Finite-State Technology

Vorlesung 5:

Transducer und reguläre Relationen

FST's als Klasse von Graphen

- Ein FST ist ein Sixtupel (Σ_1, Σ_2, Q, i, F, E) mit:

  Σ_1 ist ein finites Alphabet, das Eingabealphabet
  Σ_2 ist ein finites Alphabet, das Ausgabealphabet
  Q ist eine endliche Menge von Zuständen
  $i \in Q$ ist der Startzustand
  $F \subseteq Q$ ist die Menge des Endzustände
  $E \subseteq Q \times \Sigma_1^* \times \Sigma_2^* \times Q$ ist die Menge der Ecken

Ersetzung von E durch d und δ

- Übergangsfunktion $d = Q \times \Sigma_1^* \times 2^Q$
- Ausgabefunktion $\delta = Q \times \Sigma_1^* \times Q \times 2^{\Sigma 2^*}$

Verbindung zwischen E und d, δ:

- $d(q, a) = \{q' \in Q \mid \exists (q, a, b, q') \in E\}$
- $\delta(q, a) = \{b \in \Sigma_2^* \mid \exists (q, a, b, q') \in E\}$

Definition eines FST als FSA

- Wenn T ein FST ist $(\Sigma_1, \Sigma_2, Q, i, F, E)$, ist sein zugrundeliegender Automat (Σ, Q, i, F, E') folgendermaßen definiert:

  $\Sigma = \Sigma_1 \times \Sigma_2$

  $(q_1, (a, b), q_2) \in E'$ genau dann wenn $(q_1, a, b, q_2) \in E$.

FST‘s als Relationen

- Zuerst muß E zu Ê erweitert werden:

  a) $\forall q \in Q, (q, \varepsilon, \varepsilon, q) \in \hat{E}$

  b) $\forall w_1 \in \Sigma_1^*, \forall w_2 \in \Sigma_2^*$, wenn $(q_1, w_1, w_2, q_2) \in \hat{E}$ und $(q_2, a, b, q_3) \in E$, dann $(q_1, w_1a, w_2b, q_3) \in \hat{E}$

- Nun kann man sagen, eine Relation L(T) über $\Sigma_1^* \times \Sigma_2^*$ von einem Transducer T ist:

  $L(T) = \{(w_1, w_2) \in \Sigma_1^* \times \Sigma_2^* \mid \exists (i, w_1, w_2, q) \in \hat{E} \text{ mit } q \in F\}$

FST'S als Abbildung $|T|$

- Abbildung von der Menge der Zeichenreihen in $\Sigma_1{}^*$ in die Potenzmenge der Zeichenreihen in $2^{\Sigma 2*}$

  $|T|(u) = \{v \in \Sigma_2{}^* \mid (u, v) \in L(T)\}$

- Dies kann auch auf Zeichenreihen erweitert werden:

  Wenn $V \subseteq \Sigma^*$, dann $|T|(V) = \bigcup_{v \in V} |T|(v)$

Transducer, *transductions* und *rational functions*

- Ein Transducer repräsentiert eine *Transduction* (egal, ob rational oder nicht), genau wie ein Automat eine reguläre Sprache repräsentiert.
- Eine *Transduction* ist eine rationale *Transduction*, wenn es einen FST T gibt, so daß $\tau = |T|$.
- Wenn für jede Zeichenreihe u aus dem Eingabealphabet Σ_1* gilt, daß $|T|(u)$ entweder die leere Menge oder eine Einermenge ist, dann ist $|T|$ eine rationale Funktion.
- Es gilt: rationale Funktionen werden durch funktionale rationale *Transducers* repräsentiert.

Abgeschlossenheitseigenschaften

- Das Problem der Intersektion: Reguläre Relationen sind nur bei gleicher Länge geschlossen unter Intersektion (Schnitt).

- Problembeispiel: $P = [a:b]^* \; [\varepsilon:c]^*$, $Q = [\varepsilon:b]^* \; [a:c]^*$

  Was ist das Ergebnis von $P \cap Q$? Welche Sprache steht in der unteren Projektion?

Funktionalität und Ambiguität

- In der Analyse von Sprache möchte man für eine Eingabe genau eine mögliche Ausgabe haben. Dies geht natürlich nicht immer in beide Richtungen und manchmal nicht einmal für eine Richtung.

