
Einführung in die Computerlinguistik

Morphologie und Automaten II

SS 2009

Sebastian Pado

▶ 1

Rekapitulation

- ▶ Beobachtung: Morphologische Paradigmen zeigen ein hohes Maß an Systematik
 - ▶ Adjektivendungen des Deutschen:
 - ▶ optional nichts *oder* -st- *oder* -er-
 - ▶ dann -e *oder* -er *oder* -es *oder* -en *oder* -em
- ▶ Computerlinguistische Fragestellung: Wie man kann automatisch Adjektivsuffixe erkennen und abtrennen?

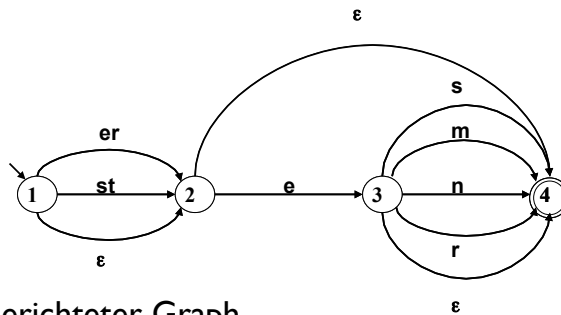
▶ 2

Formale Sprachen und Automaten

- ▶ Schritt I: Formulierung der Aufgabe als formale Sprache
 - ▶ Beschreibung einer Menge an "erlaubten" Zeichenketten
 - ▶ (und damit implizit auch einer Menge an "unerlaubten" Zeichenketten)
 - ▶ funktioniert auch für unendlich große Mengen
 - "Menge aller Worte, die auf -er enden"
- ▶ Schritt II: Darstellung der formalen Sprache als endlicher Automat
 - ▶ Automat erkennt die "erlaubten" Zeichenketten:
 - ▶ Eine Zeichenkette ist "erlaubt" genau dann wenn es einen Pfad durch den Automaten gibt, der diese Zeichenkette buchstabiert

▶ 3

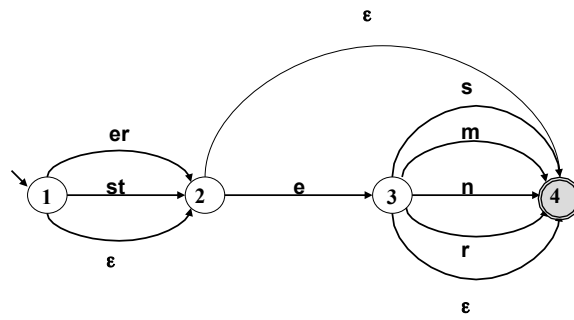
Was macht einen Automaten aus?



- ▶ Gerichteter Graph
- ▶ Kanteninschriften: Übergangsrelation
- ▶ Ein Startzustand - durch Pfeil gekennzeichnet
- ▶ Ein oder mehrere Endzustände - durch Doppelumriß gekennzeichnet

▶ 4

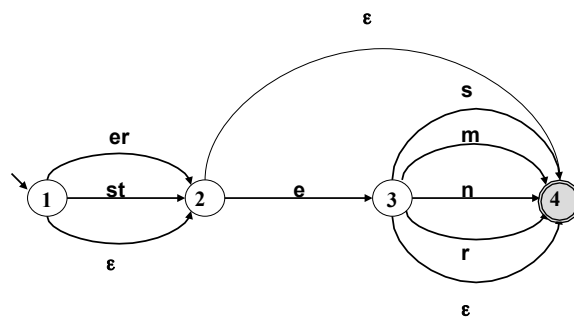
Ein Pfad durchs Diagramm



klein eres|

► 5

Ein alternativer Weg durchs Diagramm



klein er|es

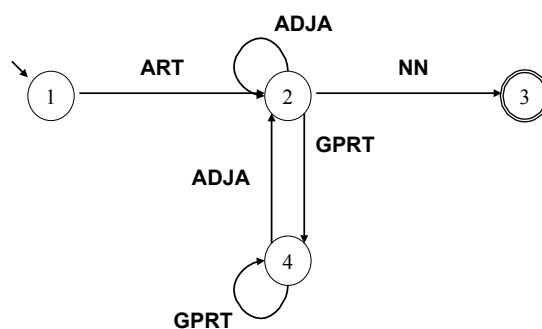
► 6

Das Problem

- ▶ Das Diagramm erlaubt typischerweise an einem Knoten mehrere Übergänge bei derselben Eingabe (deshalb „nicht-deterministisch“).
- ▶ Die zufällige Wahl einer Kante kann sich erst viel später als falsch herausstellen
- ▶ Vollständige Suche muß daher alle möglichen Entscheidungen bis zum Ende verfolgen
 - ▶ Tiefensuche-Algorithmus mit Stack/Agenda vom letzten Mal
- ▶ Der Suchbaum, der durchsucht werden muss, hat Größe $n^{|w|}$ -- der Zeitbedarf wächst exponentiell mit der Wortlänge. Das ist sehr ineffizient.

