Automatentheorie und formale Sprachen
Pumping-Lemma für reguläre Sprachen

Dozentin: Wiebke Petersen

10.6.2009

Finite-state automatons accept regular languages

Theorem (Kleene)

Every language accepted by a DFSA is regular and every regular language is accepted by some DFSA.

Finite-state automatons accept regular languages

Theorem (Kleene)

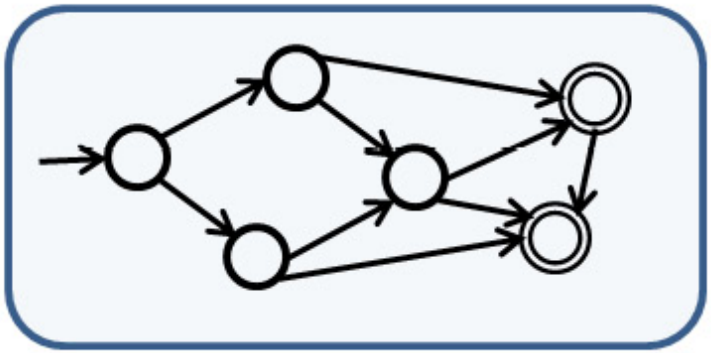
Every language accepted by a DFSA is regular and every regular language is accepted by some DFSA.

proof idea (one direction): Each regular language is accepted by a NDFSA (and therefore by a DFSA):

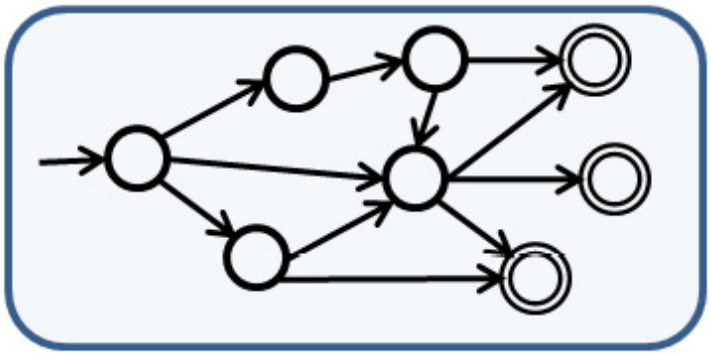
$L(\underline{\emptyset})$ $L(\epsilon)$ $L(a_i)$

Proof of Kleene's theorem (cont.)

If R_1 and R_2 are two regular expressions such that the languages $L(R_1)$ and $L(R_2)$ are accepted by the automatons $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively, then $L(R_1 + R_2)$ is accepted by:



