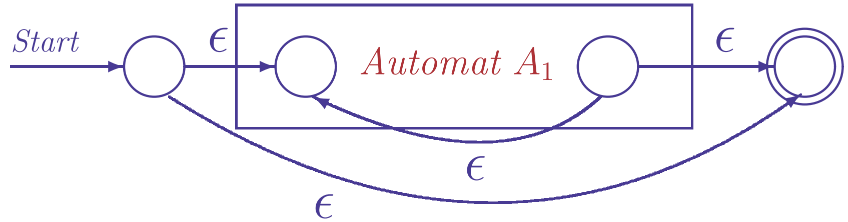


  - Für $E = E_1{\circ}E_2$ wähle

$A =$



  - Für $E = E_1^*$ wähle $A =$



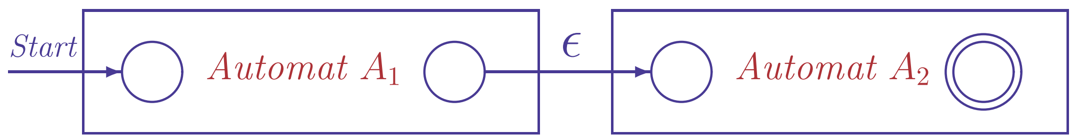
  - Für $E = (E_1)$ wähle $A = A_1$

Korrektheit der Umwandlungen

- **Klammern ändern nichts**
  - Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$

Korrektheit der Umwandlungen

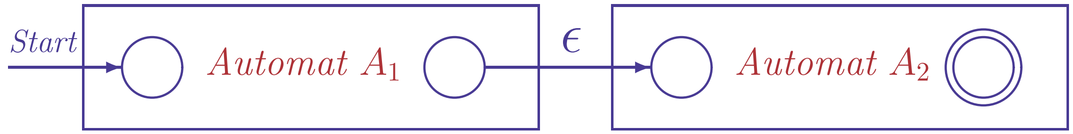
- **Klammern ändern nichts**
  – Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$
- **Verkettung ist Verschaltung von Automaten**



Es gilt $w \in L(E_1 \circ E_2)$

Korrektheit der Umwandlungen

- **Klammern ändern nichts**
  - Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$
- **Verkettung ist Verschaltung von Automaten**



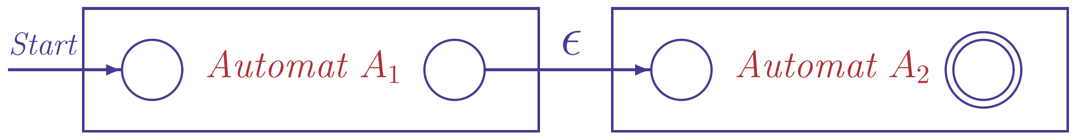
Es gilt $w \in L(E_1 \circ E_2)$

$\Rightarrow\ w \in L(E_1) \circ L(E_2) = L(A_1) \circ L(A_2)$

Korrektheit der Umwandlungen

- **Klammern ändern nichts**
  – Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$

- **Verkettung ist Verschaltung von Automaten**



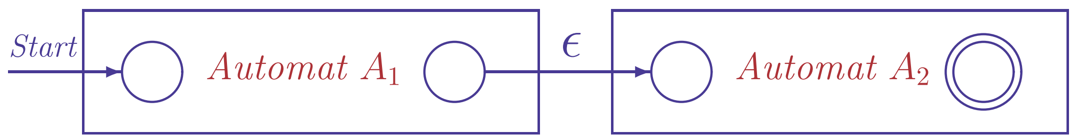
Es gilt $w \in L(E_1 \circ E_2)$

$\Rightarrow\ w \in L(E_1) \circ L(E_2) = L(A_1) \circ L(A_2)$

$\Rightarrow\ \exists u \in L(A_1). \exists v \in L(A_2). w = uv$

Korrektheit der Umwandlungen

- **Klammern ändern nichts**
  - Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$
- **Verkettung ist Verschaltung von Automaten**



Es gilt $w \in L(E_1 \circ E_2)$

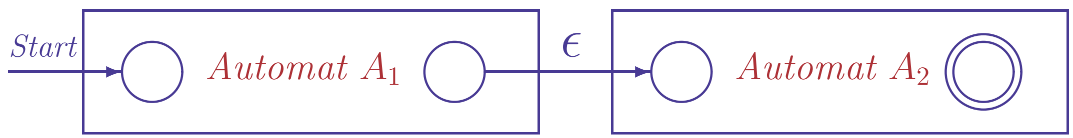
$$\Rightarrow \; w \in L(E_1) \circ L(E_2) = L(A_1) \circ L(A_2)$$

$$\Rightarrow \; \exists u \in L(A_1).\exists v \in L(A_2).w = uv$$

$$\Rightarrow \; \exists u, v \in \Sigma^*.w = uv \wedge q_{f,1} \in \hat{\delta}_1(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}_2(q_{0,2}, v)$$

Korrektheit der Umwandlungen

- **Klammern ändern nichts**
  - Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$
- **Verkettung ist Verschaltung von Automaten**

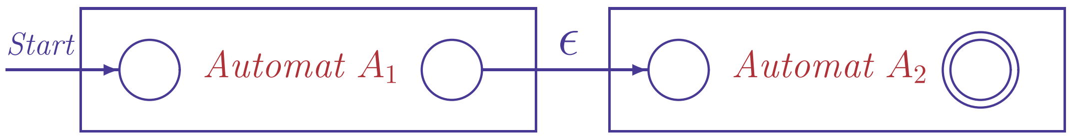


Es gilt $w \in L(E_1 \circ E_2)$

$$\Rightarrow\ w \in L(E_1) \circ L(E_2) = L(A_1) \circ L(A_2)$$
$$\Rightarrow\ \exists u \in L(A_1). \exists v \in L(A_2). w = uv$$
$$\Rightarrow\ \exists u, v \in \Sigma^*. w = uv \wedge q_{f,1} \in \hat{\delta}_1(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}_2(q_{0,2}, v)$$
$$\Rightarrow\ \exists u, v \in \Sigma^*. w = uv \wedge q_{0,2} \in \hat{\delta}(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}(q_{0,2}, v) \qquad (q_{0,2} \in \epsilon\text{-Hülle}(q_{f,1}))$$

- **Klammern ändern nichts**
  – Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$

- **Verkettung ist Verschaltung von Automaten**



Es gilt $w \in L(E_1 \circ E_2)$

$\Rightarrow\ w \in L(E_1) \circ L(E_2) = L(A_1) \circ L(A_2)$

$\Rightarrow\ \exists u \in L(A_1). \exists v \in L(A_2). w = uv$

$\Rightarrow\ \exists u, v \in \Sigma^*. w = uv \wedge q_{f,1} \in \hat{\delta}_1(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}_2(q_{0,2}, v)$

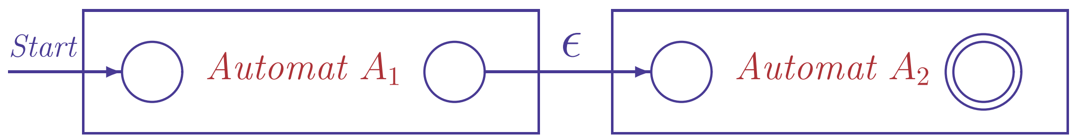
$\Rightarrow\ \exists u, v \in \Sigma^*. w = uv \wedge q_{0,2} \in \hat{\delta}(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}(q_{0,2}, v)$ ($q_{0,2} \in \epsilon$-Hülle($q_{f,1}$))

$\Rightarrow\ q_{f,2} \in \hat{\delta}(q_{0,1}, w)$ (Definition $\hat{\delta}$)

Korrektheit der Umwandlungen

- **Klammern ändern nichts**
  – Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$

- **Verkettung ist Verschaltung von Automaten**



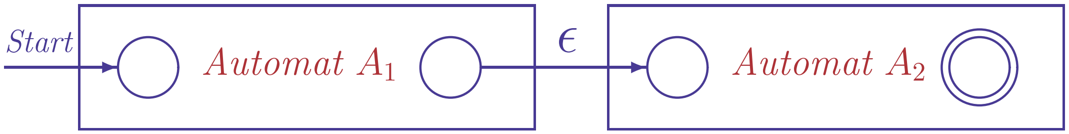
Es gilt $w \in L(E_1 \circ E_2)$

$$
\begin{aligned}
&\Rightarrow\ w \in L(E_1) \circ L(E_2) = L(A_1) \circ L(A_2)\\
&\Rightarrow\ \exists u \in L(A_1).\exists v \in L(A_2).w = uv\\
&\Rightarrow\ \exists u, v \in \Sigma^*.w = uv \wedge q_{f,1} \in \hat{\delta}_1(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}_2(q_{0,2}, v)\\
&\Rightarrow\ \exists u, v \in \Sigma^*.w = uv \wedge q_{0,2} \in \hat{\delta}(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}(q_{0,2}, v) && (q_{0,2} \in \epsilon\text{-Hülle}(q_{f,1}))\\
&\Rightarrow\ q_{f,2} \in \hat{\delta}(q_{0,1}, w) && (\text{Definition } \hat{\delta})\\
&\Rightarrow\ w \in L(A)
\end{aligned}
$$

Korrektheit der Umwandlungen

- **Klammern ändern nichts**
  - Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$

- **Verkettung ist Verschaltung von Automaten**



Es gilt $w \in L(E_1 \circ E_2)$

$\Rightarrow\ w \in L(E_1) \circ L(E_2) = L(A_1) \circ L(A_2)$

$\Rightarrow\ \exists u \in L(A_1). \exists v \in L(A_2). w = uv$

$\Rightarrow\ \exists u, v \in \Sigma^*. w = uv \wedge q_{f,1} \in \hat{\delta}_1(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}_2(q_{0,2}, v)$

$\Rightarrow\ \exists u, v \in \Sigma^*. w = uv \wedge q_{0,2} \in \hat{\delta}(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}(q_{0,2}, v)$ ($q_{0,2} \in \epsilon$-Hülle$(q_{f,1})$)

$\Rightarrow\ q_{f,2} \in \hat{\delta}(q_{0,1}, w)$ (Definition $\hat{\delta}$)

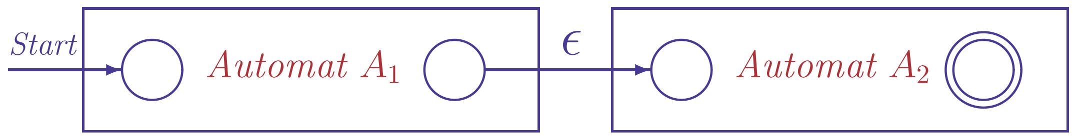
$\Rightarrow\ w \in L(A)$

Argument ist umkehrbar, also $w \in L(A) \ \Rightarrow\ w \in L(E_1 \circ E_2)$

Korrektheit der Umwandlungen

- **Klammern ändern nichts**
  – Es ist $L((E_1)) = L(E_1) = L(A_1) = L(A)$

- **Verkettung ist Verschaltung von Automaten**



Es gilt $w \in L(E_1 \circ E_2)$

$\Rightarrow\ w \in L(E_1) \circ L(E_2) = L(A_1) \circ L(A_2)$

$\Rightarrow\ \exists u \in L(A_1). \exists v \in L(A_2). w = uv$

$\Rightarrow\ \exists u, v \in \Sigma^*. w = uv \wedge q_{f,1} \in \hat{\delta}_1(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}_2(q_{0,2}, v)$

$\Rightarrow\ \exists u, v \in \Sigma^*. w = uv \wedge q_{0,2} \in \hat{\delta}(q_{0,1}, u) \wedge q_{f,2} \in \hat{\delta}(q_{0,2}, v)$ ($q_{0,2} \in \epsilon$-Hülle$(q_{f,1})$)

$\Rightarrow\ q_{f,2} \in \hat{\delta}(q_{0,1}, w)$ (Definition $\hat{\delta}$)

$\Rightarrow\ w \in L(A)$

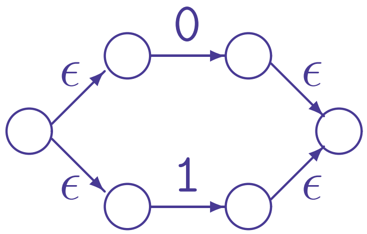
Argument ist umkehrbar, also $w \in L(A) \Rightarrow w \in L(E_1 \circ E_2)$

- **Sternbildung und Vereinigung ähnlich**

Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

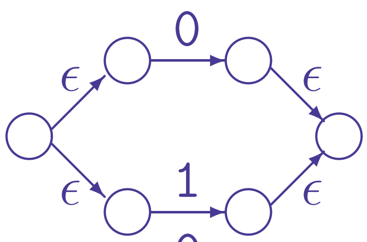
- **Teilautomat für** (0+1)



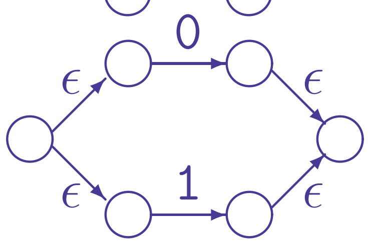
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

- **Teilautomat für** (0+1)



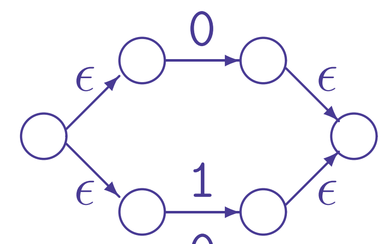
- **Teilautomat für** (0+1)*



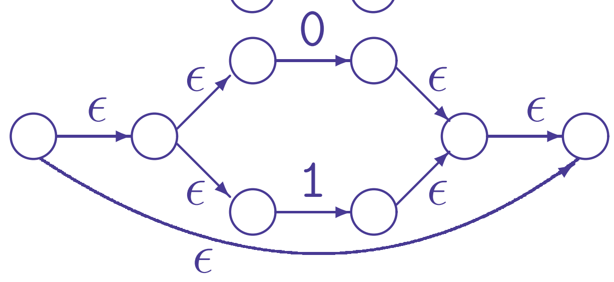
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für `(0+1)*1(0+1)`

- **Teilautomat für** `(0+1)`



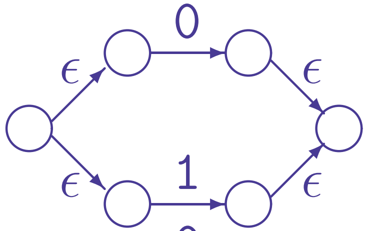
- **Teilautomat für** `(0+1)*`



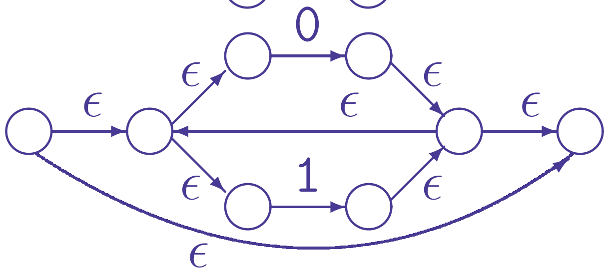
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für `(0+1)`*`1(0+1)`

- **Teilautomat für** `(0+1)`



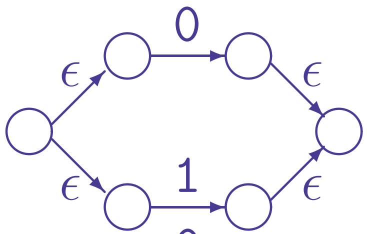
- **Teilautomat für** `(0+1)`*



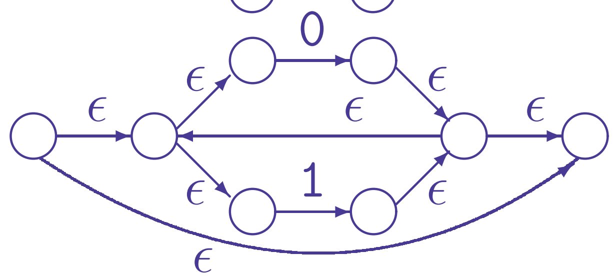
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

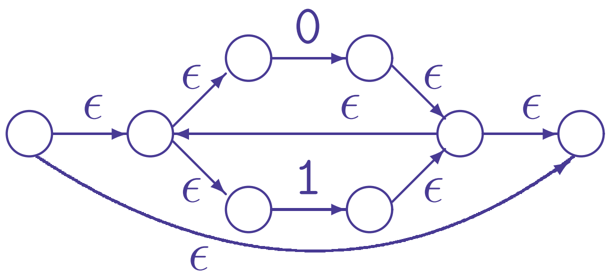
- **Teilautomat für** (0+1)



- **Teilautomat für** (0+1)*



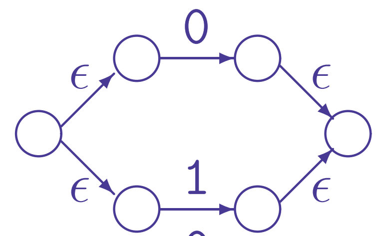
- **Automat für** (0+1)*1(0+1)



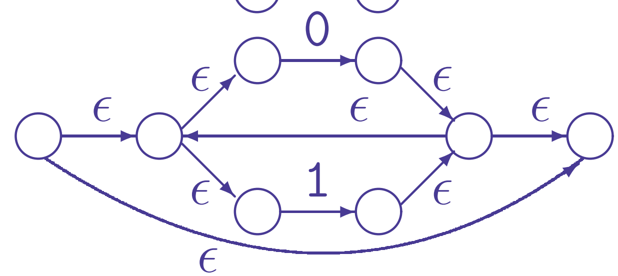
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für `(0+1)*1(0+1)`

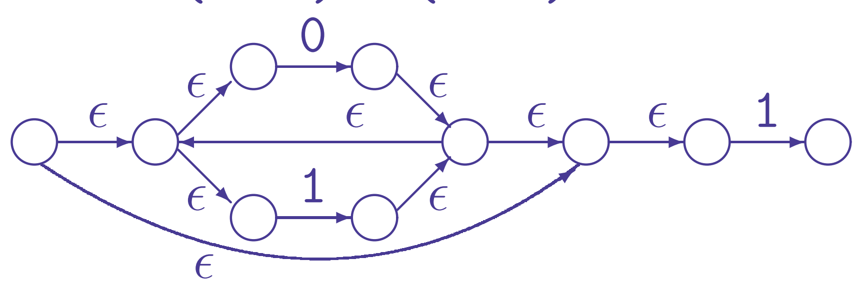
- **Teilautomat für** `(0+1)`



- **Teilautomat für** `(0+1)*`

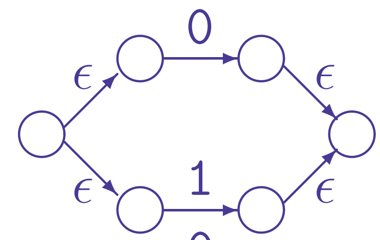


- **Automat für** `(0+1)*1(0+1)`

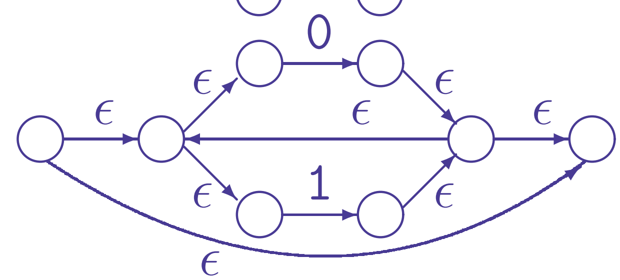


Konstruiere endlichen Automaten für `(0+1)*1(0+1)`

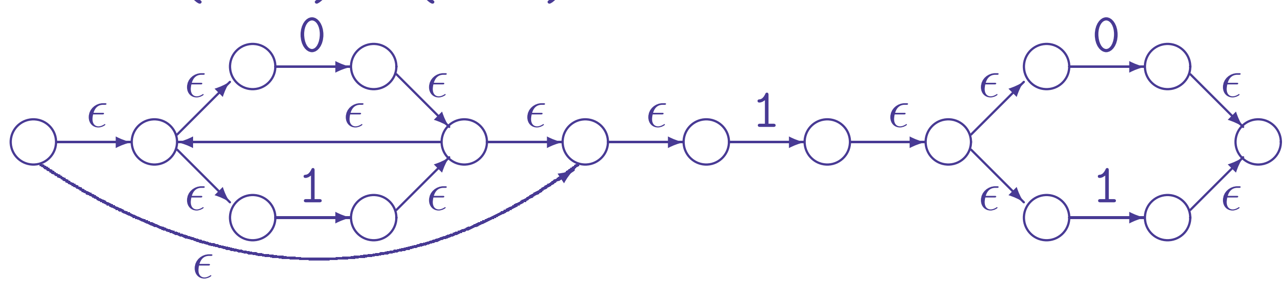
- **Teilautomat für** `(0+1)`



- **Teilautomat für** `(0+1)*`



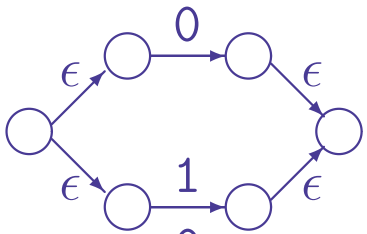
- **Automat für** `(0+1)*1(0+1)`



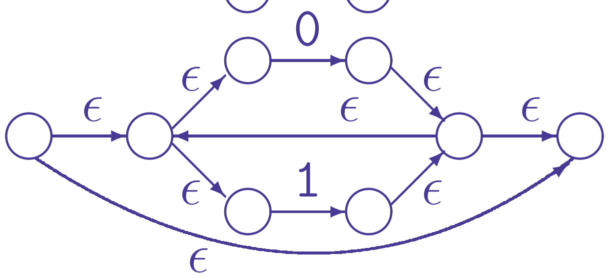
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