▶ 7

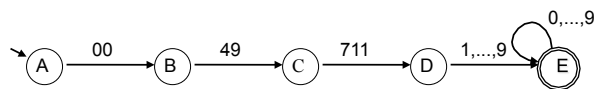
Ein deterministisches Diagramm



Beobachtung: Bestimmte Diagramme erfordern keine Suche, weil Übergänge bei gegebenem Zustand und Eingabesymbol **eindeutig festgelegt** sind.

▶ 8

Stuttgarter Telefonnummern (international)



▶ 9

Deterministische endliche Automaten

- ▶ Die beiden Diagramme unterscheiden sich von dem Adjektiv-Diagramm in einem wesentlichen Punkt: Für jeden Zustand/Knoten und jede Eingabe gibt es höchstens eine Kante, die beschriften werden kann. Sie sind deterministisch.
- ▶ Die Definition des „deterministischen endlichen Automaten“ (DEA oder DFA, für „deterministic finite-state automaton“) führt einige weitere, weniger wesentliche, aber nützliche Beschränkungen gegenüber dem NEA ein.

▶ 10

Deterministische und nicht-deterministische Automaten

- ▶ NEA erlaubt beliebige Worte (incl. ϵ) als Kanteninschrift
- ▶ DEA hat nur Einzelsymbole als Kanten-Inschriften, insbesondere sind Leerwort-Kanten nicht zulässig.
- ▶ NEA erlaubt für einen Ausgangszustand und eine Eingabe mehrere oder gar keinen Zielzustand
- ▶ DEA hat zu jedem Zustand und zu jedem Symbol genau eine wegführende Kante
- ▶ D.h.: NEA hat eine Übergangsrelation.
- ▶ D.h.: DEA hat eine Übergangsfunktion.

▶ 11

Definition: Deterministischer endlicher Automat

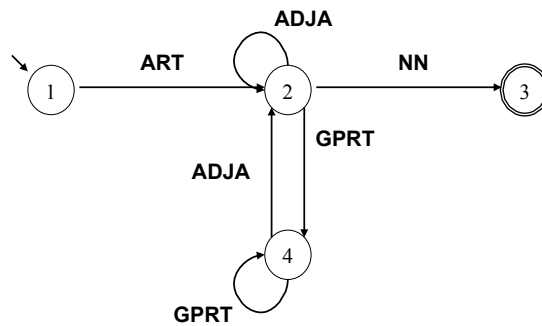
Ein deterministischer endlicher Automat ist ein Quintupel

$A = \langle K, \Sigma, \delta, s, F \rangle$, wobei

- ▶ K nicht-leere endliche Menge von Knoten (Zuständen)
- ▶ Σ nicht-leeres Alphabet
- ▶ $s \in K$ Startzustand
- ▶ $F \subseteq K$ Menge von Endzuständen
- ▶ $\delta : K \times \Sigma \rightarrow K$ Übergangsfunktion
- ▶ (war beim NEA: $\Delta : K \times \Sigma^* \times K$ Übergangsrelation)

▶ 12

Ein deterministisches Diagramm



Beobachtung: Bestimmte Diagramme erfordern keine Suche, weil Übergänge bei gegebenem Zustand und Eingabesymbol **eindeutig festgelegt** sind.

▶ 13

Beispiel: Der DEA für Wortartmuster [1]

DEA $A = \langle K, \Sigma, \delta, s, F \rangle$ mit

- ▶ $K = \{1, 2, 3, 4\}$
- ▶ $\Sigma = \{\text{ART}, \text{ADJA}, \text{NN}, \text{GPRT}\}$
- ▶ $s = 1$
- ▶ $F = \{3\}$
- ▶ δ definiert durch:

$$\begin{aligned}\delta(1, \text{ART}) &= 2 \\ \delta(2, \text{ADJA}) &= 2 \\ \delta(2, \text{NN}) &= 3 \\ \delta(2, \text{GPRT}) &= 4 \\ &\dots \quad \dots\end{aligned}$$

▶ 14

Beispiel [2]: Übergangstabelle für δ

δ :	ART	ADJA	NN	GPRT
1	2			
2		2	3	4
3				
4		2		4

► 15

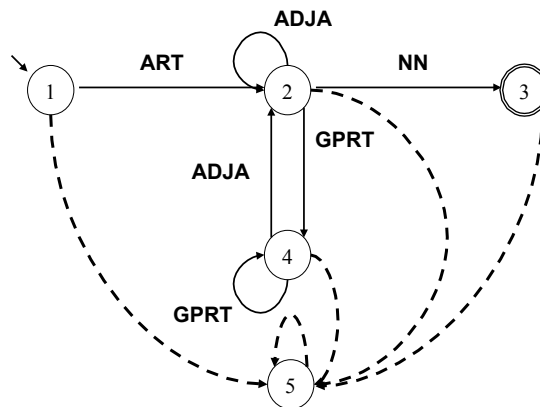
Beispiel [3]: Übergangstabelle für δ , komplettiert

δ :	ART	ADJA	NN	GPRT
1	2	5	5	5
2	5	2	3	4
3	5	5	5	5
4	5	2	5	4
5	5	5	5	5

- Der Zustand eines DEA, aus dem es keine Möglichkeit gibt, in einen Endzustand zu gelangen, heißt „Senke“ oder engl. „trap state“: Falle.