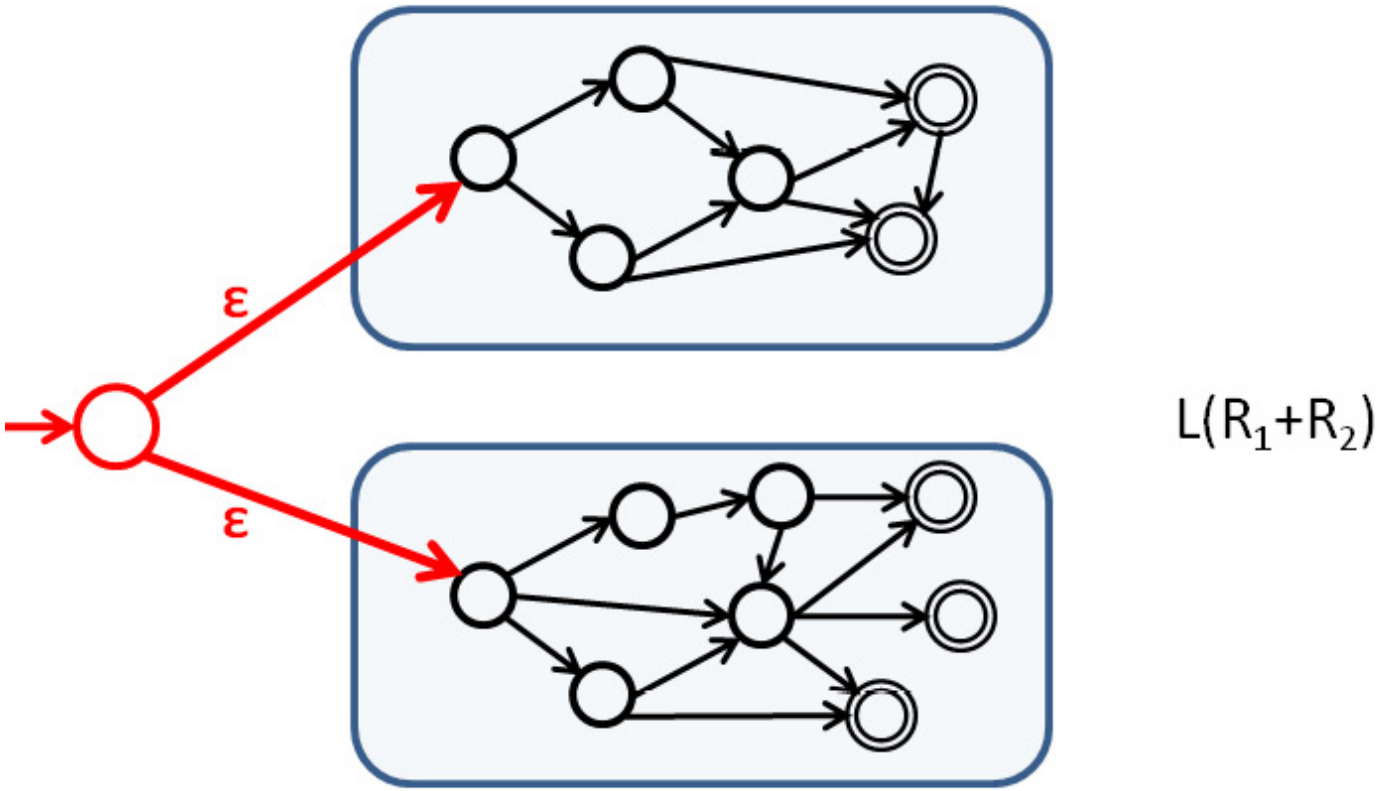
$L(R_1)$



$L(R_2)$

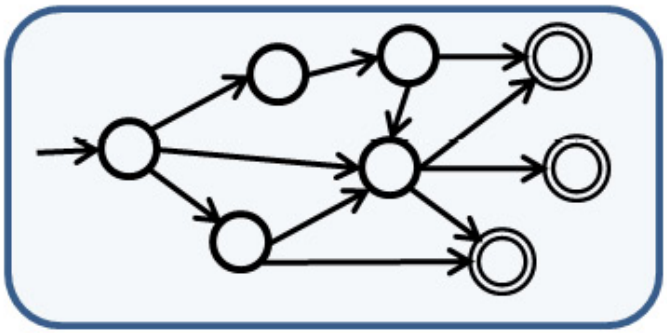
Proof of Kleene's theorem (cont.)

If R_1 and R_2 are two regular expressions such that the languages $L(R_1)$ and $L(R_2)$ are accepted by the automatons $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively, then $L(R_1 + R_2)$ is accepted by:

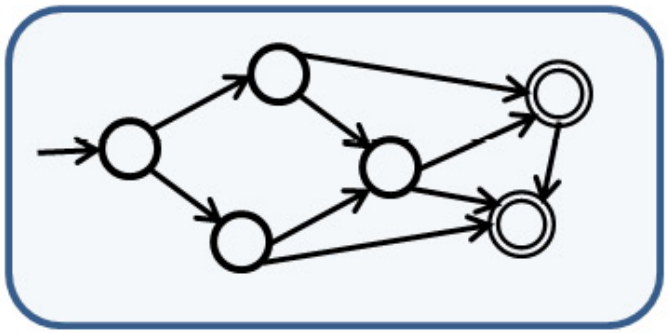


Proof of Kleene's theorem (cont.)