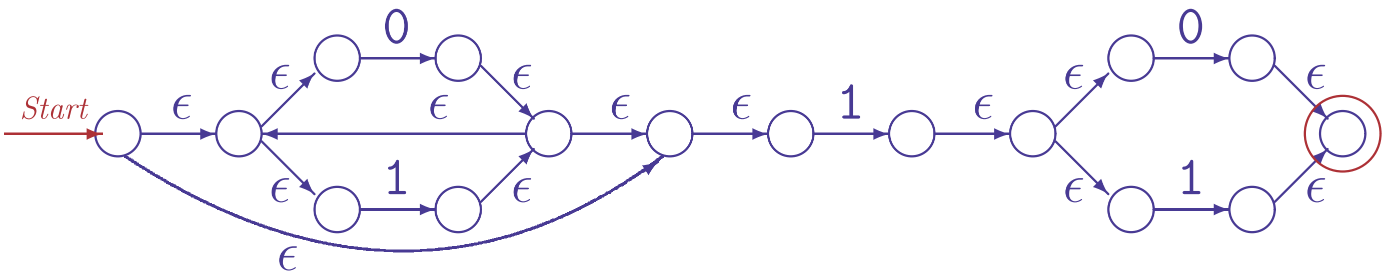
- **Teilautomat für** (0+1)



- **Teilautomat für** (0+1)*



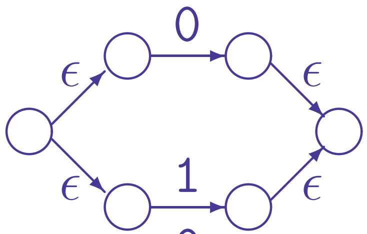
- **Automat für** (0+1)*1(0+1)



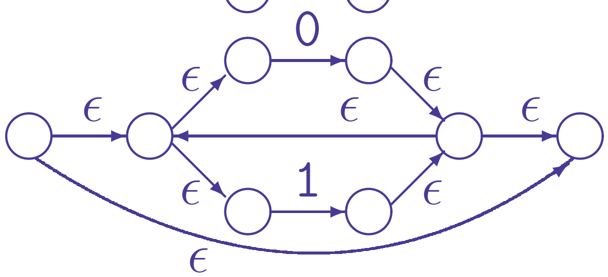
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

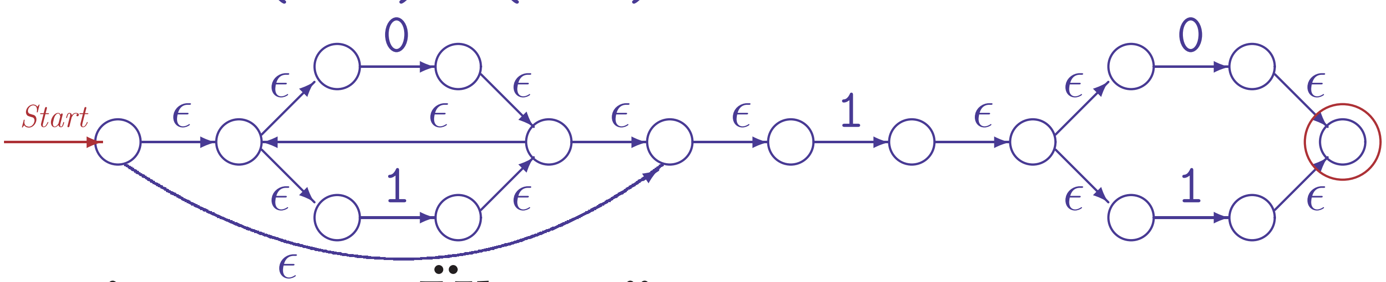
- **Teilautomat für** (0+1)



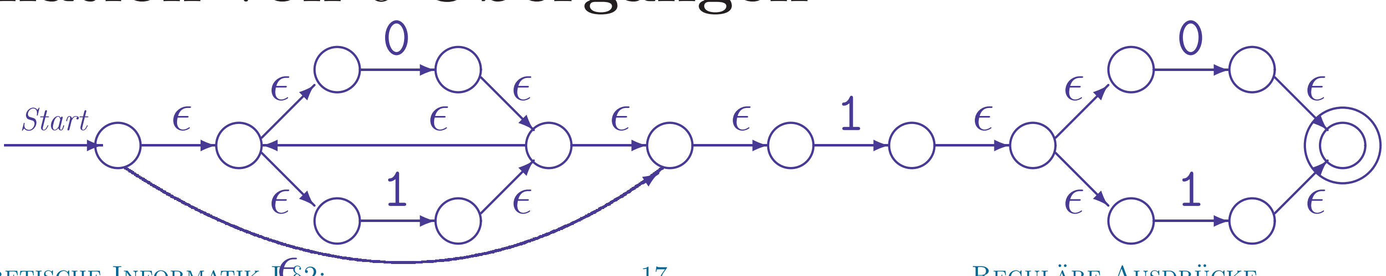
- **Teilautomat für** (0+1)*



- **Automat für** (0+1)*1(0+1)



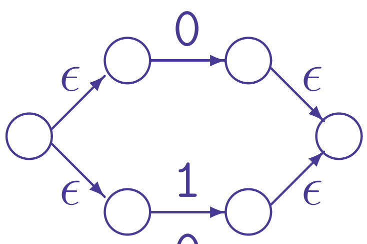
- **Elimination von ϵ-Übergängen**



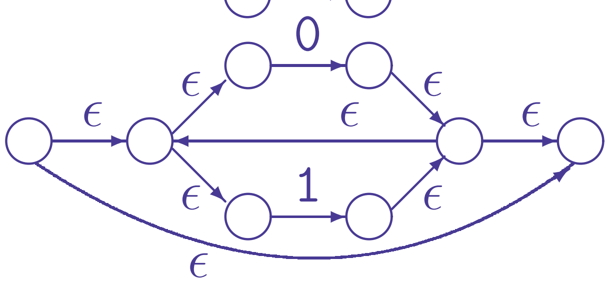
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

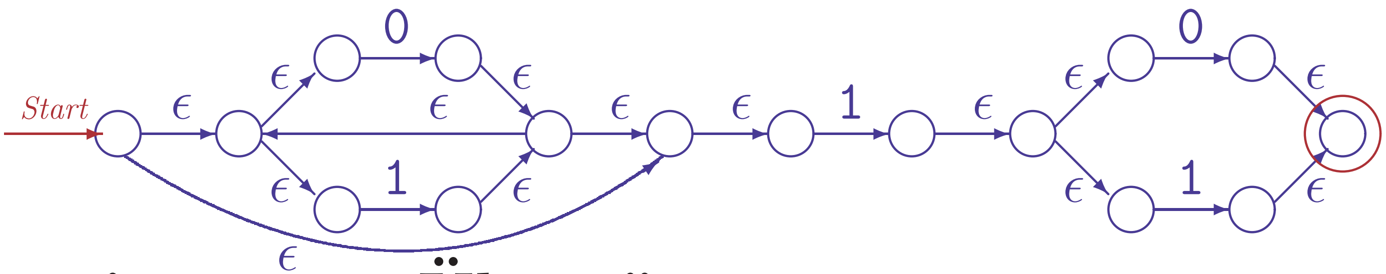
- **Teilautomat für** (0+1)



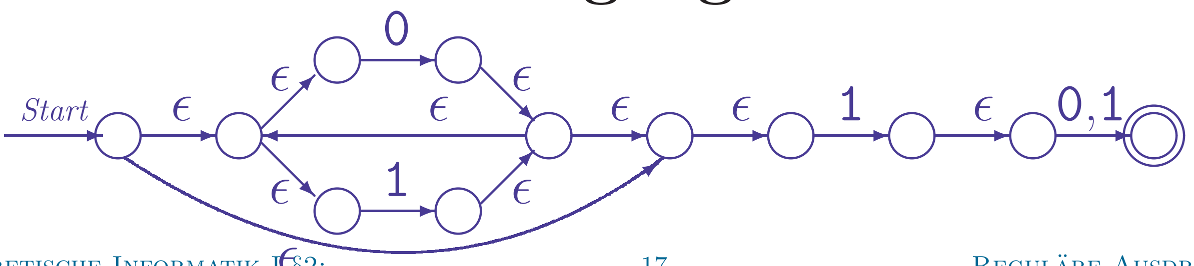
- **Teilautomat für** (0+1)*



- **Automat für** (0+1)*1(0+1)



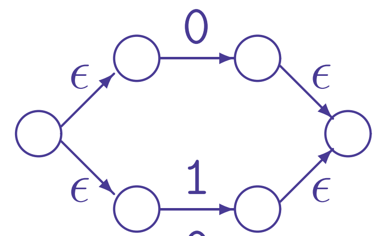
- **Elimination von ϵ-Übergängen**



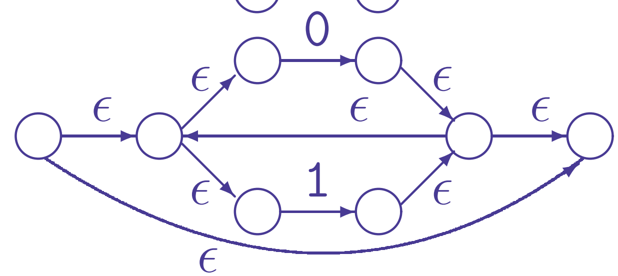
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

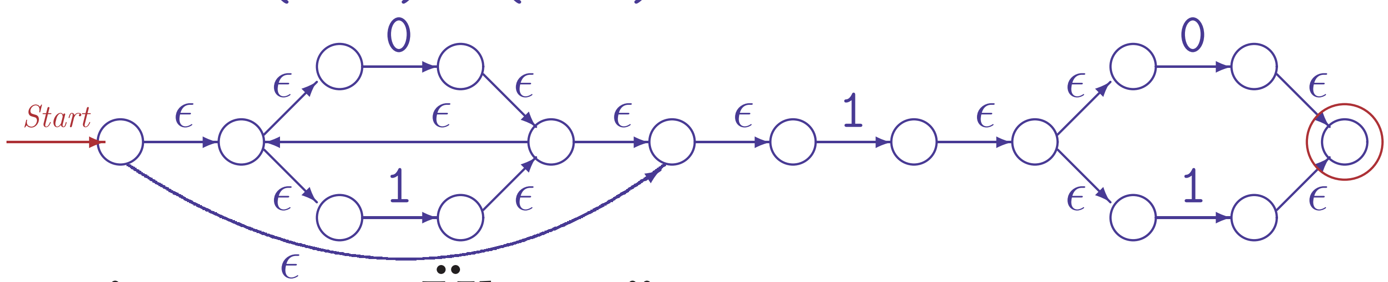
- **Teilautomat für** (0+1)



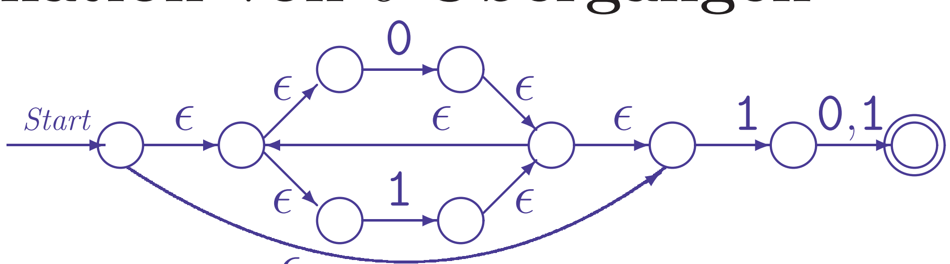
- **Teilautomat für** (0+1)*



- **Automat für** (0+1)*1(0+1)



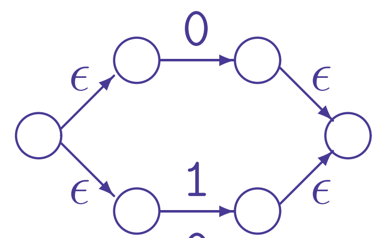
- **Elimination von ϵ-Übergängen**



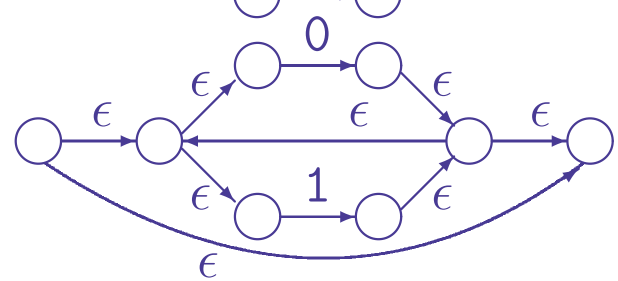
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

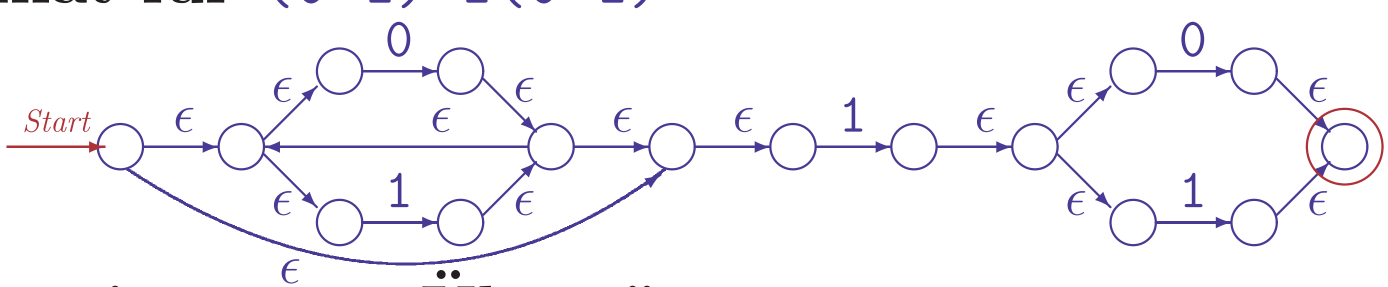
- **Teilautomat für** (0+1)



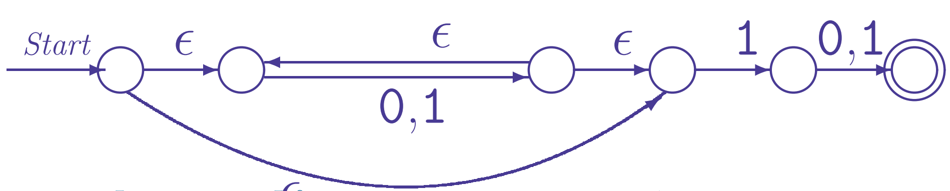
- **Teilautomat für** (0+1)*



- **Automat für** (0+1)*1(0+1)



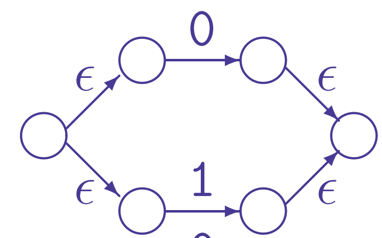
- **Elimination von ϵ-Übergängen**



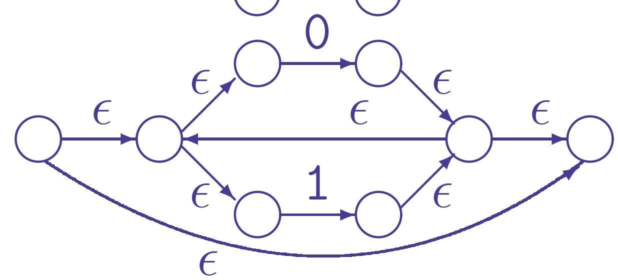
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für `(0+1)*1(0+1)`

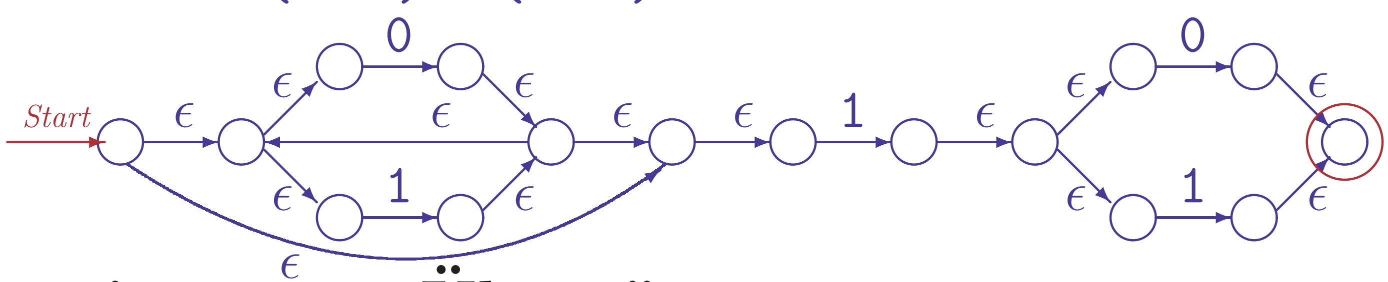
- **Teilautomat für** `(0+1)`



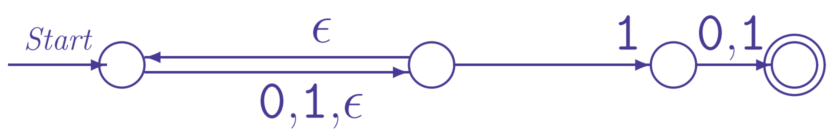
- **Teilautomat für** `(0+1)*`



- **Automat für** `(0+1)*1(0+1)`



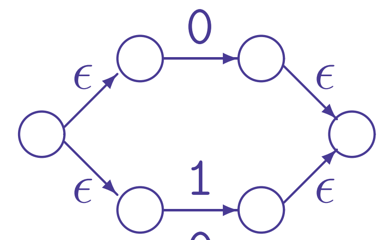
- **Elimination von ϵ-Übergängen**



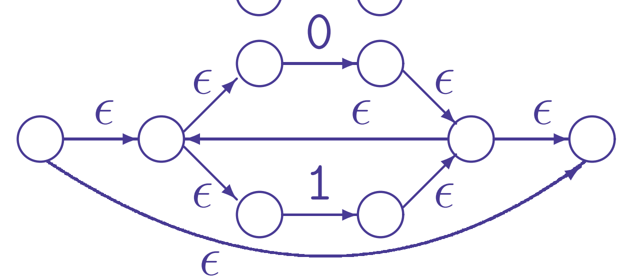
Umwandlung regulärer Ausdrücke am Beispiel

