



F2 — Automaten und formale Sprachen

Matthias Jantzen

(nach und mit Folienvorlagen von Berndt Farwer)

Fachbereich Informatik

AB „Theoretische Grundlagen der Informatik“ (TGI)

Universität Hamburg

jantzen@informatik.uni-hamburg.de



Themen der heutigen Vorlesung

Wir werden uns beschäftigen mit:

- Eigenschaften von Relationen
 - Reflexivität, Irreflexivität
 - Symmetrie, Asymmetrie, Antisymmetrie
 - Transitivität



Themen der heutigen Vorlesung

Wir werden uns beschäftigen mit:

- Eigenschaften von Relationen
 - Reflexivität, Irreflexivität
 - Symmetrie, Asymmetrie, Antisymmetrie
 - Transitivität
- Ordnungen
 - auf beliebigen Mengen
 - speziell auf Wortmengen



Themen der heutigen Vorlesung

Wir werden uns beschäftigen mit:

- Eigenschaften von Relationen
 - Reflexivität, Irreflexivität
 - Symmetrie, Asymmetrie, Antisymmetrie
 - Transitivität
- Ordnungen
 - auf beliebigen Mengen
 - speziell auf Wortmengen
- formale Sprachen



Themen der heutigen Vorlesung

Wir werden uns beschäftigen mit:

- Eigenschaften von Relationen
 - Reflexivität, Irreflexivität
 - Symmetrie, Asymmetrie, Antisymmetrie
 - Transitivität
- Ordnungen
 - auf beliebigen Mengen
 - speziell auf Wortmengen
- formale Sprachen
- endliche Automaten



Reflexivität

- Sei $R \subseteq A \times A$, dann ist R **reflexiv** gdw.
 $\forall x \in A : (x, x) \in R$ gilt.



Reflexivität

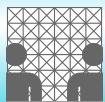
- Sei $R \subseteq A \times A$, dann ist R **reflexiv** gdw.
 $\forall x \in A : (x, x) \in R$ gilt.
- Typisches **Beispiel**: Erreichbarkeitsrelationen



-

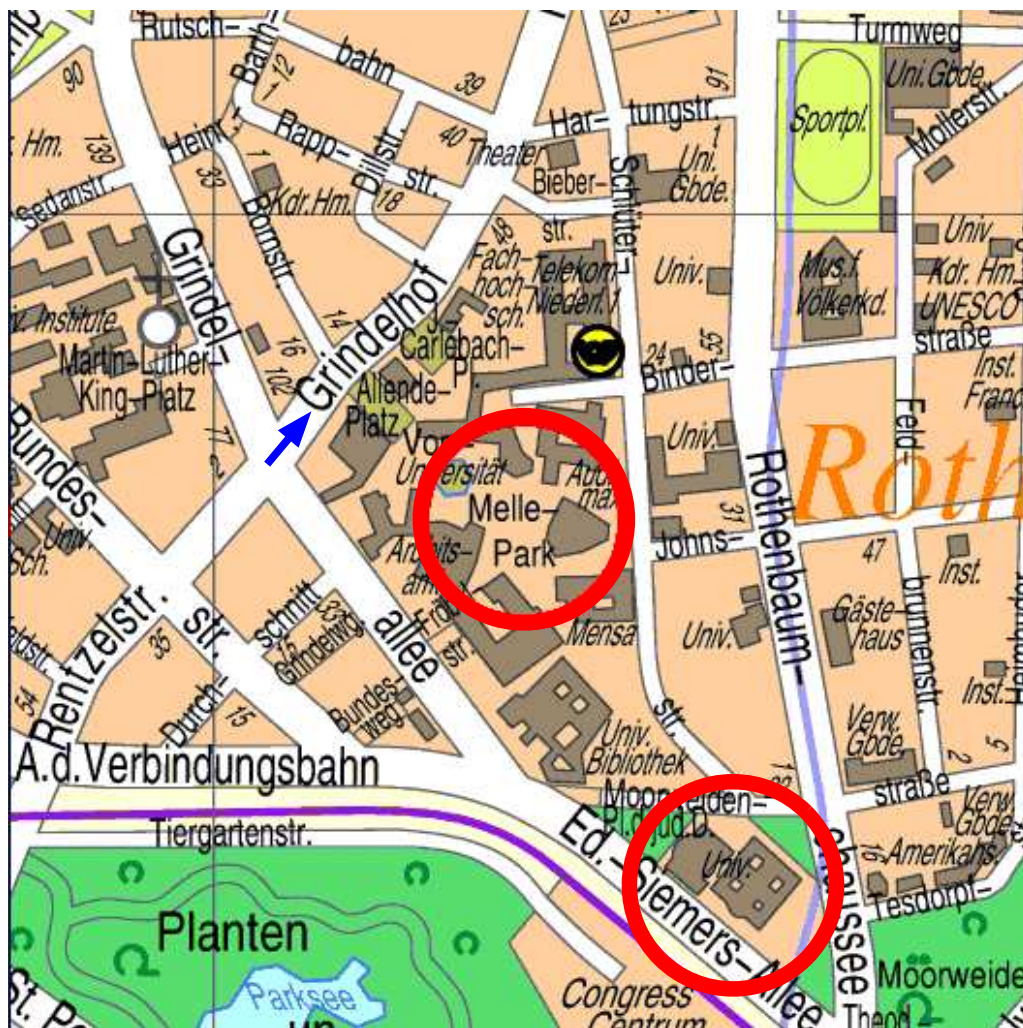


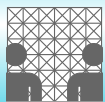
-



Reflexivität

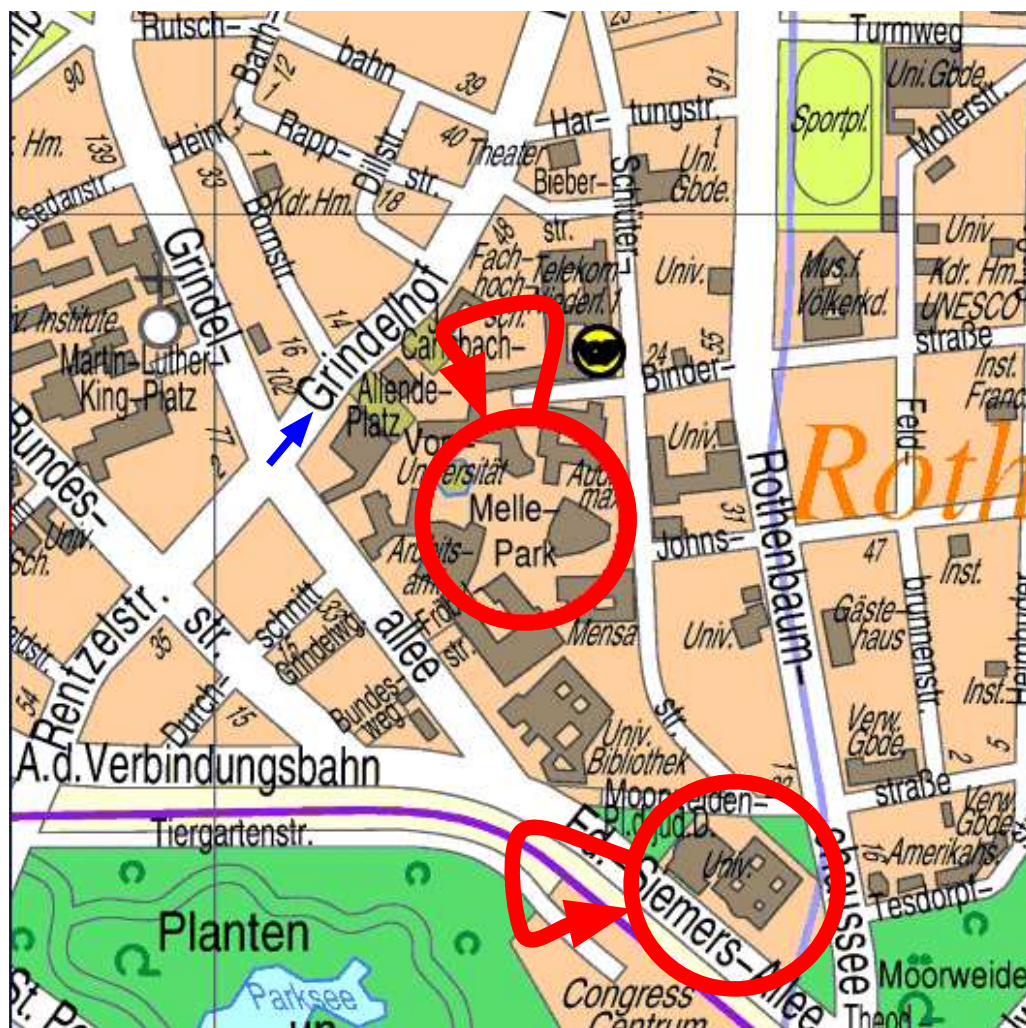
- Sei $R \subseteq A \times A$, dann ist R **reflexiv** gdw.
 $\forall x \in A : (x, x) \in R$ gilt.
- Typisches **Beispiel**: Erreichbarkeitsrelationen





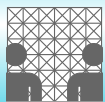
Reflexivität

- Sei $R \subseteq A \times A$, dann ist R **reflexiv** gdw.
 $\forall x \in A : (x, x) \in R$ gilt.
- Typisches **Beispiel**: Erreichbarkeitsrelationen



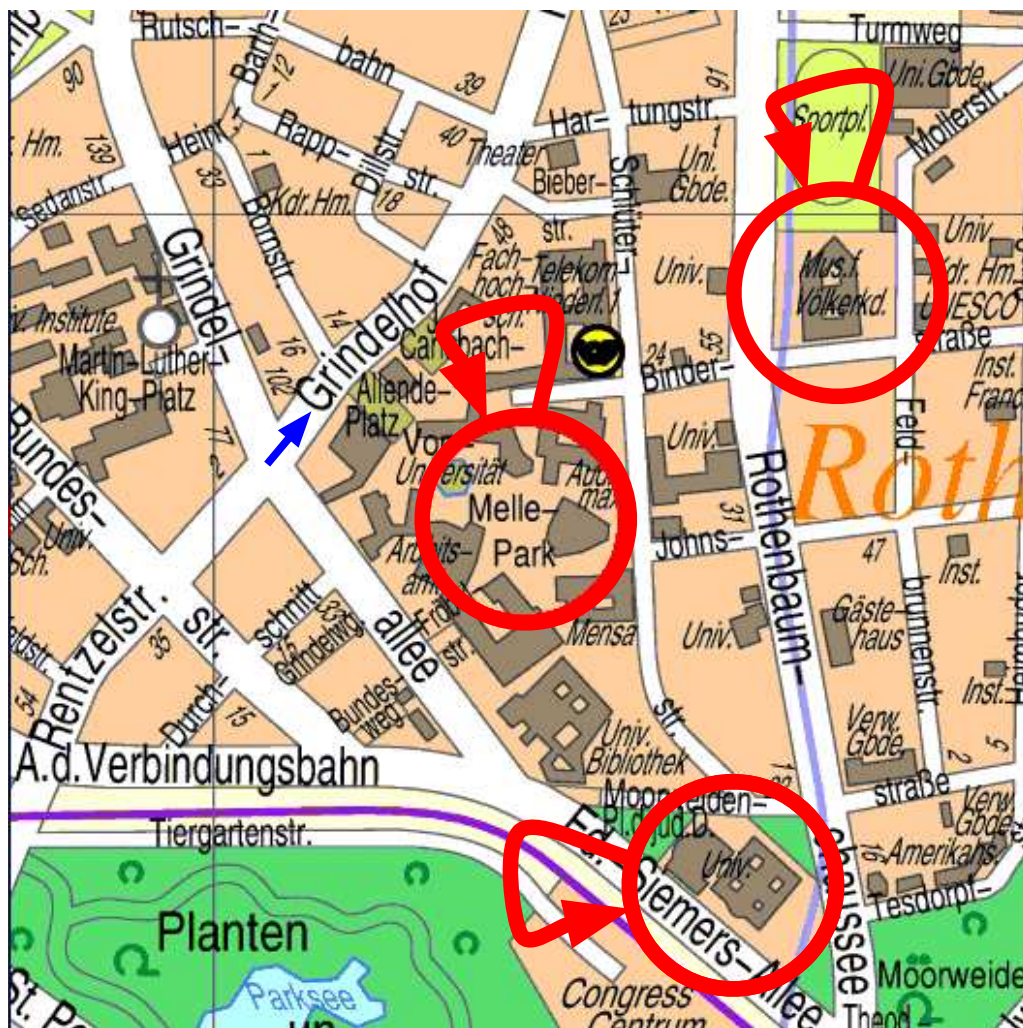


-



Reflexivität

- Sei $R \subseteq A \times A$, dann ist R **reflexiv** gdw.
 $\forall x \in A : (x, x) \in R$ gilt.
- Typisches **Beispiel**: Erreichbarkeitsrelationen





Transitivität

- Sei $R \subseteq A \times A$, dann ist R **transitiv** gdw.
 $\forall x, y, z \in A : (x, y) \in R \wedge (y, z) \in R \rightarrow (x, z) \in R$
gilt.