$L(R_1 \bullet R_2)$ is accepted by:



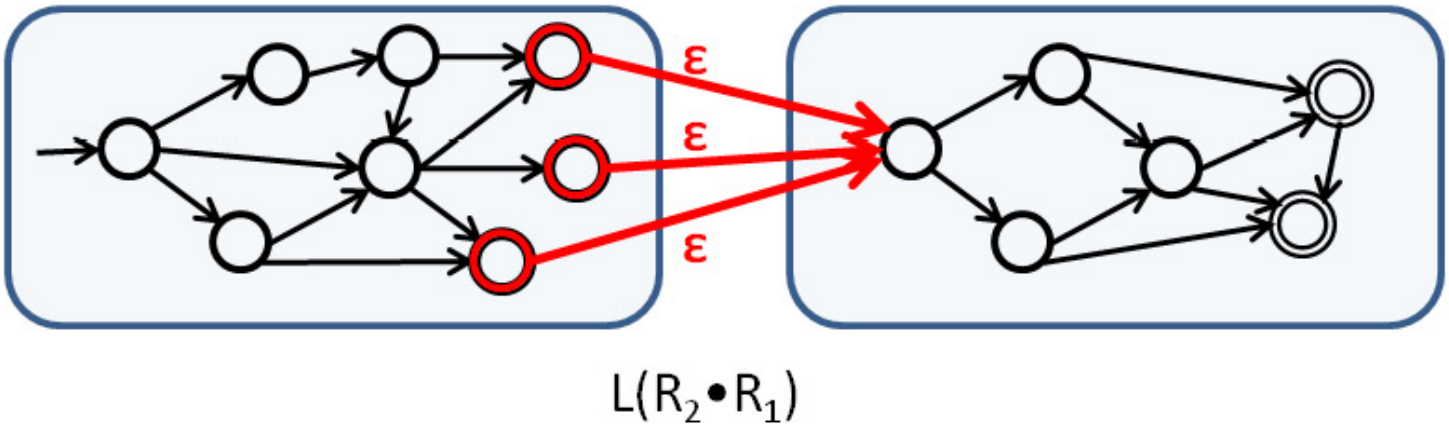
$L(R_2)$



$L(R_1)$

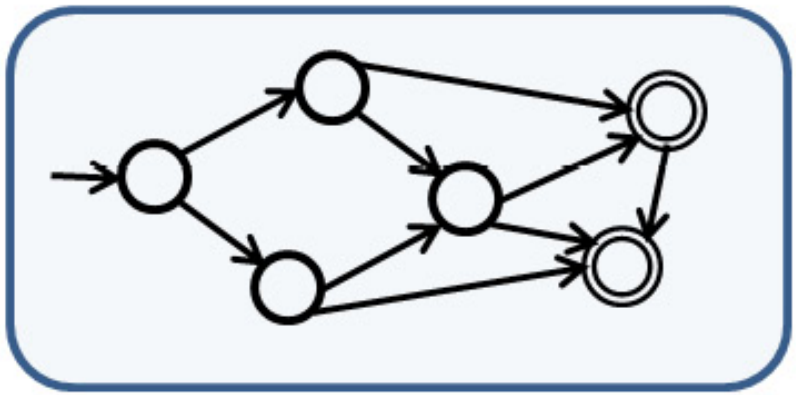
Proof of Kleene's theorem (cont.)

$L(R_1 \bullet R_2)$ is accepted by:



Proof of Kleene's theorem (cont.)

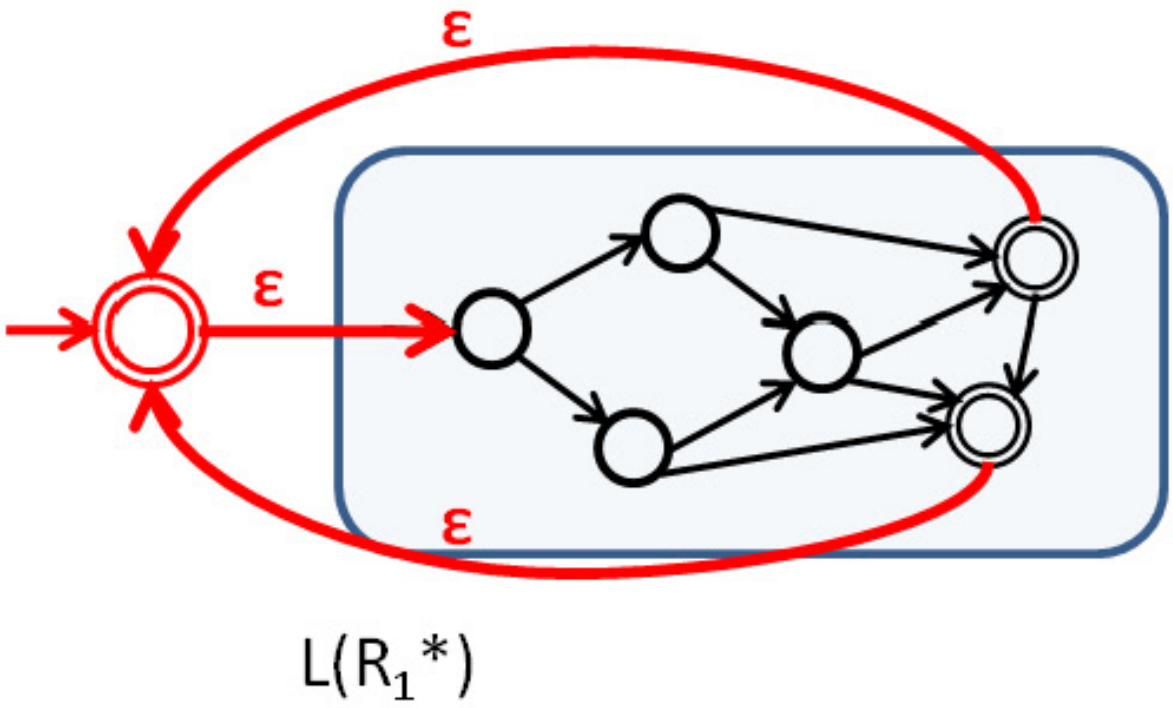
$L(R_1^*)$ is accepted by:



$L(R_1)$

Proof of Kleene's theorem (cont.)

$L(R_1^*)$ is accepted by:



Closure properties of regular languages

Theorem

1. *If L_1 and L_2 are two regular languages, then*
   - *the union of L_1 and L_2 ($L_1 \cup L_2$) is a regular language too.*
   - *the intersection of L_1 and L_2 ($L_1 \cap L_2$) is a regular language too.*
   - *the concatenation of L_1 and L_2 ($L_1 \circ L_2$) is a regular language too.*
2. *The complement of every regular language is a regular language too.*
3. *If L is a regular language, then L^* is a regular language too.*

Pumping lemma for regular languages

Lemma (Pumping-Lemma)

If L is an infinite regular language over Σ, then there exists a number $n \in \mathbb{N}$ such that each word $z \in L$ with $|z| \geq n$ can be decomposed into $z = uvw$ with $u, v, w \in \Sigma^*$ and $v \neq \epsilon$ such that $uv^i w \in L$ for any $i \geq 0$.

proof sketch:

Pumping lemma for regular languages

Lemma (Pumping-Lemma)

If L is an infinite regular language over Σ, then there exists a number $n \in \mathbb{N}$ such that each word $z \in L$ with $|z| \geq n$ can be decomposed into $z = uvw$ with $u, v, w \in \Sigma^*$ and $v \neq \epsilon$ such that $uv^i w \in L$ for any $i \geq 0$.

proof sketch:

- Any regular language is accepted by a DFSA with a finite number n of states.

Pumping lemma for regular languages

Lemma (Pumping-Lemma)

If L is an infinite regular language over Σ, then there exists a number $n \in \mathbb{N}$ such that each word $z \in L$ with $|z| \geq n$ can be decomposed into $z = uvw$ with $u, v, w \in \Sigma^*$ and $v \neq \epsilon$ such that $uv^i w \in L$ for any $i \geq 0$.

proof sketch:

- Any regular language is accepted by a DFSA with a finite number n of states.
- Any infinite language contains a word z which has at least length n ($|z| \geq n$).

Pumping lemma for regular languages

Lemma (Pumping-Lemma)

If L is an infinite regular language over Σ, then there exists a number $n \in \mathbb{N}$ such that each word $z \in L$ with $|z| \geq n$ can be decomposed into $z = uvw$ with $u, v, w \in \Sigma^*$ and $v \neq \epsilon$ such that $uv^i w \in L$ for any $i \geq 0$.

proof sketch:

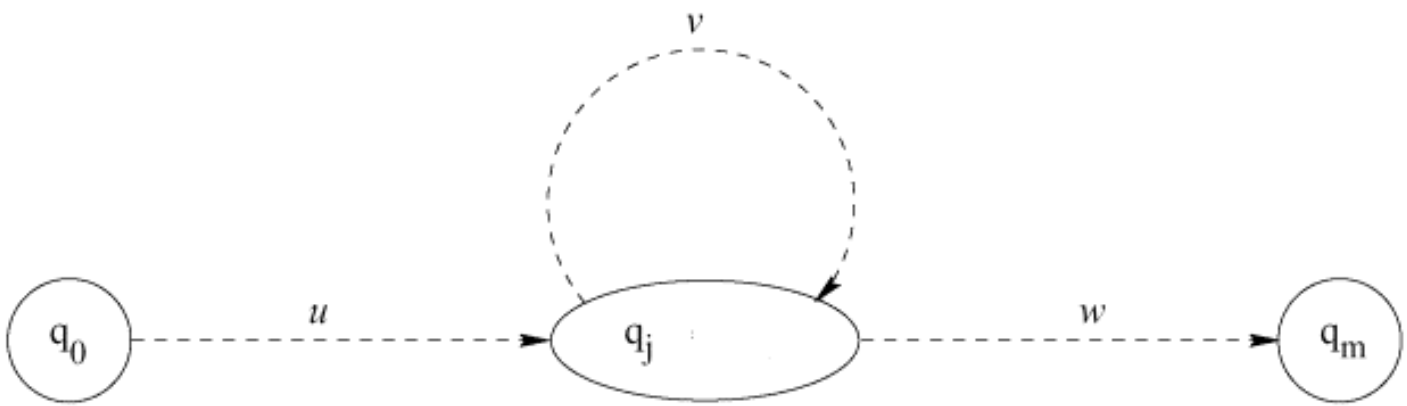
- Any regular language is accepted by a DFSA with a finite number n of states.
- Any infinite language contains a word z which has at least length n ($|z| \geq n$).
- While reading z, the DFSA passes at least one state q_j twice.

Pumping lemma for regular languages (cont.)

Lemma (Pumping-Lemma)

If L is an infinite regular language over Σ, then there exists a number $n \in \mathbb{N}$ such that each word $z \in L$ with $|z| \geq n$ can be decomposed into $z = uvw$ with $u, v, w \in \Sigma^*$ and $v \neq \epsilon$ such that $uv^i w \in L$ for any $i \geq 0$.

proof sketch:



$L = \{a^n b^n : n \geq 0\}$ ist nicht regulär

- $L = \{a^n b^n : n \geq 0\}$:
  L ist unendlich. Wäre L regulär, dann gäbe es $u, v, w \in \{a, b\}^*$, $v \neq \epsilon$ mit $uv^n w \in L$ für alle $n \geq 0$. Für v ergeben sich folgende Fälle:

$L = \{a^n b^n : n \geq 0\}$ ist nicht regulär

- $L = \{a^n b^n : n \geq 0\}$:
  L ist unendlich. Wäre L regulär, dann gäbe es $u, v, w \in \{a, b\}^*$, $v \neq \epsilon$ mit $uv^n w \in L$ für alle $n \geq 0$. Für v ergeben sich folgende Fälle:
  1. angenommen v besteht aus a's und b's, dann gibt es in v^n ($n \geq 2$) b's, die vor a's stehen (Widerspruch),

$L = \{a^n b^n : n \geq 0\}$ ist nicht regulär

- $L = \{a^n b^n : n \geq 0\}$:
  L ist unendlich. Wäre L regulär, dann gäbe es $u, v, w \in \{a, b\}^*$, $v \neq \epsilon$ mit $uv^n w \in L$ für alle $n \geq 0$. Für v ergeben sich folgende Fälle:
  1. angenommen v besteht aus a's und b's, dann gibt es in v^n ($n \geq 2$) b's, die vor a's stehen (Widerspruch),
  2. angenommen v besteht nur aus a's, dann sind alle b's in dem w-Teil; wenn man v aufpumpt, steigt die Zahl der a's ohne daß die der b's steigen würde (Widerspruch).