► 16

Das Zustandsdiagramm für Wortartmuster: Übergangsfunktion δ komplettiert



► 17

Deterministische und nicht- deterministische Automaten [1]

- ▶ DEAs erlauben den Test von Eingabeketten in linearer Zeit: Jedes Wort der Länge n wird in genau n Schritten abgearbeitet.
- ▶ DEAs haben allerdings ein eingeschränkteres Beschreibungs-Inventar als NEAs.
- ▶ Frage: Ist deshalb die Ausdrucksstärke des DEA-Formalismus eingeschränkter als die von NEAs? Das heißt, gibt es Sprachen, die durch einen NEA, aber nicht durch einen DEA beschrieben werden?
- ▶ Die Antwort: Nein.

► 18

Deterministische und nicht-deterministische Automaten [2]

- ▶ Jede Sprache, die von einem NEA akzeptiert wird, kann auch durch einen DEA beschrieben werden (und, trivialerweise, auch umgekehrt: ein DEA ist ein spezieller NEA). NEAs und DEAs besitzen die gleiche Ausdruckstärke, die Formalismen sind beschreibungsäquivalent.
- ▶ Das ist beweisbar. Noch wichtiger: Der Beweis ist konstruktiv, d.h.:
- ▶ Es gibt ein Konstruktionsverfahren, das es erlaubt, zu jedem NEA A einen DEA A' zu konstruieren, so dass $L(A') = L(A)$.

▶ 19

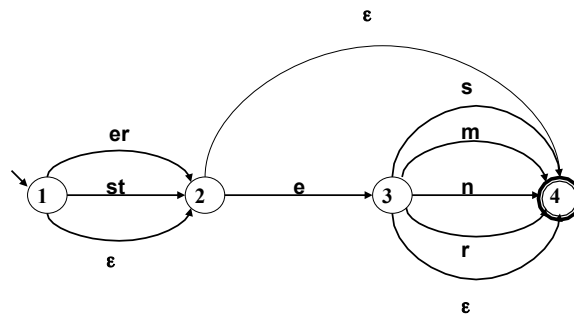
Die NEA-DEA-Überführung

Der Algorithmus zur NEA-DEA-Überführung besteht aus drei Schritten:

1. Beseitigung von Mehrsymbol-Kanten
2. Beseitigung von ε -Kanten
3. Die „Potenz-Automaten“-Konstruktion

▶ 20

Adjektivendungen: Zustandsdiagramm



► 21

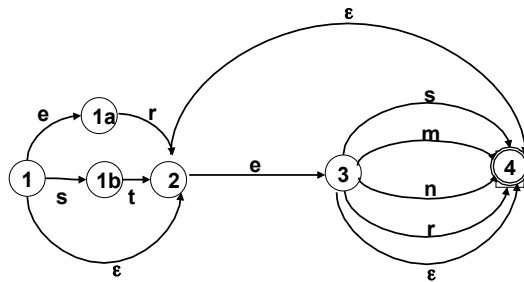
Schritt 1: Beseitigung von Mehrsymbolkanten

Gegeben sei der NEA $A = \langle K, \Sigma, \Delta, s, F \rangle$.

- Für alle Kanten $\langle q, w, q' \rangle$ mit $w = a_1 \dots a_n, n > 1$:
Entferne $\langle q, w, q' \rangle$ aus Δ .
- Erweitere K um neue Zustände q_1, \dots, q_{n-1} .
- Erweitere Δ um neue Kanten
 $\langle q, a_1, q_1 \rangle, \langle q_1, a_2, q_2 \rangle, \dots, \langle q_{n-1}, a_n, q' \rangle$

► 22

Beispiel-Automat nach Schritt 1:



▶ 23

Schritt 2: Beseitigung von ϵ -kanten

- ▶ Wir definieren zunächst als Hilfsbegriffe den „ ϵ -Vorbereich“ $V_\epsilon(p)$ und den „ ϵ -Nachbereich“ $N_\epsilon(p)$ von Zuständen:

- ▶ $V_\epsilon(p) = \{q \mid p \text{ ist von } q \text{ aus ohne Abarbeiten eines Symbols erreichbar}\}$

- ▶ $N_\epsilon(p) = \{q \mid q \text{ ist von } p \text{ aus ohne Abarbeiten eines Symbols erreichbar}\}$

Anmerkung: $V_\epsilon(p)$ und $N_\epsilon(p)$ enthalten insbesondere p selbst.