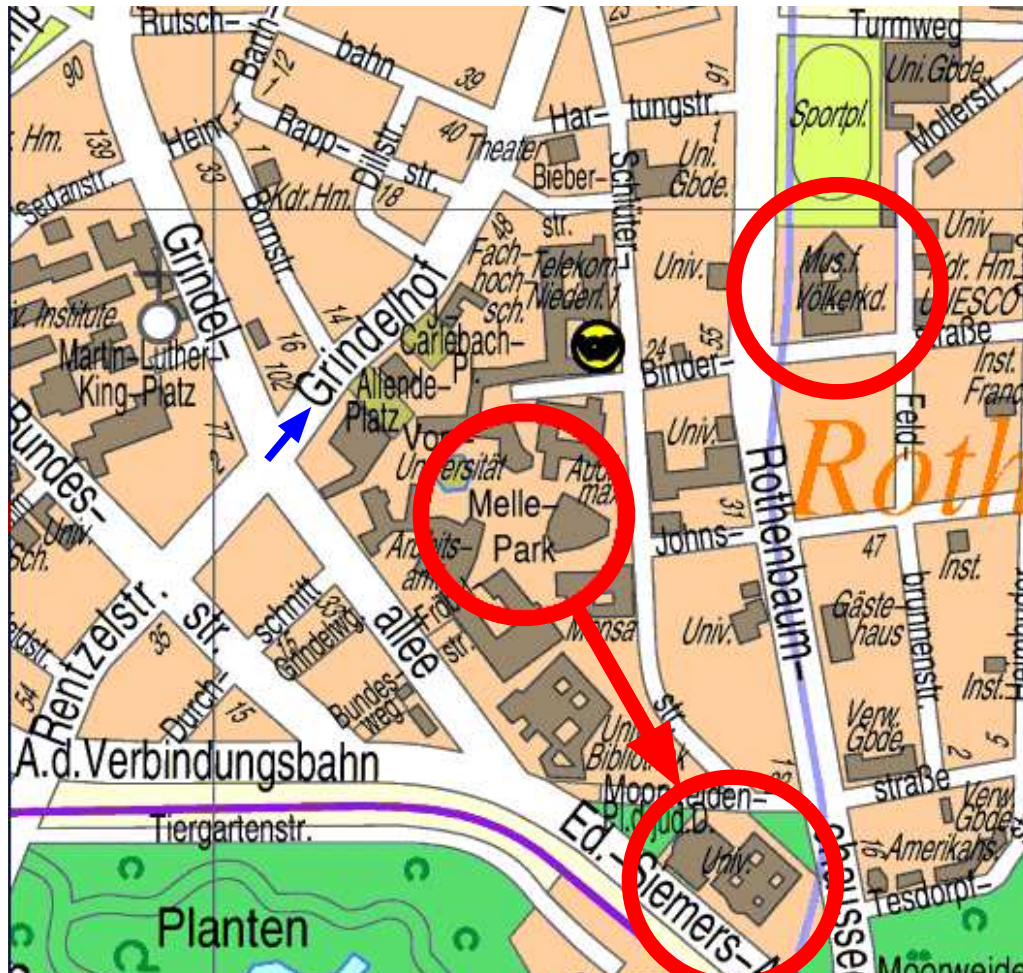


-



Transitivität

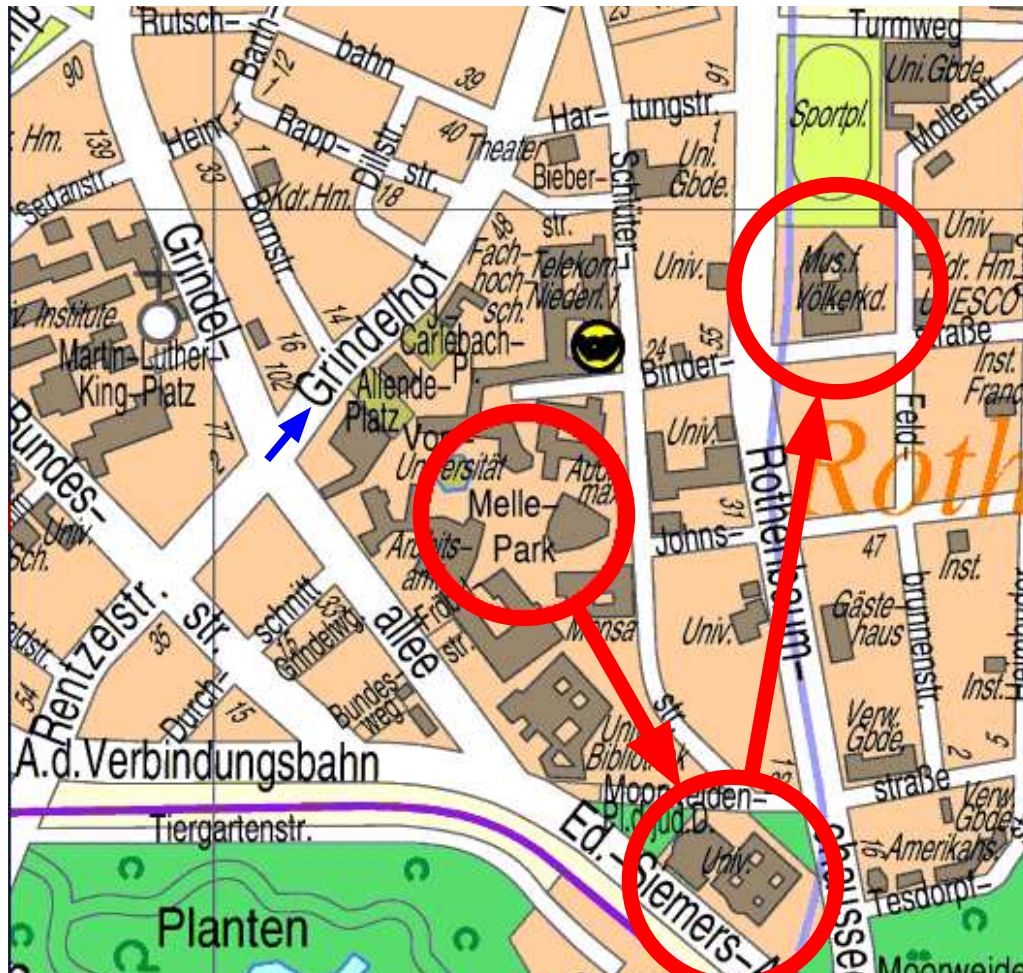
- Sei $R \subseteq A \times A$, dann ist R **transitiv** gdw.
 $\forall x, y, z \in A : (x, y) \in R \wedge (y, z) \in R \rightarrow (x, z) \in R$ gilt.
- Beispiel:** unsere Erreichbarkeitsrelation





Transitivität

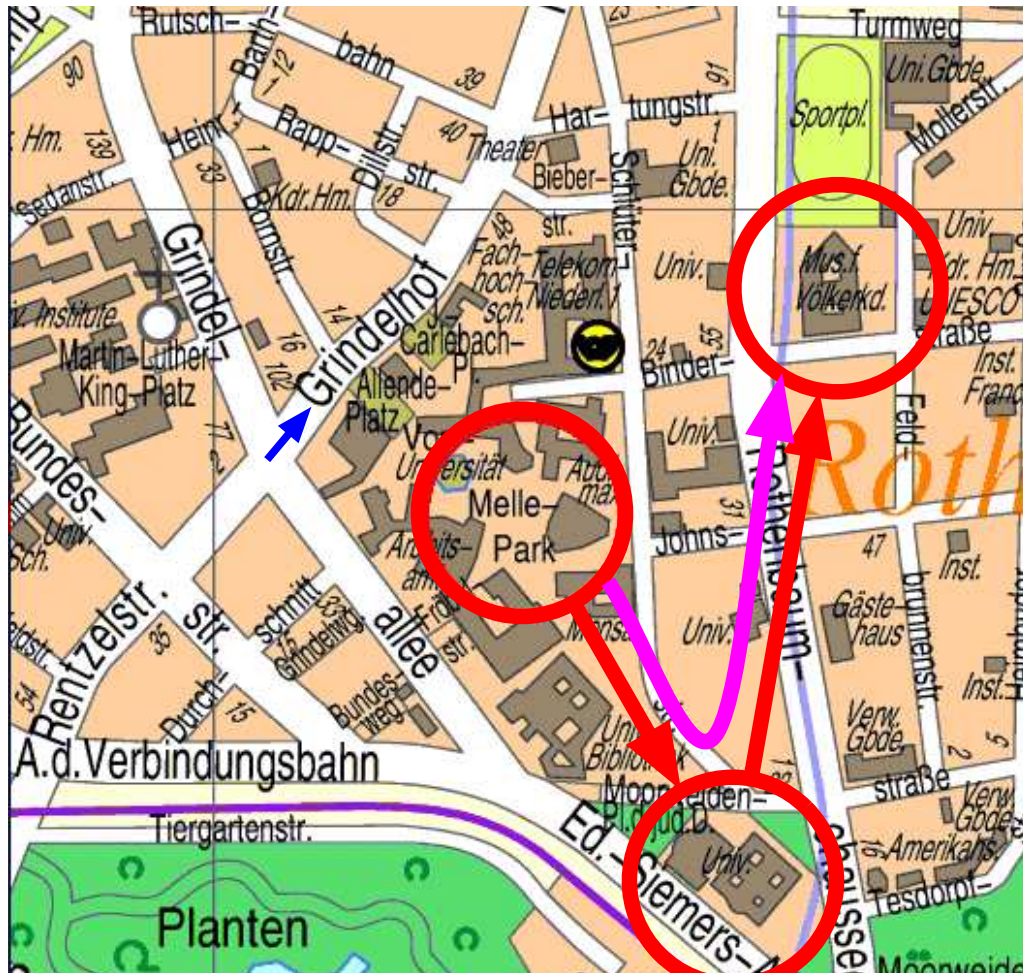
- Sei $R \subseteq A \times A$, dann ist R **transitiv** gdw.
 $\forall x, y, z \in A : (x, y) \in R \wedge (y, z) \in R \rightarrow (x, z) \in R$ gilt.
- Beispiel:** unsere Erreichbarkeitsrelation





Transitivität

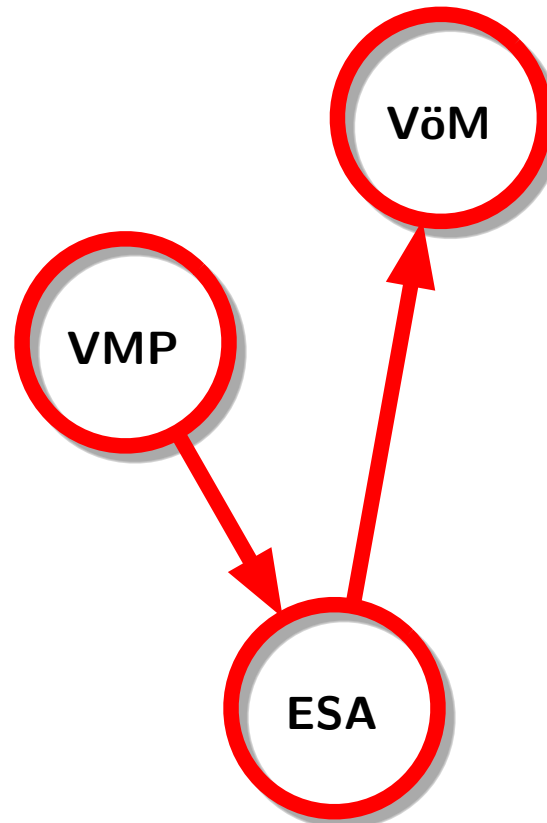
- Sei $R \subseteq A \times A$, dann ist R **transitiv** gdw.
 $\forall x, y, z \in A : (x, y) \in R \wedge (y, z) \in R \rightarrow (x, z) \in R$ gilt.
- Beispiel:** unsere Erreichbarkeitsrelation





Transitivität

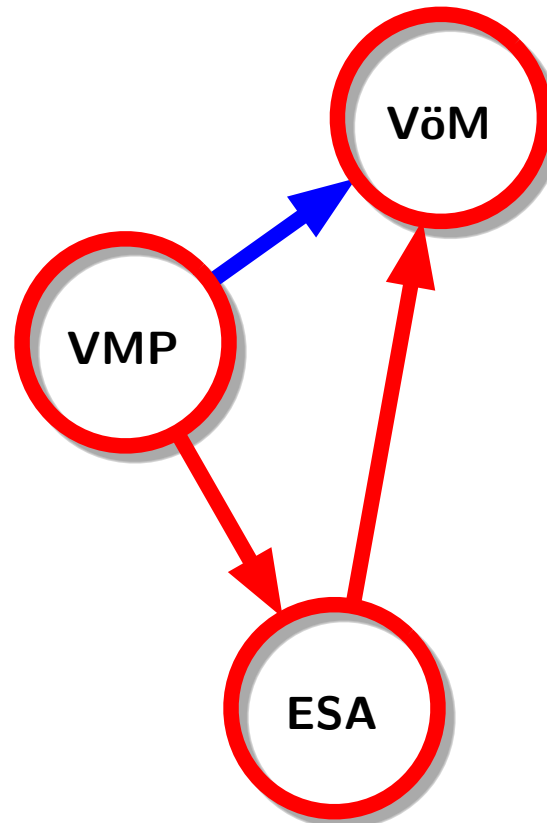
- Fortsetzung des Beispiels:





Transitivität

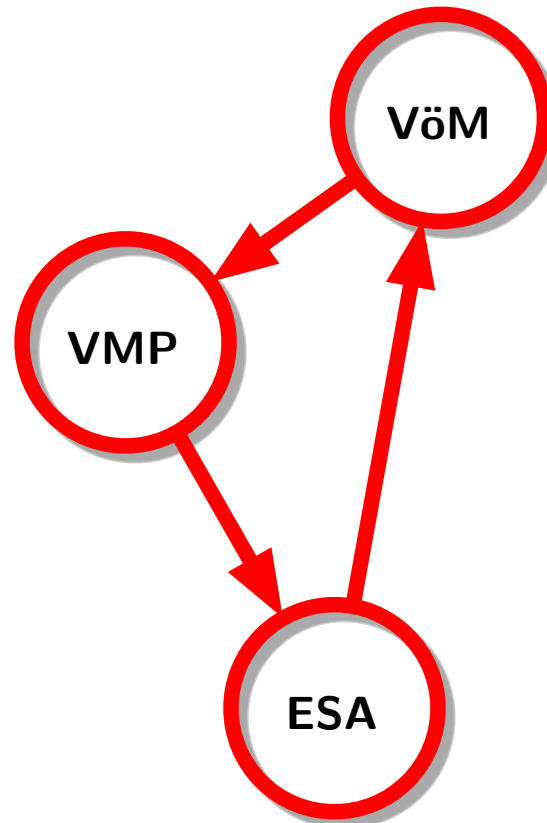
- Fortsetzung des Beispiels:





Transitivität

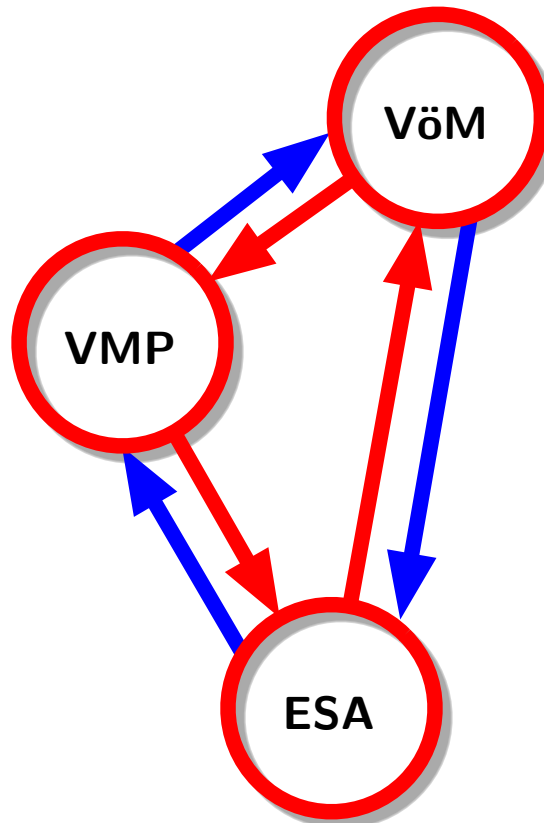
- Fortsetzung des Beispiels:

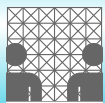




Transitivität

- Fortsetzung des Beispiels: (zusätzliche Kante)

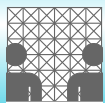




Transitiver Abschluss

- Sei $R \subseteq A \times A$ eine Relation auf der Menge A .
Dann seien R^+ , der **transitive Abschluss** (auch **transitive Hülle**), und R^* , der **reflexive, transitive Abschluss**, von R wie folgt erklärt:

$$R^+ := \bigcup_{i \geq 1} R_i \text{ mit } R_1 := R \quad \text{und} \quad R_{i+1} := R_i \cdot R.$$



Transitiver Abschluss

- Sei $R \subseteq A \times A$ eine Relation auf der Menge A .
Dann seien R^+ , der **transitive Abschluss** (auch **transitive Hülle**), und R^* , der **reflexive, transitive Abschluss**, von R wie folgt erklärt:

$$R^+ := \bigcup_{i \geq 1} R_i \text{ mit } R_1 := R \text{ und } R_{i+1} := R_i \cdot R.$$

- Für eine Teilmenge $M \subseteq H$ einer Halbgruppe (H, \odot) seien der **transitive Abschluss** M^+ sowie der **transitive und reflexive Abschluss** M^*

$$M^+ := \bigcup_{i \geq 1} M^i \text{ mit } M^1 := M \text{ und } M^{i+1} := M^i \cdot M$$

$$M^* := M^+ \cup \{e_H\}$$



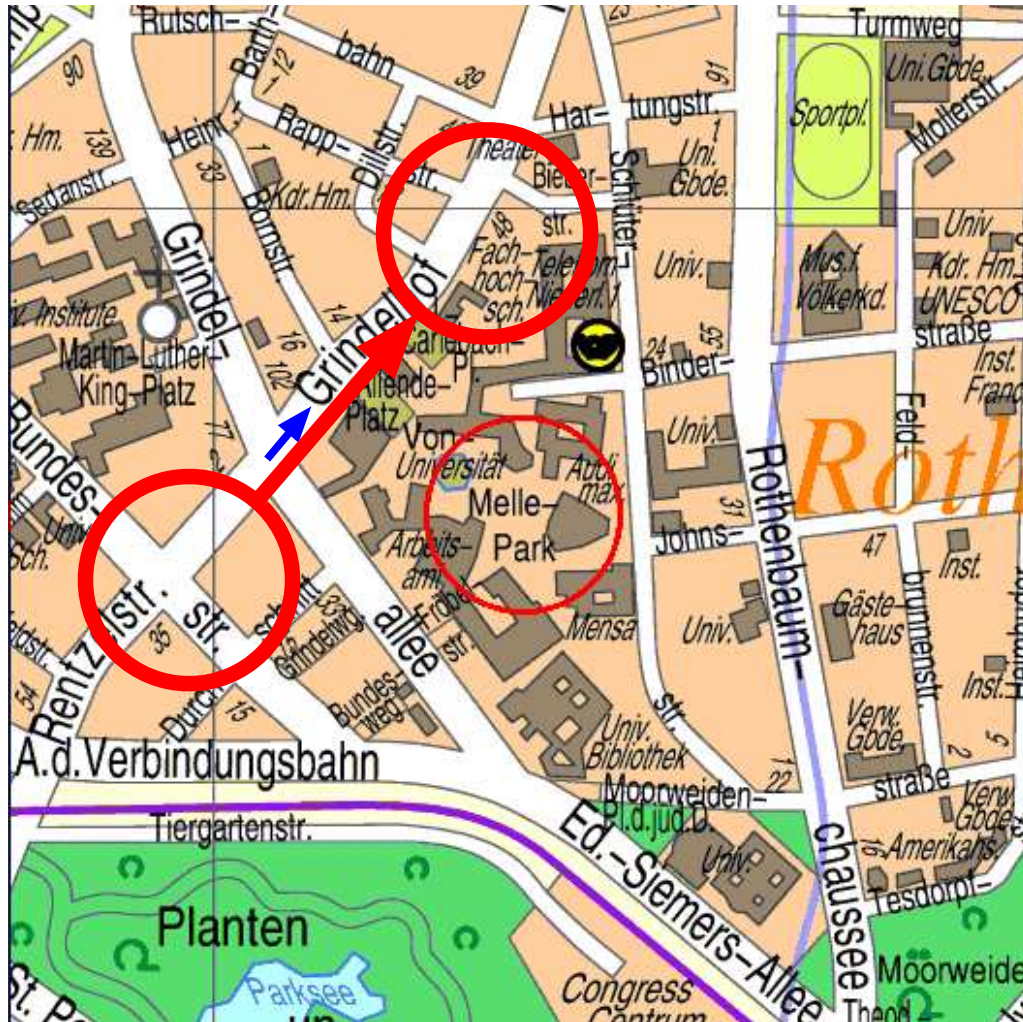
Symmetrie

- Sei $R \subseteq A \times A$, dann ist R **symmetrisch** gdw.
 $\forall x, y \in A : (x, y) \in R \rightarrow (y, x) \in R$ gilt.



Symmetrie

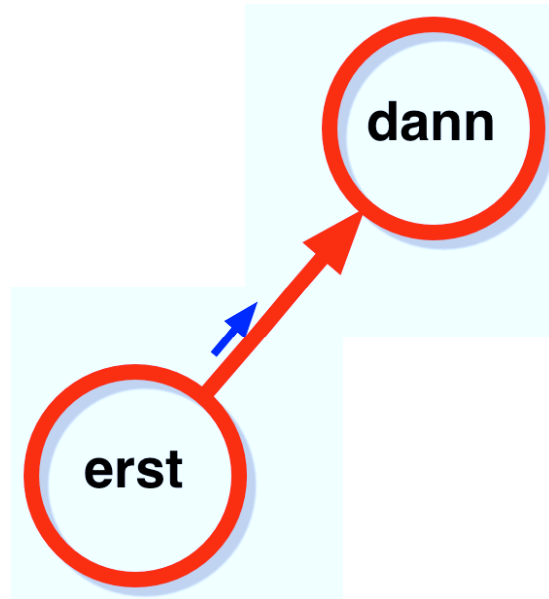
- Sei $R \subseteq A \times A$, dann ist R **symmetrisch** gdw.
 $\forall x, y \in A : (x, y) \in R \rightarrow (y, x) \in R$ gilt.



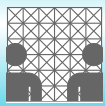


Symmetrie

- Sei $R \subseteq A \times A$, dann ist R **symmetrisch** gdw.
 $\forall x, y \in A : (x, y) \in R \rightarrow (y, x) \in R$ gilt.



Erreichbarkeitsrelationen müssen nicht symmetrisch sein!



Weitere Eigenschaften

Sei $R \subseteq A \times A$ eine Relation auf A . R heißt

- **asymmetrisch** gdw. aus $(a, b) \in R$ immer $(b, a) \notin R$ folgt.

Eine asymmetrische Relation ist also immer irreflexiv, aber eine irreflexive Relation braucht nicht asymmetrisch zu sein!



Weitere Eigenschaften

Sei $R \subseteq A \times A$ eine Relation auf A . R heißt

- **asymmetrisch** gdw. aus $(a, b) \in R$ immer $(b, a) \notin R$ folgt.

Eine asymmetrische Relation ist also immer irreflexiv, aber eine irreflexive Relation braucht nicht asymmetrisch zu sein!

- **antisymmetrisch**, falls für alle $a, b \in A$ gilt:

$$(a, b) \in R \wedge (b, a) \in R \rightarrow a = b$$



Weitere Eigenschaften

Sei $R \subseteq A \times A$ eine Relation auf A . R heißt

- **asymmetrisch** gdw. aus $(a, b) \in R$ immer $(b, a) \notin R$ folgt.

Eine asymmetrische Relation ist also immer irreflexiv, aber eine irreflexive Relation braucht nicht asymmetrisch zu sein!

- **antisymmetrisch**, falls für alle $a, b \in A$ gilt:

$$(a, b) \in R \wedge (b, a) \in R \rightarrow a = b$$

- **linear** genau dann, wenn für beliebige $a, b \in A$ genau eine der drei Bedingungen:
 $a = b$ oder $(a, b) \in R$ oder $(b, a) \in R$ erfüllt ist.



Begriffe

- Keine zwei der folgenden Eigenschaften sind identisch:



Begriffe

- Keine zwei der folgenden Eigenschaften sind identisch:
 - Die Relation R ist **symmetrisch**.



Begriffe

- Keine zwei der folgenden Eigenschaften sind identisch:
 - Die Relation R ist **symmetrisch**.
 - Die Relation R ist **asymmetrisch**.



Begriffe

- Keine zwei der folgenden Eigenschaften sind identisch:
 - Die Relation R ist **symmetrisch**.
 - Die Relation R ist **asymmetrisch**.
 - Die Relation R ist **antisymmetrisch**.



Begriffe

- Keine zwei der folgenden Eigenschaften sind identisch:
 - Die Relation R ist **symmetrisch**.
 - Die Relation R ist **asymmetrisch**.
 - Die Relation R ist **antisymmetrisch**.
 - Die Relation R ist **nicht symmetrisch**.



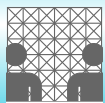
Begriffe

- Keine zwei der folgenden Eigenschaften sind identisch:
 - Die Relation R ist **symmetrisch**.
 - Die Relation R ist **asymmetrisch**.
 - Die Relation R ist **antisymmetrisch**.
 - Die Relation R ist **nicht symmetrisch**.
- Beispiele dazu im Skript und in den Übungen!