$L = \{a^n b^n : n \geq 0\}$ ist nicht regulär

- $L = \{a^n b^n : n \geq 0\}$:
  L ist unendlich. Wäre L regulär, dann gäbe es $u, v, w \in \{a, b\}^*$, $v \neq \epsilon$ mit $uv^n w \in L$ für alle $n \geq 0$. Für v ergeben sich folgende Fälle:
  1. angenommen v besteht aus a's und b's, dann gibt es in v^n ($n \geq 2$) b's, die vor a's stehen (Widerspruch),
  2. angenommen v besteht nur aus a's, dann sind alle b's in dem w-Teil; wenn man v aufpumpt, steigt die Zahl der a's ohne daß die der b's steigen würde (Widerspruch).
  3. angenommen v besteht nur aus b's, dann analog zu v besteht nur aus a's,

Hausaufgaben

Welche Aussagen kann man mit Hilfe der Abschlußeigenschaften der regulären Sprachen und dem Pumping-Lemma über die Komplexität folgender formaler Sprachen machen:

1. $L_1 = \{w \in \{a,b\}^* \mid w \text{ enthält eine ungerade Anzahl von } b's\}$.
2. $L_2 = \{w \in \{a,b\}^* \mid w \text{ enthält die gleiche Anzahl von } b's \text{ und } a's\}$.
3. w^R ist das Wort w in umgekehrter Reihenfolge (ww^R ist ein Palindrom).
   $L_3 = \{ww^R \mid w \in \{a,b\}^*\}$.

Natürliche Sprachen sind nicht regulär (1)

1. Der Hund starb.
2. Der Hund, der den Vogel jagte, starb.
3. Der Hund, der den Vogel, der den Wurm fraß, jagte, starb.
4. Der Hund, der den Vogel, der den Wurm, der den Rasen durchquerte, fraß, jagte, starb.
5. Der Hund, der den Vogel, der den Wurm, der den Rasen, der den Garten bedeckte, durchquerte, fraß, jagte, starb.

   ...

Natürliche Sprachen sind nicht regulär (1)

1. Der Hund starb.
2. Der Hund, der den Vogel jagte, starb.
3. Der Hund, der den Vogel, der den Wurm fraß, jagte, starb.
4. Der Hund, der den Vogel, der den Wurm, der den Rasen durchquerte, fraß, jagte, starb.
5. Der Hund, der den Vogel, der den Wurm, der den Rasen, der den Garten bedeckte, durchquerte, fraß, jagte, starb.

   ...

- Allgemeine Form: der Hund (der den *maskulines Nomen*)^n (*transitives Verb*)^n starb.

Natürliche Sprachen sind nicht regulär (2)

- Sei $A = \left\{ \begin{array}{r} \text{der den Hund, der den Vogel, der den Kühlschrank,} \\ \text{der den Wurm, der den Rasen, der den Garten} \end{array} \right\}$ und
- $B = \{$fraß,beschenkte,durchquerte, jagte, liebte, sah,trank$\}$ und
- $w =$ *der Hund* und $v =$ *starb*.

Natürliche Sprachen sind nicht regulär (2)

- Sei $A = \left\{ \begin{array}{r} \text{der den Hund, der den Vogel, der den Kühlschrank,} \\ \text{der den Wurm, der den Rasen, der den Garten} \end{array} \right\}$ und
- $B = \{$fraß,beschenkte,durchquerte, jagte, liebte, sah,trank$\}$ und
- $w =$ *der Hund* und $v =$ *starb*.
- $L(wx^*y^*v)$ mit $x \in A$ und $y \in B$ ist eine reguläre Sprache.

Natürliche Sprachen sind nicht regulär (2)

- Sei $A = \left\{ \begin{array}{r} \text{der den Hund, der den Vogel, der den Kühlschrank,} \\ \text{der den Wurm, der den Rasen, der den Garten} \end{array} \right\}$ und
- $B = \{$fraß,beschenkte,durchquerte, jagte, liebte, sah,trank$\}$ und
- $w =$ *der Hund* und $v =$ *starb*.
- $L(wx^*y^*v)$ mit $x \in A$ und $y \in B$ ist eine reguläre Sprache.
- Deutsch $\cap\, L(wx^*y^*v) = L(wx^ny^nv)$.

Natürliche Sprachen sind nicht regulär (2)

- Sei $A = \left\{ \begin{array}{c} \text{der den Hund, der den Vogel, der den Kühlschrank,} \\ \text{der den Wurm, der den Rasen, der den Garten} \end{array} \right\}$ und
- $B = \{$fraß,beschenkte,durchquerte, jagte, liebte, sah,trank$\}$ und
- $w =$ *der Hund* und $v =$ *starb*.
- $L(wx^*y^*v)$ mit $x \in A$ und $y \in B$ ist eine reguläre Sprache.
- Deutsch $\cap\, L(wx^*y^*v) = L(wx^ny^nv)$.
- Wäre Deutsch regulär, dann müßte auch $L(wx^ny^nv)$ regulär sein, da die Schnittmenge zweier regulärer Sprachen regulär ist (Widerspruch).