Konstruiere endlichen Automaten für (0+1)*1(0+1)

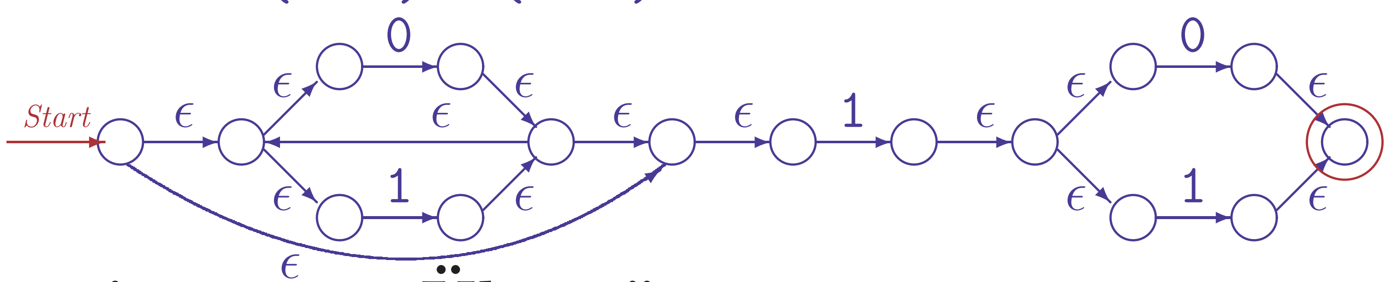
- **Teilautomat für** (0+1)



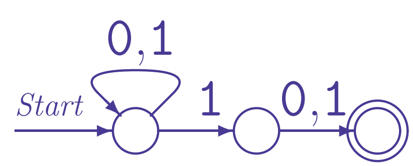
- **Teilautomat für** (0+1)*



- **Automat für** (0+1)*1(0+1)



- **Elimination von ϵ-Übergängen**



Umwandlung von NEAs in reguläre Ausdrücke

- **Ursprünglich: Pfadanalyse im Übergangsdiagramm**
  - Spezialisierung eines allgemeinen Verfahrens für Pfadanalyse in Graphen
  - Definiere reguläre Ausdrücke für Pfade durch Automaten
  - Berechnung Ausdrücke iterativ und kombiniere alle relevanten Ausdrücke
  - Kompliziertes und aufwendiges Verfahren

Mehr dazu im Anhang

Umwandlung von NEAs in reguläre Ausdrücke

- **Ursprünglich: Pfadanalyse im Übergangsdiagramm**
  - Spezialisierung eines allgemeinen Verfahrens für Pfadanalyse in Graphen
  - Definiere reguläre Ausdrücke für Pfade durch Automaten
  - Berechnung Ausdrücke iterativ und kombiniere alle relevanten Ausdrücke
  - Kompliziertes und aufwendiges Verfahren

Mehr dazu im Anhang

- **Effizienterer Zugang: Elimination von Zuständen**
  - Beschreibe Übergänge $q_i \xrightarrow{a\,\in\Sigma} q_j$ durch reguläre Ausdrücke
  - Beginne mit regulären Ausdrücken für direkte Übergänge
  - Entferne einzelne Zustände und beschreibe die entstehenden Ausdrücke
  - Liefert Ausdrücke für Übergänge zwischen Start- und Endzuständen

Umwandlung von NEAs in reguläre Ausdrücke

- **Ursprünglich: Pfadanalyse im Übergangsdiagramm**
  - Spezialisierung eines allgemeinen Verfahrens für Pfadanalyse in Graphen
  - Definiere reguläre Ausdrücke für Pfade durch Automaten
  - Berechnung Ausdrücke iterativ und kombiniere alle relevanten Ausdrücke
  - Kompliziertes und aufwendiges Verfahren

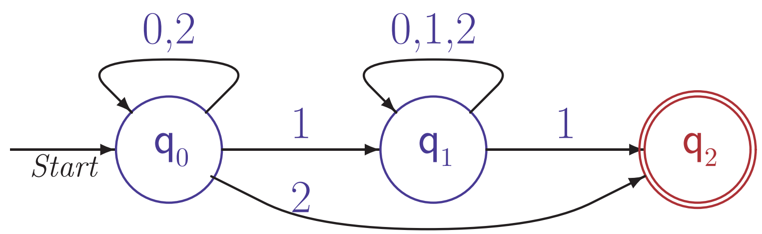
Mehr dazu im Anhang

- **Effizienterer Zugang: Elimination von Zuständen**
  - Beschreibe Übergänge $q_i \xrightarrow{a\,\in\Sigma} q_j$ durch reguläre Ausdrücke
  - Beginne mit regulären Ausdrücken für direkte Übergänge
  - Entferne einzelne Zustände und beschreibe die entstehenden Ausdrücke
  - Liefert Ausdrücke für Übergänge zwischen Start- und Endzuständen

- **Hilfsmittel: verallgemeinerte NEAs (VNEAs)**
  - NEA, dessen Überführungsfunktion δ auf regulären Ausdrücken arbeitet
  - A akzeptiert w, wenn es einen Pfad $w = v_1..v_m$ von q_0 zu einem $q \in F$ gibt und alle v_i in der Sprache des entsprechenden regulären Ausdrucks liegen
  - Konsistente Formalisierung mühsam und ohne Erkenntnisgewinn

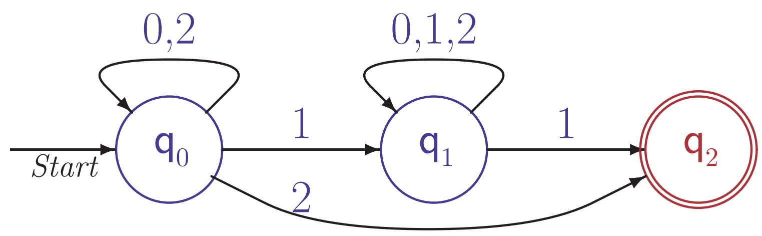
Zustandselimination in VNEAs

- **Ursprünglicher NEA**

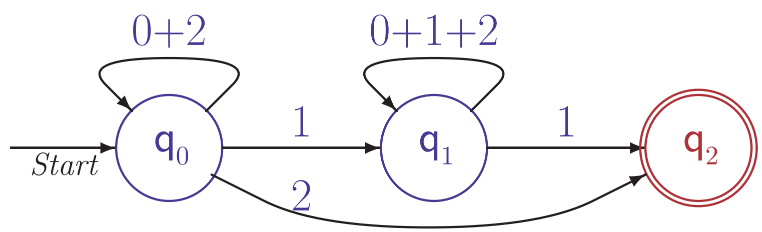


Zustandselimination in VNEAs

- **Ursprünglicher NEA**

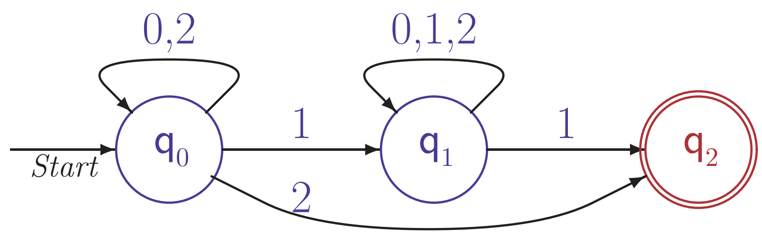


- **Zugehöriger VNEA**

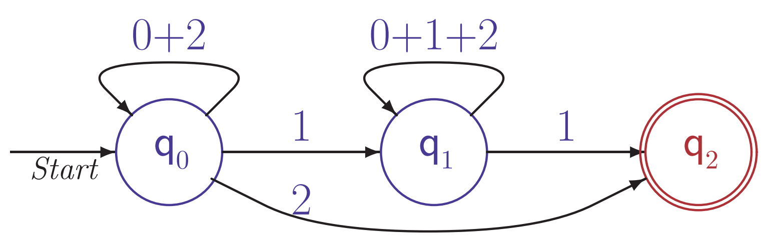


Zustandselimination in VNEAs

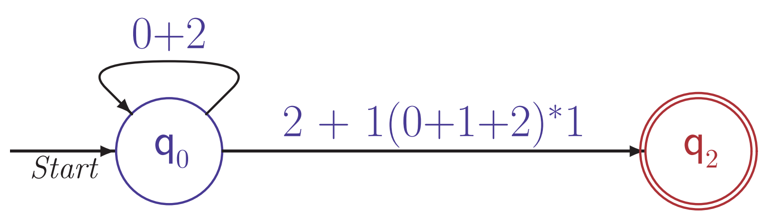
- **Ursprünglicher NEA**



- **Zugehöriger VNEA**

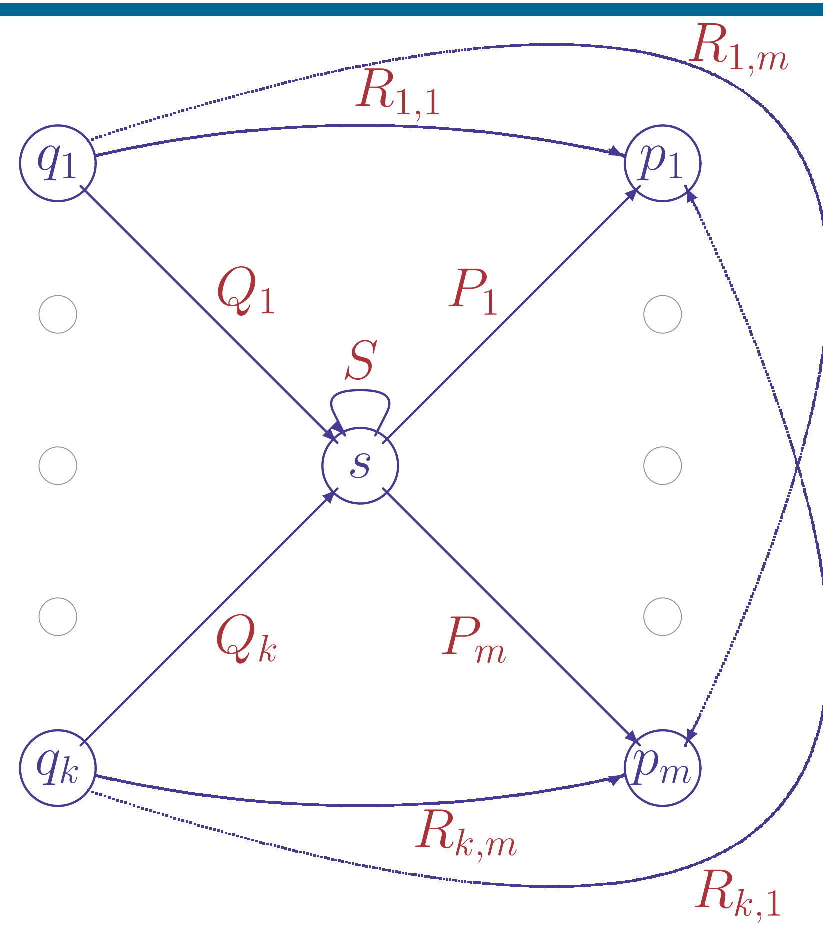


- **Nach Elimination von q_1**



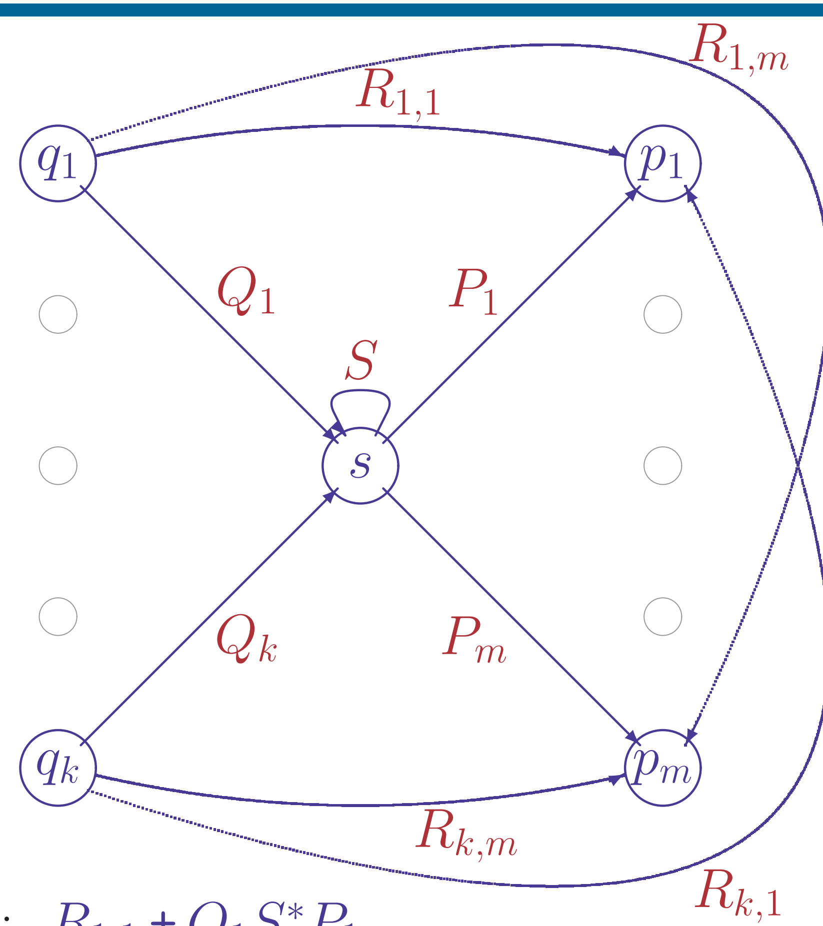
  – Ausdruck für Übergang von q_0 nach q_2 ergibt sich aus Übergang q_0 nach q_1, Schleife bei q_1, Übergang q_1 nach q_2 und existeriendem Ausdruck für direkten Übergang von q_0 nach q_2

Allgemeine Zustandselimination in VNEAs



Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

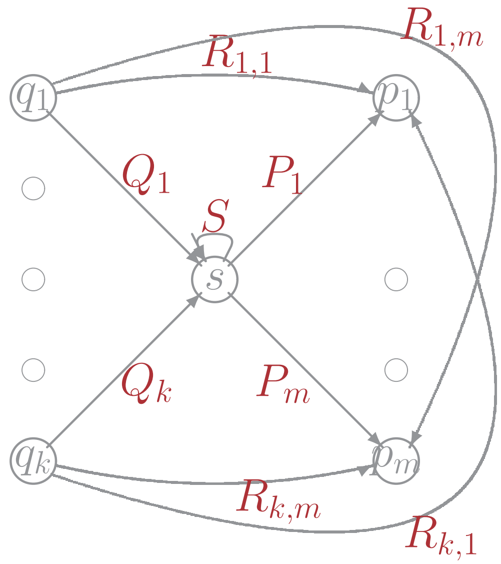
Allgemeine Zustandselimination in VNEAs

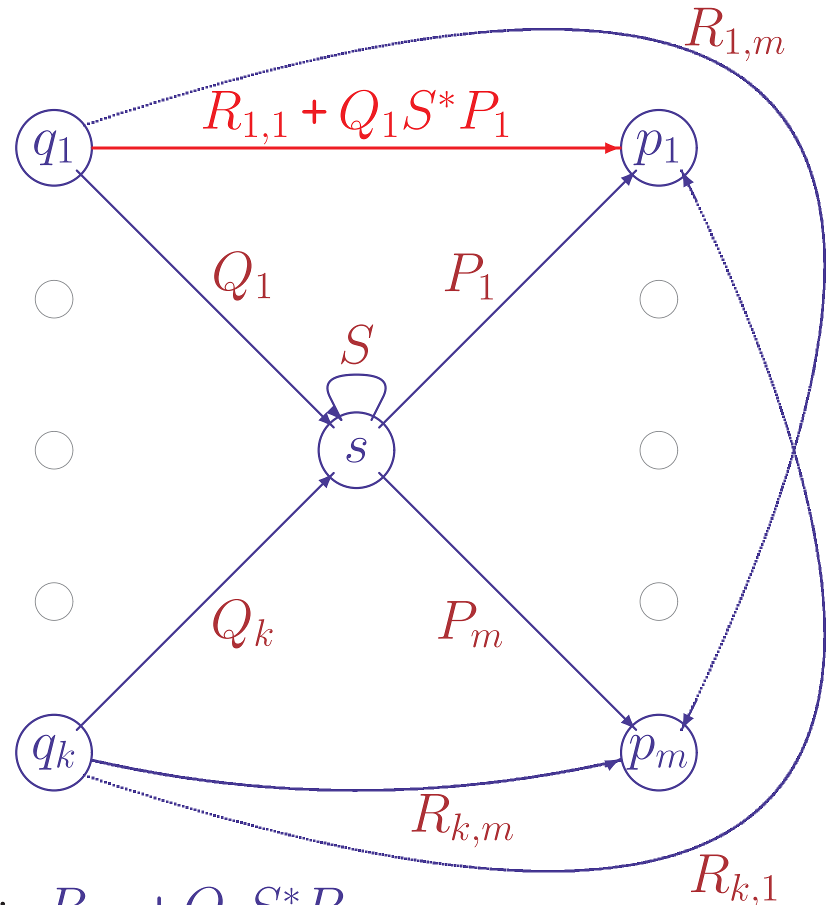


Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

– Eliminiere Pfad von q_1 nach p_1 über s: $R_{1,1} + Q_1 S^* P_1$

Allgemeine Zustandselimination in VNEAs

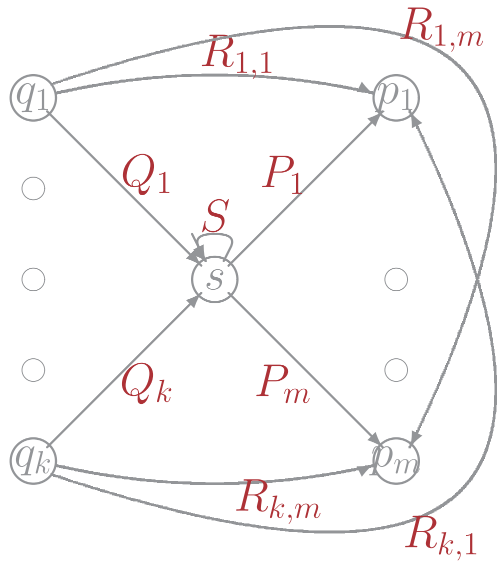


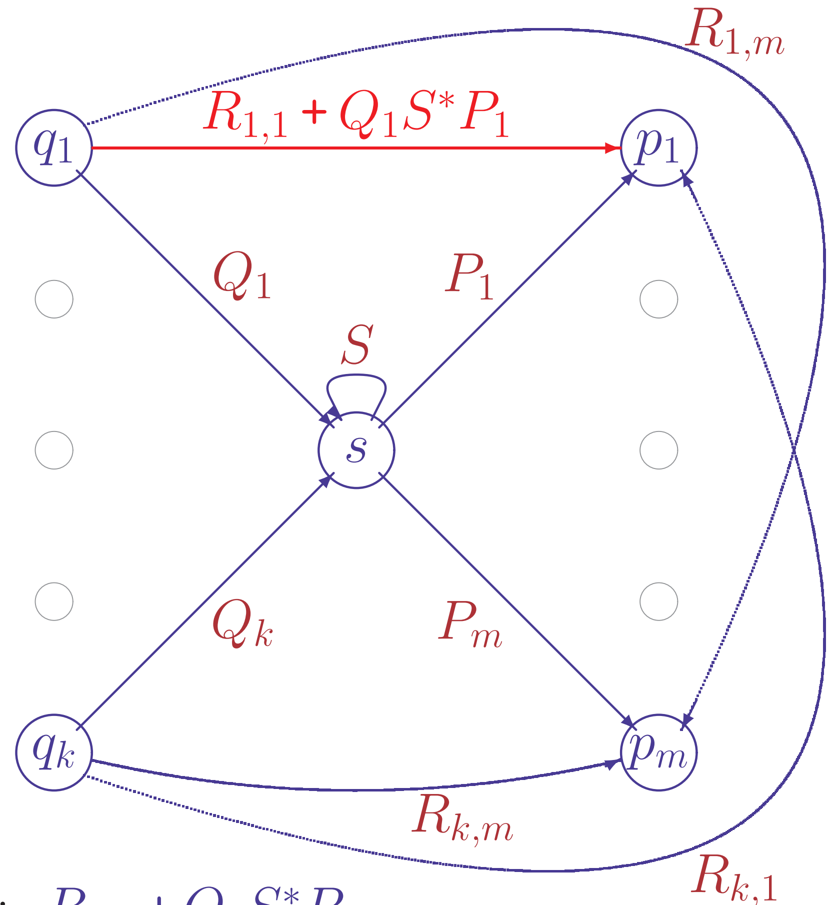


Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

– Eliminiere Pfad von q_1 nach p_1 über s: $R_{1,1} + Q_1 S^* P_1$

Allgemeine Zustandselimination in VNEAs





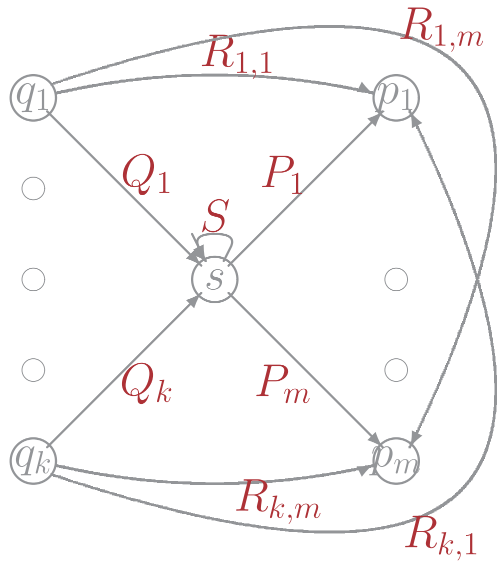
Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