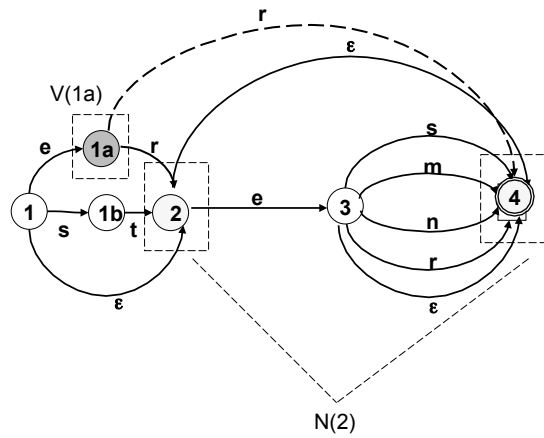
- ▶ Für jede nicht-leere Kante $\langle p, a, q \rangle \in \Delta$: Erweitere Δ um alle $\langle p', a, q' \rangle$ mit $p' \in V_\epsilon(p)$, $q' \in N_\epsilon(q)$.

- ▶ Korollar: Es genügt, alle Kanten zu betrachten, in bei denen entweder eine ϵ -Kante im Startzustand endet oder eine ϵ -Kante im Zielzustand endet

- ▶ Entferne alle leeren Kanten aus Δ .

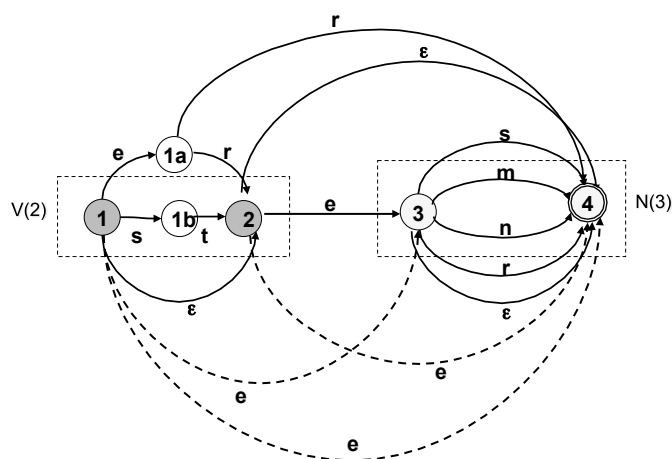
▶ 24

Schritt 2: Beseitigung von ϵ -Kanten



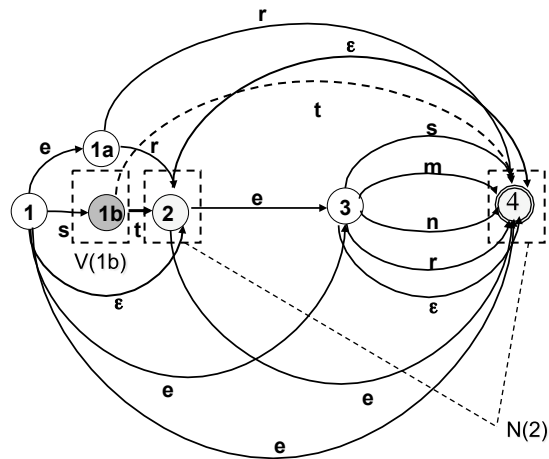
► 25

Schritt 2: Beseitigung von ϵ -kanten



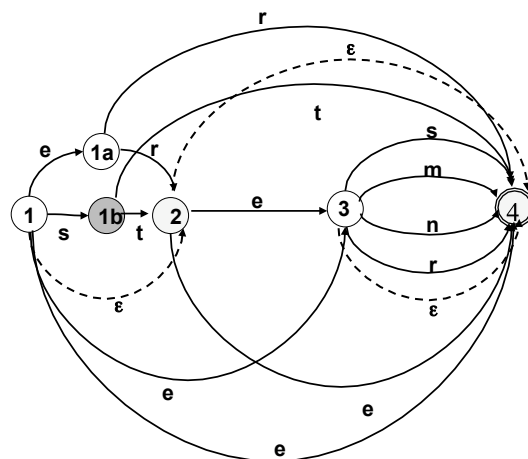
► 26

Schritt 2: Beseitigung von ϵ -kanten



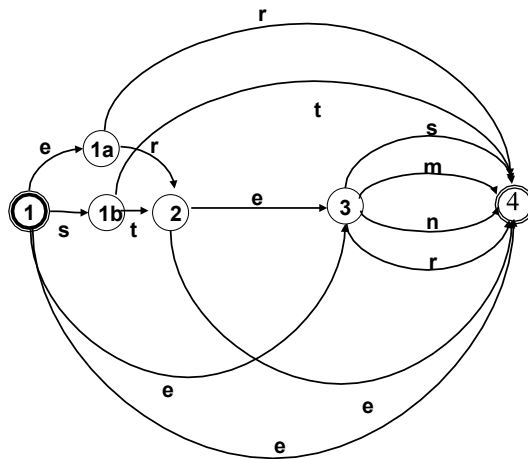
► 27

Schritt 2: Beseitigung von ϵ -kanten



► 28

Schritt 2: Beseitigung von ϵ -kanten: Resultat ist „buchstabierender Automat“



► 29

Schritt 2: Beseitigung von ϵ -kanten

- Wir definieren zunächst als Hilfsbegriffe den „ ϵ -Vorbereich“ $V_\epsilon(p)$ und den „ ϵ -Nachbereich“ $N_\epsilon(p)$ von Zuständen:

- $V_\epsilon(p) = \{q \mid p \text{ ist von } q \text{ aus ohne Abarbeiten eines Symbols erreichbar}\}$
- $N_\epsilon(p) = \{q \mid q \text{ ist von } p \text{ aus ohne Abarbeiten eines Symbols erreichbar}\}$

Anmerkung: $V_\epsilon(p)$ und $N_\epsilon(p)$ enthalten insbesondere p selbst.