- Theorem von Schützenberger: Für jeden Transducer T kann man entscheiden , ob |T| funktional ist. (Roche/Schabes 1997: 28)

- Allerdings muß man dazu einen kleinen Umweg gehen: Umwandlung des Transducers T1 in einen nichtambigen Transducer T2 mit anschließendem Äquivalenztest zwischen T1 und T2.

Definition von Ambiguität

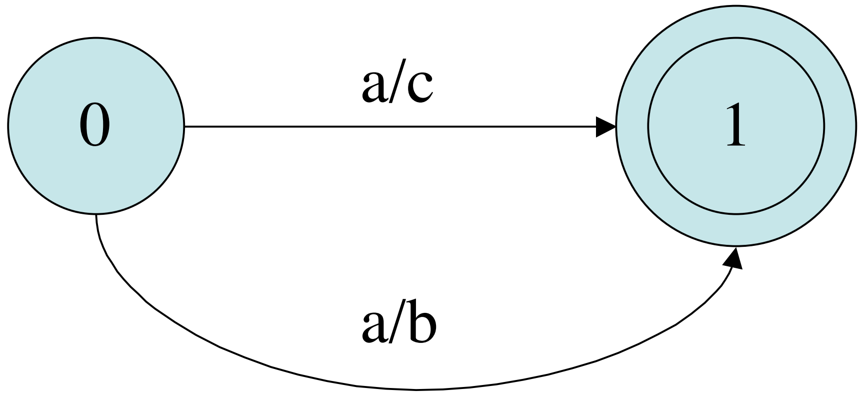
- “An unambiguous transducer is a transducer for which each input is the label of at most one successful path”. (Roche/Schabes 1997:28)

- Das folgende Beispiel zeigt einen nichtfunktionalen ambigen Transducer.

  Nichtfunktional: Mit Eingabe a gibt es mehr als eine Ausgabe.

  Ambig: Für die Eingabe a gibt es mehr als einen erfolgreichen Pfad.

Beispiel: Nichtfunktional und ambig



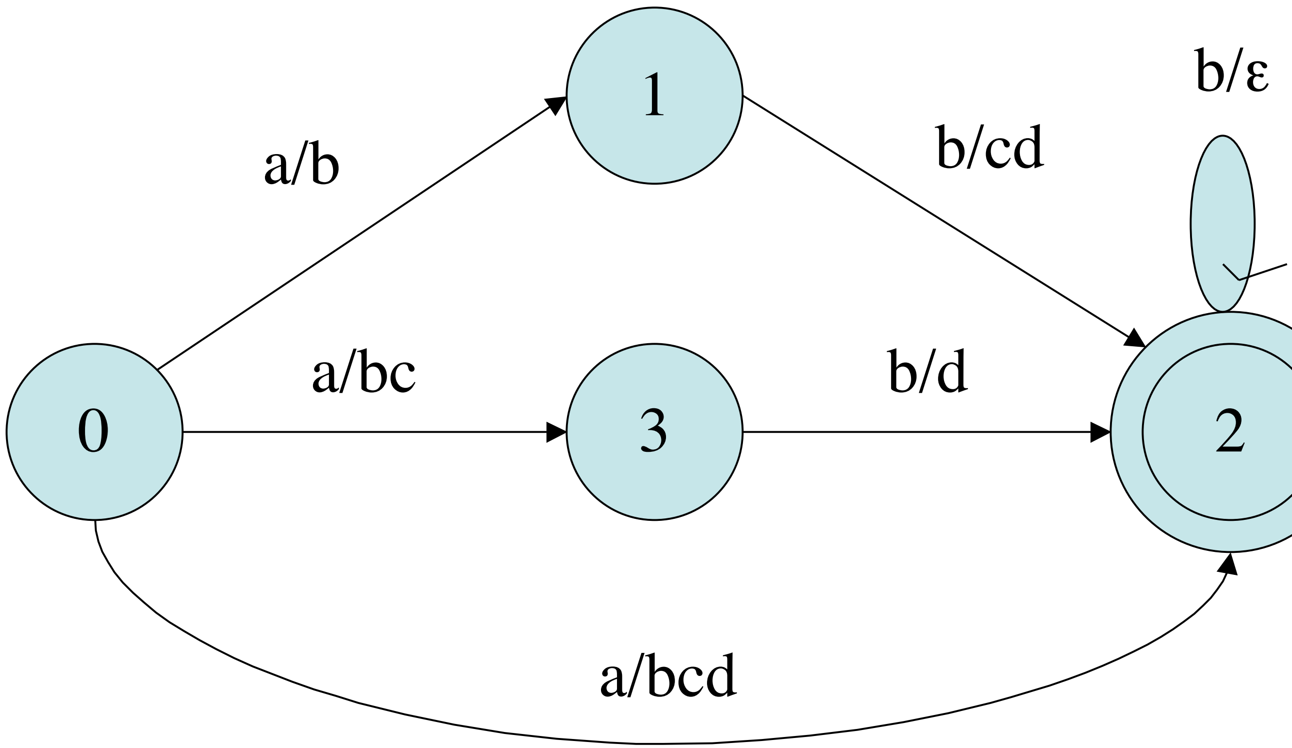
Beispiel

- Das folgende Beispiel zeigt einen funktionalen ambigen Transducer.