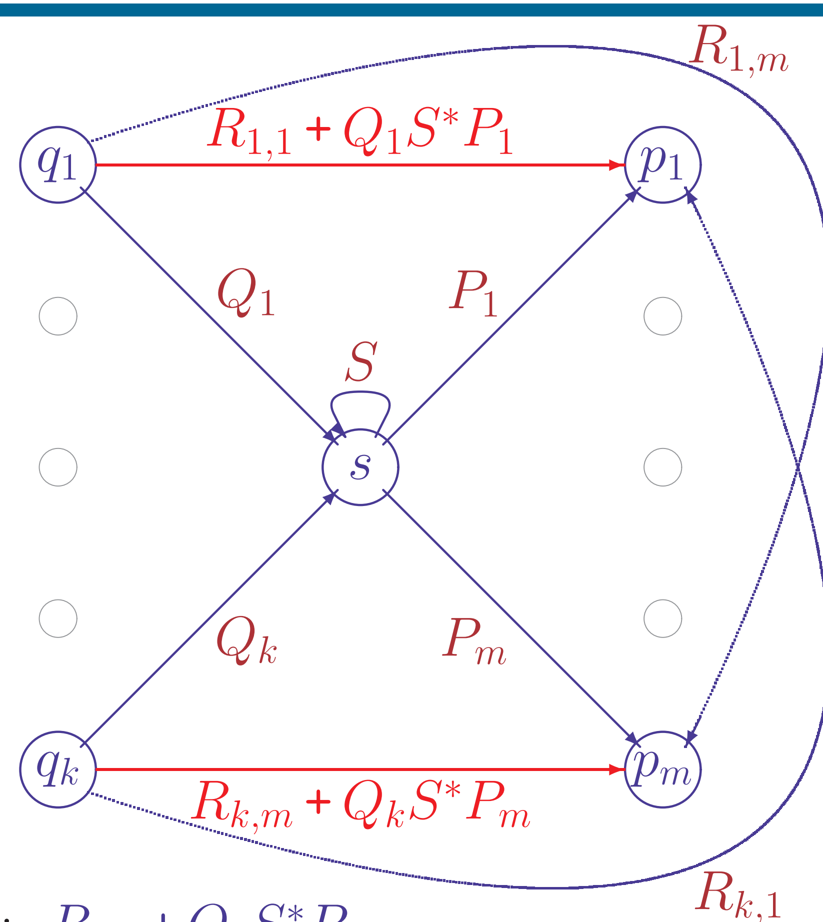
– Eliminiere Pfad von q_1 nach p_1 über s: $R_{1,1} + Q_1 S^* P_1$

$\vdots$

– Eliminiere Pfad von q_1 nach p_m über s: $R_{1,m} + Q_1 S^* P_m$

Allgemeine Zustandselimination in VNEAs





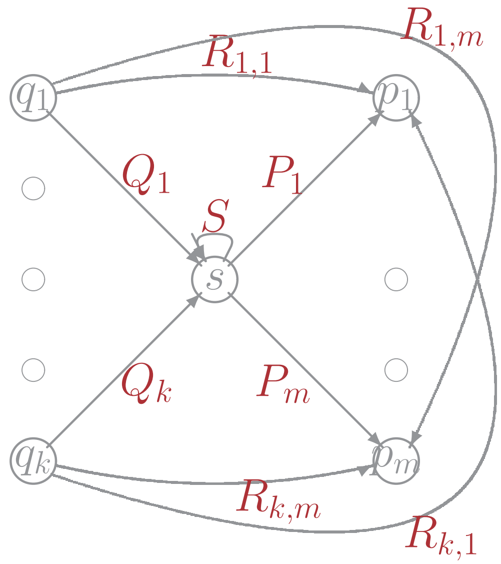
Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

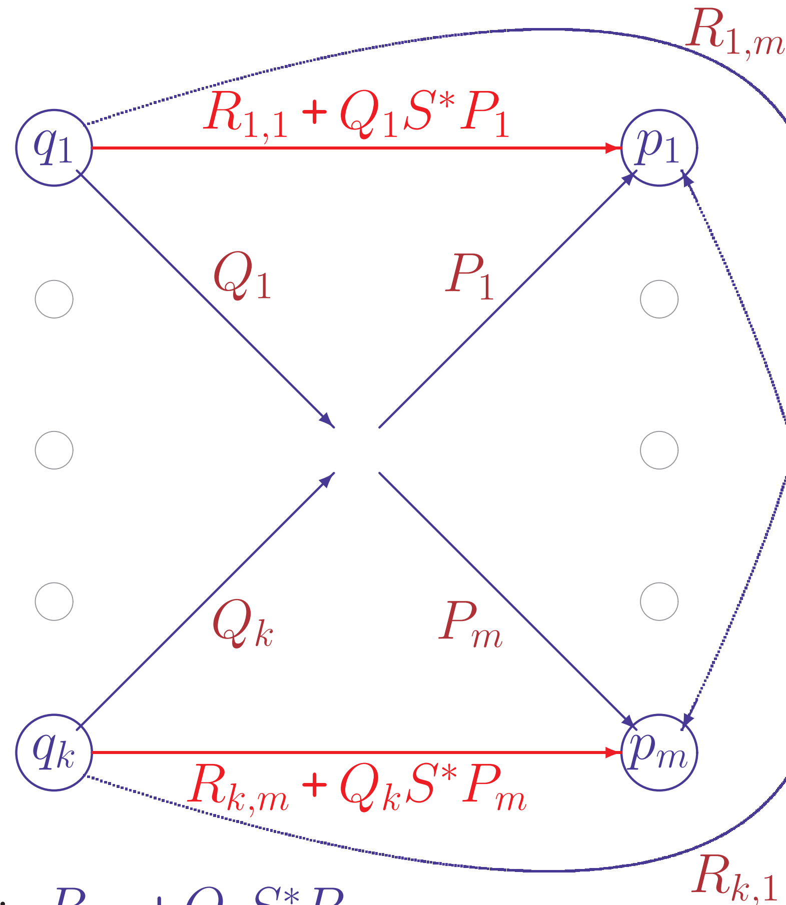
– Eliminiere Pfad von q_1 nach p_1 über s: $R_{1,1}+Q_1S^*P_1$

⋮

– Eliminiere Pfad von q_1 nach p_m über s: $R_{1,m}+Q_1S^*P_m$

Allgemeine Zustandselimination in VNEAs





Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

– Eliminiere Pfad von q_1 nach p_1 über s: $R_{1,1} + Q_1 S^* P_1$

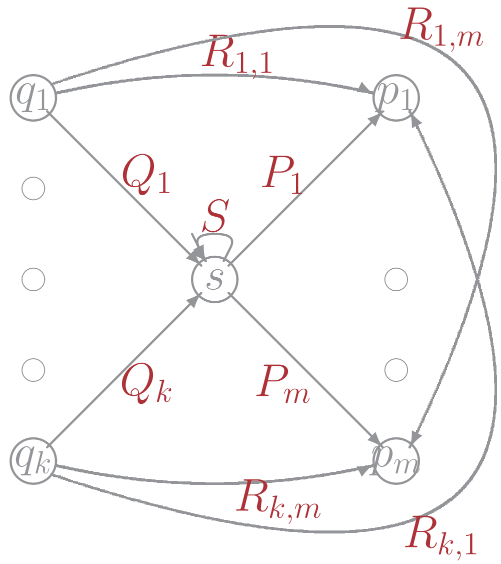
⋮

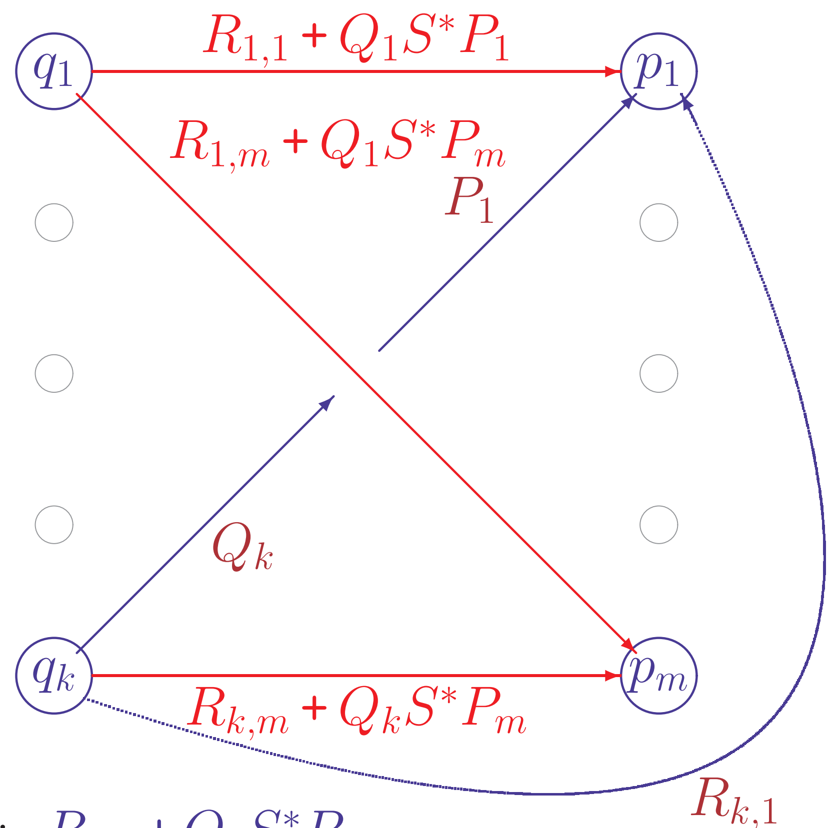
– Eliminiere Pfad von q_1 nach p_m über s: $R_{1,m} + Q_1 S^* P_m$

⋮

– Eliminiere Pfad von q_k nach p_1 über s: $R_{k,1} + Q_k S^* P_1$

Allgemeine Zustandselimination in VNEAs





Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

– Eliminiere Pfad von q_1 nach p_1 über s: $R_{1,1}+Q_1S^*P_1$

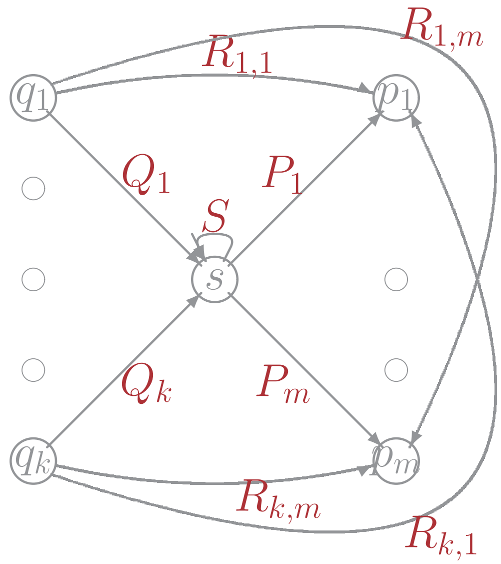
$\vdots$

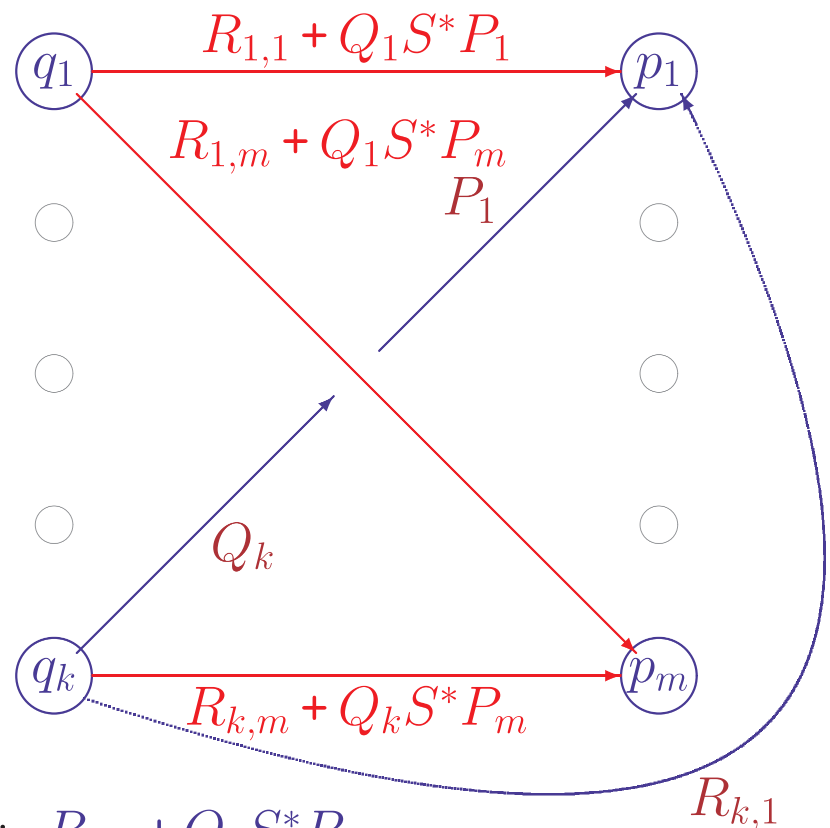
– Eliminiere Pfad von q_1 nach p_m über s: $R_{1,m}+Q_1S^*P_m$

$\vdots$

– Eliminiere Pfad von q_k nach p_1 über s: $R_{k,1}+Q_kS^*P_1$

Allgemeine Zustandselimination in VNEAs





Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

– Eliminiere Pfad von q_1 nach p_1 über s: $R_{1,1}+Q_1S^*P_1$

⋮

– Eliminiere Pfad von q_1 nach p_m über s: $R_{1,m}+Q_1S^*P_m$

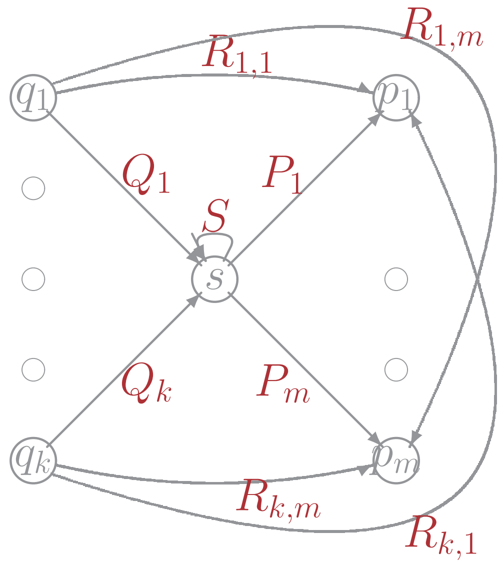
⋮

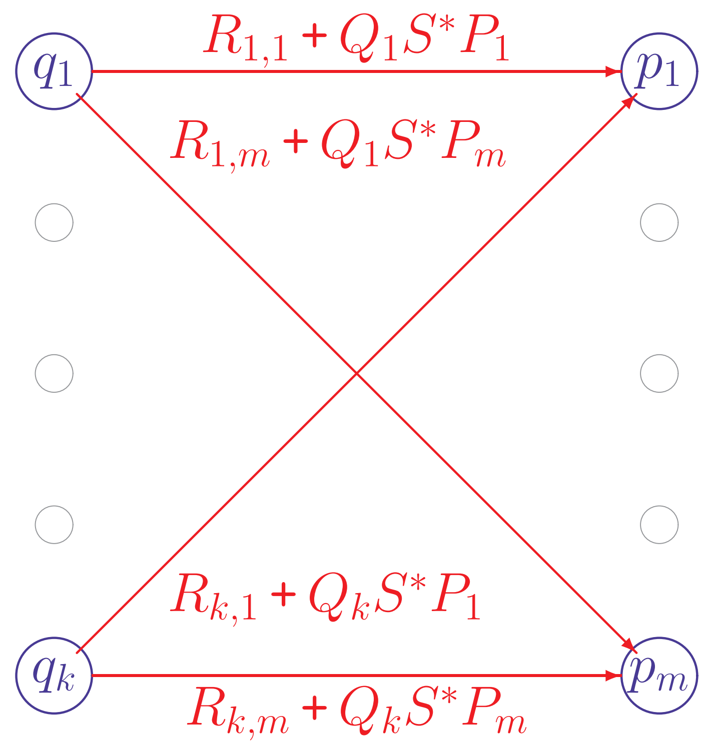
– Eliminiere Pfad von q_k nach p_1 über s: $R_{k,1}+Q_kS^*P_1$

⋮

– Eliminiere Pfad von q_k nach p_m über s: $R_{k,m}+Q_kS^*P_m$

Allgemeine Zustandselimination in VNEAs





Eliminiere Zustand s
mit Vorgängern $q_1, .., q_k$
und Nachfolgern $p_1, .., p_m$

– Eliminiere Pfad von q_1 nach p_1 über s: $R_{1,1}+Q_1S^*P_1$

$\vdots$

– Eliminiere Pfad von q_1 nach p_m über s: $R_{1,m}+Q_1S^*P_m$

$\vdots$

– Eliminiere Pfad von q_k nach p_1 über s: $R_{k,1}+Q_kS^*P_1$

$\vdots$

– Eliminiere Pfad von q_k nach p_m über s: $R_{k,m}+Q_kS^*P_m$

Umwandlung durch Zustandselimination

1. Transformiere endlichen Automaten in VNEA

– Ersetze Beschriftungen mit Symbolen $a \in \Sigma$ durch reguläre Ausdrücke

Umwandlung durch Zustandselimination

1. **Transformiere endlichen Automaten in VNEA**
   - Ersetze Beschriftungen mit Symbolen $a \in \Sigma$ durch reguläre Ausdrücke