- Für jede nicht-leere Kante $\langle p, a, q \rangle \in \Delta$: Erweitere Δ um alle $\langle p', a, q' \rangle$ mit $p' \in V_\epsilon(p)$, $q' \in N_\epsilon(q)$.
 - Korollar: Es genügt, alle Kanten zu betrachten, in bei denen entweder eine ϵ -Kante im Startzustand endet oder eine ϵ -Kante im Zielzustand endet
- Entferne alle leeren Kanten aus Δ .
- Wenn sich ein Endzustand im ϵ -Nachbereich des Startzustandes s befindet, füge s zu den Endzuständen hinzu.

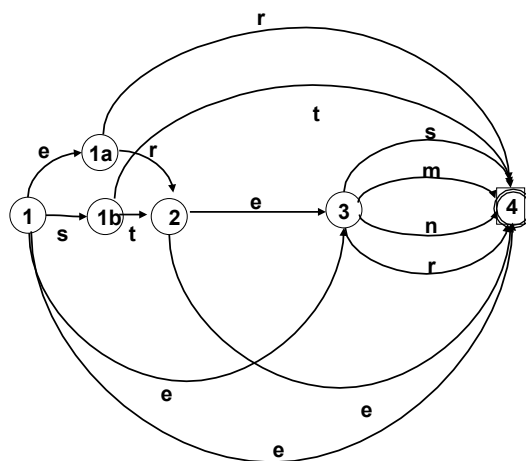
► 30

Schritt 3: Potenzautomaten-Konstruktion, Vorüberlegung

- ▶ Wir haben einen Algorithmus zur Pfadsuche am Beispiel des unbearbeiteten Adjektivendungs-Diagramms kennengelernt: „Tiefensuche mit Backtracking“. Durch die Organisation der Agenda als Stapel/Stack („last in – first out“) wird eine Alternative so weit wie möglich verfolgt; bei endgültigem Scheitern wird das System zurückgesetzt.
- ▶ Durch die Organisation der Agenda als Warteschlange (queue), bei der die Aufgaben in der Reihenfolge ihrer Generierung abgearbeitet werden („first in – first out“), erhalten wir Breitensuche. Die alternativen Pfade werden (quasi) parallel verfolgt.

▶ 31

Pfadsuche als Breitensuche

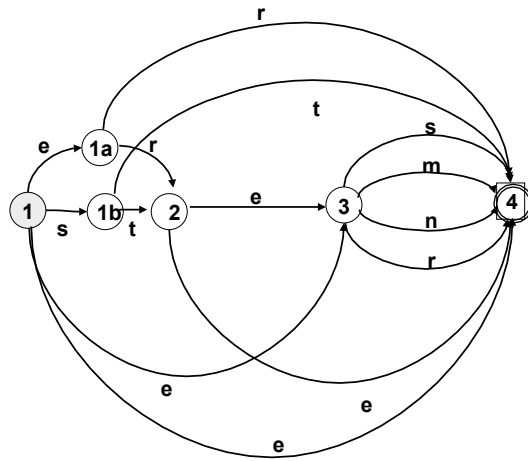


Eingabewort:

Agenda: 1 -- klein_eres

▶ 32

Pfadsuche als Breitensuche

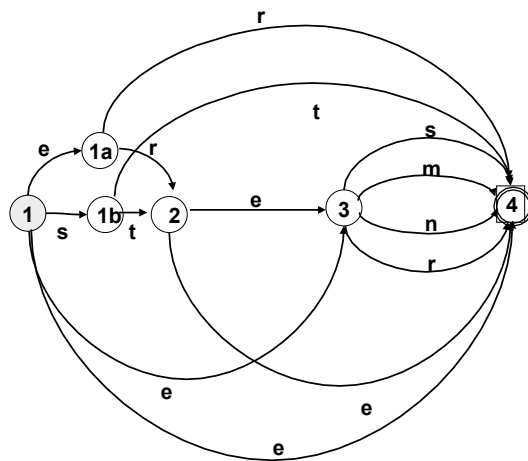


Eingabewort: klein eres

Agenda: _____

▶ 33

Pfadsuche als Breitensuche

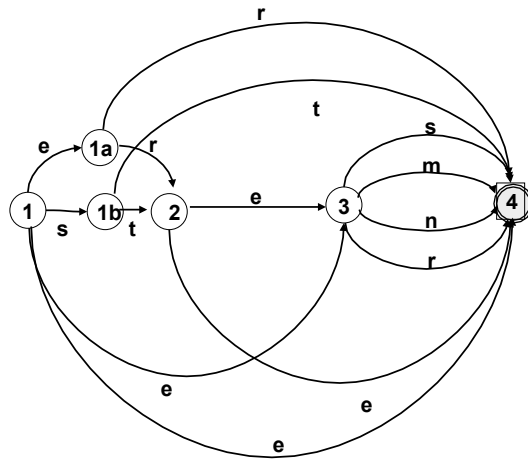


Eingabewort: klein eres

Agenda: 4 -- klein eres

▶ 34

Pfadsuche als Breitensuche

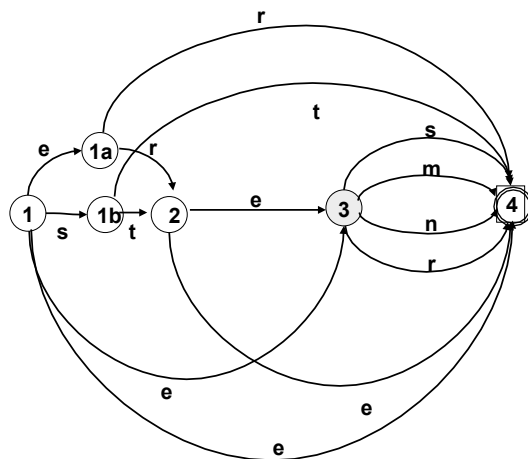


Eingabewort: klein_eres

Agenda: 3 -- klein_eres

▶ 35

Pfadsuche als Breitensuche

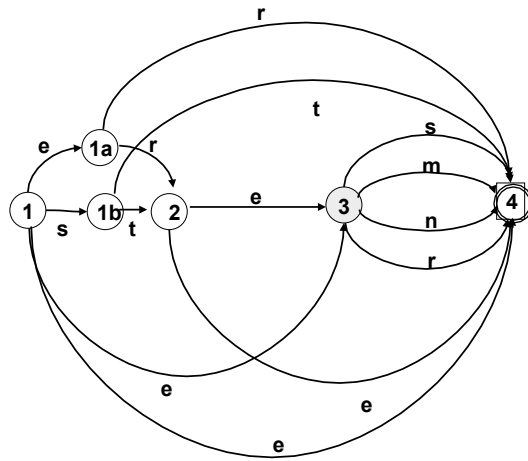


Eingabewort: klein_eres

Agenda: 1a -- klein_eres

▶ 36

Pfadsuche als Breitensuche

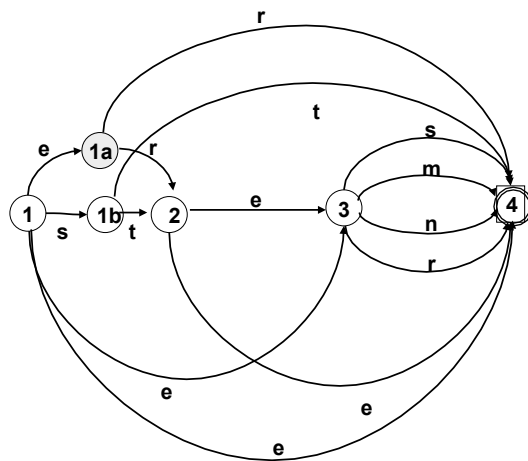


Eingabewort: klein eres

Agenda: 1a -- klein eres

▶ 37

Pfadsuche als Breitensuche

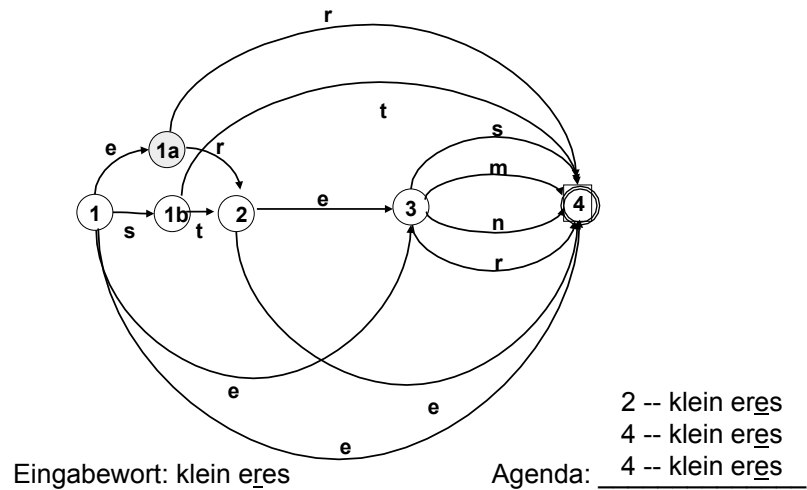


Eingabewort: klein eres

Agenda: 4 -- klein eres

▶ 38

Pfadsuche als Breitensuche



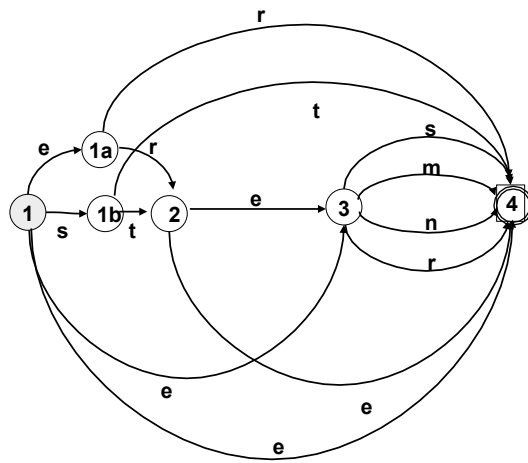
▶ 39

Schritt 3: Potenzautomaten-Konstruktion, Vorüberlegung [2]

- ▶ Wir können „getaktete“ Breitensuche in einem buchstabierenden NEA so beschreiben:
 - ▶ Wir ermitteln alle Zustände, die durch die Abarbeitung des ersten Eingabesymbols vom Startzustand aus erreicht werden können.
 - ▶ Wir ermitteln alle Zustände, die durch die Abarbeitung des zweiten Eingabesymbols von einem Zustand dieser Zustandsmenge erreicht werden können, usf.
 - ▶ Wenn die Zustandsmenge, die wir auf diese Weise nach Abarbeiten des kompletten Wortes w erhalten, einen Endzustand des NEA enthält, wird w akzeptiert.