  Funktional: Für jede mögliche Eingabe (hier a oder ab) gibt es nur eine mögliche Ausgabe (bcd).

  Ambig: Für die Eingabe ab gibt es mehr als einen erfolgreichen Pfad: (0,1,2), (0,3,2), (0,2,2).

Beispiel: Funktional und ambig



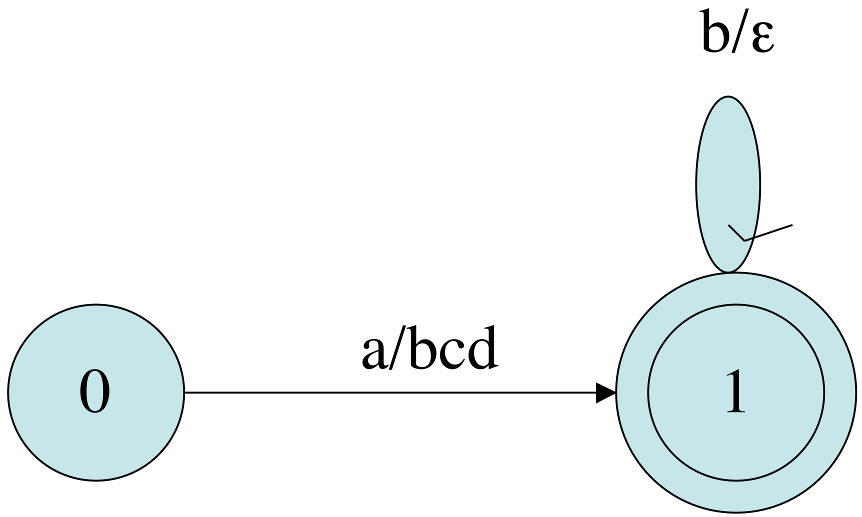
Beispiel

- Das folgende Beispiel zeigt einen funktionalen nichtambigen Transducer.

  Funktional: Für jede mögliche Eingabe (hier a oder ab) gibt es nur eine mögliche Ausgabe (bcd).

  Ambig: Für die Eingabe ab gibt es nur einen erfolgreichen Pfad: (0,1).

Beispiel: Funktional und nichtambig



Das Problem Indeterminismus

- Bei Transducern unterscheidet man zwei Arten von Indeterminismus:

  a) bei Paaren von Symbolen

  b) bei einzelnen Projektionen.

- Wie bei Automaten möchte man auch bei Transducern Indeterminismus vermeiden, um Eingaben schneller verarbeiten zu können.
- Warnung: Die folgenden Begriffe werden je nach Autor unterschiedliche definiert. Hier wurde als Grundlage Roche/Schabes (1997) benutzt.

Deterministische FST‘s

- Transducer mit $E \subseteq Q \times \Sigma_1 \times \Sigma_2{}^* \times Q$ so daß gilt:

  Für jedes $q \in Q$, $a \in \Sigma_1$ $w \in \Sigma_2{}^*$ gibt es höchstens ein $q' \in Q$ so daß $(q, a, w, q') \in Q$ (Roche/Schabes 1997:46)

- Mit etwas anderen Worten: “If two paths have the same input, they have different outputs.” (Roche/Schabes 1997:31)

Subsequentielle FST‘s

- Definition (Roche/Schabes 1997:45):

  Ein subsequentieller Transducer T ist ein Oktupel (Σ_1, Σ_2, Q, i, F, $\otimes$, *, ρ), wobei:

  Σ_1, Σ_2, Q, i sind wie üblich definiert

  $\otimes$ ist eine deterministische Übergangsfunktion, die $Q \times \Sigma_1$ auf Q abbildet (geschrieben: $q \otimes a = q'$)

  * ist eine deterministische Ausgabefunktion, die $Q \times \Sigma_1$ auf Σ_2^* abbildet (geschrieben: $q * a = w$)

  ρ ist die abschließende Ausgabefunktion, die F auf Σ^* abbildet (geschrieben: $\rho(q) = w$).