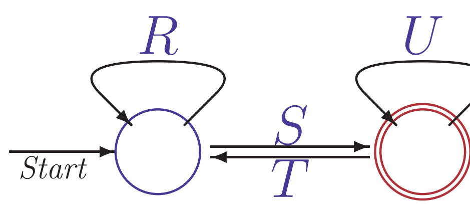
2. **Für $q \in F$ eliminiere alle Zustände außer q_0 und q**
   - Iterative Anwendung des Eliminationsverfahrens

Umwandlung durch Zustandselimination

1. **Transformiere endlichen Automaten in VNEA**
   – Ersetze Beschriftungen mit Symbolen $a \in \Sigma$ durch reguläre Ausdrücke

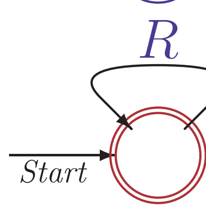
2. **Für $q \in F$ eliminiere alle Zustände außer q_0 und q**
   – Iterative Anwendung des Eliminationsverfahrens

3. **Bilde regulären Ausdruck aus finalem Automaten**
   – $q_0 \neq q$:



$(R^* + SU^*T)^*SU^*$

   – $q_0 = q$:



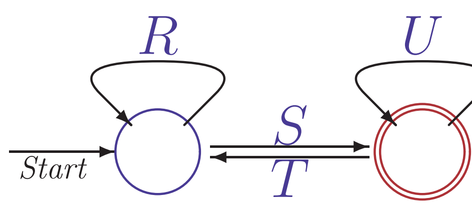
R^*

Umwandlung durch Zustandselimination

1. **Transformiere endlichen Automaten in VNEA**
   – Ersetze Beschriftungen mit Symbolen $a \in \Sigma$ durch reguläre Ausdrücke

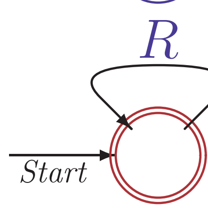
2. **Für $q \in F$ eliminiere alle Zustände außer q_0 und q**
   – Iterative Anwendung des Eliminationsverfahrens

3. **Bilde regulären Ausdruck aus finalem Automaten**
   – $q_0 \neq q$:



$(R^* + SU^*T)^*SU^*$

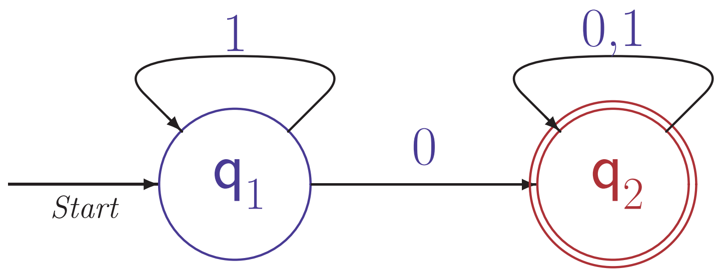
   – $q_0 = q$:



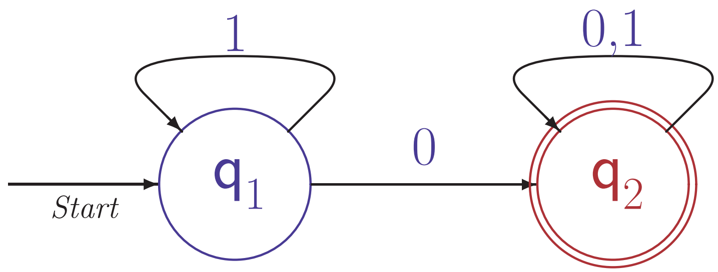
R^*

4. **Vereinige Ausdrücke aller Endzustände**
   – Bilde Summe aller entstandenen regulären Ausdrücke

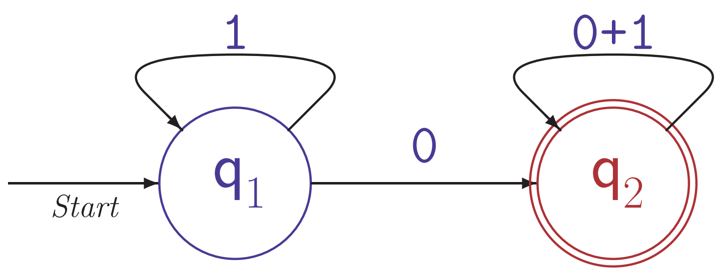
Umwandlung durch Zustandselimination: Beispiel



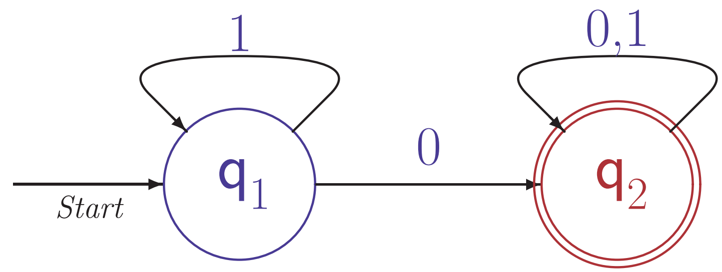
Umwandlung durch Zustandselimination: Beispiel



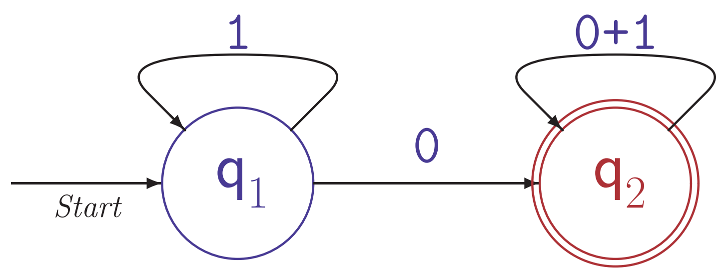
- **Transformiere in RA-Automaten**



Umwandlung durch Zustandselimination: Beispiel

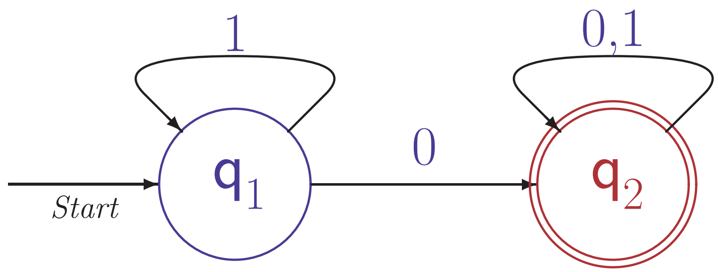


- **Transformiere in RA-Automaten**

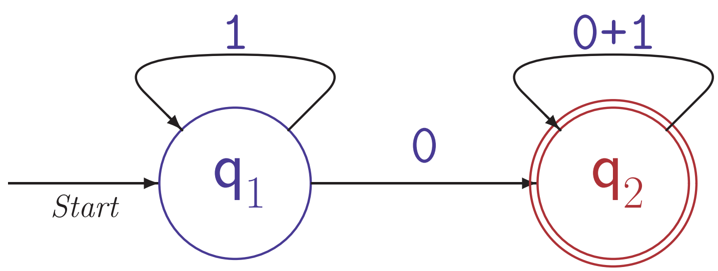


- **Keine Zustände zu eliminieren**

Umwandlung durch Zustandselimination: Beispiel



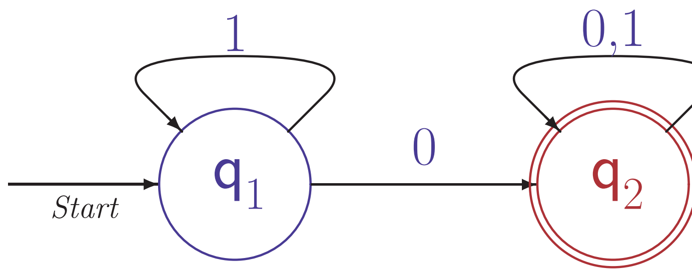
- **Transformiere in RA-Automaten**



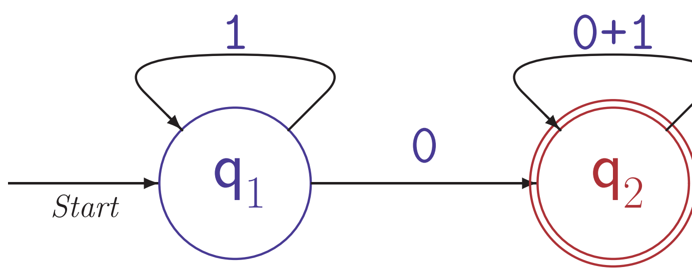
- **Keine Zustände zu eliminieren**

- **Bilde regulären Ausdruck aus finalem Automaten**
  - Extrahierter Ausdruck: $(1^* + 0(0+1)^*\emptyset)^*0(0+1)^*$
  - Nach Vereinfachung: $\boxed{1^*0(0+1)^*}$

Umwandlung durch Zustandselimination: Beispiel



- **Transformiere in RA-Automaten**

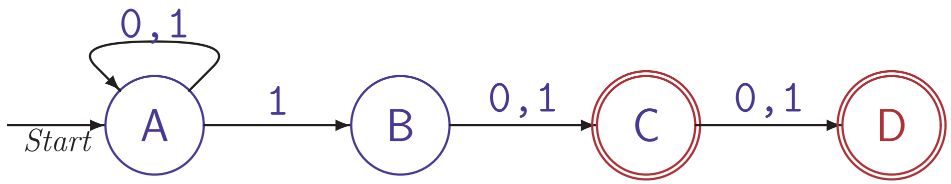


- **Keine Zustände zu eliminieren**

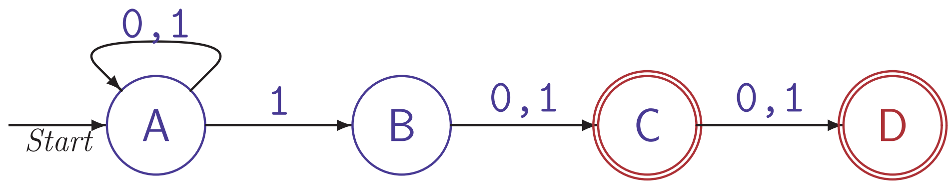
- **Bilde regulären Ausdruck aus finalem Automaten**
  - Extrahierter Ausdruck: $(1^* + 0(0+1)^*\emptyset)^*0(0+1)^*$
  - Nach Vereinfachung: $\boxed{1^*0(0+1)^*}$

Umwandlung mit Pfadanalyseverfahren erfordert 12 aufwendige Schritte

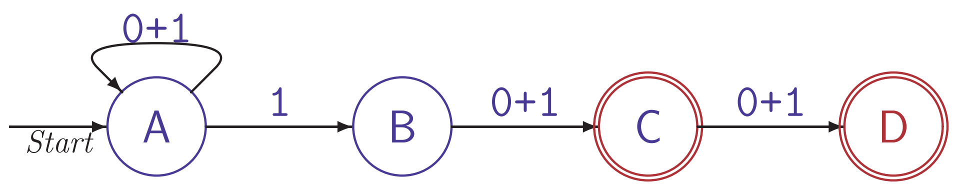
Umwandlung durch Zustandselimination II



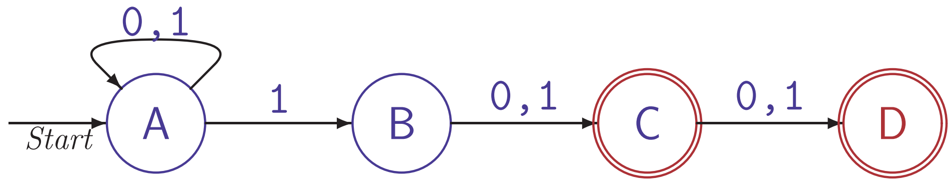
Umwandlung durch Zustandselimination II



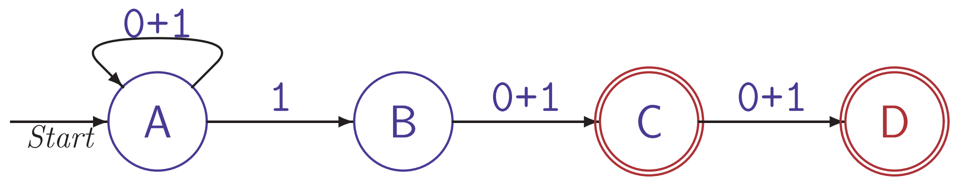
- **Transformiere in RA-Automaten**



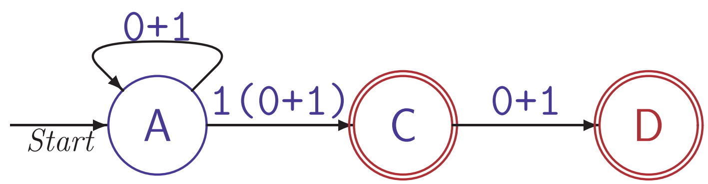
Umwandlung durch Zustandselimination II



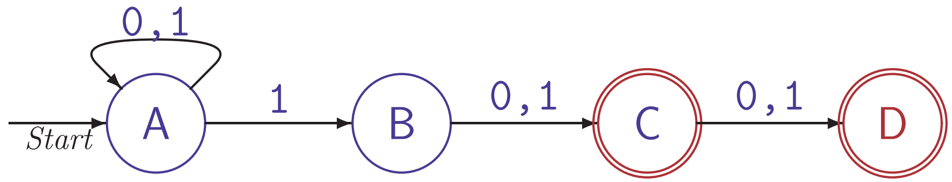
- **Transformiere in RA-Automaten**



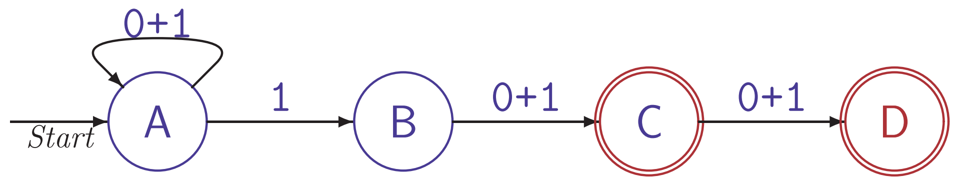
- **Elimination von Zustand *B***



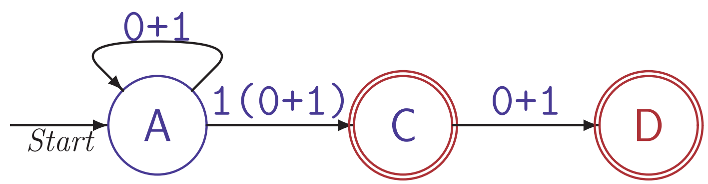
Umwandlung durch Zustandselimination II



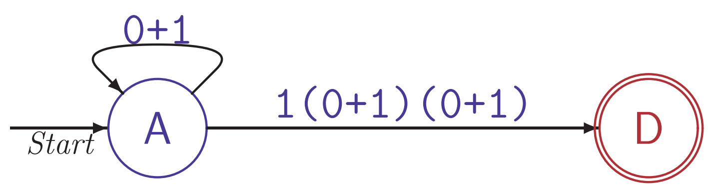
- **Transformiere in RA-Automaten**



- **Elimination von Zustand *B***

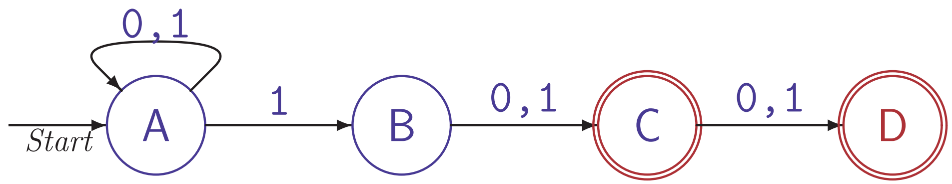


- **Elimination von Zustand *C* für Endzustand *D***

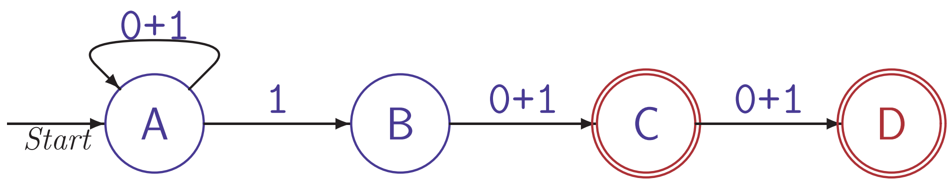


$(0+1)^*1(0+1)(0+1)$

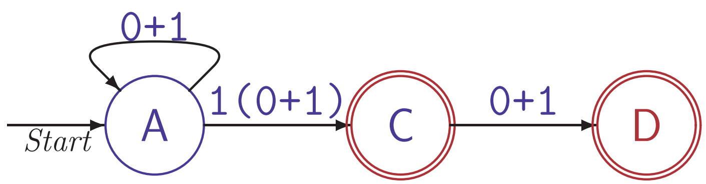
Umwandlung durch Zustandselimination II



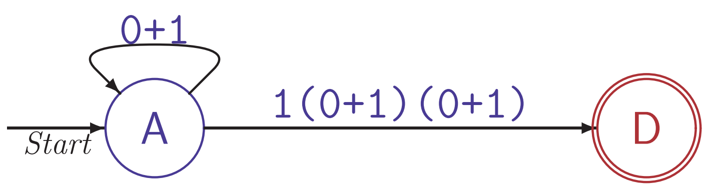
- **Transformiere in RA-Automaten**



- **Elimination von Zustand *B***

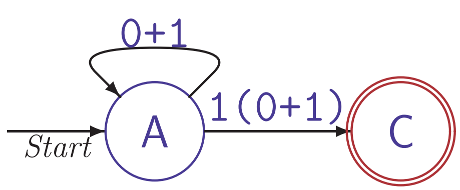


- **Elimination von Zustand *C* für Endzustand *D***



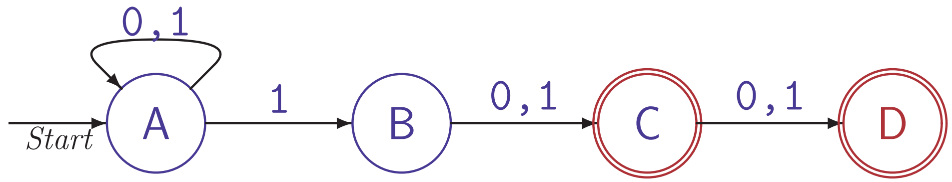
$(0+1)^*1(0+1)(0+1)$

- **Elimination von Zustand *D* für Endzustand *C***

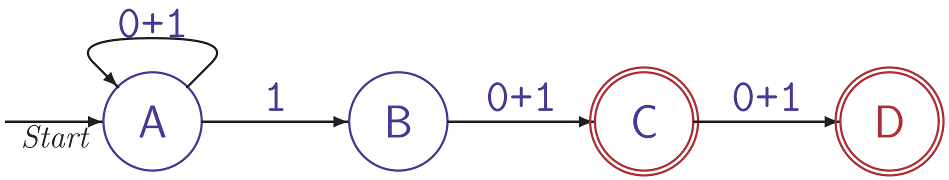


$(0+1)^*1(0+1)$

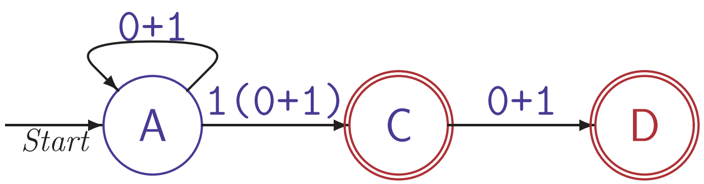
Umwandlung durch Zustandselimination II



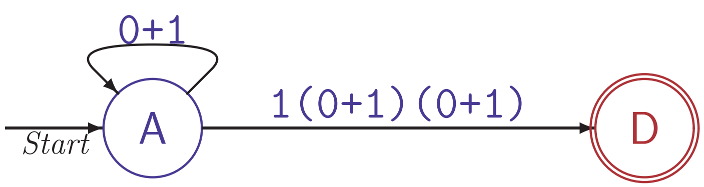
- **Transformiere in RA-Automaten**



- **Elimination von Zustand *B***

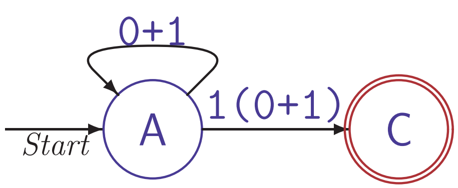


- **Elimination von Zustand *C* für Endzustand *D***



$(0+1)^*1(0+1)(0+1)$

- **Elimination von Zustand *D* für Endzustand *C***



$(0+1)^*1(0+1)$

- **Gesamter Ausdruck:** $\boxed{(0+1)^*1(0+1) + (0+1)^*1(0+1)(0+1)}$

Reguläre Ausdrücke – Zusammenfassung

- **Algebraische Notation für Sprachen**
  – ϵ, $\emptyset$, Symbole des Alphabets, Vereinigung, Verkettung, Sternoperator
  – Äquivalent zu endlichen Automaten
  – Gut zum Nachweis algebraischer Gesetze von Sprachen
  – Anwendung in Programmiersprachen und Suchmaschinen