▶ 40

Pfadsuche als Breitensuche

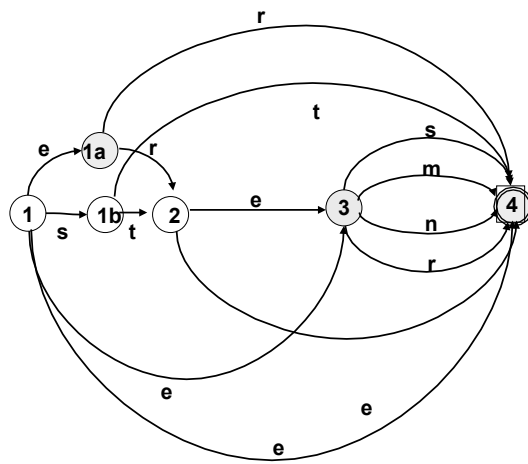


Eingabewort: klein eres

Agenda: 1a -- klein eres
 3 -- klein eres
 4 -- klein eres

► 41

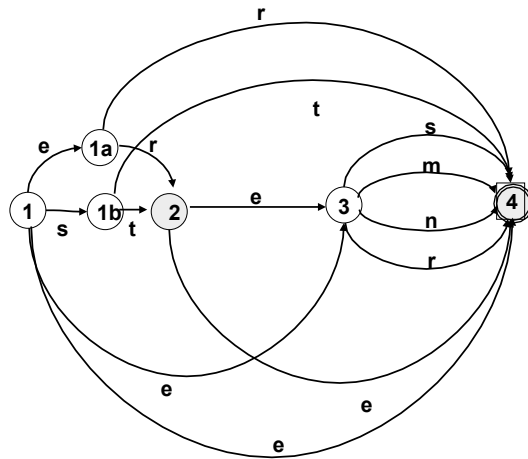
Pfadsuche als Breitensuche



Eingabewort: klein eres

► 42

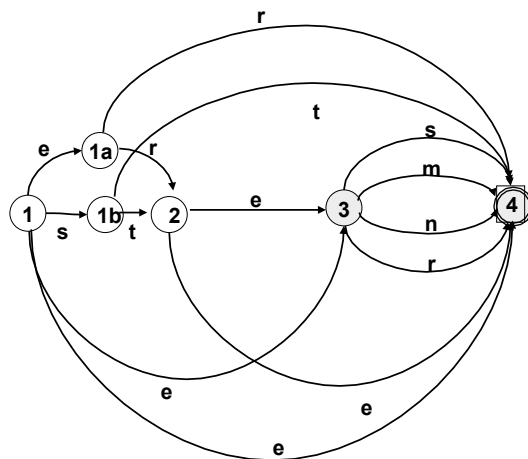
Pfadsuche als Breitensuche



Eingabewort: klein ers

► 43

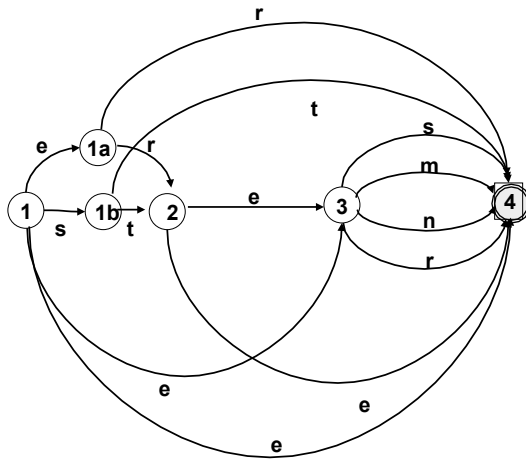
Pfadsuche als Breitensuche



Eingabewort: klein ers

► 44

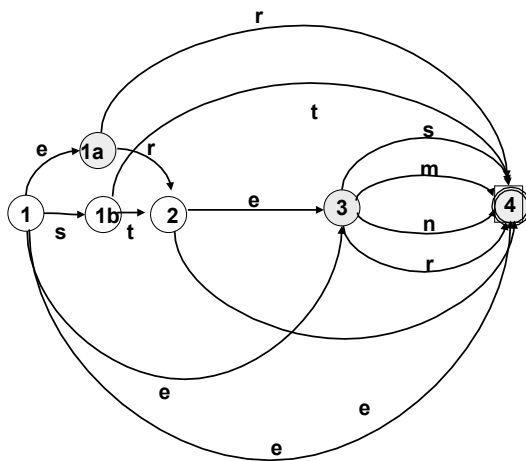
Pfadsuche als Breitensuche



Eingabewort: klein eres_

► 45

Pfadsuche als Breitensuche



Eingabewort: klein er_

► 46

Schritt 3: Potenzautomaten-Konstruktion, Vorüberlegung [3]

- ▶ Wir können diese "getaktete Suche" selbst mit einem endlichen Automaten beschreiben:
 - ▶ Zustände des neuen Automaten lassen sich als Mengen von Zuständen des NEA beschreiben. Am Beispiel: Nach Abarbeiten des ersten Symbols „e“ befindet er sich in dem Zustand, dass es die Zustandsmenge des NEA $\{1, 2, 4\}$ als mögliche aktuelle Zustände erkannt hat.
 - ▶ Wenn die Eingabekette abgearbeitet ist, und der Automat sich in einem Zustand befindet, der einen Endzustand des NEA enthält, ist die Eingabe akzeptiert.
 - ▶ Die „möglichen Zustände“ des NEA, die sich durch ein bestimmtes Eingabe-Symbol erreichen lassen, sind eindeutig definiert. Der neue Automat ist also ein DEA.

▶ 47