Subsequentielle FST‘s

- Mit anderen Worten:

  “For each state there is at most one outgoing transition whose input label corresponds to a given input symbol. ” (Roche/Schabe 1997:43)

Beispiel

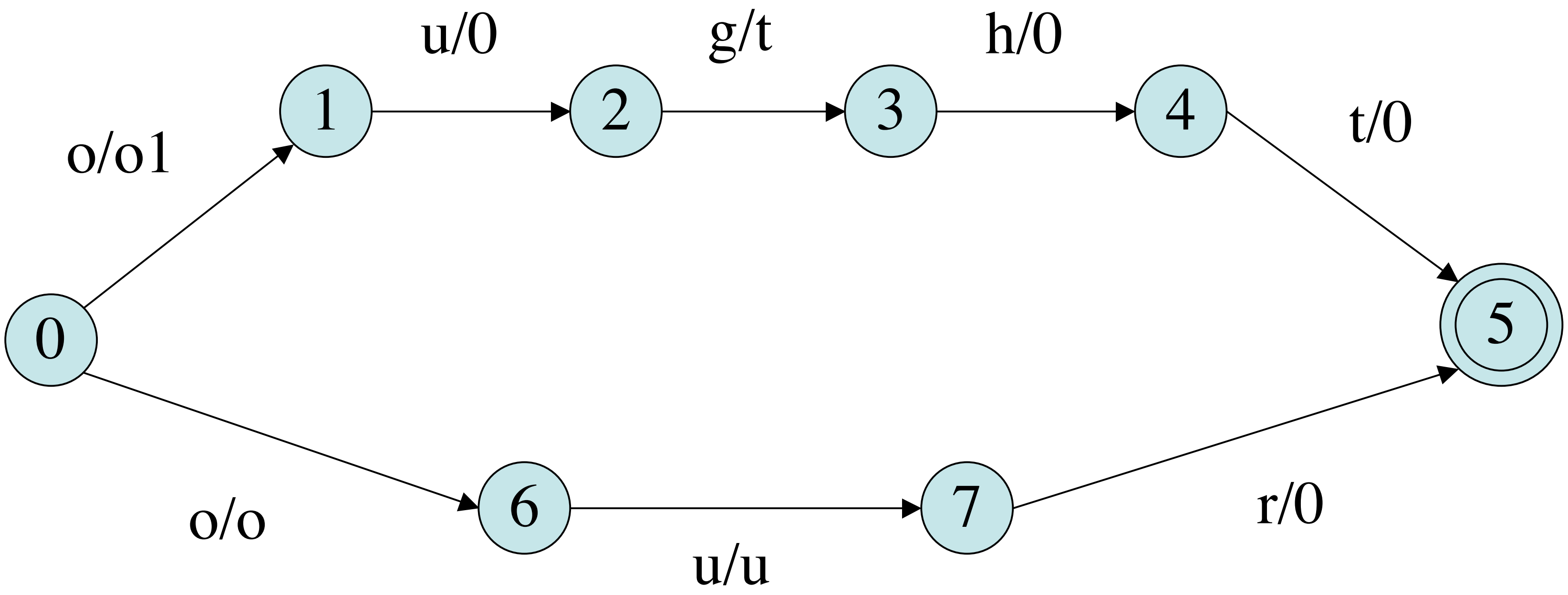
- Das folgende Beispiel zeigt einen deterministischen nichtsubsequentionellen Transducer.

  Deterministisch: Für jedes mögliche Symbolpaar gibt es genau einen definierten Übergang von einem Zustand zum nächsten.

  Nichtsubsequentionell: Für die Eingabe o gibt es mehr als eine mögliche Paarung mit einem Ausgabesymbol (o:o1; o:o).

- Frage: Ist der Transducer funktional? Ist er ambig?