Reguläre Ausdrücke – Zusammenfassung

- **Algebraische Notation für Sprachen**
  - ϵ, $\emptyset$, Symbole des Alphabets, Vereinigung, Verkettung, Sternoperator
  - Äquivalent zu endlichen Automaten
  - Gut zum Nachweis algebraischer Gesetze von Sprachen
  - Anwendung in Programmiersprachen und Suchmaschinen
- **Transformation in endliche Automaten**
  - Iterative Konstruktion von ϵ-NEAs
  - Nachträgliche Optimierung durch Elimination von ϵ-Übergängen

Reguläre Ausdrücke – Zusammenfassung

- **Algebraische Notation für Sprachen**
  - ϵ, $\emptyset$, Symbole des Alphabets, Vereinigung, Verkettung, Sternoperator
  - Äquivalent zu endlichen Automaten
  - Gut zum Nachweis algebraischer Gesetze von Sprachen
  - Anwendung in Programmiersprachen und Suchmaschinen
- **Transformation in endliche Automaten**
  - Iterative Konstruktion von ϵ-NEAs
  - Nachträgliche Optimierung durch Elimination von ϵ-Übergängen
- **Transformation von Automaten in Ausdrücke**
  - Konstruktion durch Elimination von Zuständen in VNEAs
  - Historisch: Konstruktion von Ausdrücken für Abarbeitungspfade
  - Nachträgliche Optimierungen durch Anwendung algebraischer Gesetze

ANHANG

Originalmethode: allgemeines Graphanalyseverfahren

Originalmethode: allgemeines Graphanalyseverfahren

- **Gegeben DEA** $A = (\{q_1, ..q_n\}, \Sigma, \delta, q_1, \{q_{f_1}, ..q_{f_m}\})$

Originalmethode: allgemeines Graphanalyseverfahren

- **Gegeben DEA $A = (\{q_1, ..q_n\}, \Sigma, \delta, q_1, \{q_{f_1}, ..q_{f_m}\})$**
- **Definiere Ausdrücke für Pfade durch A**
  - R_{ij}^k : Regulärer Ausdruck für Menge der Wörter w mit $\hat{\delta}(q_i, w) = q_j$,
    so dass für alle $\epsilon{\neq}v{\sqsubseteq}w$ $(v{\neq}w)$ gilt: $\hat{\delta}(q_i, v){=}q_m \Rightarrow m{\leq}k$
    (Abarbeitung von w berührt keinen Zustand größer als k)

Originalmethode: allgemeines Graphanalyseverfahren

- **Gegeben DEA $A = (\{q_1, ..q_n\}, \Sigma, \delta, q_1, \{q_{f_1}, ..q_{f_m}\})$**
- **Definiere Ausdrücke für Pfade durch A**
  - R_{ij}^k : Regulärer Ausdruck für Menge der Wörter w mit $\hat{\delta}(q_i, w) = q_j$,
    so dass für alle $\epsilon{\neq}v{\sqsubseteq}w$ $(v{\neq}w)$ gilt: $\hat{\delta}(q_i, v){=}q_m \Rightarrow m{\leq}k$
    (Abarbeitung von w berührt keinen Zustand größer als k)
- **Setze die R_{ij}^k zu Ausdruck für $L(A)$ zusammen**
  - Per Definition ist R_{ij}^n ein Ausdruck für Wörter w mit $\hat{\delta}(q_i, w) = q_j$

Originalmethode: allgemeines Graphanalyseverfahren

- **Gegeben DEA $A = (\{q_1, ..q_n\}, \Sigma, \delta, q_1, \{q_{f_1}, ..q_{f_m}\})$**
- **Definiere Ausdrücke für Pfade durch A**
  - R_{ij}^k : Regulärer Ausdruck für Menge der Wörter w mit $\hat{\delta}(q_i, w) = q_j$, so dass für alle $\epsilon{\neq}v{\sqsubseteq}w$ $(v{\neq}w)$ gilt: $\hat{\delta}(q_i, v){=}q_m \Rightarrow m{\leq}k$ (Abarbeitung von w berührt keinen Zustand größer als k)
- **Setze die R_{ij}^k zu Ausdruck für $L(A)$ zusammen**
  - Per Definition ist R_{ij}^n ein Ausdruck für Wörter w mit $\hat{\delta}(q_i, w) = q_j$
  - Setze $R = R_{1f_1}^n + ... + R_{1f_m}^n$

Originalmethode: allgemeines Graphanalyseverfahren

- **Gegeben DEA $A = (\{q_1, ..q_n\}, \Sigma, \delta, q_1, \{q_{f_1}, ..q_{f_m}\})$**
- **Definiere Ausdrücke für Pfade durch A**
  - R^k_{ij} : Regulärer Ausdruck für Menge der Wörter w mit $\hat{\delta}(q_i, w) = q_j$,
    so dass für alle $\epsilon{\neq}v{\sqsubseteq}w$ $(v{\neq}w)$ gilt: $\hat{\delta}(q_i, v){=}q_m \Rightarrow m{\leq}k$
    (Abarbeitung von w berührt keinen Zustand größer als k)
- **Setze die R^k_{ij} zu Ausdruck für $L(A)$ zusammen**
  - Per Definition ist R^n_{ij} ein Ausdruck für Wörter w mit $\hat{\delta}(q_i, w) = q_j$
  - Setze $R = R^n_{1f_1} + ... + R^n_{1f_m}$
  - Dann gilt $L(R) = \bigcup_{j=1}^m \{w \in \Sigma^* \mid \hat{\delta}(q_1, w) = q_{f_j}\}$

Originalmethode: allgemeines Graphanalyseverfahren

- **Gegeben DEA $A = (\{q_1, ..q_n\}, \Sigma, \delta, q_1, \{q_{f_1}, ..q_{f_m}\})$**
- **Definiere Ausdrücke für Pfade durch A**
  - R^k_{ij} : Regulärer Ausdruck für Menge der Wörter w mit $\hat{\delta}(q_i, w) = q_j$, so dass für alle $\epsilon{\neq}v{\sqsubseteq}w$ $(v{\neq}w)$ gilt: $\hat{\delta}(q_i, v){=}q_m \Rightarrow m{\leq}k$ (Abarbeitung von w berührt keinen Zustand größer als k)
- **Setze die R^k_{ij} zu Ausdruck für $L(A)$ zusammen**
  - Per Definition ist R^n_{ij} ein Ausdruck für Wörter w mit $\hat{\delta}(q_i, w) = q_j$
  - Setze $R = R^n_{1f_1} + ... + R^n_{1f_m}$
  - Dann gilt $L(R) = \bigcup_{j=1}^m \{w \in \Sigma^* \mid \hat{\delta}(q_1, w) = q_{f_j}\}$
    $= \{w \in \Sigma^* \mid \exists q \in \{q_{f_1}, .., q_{f_m}\}.\ \hat{\delta}(q_1, w) = q\}$

Originalmethode: allgemeines Graphanalyseverfahren

- **Gegeben DEA $A = (\{q_1, ..q_n\}, \Sigma, \delta, q_1, \{q_{f_1}, ..q_{f_m}\})$**
- **Definiere Ausdrücke für Pfade durch A**
  - R_{ij}^k : Regulärer Ausdruck für Menge der Wörter w mit $\hat{\delta}(q_i, w) = q_j$,
    so dass für alle $\epsilon{\neq}v{\sqsubseteq}w$ $(v{\neq}w)$ gilt: $\hat{\delta}(q_i, v){=}q_m \;\Rightarrow\; m{\leq}k$
    (Abarbeitung von w berührt keinen Zustand größer als k)
- **Setze die R_{ij}^k zu Ausdruck für $L(A)$ zusammen**
  - Per Definition ist R_{ij}^n ein Ausdruck für Wörter w mit $\hat{\delta}(q_i, w) = q_j$
  - Setze $R = R_{1f_1}^n + ... + R_{1f_m}^n$
  - Dann gilt $L(R) = \bigcup_{j=1}^m \{w \in \Sigma^* \mid \hat{\delta}(q_1, w) = q_{f_j}\}$
    $= \{w \in \Sigma^* \mid \exists q \in \{q_{f_1}, .., q_{f_m}\}.\ \hat{\delta}(q_1, w) = q\} = L(A)$

Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren

Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall $\boldsymbol{R_{ij}^0}$:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$

Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für i=j): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$

Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $R_{ii}^0 = \epsilon + \mathtt{a}_1 + \ldots + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$

    $R_{ij}^0 = \emptyset + \mathtt{a}_1 + \ldots + \mathtt{a}_k \quad (i{\neq}j)$

ITERATIVE BESTIMMUNG DER AUSDRÜCKE R_{ij}^k

- **Basisfall $\boldsymbol{R_{ij}^0}$:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $\boldsymbol{R_{ii}^0} = \epsilon + \mathtt{a}_1 + \ldots + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$

    $\boldsymbol{R_{ij}^0} = \emptyset + \mathtt{a}_1 + \ldots + \mathtt{a}_k \quad (i{\neq}j)$

- **Schrittfall $\boldsymbol{R_{ij}^k}$** $(0{<}k{\leq}n)$: zwei Alternativen

Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $R_{ii}^0 = \epsilon + \mathtt{a}_1 + .. + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$
    $R_{ij}^0 = \emptyset + \mathtt{a}_1 + .. + \mathtt{a}_k$ $(i{\neq}j)$

- **Schrittfall R_{ij}^k** $(0{<}k{\leq}n)$: zwei Alternativen
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k nicht enthält, gehören zu $L(R_{ij}^{k-1})$

Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $\boldsymbol{R_{ii}^0} = \epsilon + \mathtt{a}_1 + \mathtt{..} + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$
    $\boldsymbol{R_{ij}^0} = \emptyset + \mathtt{a}_1 + \mathtt{..} + \mathtt{a}_k$ $(i{\neq}j)$

- **Schrittfall R_{ij}^k** $(0{<}k{\leq}n)$: zwei Alternativen
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k nicht enthält, gehören zu $L(\boldsymbol{R_{ij}^{k-1}})$
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k enthält:
    Zerlege w in $uz_1..z_pv$ mit $\hat{\delta}(q_i, u){=}q_k \wedge \forall l{\leq}p.\hat{\delta}(q_k, z_l){=}q_k \wedge \hat{\delta}(q_k, v){=}q_j$

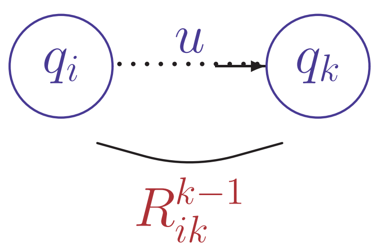
Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $\boldsymbol{R_{ii}^0} = \epsilon + \mathtt{a}_1 + \ldots + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$

    $\boldsymbol{R_{ij}^0} = \emptyset + \mathtt{a}_1 + \ldots + \mathtt{a}_k \quad (i{\neq}j)$

- **Schrittfall R_{ij}^k** $(0{<}k{\leq}n)$: zwei Alternativen
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k nicht enthält, gehören zu $L(\boldsymbol{R_{ij}^{k-1}})$
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k enthält:

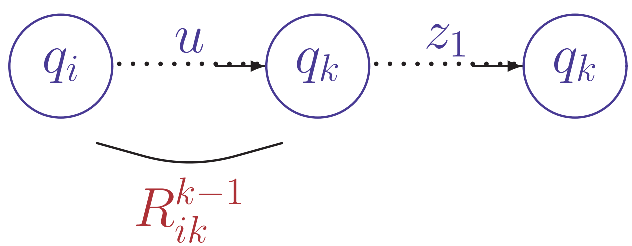
    Zerlege w in $uz_1..z_pv$ mit $\hat{\delta}(q_i, u){=}q_k \ \wedge\ \forall l{\leq}p.\hat{\delta}(q_k, z_l){=}q_k \ \wedge\ \hat{\delta}(q_k, v){=}q_j$



Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $R_{ii}^0 = \epsilon + \mathtt{a}_1 + \ldots + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$
    $R_{ij}^0 = \emptyset + \mathtt{a}_1 + \ldots + \mathtt{a}_k$ $(i{\neq}j)$

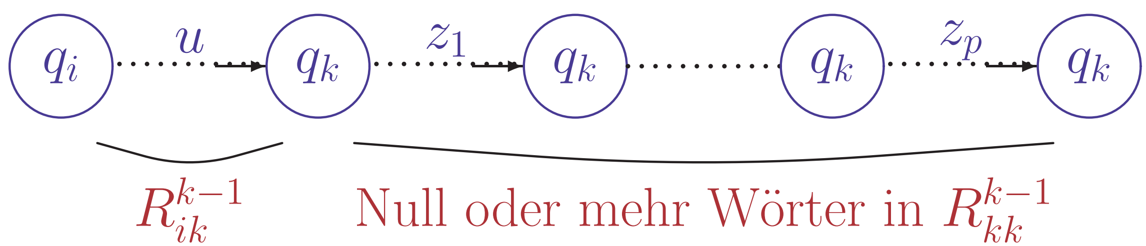
- **Schrittfall R_{ij}^k** $(0{<}k{\leq}n)$: zwei Alternativen
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k nicht enthält, gehören zu $L(R_{ij}^{k-1})$
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k enthält:
    Zerlege w in $uz_1..z_pv$ mit $\hat{\delta}(q_i, u){=}q_k \wedge \forall l{\leq}p.\hat{\delta}(q_k, z_l){=}q_k \wedge \hat{\delta}(q_k, v){=}q_j$



Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $\boldsymbol{R_{ii}^0} = \epsilon + \mathtt{a}_1 + \ldots + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$
    $\boldsymbol{R_{ij}^0} = \emptyset + \mathtt{a}_1 + \ldots + \mathtt{a}_k$ $(i{\neq}j)$

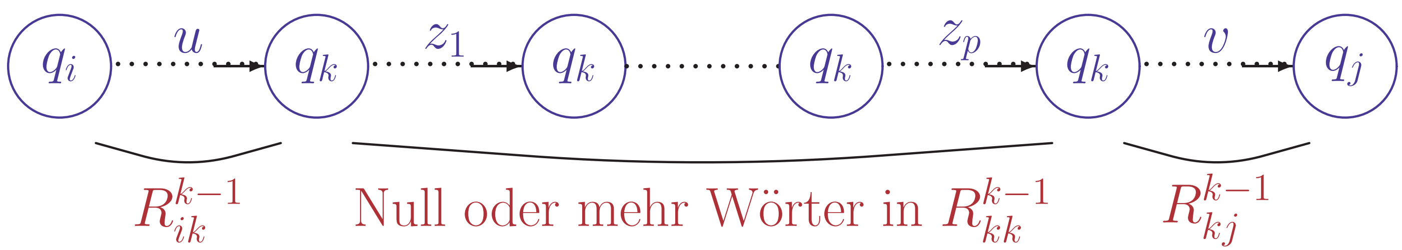
- **Schrittfall R_{ij}^k** $(0{<}k{\leq}n)$: zwei Alternativen
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k nicht enthält, gehören zu $L(\boldsymbol{R_{ij}^{k-1}})$
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k enthält:
    Zerlege w in $uz_1..z_pv$ mit $\hat{\delta}(q_i, u){=}q_k \wedge \forall l{\leq}p.\hat{\delta}(q_k, z_l){=}q_k \wedge \hat{\delta}(q_k, v){=}q_j$



Iterative Bestimmung der Ausdrücke R_{ij}^k

- **Basisfall R_{ij}^0:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $\boldsymbol{R_{ii}^0} = \epsilon + \mathtt{a}_1 + \ldots + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$
    $\boldsymbol{R_{ij}^0} = \emptyset + \mathtt{a}_1 + \ldots + \mathtt{a}_k$ $(i{\neq}j)$

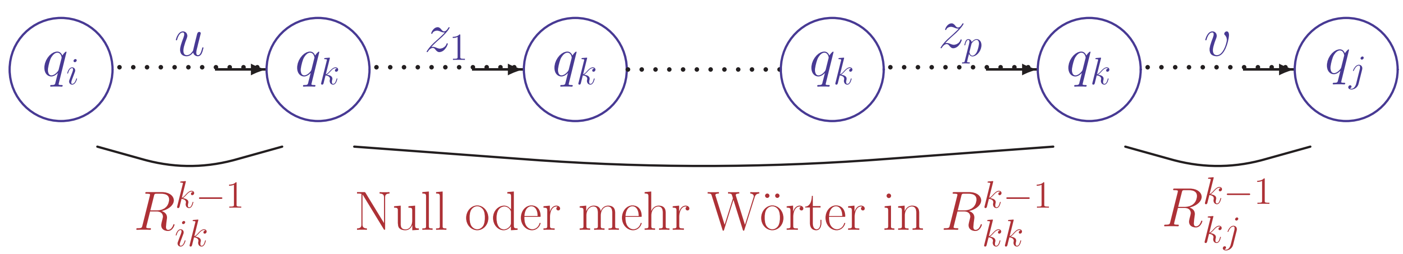
- **Schrittfall R_{ij}^k** $(0{<}k{\leq}n)$: zwei Alternativen
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k nicht enthält, gehören zu $L(\boldsymbol{R_{ij}^{k-1}})$
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k enthält:
    Zerlege w in $uz_1..z_pv$ mit $\hat{\delta}(q_i, u){=}q_k \wedge \forall l{\leq}p.\hat{\delta}(q_k, z_l){=}q_k \wedge \hat{\delta}(q_k, v){=}q_j$



ITERATIVE BESTIMMUNG DER AUSDRÜCKE R_{ij}^k

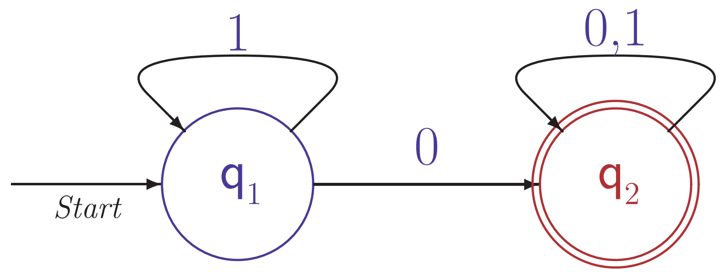
- **Basisfall $\boldsymbol{R_{ij}^0}$:** Pfad darf zwischendurch keine Zustände berühren
  - Pfadlänge 0 (nur für $i{=}j$): $\epsilon \in L(R_{ii}^0)$
  - Pfadlänge 1: $\{a \in \Sigma \mid \delta(q_i, a) = q_j\} \subseteq L(R_{ij}^0)$
  - Ergebnis: $\boldsymbol{R_{ii}^0} = \epsilon + \mathtt{a}_1 + \ldots + \mathtt{a}_k$, wobei $\{a_1, .., a_k\} = \{a \in \Sigma \mid \delta(q_i, a){=}q_j\}$
    $\boldsymbol{R_{ij}^0} = \emptyset + \mathtt{a}_1 + \ldots + \mathtt{a}_k$ $(i{\neq}j)$