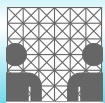
Formale Sprache

- Eine **formale Sprache** über dem Alphabet Σ ist eine Menge von Wörtern aus Σ^* .
 - *Zur Erinnerung:* Ein **Alphabet** ist eine endliche Menge von Symbolen.



Formale Sprache

- Eine **formale Sprache** über dem Alphabet Σ ist eine Menge von Wörtern aus Σ^* .
 - *Zur Erinnerung:* Ein **Alphabet** ist eine endliche Menge von Symbolen.
- Statt *formale Sprache* verwenden wir auch einfach *Sprache*.



Formale Sprache

- Eine **formale Sprache** über dem Alphabet Σ ist eine Menge von Wörtern aus Σ^* .
 - *Zur Erinnerung:* Ein **Alphabet** ist eine endliche Menge von Symbolen.
- Statt *formale Sprache* verwenden wir auch einfach *Sprache*.
- **Beispiele:**
 - $\{\lambda, a, b, aa, ab, ba, bb\}$ ist eine Sprache über dem Alphabet $\{a, b\}$.



Formale Sprache

- Eine **formale Sprache** über dem Alphabet Σ ist eine Menge von Wörtern aus Σ^* .
 - *Zur Erinnerung:* Ein **Alphabet** ist eine endliche Menge von Symbolen.
- Statt *formale Sprache* verwenden wir auch einfach *Sprache*.
- **Beispiele:**
 - $\{\lambda, a, b, aa, ab, ba, bb\}$ ist eine Sprache über dem Alphabet $\{a, b\}$.
 - $\{w \in \{a, b\}^* \mid w = av \wedge v \in \{a, b\}^*\}$ ist ebenfalls eine Sprache über dem Alphabet $\{a, b\}$.



Beschreibung formaler Sprachen

- Eine Sprache muss durch eine **endliche Repräsentation** dargestellt werden.



Beschreibung formaler Sprachen

- Eine Sprache muss durch eine **endliche Repräsentation** dargestellt werden.
- Die Beschreibung einer Sprache muss **eindeutig** sein.



Beschreibung formaler Sprachen

- Eine Sprache muss durch eine **endliche Repräsentation** dargestellt werden.
- Die Beschreibung einer Sprache muss **eindeutig** sein.
- Es kann **unterschiedliche Beschreibungen** für dieselbe Sprache geben.



Beschreibung formaler Sprachen

- Eine Sprache muss durch eine **endliche Repräsentation** dargestellt werden.
- Die Beschreibung einer Sprache muss **eindeutig** sein.
- Es kann **unterschiedliche Beschreibungen** für dieselbe Sprache geben.
- Wir werden dazu kennenlernen:
 - endliche Automaten
 - rationale Ausdrücke
 - rechtslineare Grammatiken



Beschreibung formaler Sprachen

- Eine Sprache muss durch eine **endliche Repräsentation** dargestellt werden.
- Die Beschreibung einer Sprache muss **eindeutig** sein.
- Es kann **unterschiedliche Beschreibungen** für dieselbe Sprache geben.
- Wir werden dazu kennenlernen:
 - **endliche Automaten**
 - rationale Ausdrücke
 - rechtslineare Grammatiken



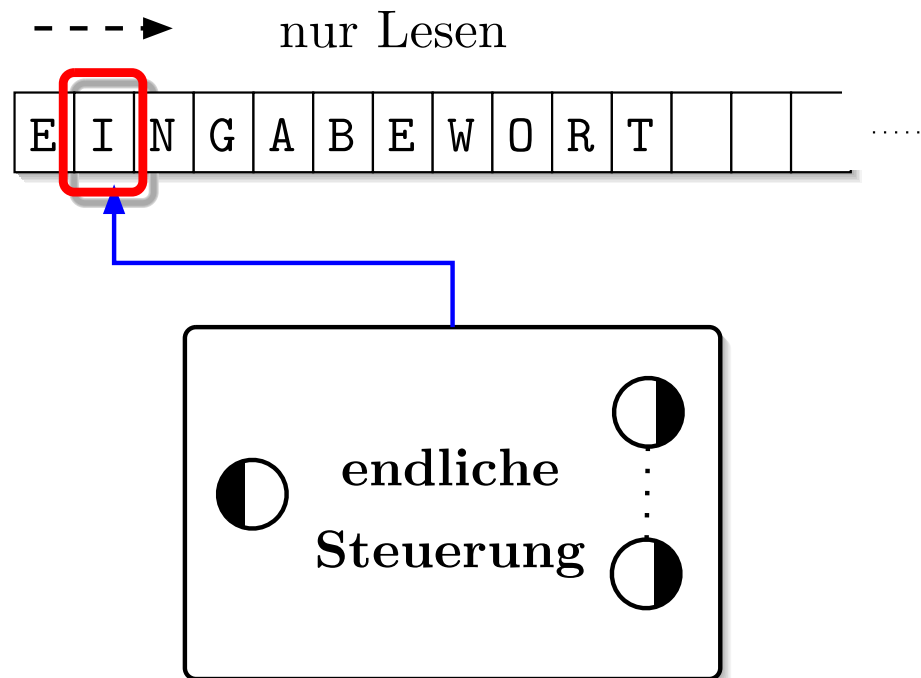
endlicher Automat (DFA)

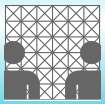
- Zustandsübergangsdiagramme werden zu **deterministischen endlichen Automaten** erweitert:



endlicher Automat (DFA)

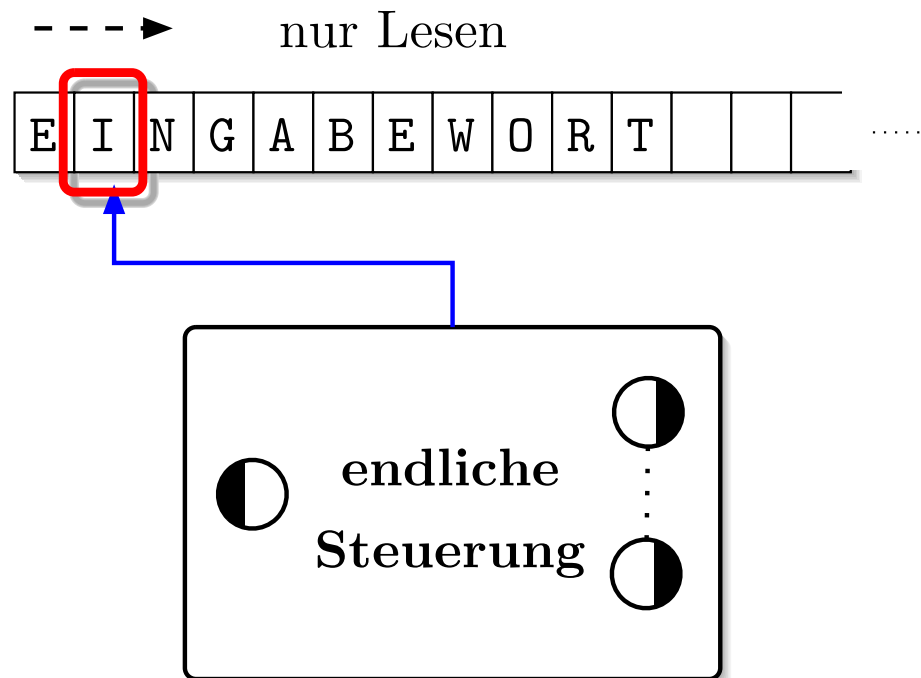
- Zustandsübergangsdiagramme werden zu **deterministischen endlichen Automaten** erweitert:





endlicher Automat (DFA)

- Zustandsübergangsdiagramme werden zu **deterministischen endlichen Automaten** erweitert:

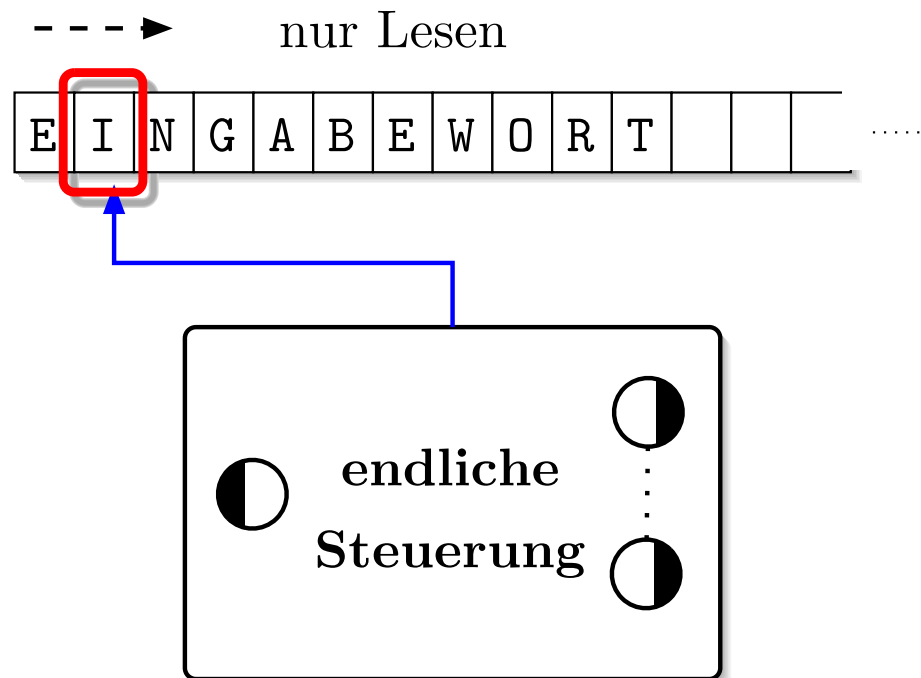


- Start- und Endzustände



endlicher Automat (DFA)

- Zustandsübergangsdiagramme werden zu **deterministischen endlichen Automaten** erweitert:



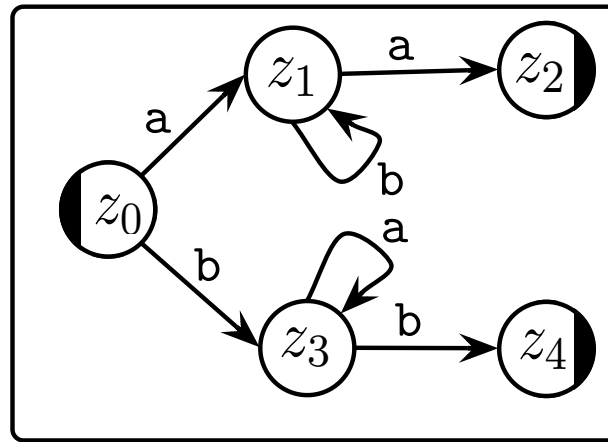
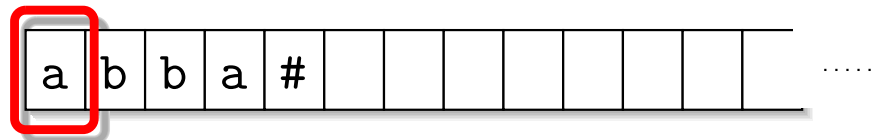
- Start- und Endzustände
- endliche Steuerung = konstanter Speicher

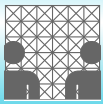


Funktionsweise des DFA

- Bei Eingabe eines Wortes wird durch die endliche Kontrolle das Verhalten des DFA bestimmt:

Eingabewort:

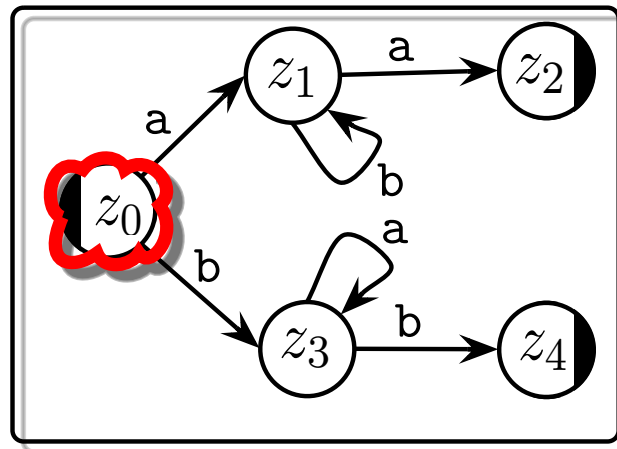
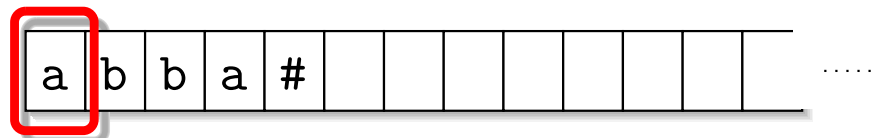