Beispiel: Deterministisch und nichtsubsequentiell



Beispiel

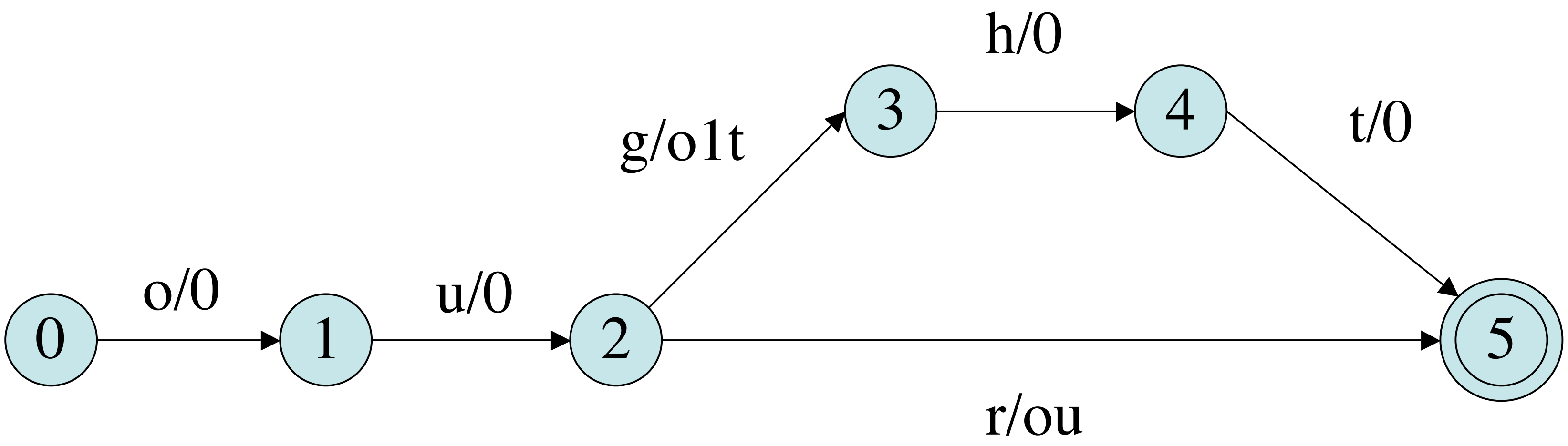
- Das folgende Beispiel zeigt einen deterministischen subsequentionellen Transducer.

  Deterministisch: Für jedes mögliche Symbolpaar gibt es genau einen definierten Übergang von einem Zustand zum nächsten.

  Subsequentionell: Für jede Eingabe gibt es genau eine mögliche Paarung mit einem Ausgabesymbol.

- Frage: Ist der Transducer funktional? Ist er ambig?