- **Schrittfall $\boldsymbol{R_{ij}^k}$** $(0{<}k{\leq}n)$: zwei Alternativen
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k nicht enthält, gehören zu $L(\boldsymbol{R_{ij}^{k-1}})$
  - Wörter $w \in L(R_{ij}^k)$, deren Pfad q_k enthält:
    Zerlege w in $uz_1..z_pv$ mit $\hat{\delta}(q_i, u){=}q_k \wedge \forall l{\leq}p.\hat{\delta}(q_k, z_l){=}q_k \wedge \hat{\delta}(q_k, v){=}q_j$



  - Ergebnis: $\boldsymbol{R_{ij}^k} = \boldsymbol{R_{ij}^{k-1}} + \boldsymbol{R_{ik}^{k-1}} \circ (\boldsymbol{R_{kk}^{k-1}})^* \circ \boldsymbol{R_{kj}^{k-1}}$

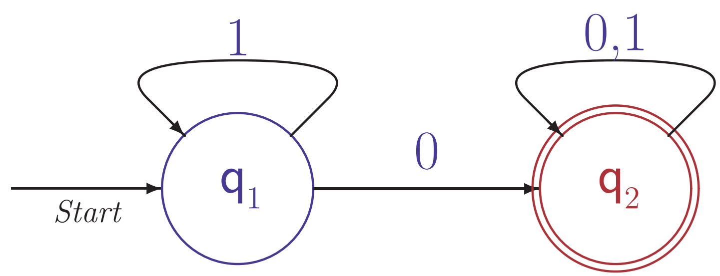
Umwandlung von Automaten am Beispiel



Umwandlung von Automaten am Beispiel

- **Basisfall**

  $R_{11}^{0} = \epsilon\ \texttt{+}\ \texttt{1}$

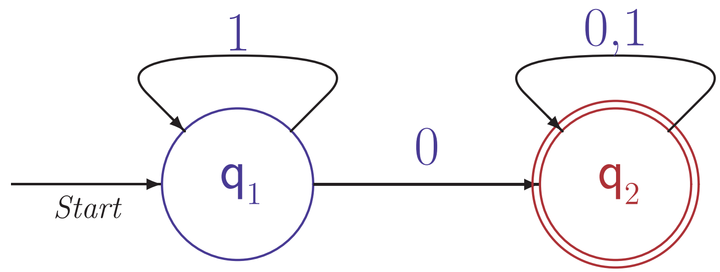


Umwandlung von Automaten am Beispiel

- **Basisfall**

  $R_{11}^0 = \epsilon\ \mathtt{+}\ \mathtt{1}$

  $R_{12}^0 = \mathtt{0}$



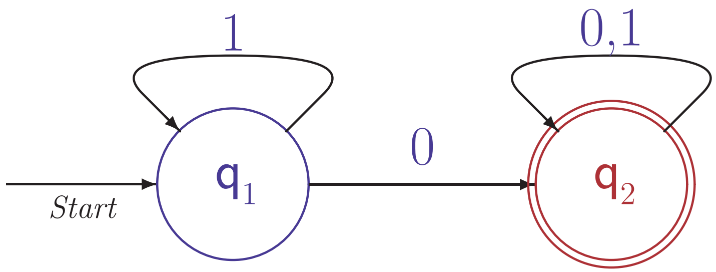
Umwandlung von Automaten am Beispiel

- **Basisfall**

  $R_{11}^{0} = \epsilon + \mathtt{1}$

  $R_{12}^{0} = \mathtt{0}$

  $R_{21}^{0} = \emptyset$



Umwandlung von Automaten am Beispiel

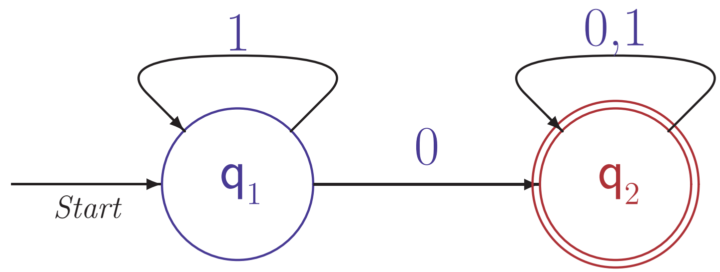
- **Basisfall**

  $R_{11}^{0} = \epsilon\ \texttt{+}\ \texttt{1}$

  $R_{12}^{0} = \texttt{0}$

  $R_{21}^{0} = \emptyset$

  $R_{22}^{0} = \epsilon\ \texttt{+}\ \texttt{0}\ \texttt{+}\ \texttt{1}$



Umwandlung von Automaten am Beispiel

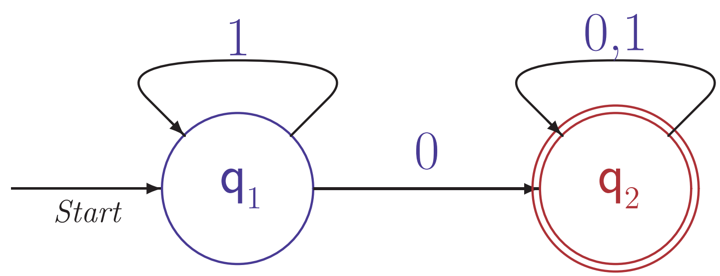
- **Basisfall**

  $R_{11}^{0} = \epsilon + 1$

  $R_{12}^{0} = 0$

  $R_{21}^{0} = \emptyset$

  $R_{22}^{0} = \epsilon + 0 + 1$



- **Stufe 1**

  $R_{11}^{1} = R_{11}^{0} + R_{11}^{0}(R_{11}^{0})^{*}R_{11}^{0} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^{*}(\epsilon + 1)$

Umwandlung von Automaten am Beispiel

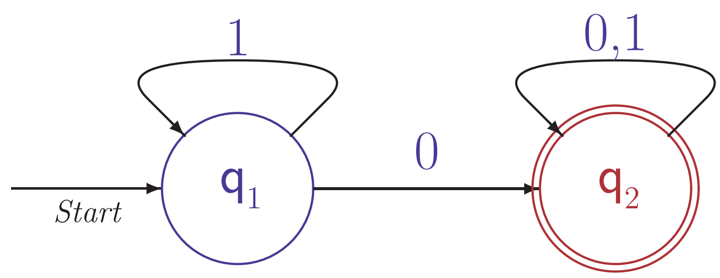
- **Basisfall**

  $R_{11}^0 = \epsilon + 1$

  $R_{12}^0 = 0$

  $R_{21}^0 = \emptyset$

  $R_{22}^0 = \epsilon + 0 + 1$



- **Stufe 1**

  $R_{11}^1 = R_{11}^0 + R_{11}^0 (R_{11}^0)^* R_{11}^0 = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1)$

  $R_{12}^1 = R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0 = 0 + (\epsilon + 1)(\epsilon + 1)^* 0$

Umwandlung von Automaten am Beispiel

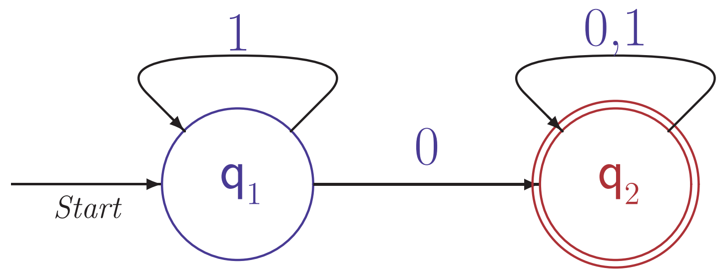
- **Basisfall**

$R^0_{11} = \epsilon + 1$

$R^0_{12} = 0$

$R^0_{21} = \emptyset$

$R^0_{22} = \epsilon + 0 + 1$



- **Stufe 1**

$R^1_{11} = R^0_{11} + R^0_{11}(R^0_{11})^* R^0_{11} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1)$

$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0$

$R^1_{21} = R^0_{21} + R^0_{21}(R^0_{11})^* R^0_{11} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1)$

Umwandlung von Automaten am Beispiel

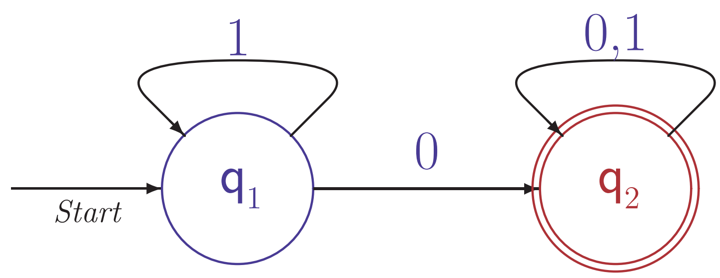
- **Basisfall**

$R^0_{11} = \epsilon + 1$

$R^0_{12} = 0$

$R^0_{21} = \emptyset$

$R^0_{22} = \epsilon + 0 + 1$



- **Stufe 1**

$R^1_{11} = R^0_{11} + R^0_{11}(R^0_{11})^* R^0_{11} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1)$

$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0$

$R^1_{21} = R^0_{21} + R^0_{21}(R^0_{11})^* R^0_{11} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1)$

$R^1_{22} = R^0_{22} + R^0_{21}(R^0_{11})^* R^0_{12} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0$

Umwandlung von Automaten am Beispiel

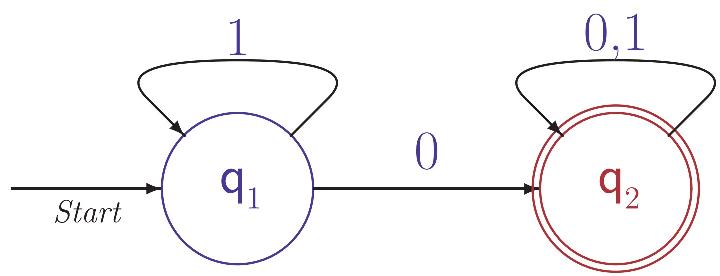
- **Basisfall**

$R_{11}^0 = \epsilon + 1$

$R_{12}^0 = 0$

$R_{21}^0 = \emptyset$

$R_{22}^0 = \epsilon + 0 + 1$



- **Stufe 1**

$R_{11}^1 = R_{11}^0 + R_{11}^0(R_{11}^0)^* R_{11}^0 = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$

$R_{12}^1 = R_{12}^0 + R_{11}^0(R_{11}^0)^* R_{12}^0 = 0 + (\epsilon + 1)(\epsilon + 1)^* 0$

$R_{21}^1 = R_{21}^0 + R_{21}^0(R_{11}^0)^* R_{11}^0 = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1)$

$R_{22}^1 = R_{22}^0 + R_{21}^0(R_{11}^0)^* R_{12}^0 = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0$

Umwandlung von Automaten am Beispiel

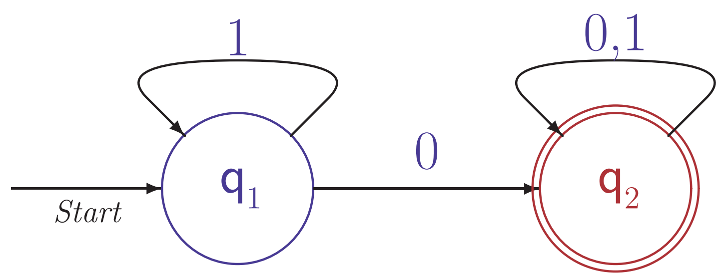
- **Basisfall**

$R_{11}^0 = \epsilon + 1$

$R_{12}^0 = 0$

$R_{21}^0 = \emptyset$

$R_{22}^0 = \epsilon + 0 + 1$



- **Stufe 1**

$R_{11}^1 = R_{11}^0 + R_{11}^0(R_{11}^0)^* R_{11}^0 = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$

$R_{12}^1 = R_{12}^0 + R_{11}^0(R_{11}^0)^* R_{12}^0 = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^*0$

$R_{21}^1 = R_{21}^0 + R_{21}^0(R_{11}^0)^* R_{11}^0 = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1)$

$R_{22}^1 = R_{22}^0 + R_{21}^0(R_{11}^0)^* R_{12}^0 = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0$

Umwandlung von Automaten am Beispiel

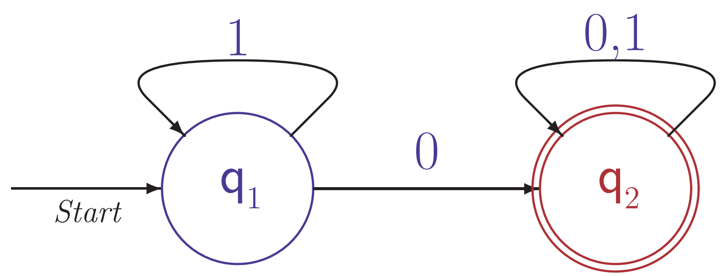
- **Basisfall**

$R_{11}^0 = \epsilon + 1$

$R_{12}^0 = 0$

$R_{21}^0 = \emptyset$

$R_{22}^0 = \epsilon + 0 + 1$



- **Stufe 1**

$$R_{11}^1 = R_{11}^0 + R_{11}^0 (R_{11}^0)^* R_{11}^0 = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \longmapsto \quad 1^*$$

$$R_{12}^1 = R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0 = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \longmapsto \quad 1^* 0$$

$$R_{21}^1 = R_{21}^0 + R_{21}^0 (R_{11}^0)^* R_{11}^0 = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \longmapsto \quad \emptyset$$

$$R_{22}^1 = R_{22}^0 + R_{21}^0 (R_{11}^0)^* R_{12}^0 = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0$$

Umwandlung von Automaten am Beispiel

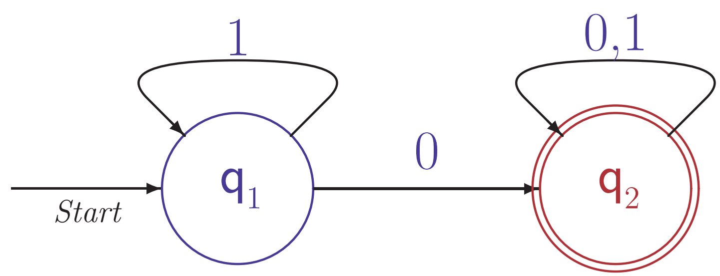
- **Basisfall**

$R_{11}^{0} = \epsilon + 1$

$R_{12}^{0} = 0$

$R_{21}^{0} = \emptyset$

$R_{22}^{0} = \epsilon + 0 + 1$



- **Stufe 1**

$R_{11}^{1} = R_{11}^{0} + R_{11}^{0}(R_{11}^{0})^{*}R_{11}^{0} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^{*}(\epsilon + 1) \quad \mapsto \quad 1^{*}$

$R_{12}^{1} = R_{12}^{0} + R_{11}^{0}(R_{11}^{0})^{*}R_{12}^{0} = 0 + (\epsilon + 1)(\epsilon + 1)^{*}0 \quad \mapsto \quad 1^{*}0$

$R_{21}^{1} = R_{21}^{0} + R_{21}^{0}(R_{11}^{0})^{*}R_{11}^{0} = \emptyset + \emptyset(\epsilon + 1)^{*}(\epsilon + 1) \quad \mapsto \quad \emptyset$

$R_{22}^{1} = R_{22}^{0} + R_{21}^{0}(R_{11}^{0})^{*}R_{12}^{0} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^{*}0 \quad \mapsto \quad \epsilon + 0 + 1$

Umwandlung von Automaten am Beispiel

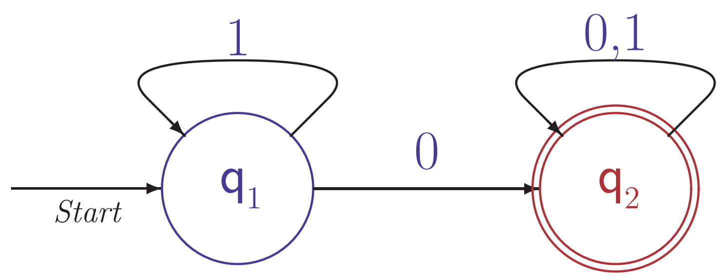
- **Basisfall**

$R^0_{11} = \epsilon + 1$

$R^0_{12} = 0$

$R^0_{21} = \emptyset$

$R^0_{22} = \epsilon + 0 + 1$



- **Stufe 1**

$R^1_{11} = R^0_{11} + R^0_{11}(R^0_{11})^* R^0_{11} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$

$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^* 0$

$R^1_{21} = R^0_{21} + R^0_{21}(R^0_{11})^* R^0_{11} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$

$R^1_{22} = R^0_{22} + R^0_{21}(R^0_{11})^* R^0_{12} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$

- **Stufe 2**

$R^2_{11} = R^1_{11} + R^1_{12}(R^1_{22})^* R^1_{21} = 1^* + 1^* 0(\epsilon + 0 + 1)^* \emptyset$

Umwandlung von Automaten am Beispiel

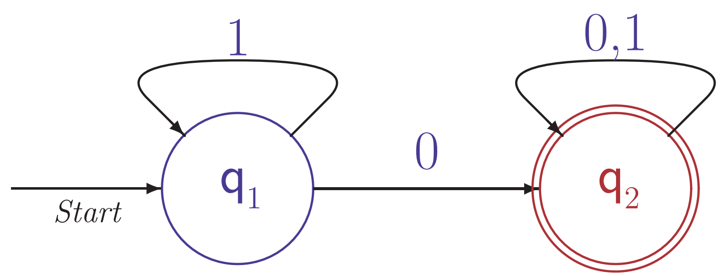
- **Basisfall**

$R^0_{11} = \epsilon + 1$

$R^0_{12} = 0$

$R^0_{21} = \emptyset$

$R^0_{22} = \epsilon + 0 + 1$



- **Stufe 1**

$R^1_{11} = R^0_{11} + R^0_{11}(R^0_{11})^* R^0_{11} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$

$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^* 0$

$R^1_{21} = R^0_{21} + R^0_{21}(R^0_{11})^* R^0_{11} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$

$R^1_{22} = R^0_{22} + R^0_{21}(R^0_{11})^* R^0_{12} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$

- **Stufe 2**

$R^2_{11} = R^1_{11} + R^1_{12}(R^1_{22})^* R^1_{21} = 1^* + 1^* 0(\epsilon+0+1)^* \emptyset$

$R^2_{12} = R^1_{12} + R^1_{12}(R^1_{22})^* R^1_{22} = 1^* 0 + 1^* 0(\epsilon+0+1)^*(\epsilon+0+1)$

Umwandlung von Automaten am Beispiel

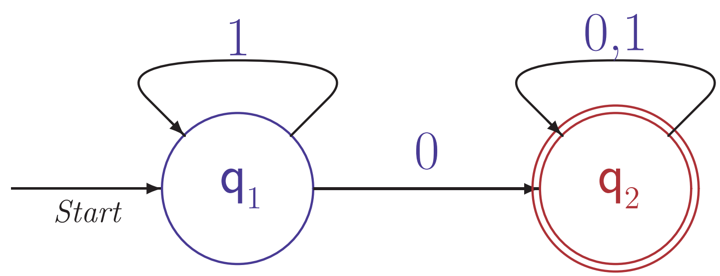
- **Basisfall**

$R^0_{11} = \epsilon + 1$

$R^0_{12} = 0$

$R^0_{21} = \emptyset$

$R^0_{22} = \epsilon + 0 + 1$



- **Stufe 1**

$$R^1_{11} = R^0_{11} + R^0_{11}(R^0_{11})^* R^0_{11} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$$

$$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^*0$$

$$R^1_{21} = R^0_{21} + R^0_{21}(R^0_{11})^* R^0_{11} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$$

$$R^1_{22} = R^0_{22} + R^0_{21}(R^0_{11})^* R^0_{12} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$$