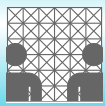


Funktionsweise des DFA

- Bei Eingabe eines Wortes wird durch die endliche Kontrolle das Verhalten des DFA bestimmt:

Eingabewort:

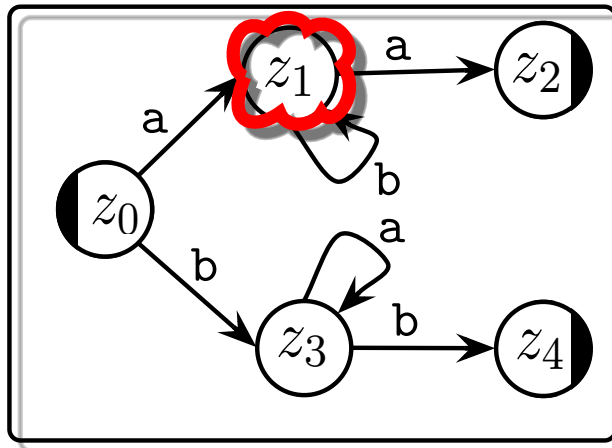
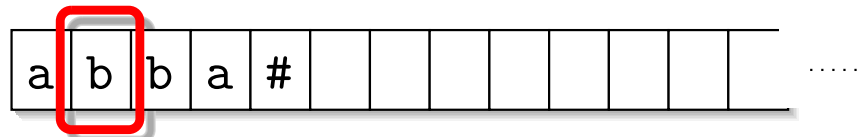




Funktionsweise des DFA

- Bei Eingabe eines Wortes wird durch die endliche Kontrolle das Verhalten des DFA bestimmt:

Eingabewort:

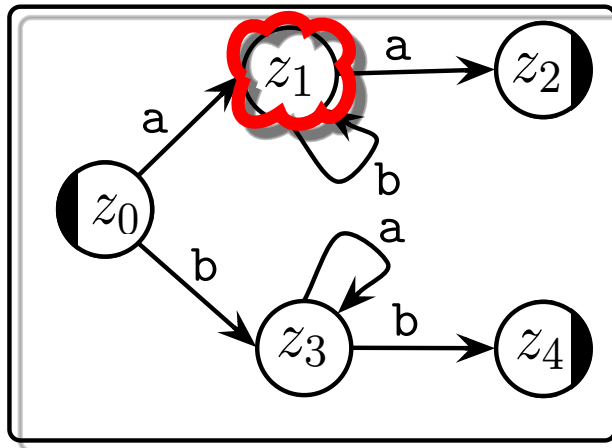
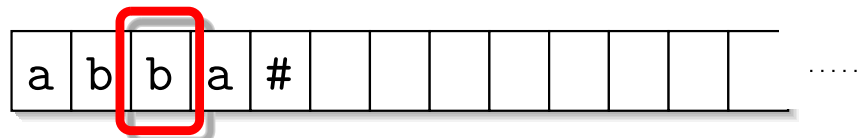




Funktionsweise des DFA

- Bei Eingabe eines Wortes wird durch die endliche Kontrolle das Verhalten des DFA bestimmt:

Eingabewort:

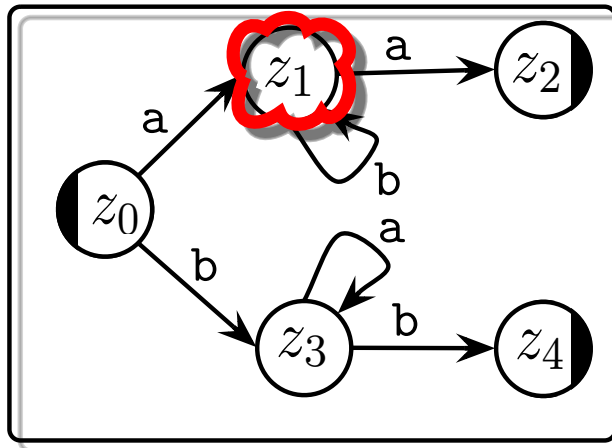
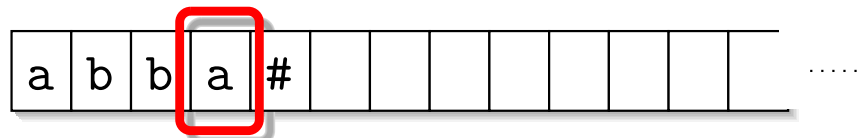


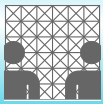


Funktionsweise des DFA

- Bei Eingabe eines Wortes wird durch die endliche Kontrolle das Verhalten des DFA bestimmt:

Eingabewort:

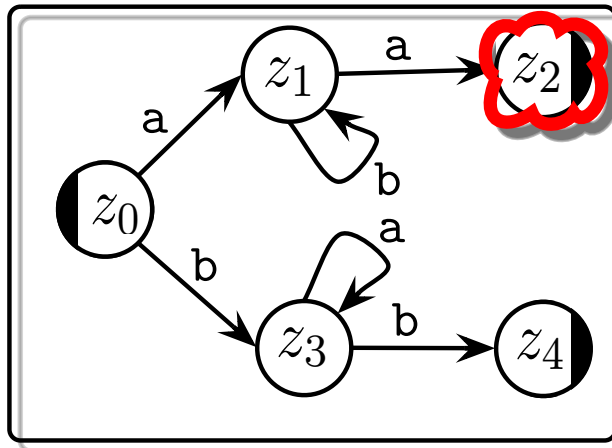
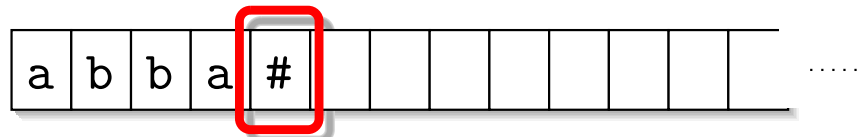




Funktionsweise des DFA

- Bei Eingabe eines Wortes wird durch die endliche Kontrolle das Verhalten des DFA bestimmt:

Eingabewort:



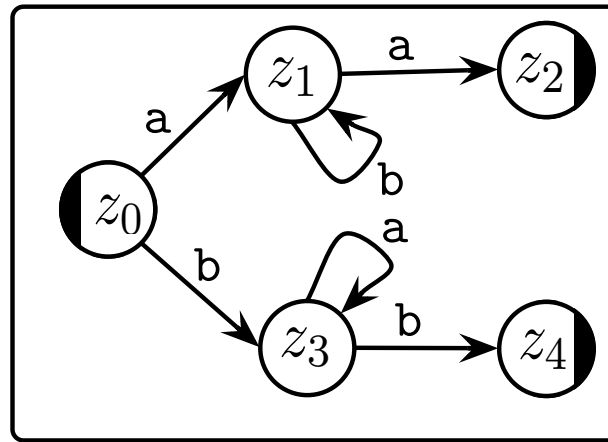
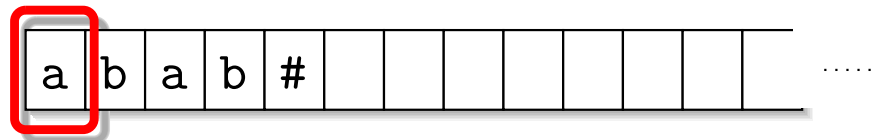
Die Rechnung des Automaten endet in z_2 , einem Endzustand. Die Eingabe ist komplett gelesen, d.h. der Lesekopf steht hinter der Eingabe.



Funktionsweise des DFA

- Eine andere Eingabe bewirkt auch ein anderes Verhalten:

Eingabewort:

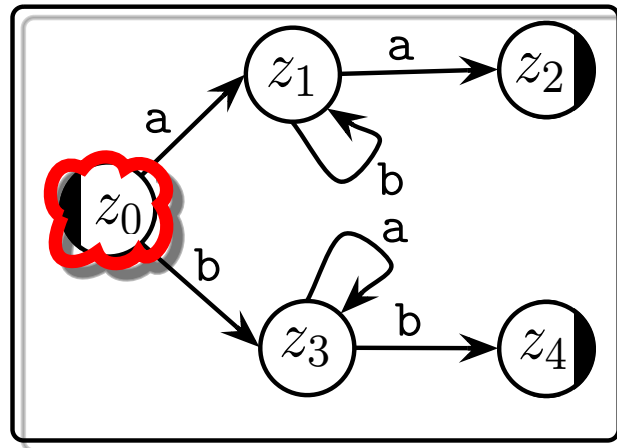
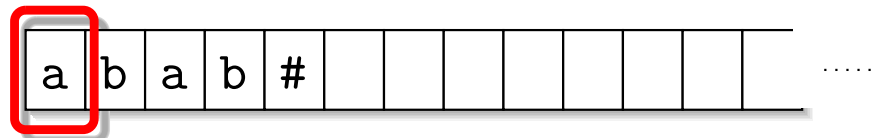




Funktionsweise des DFA

- Eine andere Eingabe bewirkt auch ein anderes Verhalten:

Eingabewort:

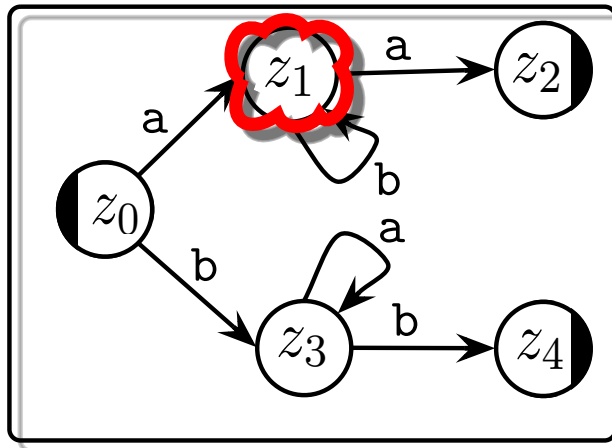
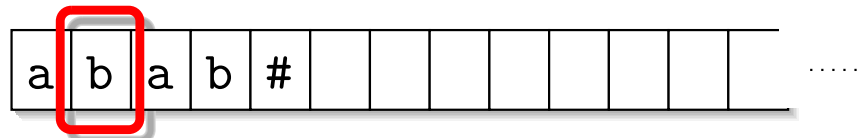


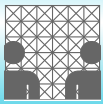


Funktionsweise des DFA

- Eine andere Eingabe bewirkt auch ein anderes Verhalten:

Eingabewort:

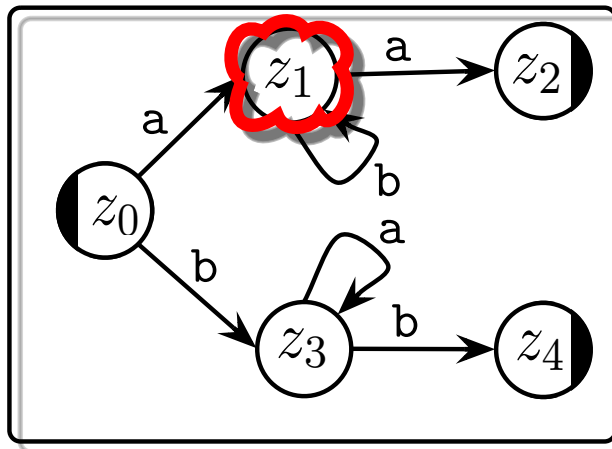
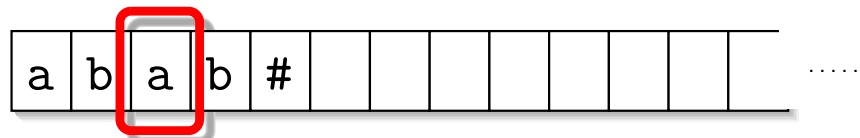




Funktionsweise des DFA

- Eine andere Eingabe bewirkt auch ein anderes Verhalten:

Eingabewort:

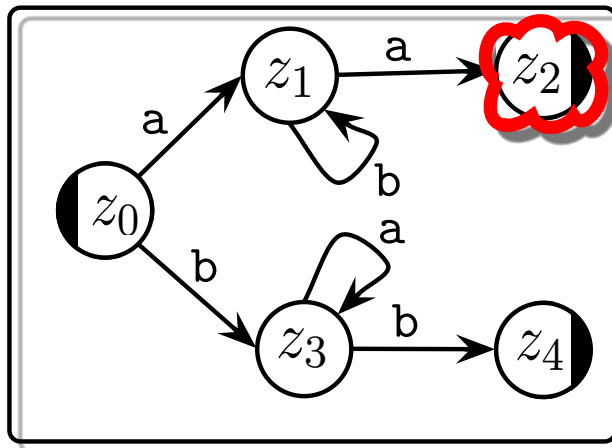
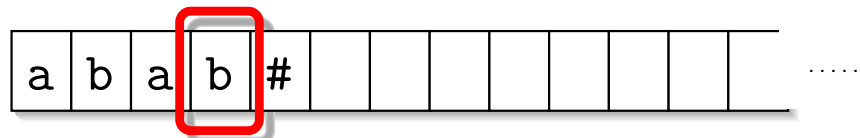




Funktionsweise des DFA

- Eine andere Eingabe bewirkt auch ein anderes Verhalten:

Eingabewort:



Die Rechnung des Automaten endet wieder in z_2 , aber die Eingabe ist nicht vollständig gelesen.