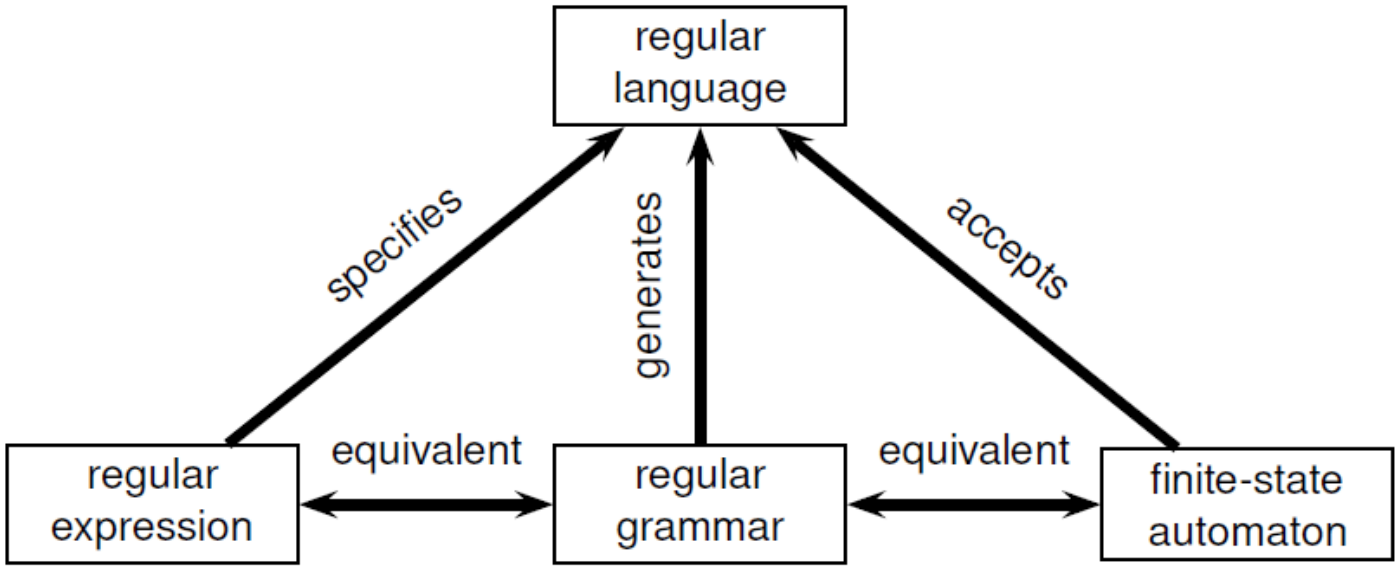
Intuitive rules for regular languages

- L is regular if it is possible to check the membership of a word simply by reading it symbol for symbol while using only a finite stack.

Intuitive rules for regular languages

- L is regular if it is possible to check the membership of a word simply by reading it symbol for symbol while using only a finite stack.
- Finite-state automatons are too weak for:
  - counting in $\mathbb{N}$ ("same number as");
  - recognizing a pattern of arbitrary length ("palindrome");
  - expressions with brackets of arbitrary depth.

Summary: regular languages



Gruppenarbeit

Gruppe 1: Führen sie das in Klabunde beschriebene Verfahren zur Konstruktion eines deterministischen endlichen Automaten aus einem nichtdeterministischen Automaten an einem kleinen Beispiel im Detail durch.

Gruppe 2: Beschreiben sie an 2-3 Beispielen, wie man effizient einen deterministischen Automaten aus einem nichtdeterministischen konstruiert.

Gruppe 3: Beschreiben sie an 2-3 Beispielen, wie man ϵ-Übergänge eliminiert.

Gruppe 4: Beschreiben sie an Beispielen, wie man endliche Automaten zur Konkatenation zweier regulärer Sprachen und zur Vereinigung und zur Schnittmenge zweier Sprachen bildet, und wie man endliche Automaten zum Kleenschen Stern einer regulären Sprache und zum Komplement einer regulären Sprache konstruiert.