- **Stufe 2**

$$R^2_{11} = R^1_{11} + R^1_{12}(R^1_{22})^* R^1_{21} = 1^* + 1^*0(\epsilon+0+1)^*\emptyset$$

$$R^2_{12} = R^1_{12} + R^1_{12}(R^1_{22})^* R^1_{22} = 1^*0 + 1^*0(\epsilon+0+1)^*(\epsilon+0+1)$$

$$R^2_{21} = R^1_{21} + R^1_{22}(R^1_{22})^* R^1_{21} = \emptyset + (\epsilon+0+1)(\epsilon+0+1)^*\emptyset$$

Umwandlung von Automaten am Beispiel

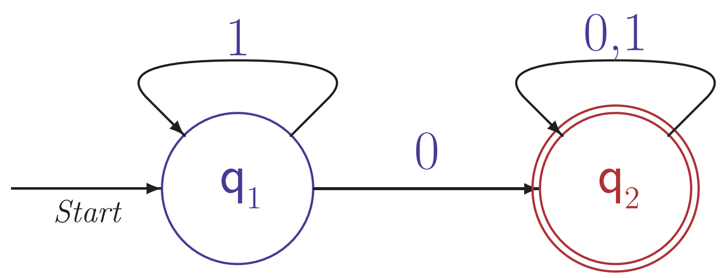
- **Basisfall**

$R_{11}^0 = \epsilon + 1$

$R_{12}^0 = 0$

$R_{21}^0 = \emptyset$

$R_{22}^0 = \epsilon + 0 + 1$



- **Stufe 1**

$$R_{11}^1 = R_{11}^0 + R_{11}^0 (R_{11}^0)^* R_{11}^0 = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$$

$$R_{12}^1 = R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0 = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^* 0$$

$$R_{21}^1 = R_{21}^0 + R_{21}^0 (R_{11}^0)^* R_{11}^0 = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$$

$$R_{22}^1 = R_{22}^0 + R_{21}^0 (R_{11}^0)^* R_{12}^0 = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$$

- **Stufe 2**

$$R_{11}^2 = R_{11}^1 + R_{12}^1 (R_{22}^1)^* R_{21}^1 = 1^* + 1^* 0 (\epsilon+0+1)^* \emptyset$$

$$R_{12}^2 = R_{12}^1 + R_{12}^1 (R_{22}^1)^* R_{22}^1 = 1^* 0 + 1^* 0 (\epsilon+0+1)^*(\epsilon+0+1)$$

$$R_{21}^2 = R_{21}^1 + R_{22}^1 (R_{22}^1)^* R_{21}^1 = \emptyset + (\epsilon+0+1)(\epsilon+0+1)^* \emptyset$$

$$R_{22}^2 = R_{22}^1 + R_{22}^1 (R_{22}^1)^* R_{22}^1 = (\epsilon+0+1) + (\epsilon+0+1)(\epsilon+0+1)^*(\epsilon+0+1)$$

Umwandlung von Automaten am Beispiel

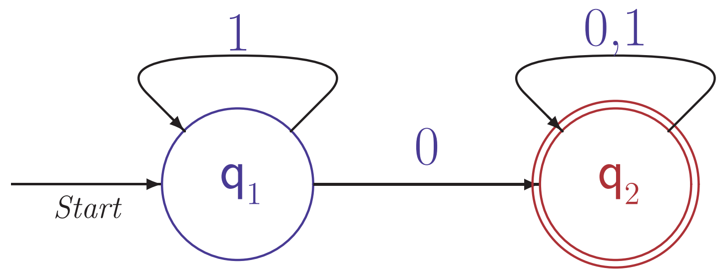
- **Basisfall**

$R^0_{11} = \epsilon + 1$

$R^0_{12} = 0$

$R^0_{21} = \emptyset$

$R^0_{22} = \epsilon + 0 + 1$



- **Stufe 1**

$R^1_{11} = R^0_{11} + R^0_{11}(R^0_{11})^* R^0_{11} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$

$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^* 0$

$R^1_{21} = R^0_{21} + R^0_{21}(R^0_{11})^* R^0_{11} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$

$R^1_{22} = R^0_{22} + R^0_{21}(R^0_{11})^* R^0_{12} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$

- **Stufe 2**

$R^2_{11} = R^1_{11} + R^1_{12}(R^1_{22})^* R^1_{21} = 1^* + 1^* 0(\epsilon+0+1)^* \emptyset \quad \mapsto \quad 1^*$

$R^2_{12} = R^1_{12} + R^1_{12}(R^1_{22})^* R^1_{22} = 1^* 0 + 1^* 0(\epsilon+0+1)^*(\epsilon+0+1)$

$R^2_{21} = R^1_{21} + R^1_{22}(R^1_{22})^* R^1_{21} = \emptyset + (\epsilon+0+1)(\epsilon+0+1)^* \emptyset$

$R^2_{22} = R^1_{22} + R^1_{22}(R^1_{22})^* R^1_{22} = (\epsilon+0+1) + (\epsilon+0+1)(\epsilon+0+1)^*(\epsilon+0+1)$

Umwandlung von Automaten am Beispiel

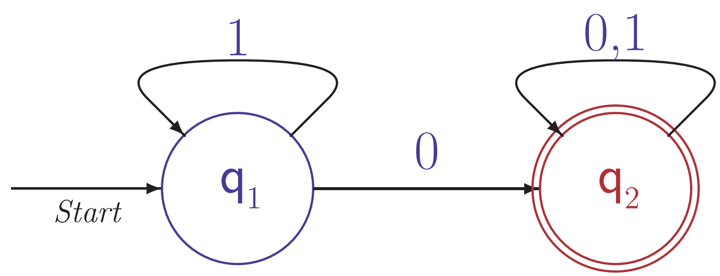
- **Basisfall**

$R^0_{11} = \epsilon + 1$

$R^0_{12} = 0$

$R^0_{21} = \emptyset$

$R^0_{22} = \epsilon + 0 + 1$



- **Stufe 1**

$$R^1_{11} = R^0_{11} + R^0_{11}(R^0_{11})^* R^0_{11} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$$

$$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^* 0$$

$$R^1_{21} = R^0_{21} + R^0_{21}(R^0_{11})^* R^0_{11} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$$

$$R^1_{22} = R^0_{22} + R^0_{21}(R^0_{11})^* R^0_{12} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$$

- **Stufe 2**

$$R^2_{11} = R^1_{11} + R^1_{12}(R^1_{22})^* R^1_{21} = 1^* + 1^* 0(\epsilon+0+1)^*\emptyset \quad \mapsto \quad 1^*$$

$$R^2_{12} = R^1_{12} + R^1_{12}(R^1_{22})^* R^1_{22} = 1^* 0 + 1^* 0(\epsilon+0+1)^*(\epsilon+0+1) \quad \mapsto \quad 1^* 0(0+1)^*$$

$$R^2_{21} = R^1_{21} + R^1_{22}(R^1_{22})^* R^1_{21} = \emptyset + (\epsilon+0+1)(\epsilon+0+1)^*\emptyset$$

$$R^2_{22} = R^1_{22} + R^1_{22}(R^1_{22})^* R^1_{22} = (\epsilon+0+1) + (\epsilon+0+1)(\epsilon+0+1)^*(\epsilon+0+1)$$

Umwandlung von Automaten am Beispiel

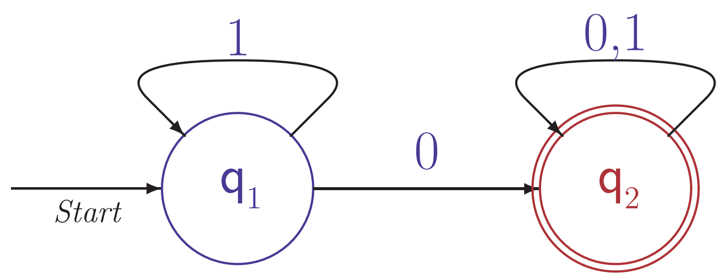
- **Basisfall**

$R^0_{11} = \epsilon + 1$

$R^0_{12} = 0$

$R^0_{21} = \emptyset$

$R^0_{22} = \epsilon + 0 + 1$



- **Stufe 1**

$R^1_{11} = R^0_{11} + R^0_{11}(R^0_{11})^* R^0_{11} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$

$R^1_{12} = R^0_{12} + R^0_{11}(R^0_{11})^* R^0_{12} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^*0$

$R^1_{21} = R^0_{21} + R^0_{21}(R^0_{11})^* R^0_{11} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$

$R^1_{22} = R^0_{22} + R^0_{21}(R^0_{11})^* R^0_{12} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$

- **Stufe 2**

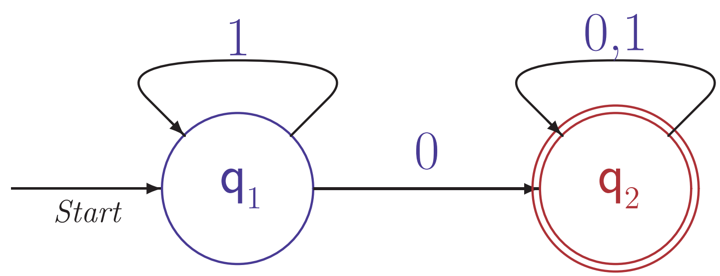
$R^2_{11} = R^1_{11} + R^1_{12}(R^1_{22})^* R^1_{21} = 1^* + 1^*0(\epsilon+0+1)^*\emptyset \quad \mapsto \quad 1^*$

$R^2_{12} = R^1_{12} + R^1_{12}(R^1_{22})^* R^1_{22} = 1^*0 + 1^*0(\epsilon+0+1)^*(\epsilon+0+1) \quad \mapsto \quad 1^*0(0+1)^*$

$R^2_{21} = R^1_{21} + R^1_{22}(R^1_{22})^* R^1_{21} = \emptyset + (\epsilon+0+1)(\epsilon+0+1)^*\emptyset \quad \mapsto \quad \emptyset$

$R^2_{22} = R^1_{22} + R^1_{22}(R^1_{22})^* R^1_{22} = (\epsilon+0+1) + (\epsilon+0+1)(\epsilon+0+1)^*(\epsilon+0+1)$

Umwandlung von Automaten am Beispiel



- **Basisfall**

$$R_{11}^0 = \epsilon + 1$$
$$R_{12}^0 = 0$$
$$R_{21}^0 = \emptyset$$
$$R_{22}^0 = \epsilon + 0 + 1$$

- **Stufe 1**

$$R_{11}^1 = R_{11}^0 + R_{11}^0(R_{11}^0)^* R_{11}^0 = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$$
$$R_{12}^1 = R_{12}^0 + R_{11}^0(R_{11}^0)^* R_{12}^0 = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^* 0$$
$$R_{21}^1 = R_{21}^0 + R_{21}^0(R_{11}^0)^* R_{11}^0 = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$$
$$R_{22}^1 = R_{22}^0 + R_{21}^0(R_{11}^0)^* R_{12}^0 = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$$

- **Stufe 2**

$$R_{11}^2 = R_{11}^1 + R_{12}^1(R_{22}^1)^* R_{21}^1 = 1^* + 1^* 0(\epsilon+0+1)^* \emptyset \quad \mapsto \quad 1^*$$
$$R_{12}^2 = R_{12}^1 + R_{12}^1(R_{22}^1)^* R_{22}^1 = 1^* 0 + 1^* 0(\epsilon+0+1)^*(\epsilon+0+1) \quad \mapsto \quad 1^* 0(0+1)^*$$
$$R_{21}^2 = R_{21}^1 + R_{22}^1(R_{22}^1)^* R_{21}^1 = \emptyset + (\epsilon+0+1)(\epsilon+0+1)^* \emptyset \quad \mapsto \quad \emptyset$$
$$R_{22}^2 = R_{22}^1 + R_{22}^1(R_{22}^1)^* R_{22}^1 = (\epsilon+0+1) + (\epsilon+0+1)(\epsilon+0+1)^*(\epsilon+0+1) \quad \mapsto \quad (0+1)^*$$

Umwandlung von Automaten am Beispiel

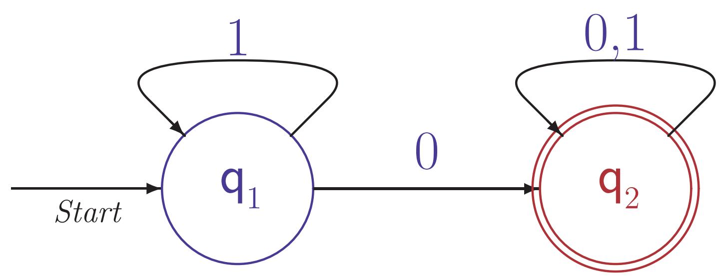
- **Basisfall**

$R_{11}^{0} = \epsilon + 1$

$R_{12}^{0} = 0$

$R_{21}^{0} = \emptyset$

$R_{22}^{0} = \epsilon + 0 + 1$



- **Stufe 1**

$R_{11}^{1} = R_{11}^{0} + R_{11}^{0}(R_{11}^{0})^* R_{11}^{0} = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$

$R_{12}^{1} = R_{12}^{0} + R_{11}^{0}(R_{11}^{0})^* R_{12}^{0} = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^* 0$

$R_{21}^{1} = R_{21}^{0} + R_{21}^{0}(R_{11}^{0})^* R_{11}^{0} = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$

$R_{22}^{1} = R_{22}^{0} + R_{21}^{0}(R_{11}^{0})^* R_{12}^{0} = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$

- **Stufe 2** — **Gebraucht wird nur R_{12}^{2}**

$R_{11}^{2} = R_{11}^{1} + R_{12}^{1}(R_{22}^{1})^* R_{21}^{1} = 1^* + 1^* 0(\epsilon+0+1)^* \emptyset \quad \mapsto \quad 1^*$

$R_{12}^{2} = R_{12}^{1} + R_{12}^{1}(R_{22}^{1})^* R_{22}^{1} = 1^* 0 + 1^* 0(\epsilon+0+1)^*(\epsilon+0+1) \quad \mapsto \quad 1^* 0(0+1)^*$

$R_{21}^{2} = R_{21}^{1} + R_{22}^{1}(R_{22}^{1})^* R_{21}^{1} = \emptyset + (\epsilon+0+1)(\epsilon+0+1)^* \emptyset \quad \mapsto \quad \emptyset$

$R_{22}^{2} = R_{22}^{1} + R_{22}^{1}(R_{22}^{1})^* R_{22}^{1} = (\epsilon+0+1) + (\epsilon+0+1)(\epsilon+0+1)^*(\epsilon+0+1) \quad \mapsto \quad (0+1)^*$

UMWANDLUNG VON AUTOMATEN AM BEISPIEL

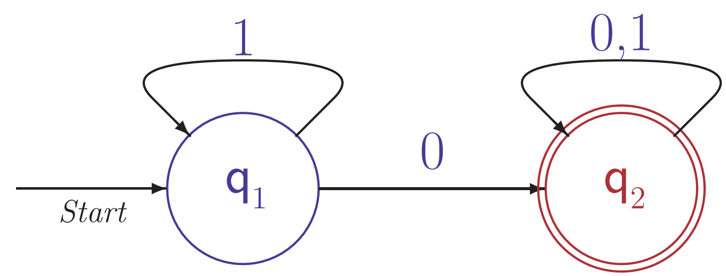
- **Basisfall**

$R_{11}^0 = \epsilon + 1$

$R_{12}^0 = 0$

$R_{21}^0 = \emptyset$

$R_{22}^0 = \epsilon + 0 + 1$



- **Stufe 1**

$R_{11}^1 = R_{11}^0 + R_{11}^0 (R_{11}^0)^* R_{11}^0 = \epsilon + 1 + (\epsilon + 1)(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad 1^*$

$R_{12}^1 = R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0 = 0 + (\epsilon + 1)(\epsilon + 1)^* 0 \quad \mapsto \quad 1^* 0$

$R_{21}^1 = R_{21}^0 + R_{21}^0 (R_{11}^0)^* R_{11}^0 = \emptyset + \emptyset(\epsilon + 1)^*(\epsilon + 1) \quad \mapsto \quad \emptyset$

$R_{22}^1 = R_{22}^0 + R_{21}^0 (R_{11}^0)^* R_{12}^0 = \epsilon + 0 + 1 + \emptyset(\epsilon + 1)^* 0 \quad \mapsto \quad \epsilon + 0 + 1$

- **Stufe 2** **Gebraucht wird nur R_{12}^2**

$R_{11}^2 = R_{11}^1 + R_{12}^1 (R_{22}^1)^* R_{21}^1 = 1^* + 1^* 0 (\epsilon+0+1)^* \emptyset \quad \mapsto \quad 1^*$

$R_{12}^2 = R_{12}^1 + R_{12}^1 (R_{22}^1)^* R_{22}^1 = 1^* 0 + 1^* 0 (\epsilon+0+1)^* (\epsilon+0+1) \quad \mapsto \quad 1^* 0 (0+1)^*$

$R_{21}^2 = R_{21}^1 + R_{22}^1 (R_{22}^1)^* R_{21}^1 = \emptyset + (\epsilon+0+1)(\epsilon+0+1)^* \emptyset \quad \mapsto \quad \emptyset$

$R_{22}^2 = R_{22}^1 + R_{22}^1 (R_{22}^1)^* R_{22}^1 = (\epsilon+0+1) + (\epsilon+0+1)(\epsilon+0+1)^*(\epsilon+0+1) \quad \mapsto \quad (0+1)^*$

Regulärer Ausdruck des Automaten: $1^* 0 (0+1)^*$

Das Pfadanalyseverfahren ist zu kompliziert

- **Konstruktion aller R_{ij}^{k} ist aufwendig**
  - Es müssen mehr als n^3 Ausdrücke R_{ij}^{k} erzeugt werden
  - Ausdrücke R_{ij}^{k} können viermal so groß wie die R_{ij}^{k-1} werden

Das Pfadanalyseverfahren ist zu kompliziert

- **Konstruktion aller R_{ij}^k ist aufwendig**
  - Es müssen mehr als n^3 Ausdrücke R_{ij}^k erzeugt werden
  - Ausdrücke R_{ij}^k können viermal so groß wie die R_{ij}^{k-1} werden
  - Ohne Vereinfachung der R_{ij}^k sind bis zu $n^3 * 4^n$ Symbole zu erzeugen

Das Pfadanalyseverfahren ist zu kompliziert

- **Konstruktion aller R_{ij}^{k} ist aufwendig**
  - Es müssen mehr als n^3 Ausdrücke R_{ij}^{k} erzeugt werden
  - Ausdrücke R_{ij}^{k} können viermal so groß wie die R_{ij}^{k-1} werden
  - Ohne Vereinfachung der R_{ij}^{k} sind bis zu $n^3 * 4^n$ Symbole zu erzeugen
- **Optimierungen des Verfahrens sind möglich**
  - Vermeide Vielfachkopien der R_{ij}^{k-1}
  - Vereinfache Ausdrücke R_{ij}^{k} direkt nach Erzeugung
  - Liefert keine grundsätzliche Verbesserung

Das Pfadanalyseverfahren ist zu kompliziert

- **Konstruktion aller R_{ij}^k ist aufwendig**
  - Es müssen mehr als n^3 Ausdrücke R_{ij}^k erzeugt werden
  - Ausdrücke R_{ij}^k können viermal so groß wie die R_{ij}^{k-1} werden
  - Ohne Vereinfachung der R_{ij}^k sind bis zu $n^3 * 4^n$ Symbole zu erzeugen
- **Optimierungen des Verfahrens sind möglich**
  - Vermeide Vielfachkopien der R_{ij}^{k-1}
  - Vereinfache Ausdrücke R_{ij}^k direkt nach Erzeugung
  - Liefert keine grundsätzliche Verbesserung

Zustandselimination ist erheblich effizienter