Definition: DFA

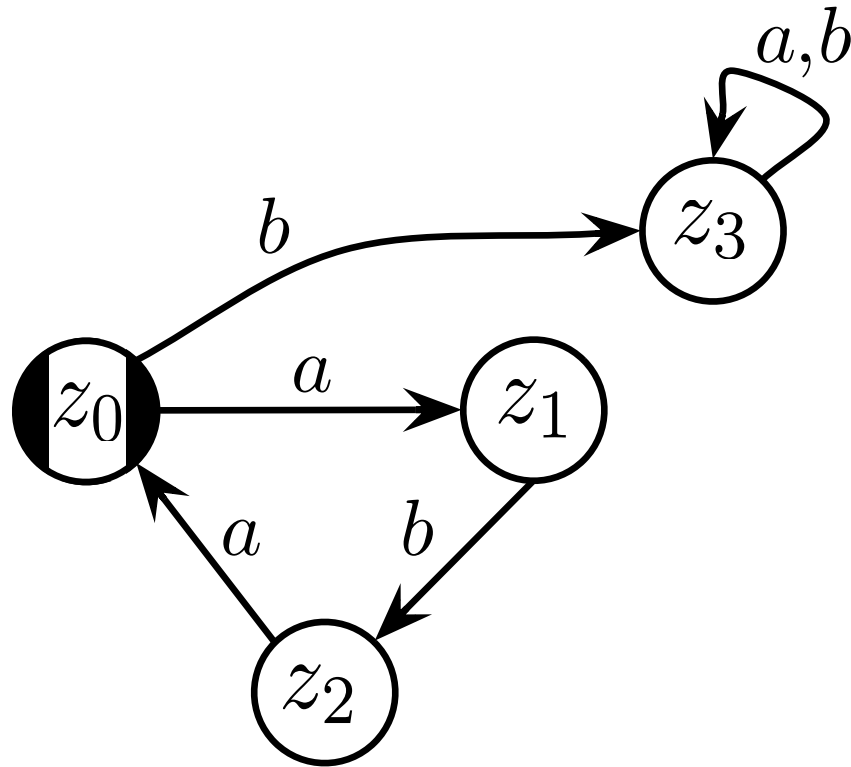
Ein **deterministischer endlicher Automat** (**DFA** für *deterministic finite automaton*) wird durch ein Quintupel $A := (Z, \Sigma, \delta, z_0, Z_{\text{end}})$ beschrieben, wobei

- Z eine endliche Menge von **Zuständen** ist,
- Σ ein endliches Alphabet von **Eingabesymbolen** ist,
- $\delta : \subseteq Z \times \Sigma \longrightarrow Z$ die nicht notwendigerweise totale **Überföhrungsfunktion** ist,
- $z_0 \in Z$ der **Startzustand** ist und
- $Z_{\text{end}} \subseteq Z$ die Menge der **Endzustände** ist.



Ein Beispiel

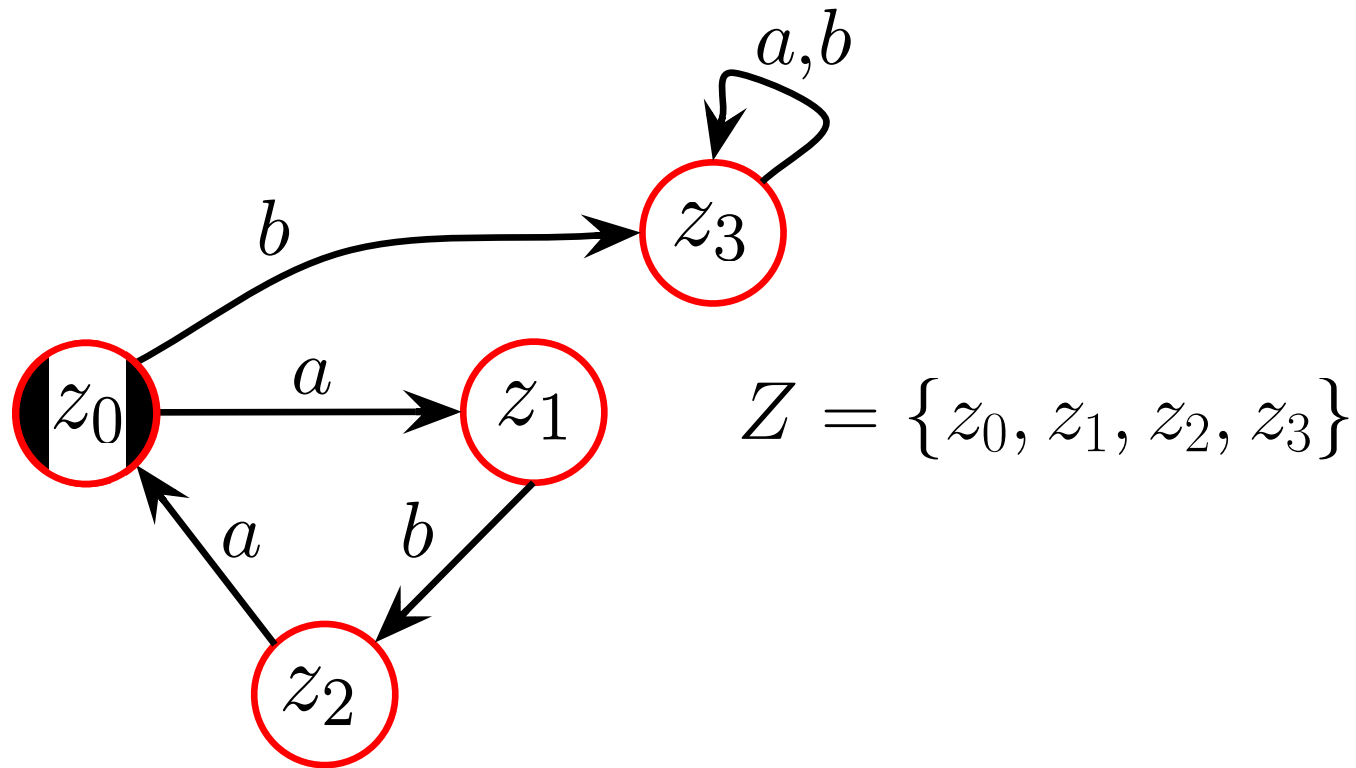
Die graphische Darstellung eines DFA:





Ein Beispiel

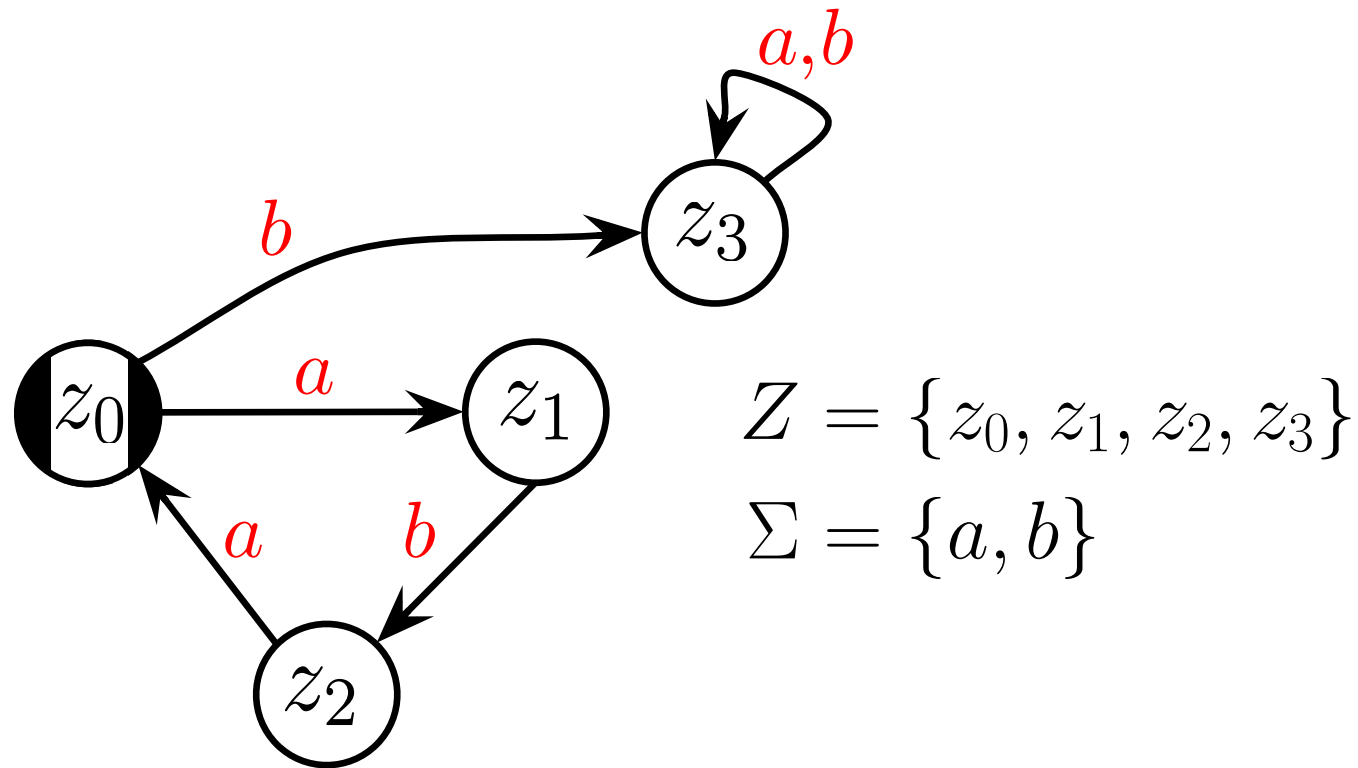
Die graphische Darstellung eines DFA:





Ein Beispiel

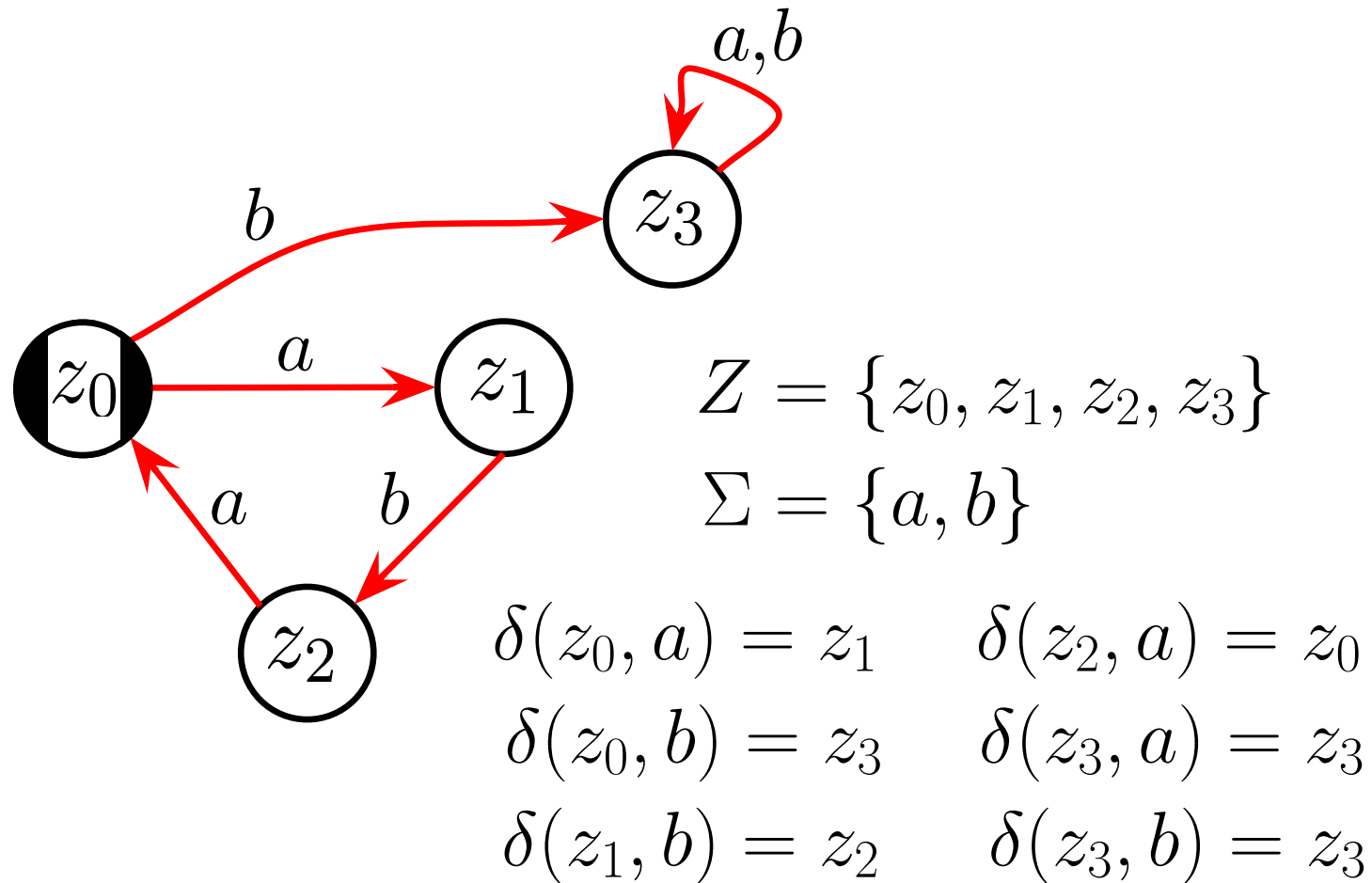
Die graphische Darstellung eines DFA:





Ein Beispiel

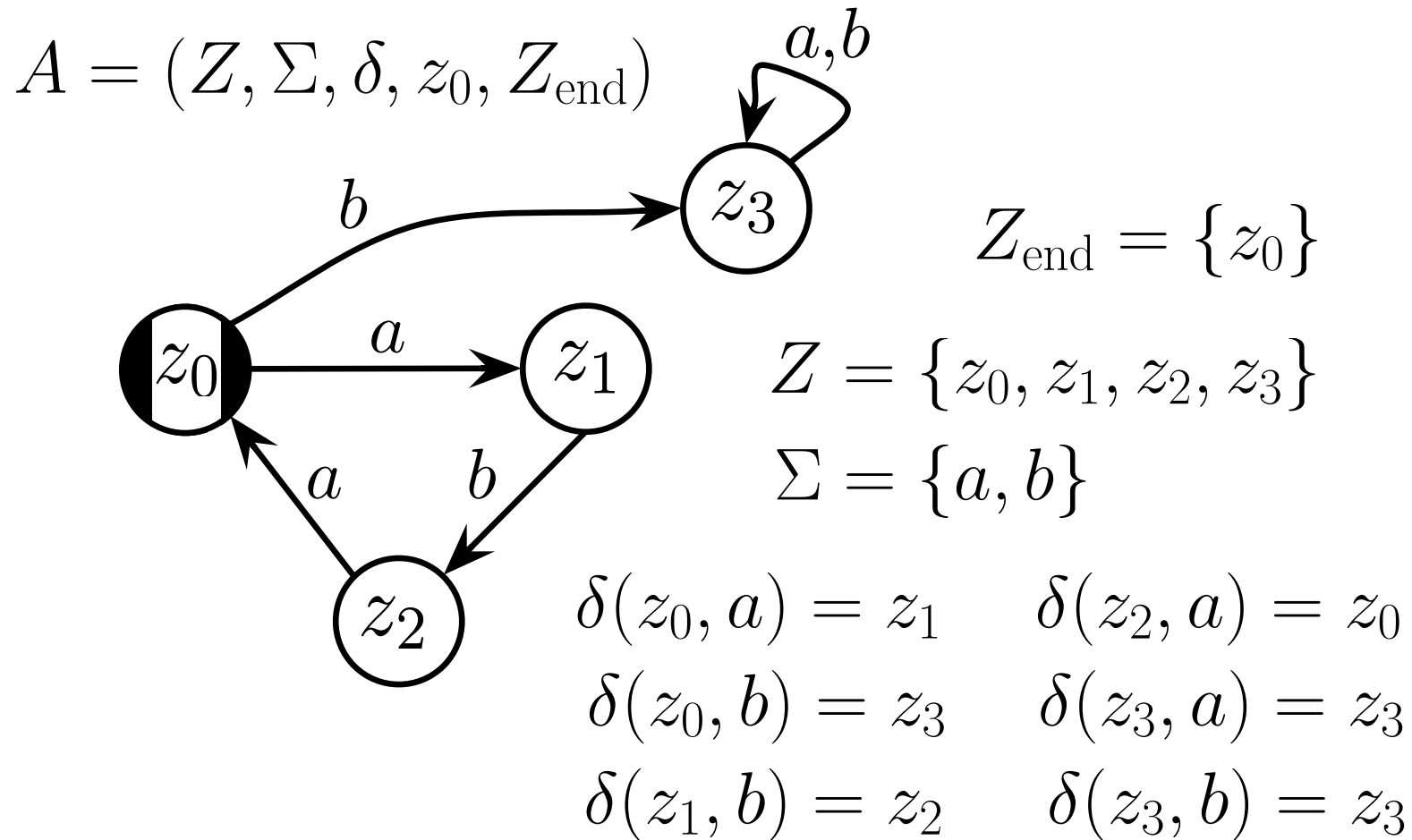
Die graphische Darstellung eines DFA:





Ein Beispiel

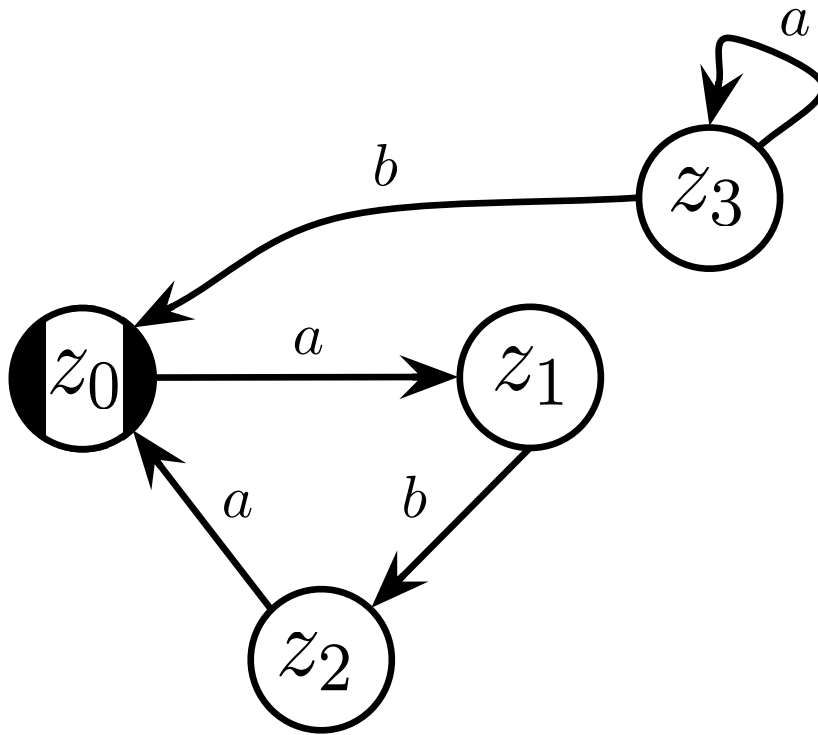
Die graphische Darstellung eines DFA:





Ein Beispiel

Die graphische Darstellung eines DFA:



... ist übrigens auch ein DFA!



Akzeptierte Sprache

- Zu einem DFA $A := (Z, \Sigma, \delta, z_0, Z_{\text{end}})$ definieren wir die **erweiterte Überföhrungsfunktion** $\hat{\delta} : Z \times \Sigma^* \longrightarrow Z$ durch:

$$\hat{\delta}(z, xw) := \hat{\delta}(\delta(z, x), w)$$

für alle Zustände $z \in Z$, alle Symbole $x \in \Sigma$ und alle Wörter $w \in \Sigma^*$, sowie

$$\forall z_i \in Z : \hat{\delta}(z_i, \lambda) := z_i.$$



Akzeptierte Sprache

- Zu einem DFA $A := (Z, \Sigma, \delta, z_0, Z_{\text{end}})$ definieren wir die **erweiterte Überföhrungsfunktion** $\hat{\delta} : Z \times \Sigma^* \longrightarrow Z$ durch:

$$\hat{\delta}(z, xw) := \hat{\delta}(\delta(z, x), w)$$

für alle Zustände $z \in Z$, alle Symbole $x \in \Sigma$ und alle Wörter $w \in \Sigma^*$, sowie

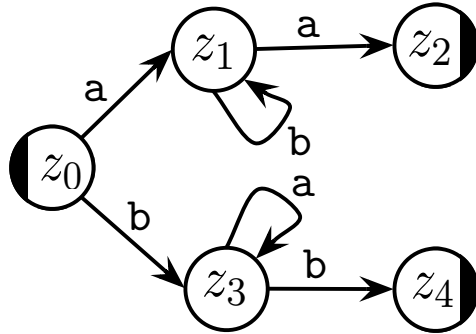
$$\forall z_i \in Z : \hat{\delta}(z_i, \lambda) := z_i.$$

- Die von dem DFA A **akzeptierte Sprache** ist die Menge

$$L(A) := \{w \in \Sigma^* \mid \hat{\delta}(z_0, w) \in Z_{\text{end}}\}.$$



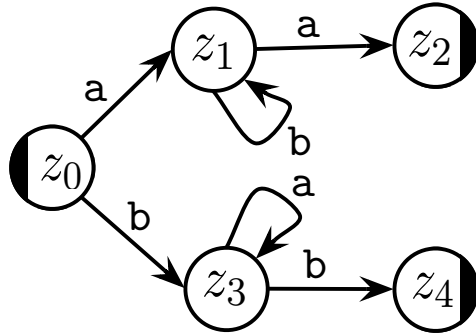
Der Beispiel-DFA als Akzeptor



$$L(A) = \{w \in \{a, b\}^* \mid \exists n \in \mathbb{N} : \\ (w = ab^n a \vee w = ba^n b)\}$$



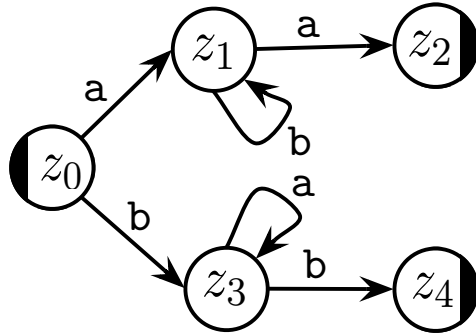
Der Beispiel-DFA als Akzeptor



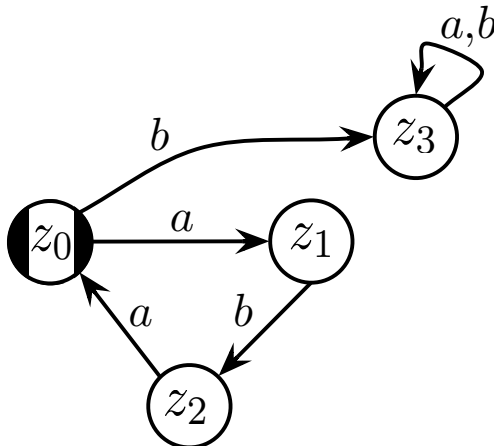
$L(A) = \{w \in \{a, b\}^* \mid \exists n \in \mathbb{N} : \dots \text{ alle W\"or}ter, \text{ die}$
 $(w = ab^n a \vee w = ba^n b)\}$
mit a beginnen und enden und dazwischen nur b 's
haben und umgekehrt.



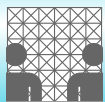
Der Beispiel-DFA als Akzeptor



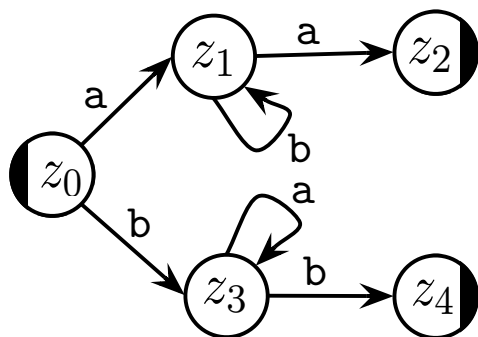
$L(A) = \{w \in \{a, b\}^* \mid \exists n \in \mathbb{N} : \dots \text{ alle Wörter, die } (w = ab^n a \vee w = ba^n b)\}$
mit a beginnen und enden und dazwischen nur b 's haben und umgekehrt.



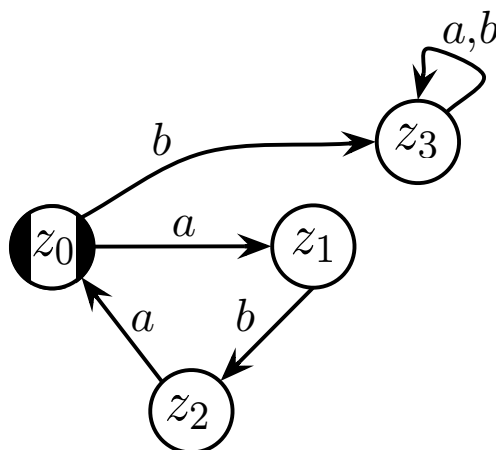
$$L(B) = \{aba\}^* = \{\lambda, aba, abaaba, abaabaaba, \dots\}$$



Der Beispiel-DFA als Akzeptor



$L(A) = \{w \in \{a, b\}^* \mid \exists n \in \mathbb{N} : (w = ab^n a \vee w = ba^n b)\}$
 ... alle Wörter, die mit a beginnen und enden und dazwischen nur b 's haben und umgekehrt.