Beispiel: Deterministisch und subsequentiell

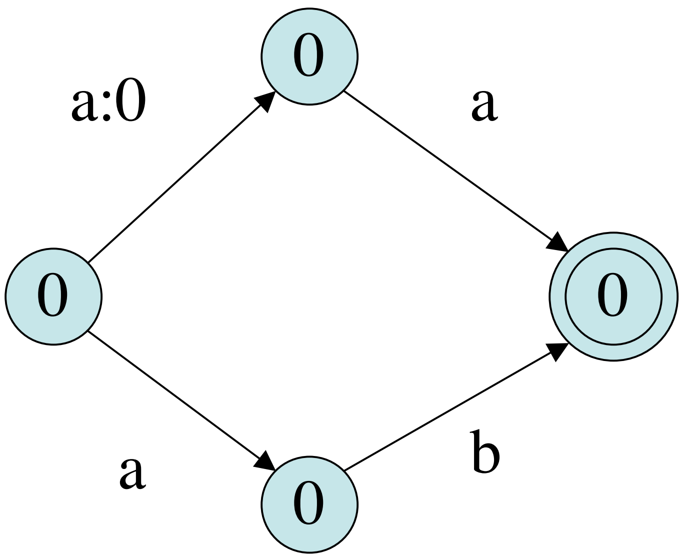


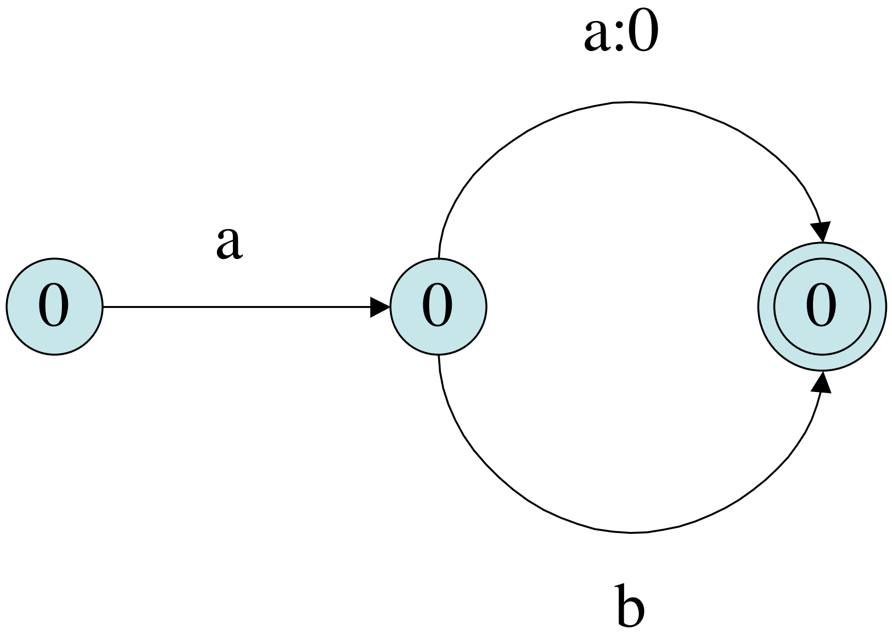
Beispiel

- Das folgende Beispiel zeigt einen deterministischen nichtsubsequentionellen Transducer (links) und einen deterministischen subsequentionellen Transducer (rechts).
- Problem: “There is no general method for deriving a unique canonical representation for a relation that pairs strings of unequal length. Two networks may encode the same relation but there is no general algorithm for deciding whether they do. In this respect regular relations are computationally more difficult to handle than regular languages. Every regular language has a unique minimal encoding as a network, but some relations do not.”

(Beesley/Karttunen 2003:53)

Beispiel: Deterministisch und (nicht)subsequentiell





Zusammenfassung Transducer

- Rational/funktional: Betrifft die Anzahl der Zeichenreihen der Ausgabe pro Eingabe (0 bzw. 1 = rational).
- Ambig/nicht ambig: Betrifft die Zeichenreihe der ersten Projektion und die Anzahl der erfolgreichen Pfade (mehr als 1 Pfad = ambig).
- Deterministisch/indeterministisch: Betrifft die Anzahl der Übergänge von Symbolpaaren (mehr als 1 Übergang für ein Symbolpaar = indeterministisch).
- Subsequentiell/nicht subsequentiell: Betrifft die Anzahl der Abbildungen eines Symbols a der ersten Projektion auf Symbole b der zweiten Projektion (wird a auf mehr als ein Symbol abgebildet: nicht subsequentiell).

Zusammenfassung Transducer

- Ist es nicht möglich, einen funktionaler Transducer in einen äquivalenten subsequentionellen umzuformen, kann man ihn in eine *bimachine* umformen.
- Dies ist eine Kombination aus zwei sequentionellen (andere Bedeutung als subsequentionell!) Transducern T1 und T2, die die Eingabe einmal von links nach rechts (T1) und von rechts nach links (T2) lesen. Die Ausgabe von T1 ist dabei die Eingabe von T2.
- Ein einfaches Beispiel dafür wird in Beesley/Karttunen (2003:2.5) vorgestellt.

Reguläre Relationen und Phonologie

- In der generativen Phonologie werden kontextsensitive Regeln benutzt, wie z.B.

  ε -> ab / _ (b). Dies ist eine einfache Epentheseregel.
- Außerdem können Regeln in der generativen Phonologie zyklisch angewendet werden, also mehrmals hintereinander. Für die obere Regel ergeben sich bei zyklischer Anwendung mehrere mögliche Ausgabemöglichkeiten.

  Ausgabemöglichkeit 1: $aa^n b^n b$

  Ausgabemöglichkeit 2: $a[ab]^n b$

Reguläre Relationen und Phonologie

- Die erste Ausgabemöglichkeit bildet eine Eingabe auf eine kontextfreie Sprache ab. Dies ist also außerhalb der regulären Sprache. (Anmerkung: Solch eine Regel ist nur der schlimmste anzunehmende Fall. Phonologische Regeln sind sonst sehr viel einfacher.)
- Bei der Untersuchung einer Reihe von phonologischen Regeln wurde jedoch bemerkt, daß die erste Ausgabemöglichkeit implizit ausgeschlossen wird. Die Ausgabe einer phonologischen Regel wird also bei erneuter Anwendung dieser Regel nicht mehr verändert.
- Mit dieser Beschränkung kann man phonologische Regeln als reguläre Relationen definieren.

Literatur

- Beesley/Karttunen (2003:II-III)
- Johnson (1972)
- Jurafsky/Martin (2000, insbesondere Kap. XIII)
- Kaplan/Kay (1994)
- Karttunen (1991)
- Roche/Schabes (1997:I)