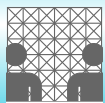


$L(B) = \{aba\}^* = \{\lambda, aba, abaaba, abaabaaba, \dots\}$



Akzeptieren vs. Berechenbarkeit

- Sei $M \subseteq C$, dann ist die **charakteristische Funktion** von M die auf ganz C definierte Funktion $\chi_M : C \longrightarrow \{0, 1\}$ mit $\chi_M(x) := [x \in M]$.



Akzeptieren vs. Berechenbarkeit

- Sei $M \subseteq C$, dann ist die **charakteristische Funktion** von M die auf ganz C definierte Funktion $\chi_M : C \longrightarrow \{0, 1\}$ mit $\chi_M(x) := [x \in M]$.
- Ist p ein Prädikat, so ist $[p]$ definiert als:

$$[p] := \begin{cases} 1 & \text{falls } p \text{ wahr ist} \\ 0 & \text{falls } p \text{ nicht wahr ist.} \end{cases}$$



Akzeptieren vs. Berechenbarkeit

- Sei $M \subseteq C$, dann ist die **charakteristische Funktion** von M die auf ganz C definierte Funktion $\chi_M : C \longrightarrow \{0, 1\}$ mit $\chi_M(x) := [x \in M]$.

- Ist p ein Prädikat, so ist $[p]$ definiert als:

$$[p] := \begin{cases} 1 & \text{falls } p \text{ wahr ist} \\ 0 & \text{falls } p \text{ nicht wahr ist.} \end{cases}$$

- ... also direkte Definition:

$$\chi_M(x) := \begin{cases} 1 & \text{falls } x \in M \\ 0 & \text{sonst} \end{cases}$$



Akzeptieren vs. Berechenbarkeit

- Sei $M \subseteq C$, dann ist die **charakteristische Funktion** von M die auf ganz C definierte Funktion $\chi_M : C \longrightarrow \{0, 1\}$ mit $\chi_M(x) := [x \in M]$.

- Ist p ein Prädikat, so ist $[p]$ definiert als:

$$[p] := \begin{cases} 1 & \text{falls } p \text{ wahr ist} \\ 0 & \text{falls } p \text{ nicht wahr ist.} \end{cases}$$

- ... also direkte Definition:

$$\chi_M(x) := \begin{cases} 1 & \text{falls } x \in M \\ 0 & \text{sonst} \end{cases}$$

- **Frage:** Ist die charakteristische Funktion *berechenbar*?



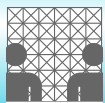
Beweis $L(A) = L$

- Das Akzeptieren oder Nicht-Akzeptieren kann als Berechnung der charakteristischen Funktion angesehen werden.



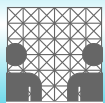
Beweis $L(A) = L$

- Das Akzeptieren oder Nicht-Akzeptieren kann als Berechnung der charakteristischen Funktion angesehen werden.
- Nur die Angabe eines DFA genügt nicht! Es muss bewiesen werden, dass er **genau** die gewünschte Sprache akzeptiert.



Beweis $L(A) = L$

- Das Akzeptieren oder Nicht-Akzeptieren kann als Berechnung der charakteristischen Funktion angesehen werden.
- Nur die Angabe eines DFA genügt nicht! Es muss bewiesen werden, dass er **genau** die gewünschte Sprache akzeptiert.
- Dies geschieht meist durch den Beweis von $L \subseteq L(A)$ und $L(A) \subseteq L$.



Beweis $L(A) = L$

- Das Akzeptieren oder Nicht-Akzeptieren kann als Berechnung der charakteristischen Funktion angesehen werden.
- Nur die Angabe eines DFA genügt nicht! Es muss bewiesen werden, dass er **genau** die gewünschte Sprache akzeptiert.
- Dies geschieht meist durch den Beweis von $L \subseteq L(A)$ und $L(A) \subseteq L$.
- Wir beweisen heute nicht die Korrektheit der Beispielautomaten und -sprachen.



Beweistechniken

- Folgende Beweistechniken werden immer wieder benötigt:



Beweistechniken

- Folgende Beweistechniken werden immer wieder benötigt:
 - vollständige Induktion



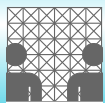
Beweistechniken

- Folgende Beweistechniken werden immer wieder benötigt:
 - vollständige Induktion
 - Widerspruchsbeweise



Beweistechniken

- Folgende Beweistechniken werden immer wieder benötigt:
 - vollständige Induktion
 - Widerspruchsbeweise
 - Diagonalisierung



Beweistechniken

- Folgende Beweistechniken werden immer wieder benötigt:
 - vollständige Induktion
 - Widerspruchsbeweise
 - Diagonalisierung
- **Feststellung:** Hierfür muss oft systematisch jedes Wort (allgemeiner: jedes Objekt) betrachtet werden!



Ordnungsrelationen

- **Quasiordnung:** reflexiv, transitiv



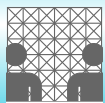
Ordnungsrelationen

- **Quasiordnung:** reflexiv, transitiv
- **partielle Ordnung:** reflexiv, transitiv, antisymmetrisch



Ordnungsrelationen

- **Quasiordnung:** reflexiv, transitiv
- **partielle Ordnung:** reflexiv, transitiv, antisymmetrisch
- **Halbordnung:** transitiv, antisymmetrisch



Ordnungsrelationen

- **Quasiordnung:** reflexiv, transitiv
- **partielle Ordnung:** reflexiv, transitiv, antisymmetrisch
- **Halbordnung:** transitiv, antisymmetrisch
- **Striktordnung:** transitiv, asymmetrisch



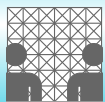
Ordnungsrelationen

- **Quasiordnung:** reflexiv, transitiv
- **partielle Ordnung:** reflexiv, transitiv, antisymmetrisch
- **Halbordnung:** transitiv, antisymmetrisch
- **Striktordnung:** transitiv, asymmetrisch
- **lineare Ordnung:** partielle Ordnung, für die jede zwei unterschiedlichen Elemente x, y entweder $(x, y) \in R$ oder $(y, x) \in R$.



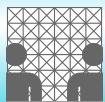
Lexikographische Ordnung

- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikographische Erweiterung** $\prec^{\text{lex}} \subseteq \Sigma^* \times \Sigma^*$ von \prec definiert durch:
 1. $\lambda \prec^{\text{lex}} w$ für alle $w \in \Sigma^*$.
 2. $\forall x, y \in \Sigma \forall u, v \in \Sigma^+ :$
 $(xu \prec^{\text{lex}} yv) \leftrightarrow (x \prec y) \vee [(x = y) \wedge (u \prec^{\text{lex}} v)]$



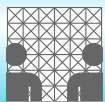
Lexikographische Ordnung

- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikographische Erweiterung** $\prec^{\text{lex}} \subseteq \Sigma^* \times \Sigma^*$ von \prec definiert durch:
 1. $\lambda \prec^{\text{lex}} w$ für alle $w \in \Sigma^*$.
 2. $\forall x, y \in \Sigma \forall u, v \in \Sigma^+ :$
$$(xu \prec^{\text{lex}} yv) \leftrightarrow (x \prec y) \vee [(x = y) \wedge (u \prec^{\text{lex}} v)]$$
- $\preceq^{\text{lex}} := \prec^{\text{lex}} \cup \text{Id}_{\Sigma^*}$ heißt **lexikographische Ordnung** auf Σ^* .



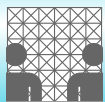
Lexikographische Ordnung

- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikographische Erweiterung** $\prec^{\text{lex}} \subseteq \Sigma^* \times \Sigma^*$ von \prec definiert durch:
 1. $\lambda \prec^{\text{lex}} w$ für alle $w \in \Sigma^*$.
 2. $\forall x, y \in \Sigma \forall u, v \in \Sigma^+ :$
$$(xu \prec^{\text{lex}} yv) \leftrightarrow (x \prec y) \vee [(x = y) \wedge (u \prec^{\text{lex}} v)]$$
- $\preceq^{\text{lex}} := \prec^{\text{lex}} \cup \text{Id}_{\Sigma^*}$ heißt **lexikographische Ordnung** auf Σ^* .
- **Beispiel:** ALLE \prec^{lex} FINDEN \prec^{lex}
THEORIE \prec^{lex} WUNDERBAR



Lexikographische Ordnung

- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikographische Erweiterung** $\prec^{\text{lex}} \subseteq \Sigma^* \times \Sigma^*$ von \prec definiert durch:
 1. $\lambda \prec^{\text{lex}} w$ für alle $w \in \Sigma^*$.
 2. $\forall x, y \in \Sigma \forall u, v \in \Sigma^+ :$
 $(xu \prec^{\text{lex}} yv) \leftrightarrow (x \prec y) \vee [(x = y) \wedge (u \prec^{\text{lex}} v)]$
- $\preceq^{\text{lex}} := \prec^{\text{lex}} \cup \text{Id}_{\Sigma^*}$ heißt **lexikographische Ordnung** auf Σ^* .
- **Beispiel:** ALLE \prec^{lex} FINDEN \prec^{lex}
THEORIE \prec^{lex} WUNDERBAR
- Wenn wir systematisch alle Wörter aus $\{a, b\}^*$ mit $a \prec b$ aufzählen wollen, eignet sich \prec^{lex} nicht!



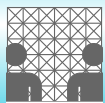
Lexikographische Ordnung

- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikographische Erweiterung** $\prec^{\text{lex}} \subseteq \Sigma^* \times \Sigma^*$ von \prec definiert durch:
 1. $\lambda \prec^{\text{lex}} w$ für alle $w \in \Sigma^*$.
 2. $\forall x, y \in \Sigma \forall u, v \in \Sigma^+ :$
 $(xu \prec^{\text{lex}} yv) \leftrightarrow (x \prec y) \vee [(x = y) \wedge (u \prec^{\text{lex}} v)]$
- $\preceq^{\text{lex}} := \prec^{\text{lex}} \cup \text{Id}_{\Sigma^*}$ heißt **lexikographische Ordnung** auf Σ^* .
- **Beispiel:** ALLE \prec^{lex} FINDEN \prec^{lex}
THEORIE \prec^{lex} WUNDERBAR
- Wenn wir systematisch alle Wörter aus $\{a, b\}^*$ mit $a \prec b$ aufzählen wollen, eignet sich \prec^{lex} nicht!



Lexikalische Ordnung

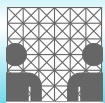
- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikalische** Erweiterung $\prec^{\text{lg-lex}}$ von \prec definiert durch:



Lexikalische Ordnung

- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikalische** Erweiterung $\prec^{\text{lg-lex}}$ von \prec definiert durch:
 - $w_1 \prec^{\text{lg-lex}} w_2$ gelte für beliebige Wörter $w_1, w_2 \in \Sigma^*$ gdw.

$$|w_1| < |w_2| \text{ oder } |w_1| = |w_2| \wedge w_1 \prec^{\text{lex}} w_2.$$



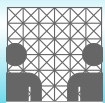
Lexikalische Ordnung

- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikalische** Erweiterung $\prec^{\text{lg-lex}}$ von \prec definiert durch:

- $w_1 \prec^{\text{lg-lex}} w_2$ gelte für beliebige Wörter $w_1, w_2 \in \Sigma^*$ gdw.

$$|w_1| < |w_2| \text{ oder } |w_1| = |w_2| \wedge w_1 \prec^{\text{lex}} w_2.$$

- $\preceq^{\text{lg-lex}} := \prec^{\text{lg-lex}} \cup \text{Id}_{\Sigma^*}$ heißt **lexikalische Ordnung** auf Σ^* .



Lexikalische Ordnung

- Sei (Σ, \prec) ein mit \prec linear geordnetes Alphabet, dann ist die **lexikalische** Erweiterung $\prec^{\text{lg-lex}}$ von \prec definiert durch:

- $w_1 \prec^{\text{lg-lex}} w_2$ gelte für beliebige Wörter $w_1, w_2 \in \Sigma^*$ gdw.

$$|w_1| < |w_2| \text{ oder } |w_1| = |w_2| \wedge w_1 \prec^{\text{lex}} w_2.$$

- $\preceq^{\text{lg-lex}} := \prec^{\text{lg-lex}} \cup \text{Id}_{\Sigma^*}$ heißt **lexikalische Ordnung** auf Σ^* .

- ... und hiermit ist das Problem gelöst:

$\lambda, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb, aaaa, \dots$



Ausblick

- Eigenschaften und Abwandlungen des endlichen Automaten
- Eigenschaften der von DFAs akzeptierten Sprachen
- $\text{FUER} \prec^{\text{lg-lex}} \text{HEUT} \prec^{\text{lg-lex}} \text{SEID} \prec^{\text{lg-lex}} \text{ERLOEST} \prec^{\text{lg-lex}} \text{ERLAUCHTE} \prec^{\text{lg-lex}} \text{HOERERSCHAFT}$