Schritt 3: Potenzautomaten-Konstruktion: Die Definition

Der Potenzautomat zum buchstabierenden NEA

$A = \langle K, \Sigma, \Delta, s, F \rangle$ ist der DEA A' :

$A' = \langle K', \Sigma, \delta, s', F' \rangle$ mit:

- ▶ $K' = \wp(K)$ (die Potenzmenge der Zustandsmenge des NEA)
- ▶ $s' = \{s\}$
- ▶ $\delta(p', a) = \{q' \mid \text{es gibt } p \in p' \text{ und } \langle p, a, q \rangle \in \Delta\}$ für jedes $p' \subseteq K, a \in \Sigma$
- ▶ $q' \in F'$ gdw. $q' \cap F \neq \emptyset$

▶ 48

Beobachtung

Der Potenzautomat A' zu $A = \langle K, \Sigma, \Delta, s, F \rangle$ hat $2^{|\Delta|}$ Zustände. In der Regel sind viele dieser Zustände unerreichbar (vom Startzustand $\{s\}$ aus) und deshalb funktionslos.

Praktisches Konstruktionsverfahren:

Beginne mit $\{s\}$, berechne die Übergangsfunktion für $\{s\}$, für alle direkt von s erreichbaren Zustände usw., bis keine neuen erreichbaren Zustände hinzukommen.

▶ 49

Beispiel: DEA für Adjektiv-Endungen

- ▶ Grundlage: der buchstabierende Automat

$$A = \langle \{1, 1a, 1b, 2, 3, 4\}, \{e, m, n, r, s, t\}, \Delta, 1, \{1, 4\} \rangle,$$

Δ wie im Diagramm Folie 40

- ▶ Potenzautomat ist $A' = \langle K', \Sigma, \delta, s', F' \rangle$

mit $K' = \wp(K)$

$$s' = \{s\}$$

$$F' = \{q' \in K' \mid 1 \in q' \text{ oder } 4 \in q'\}$$

δ s. Übergangstabelle nächste Folie

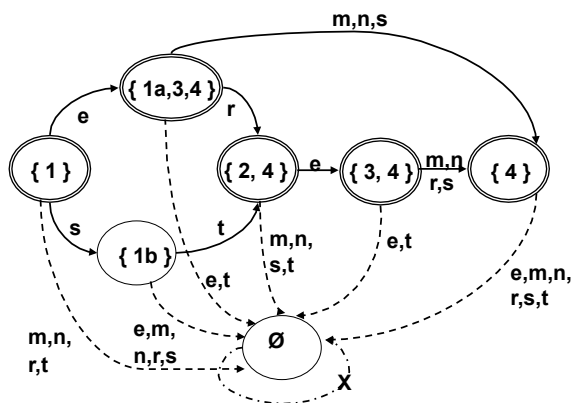
▶ 50

DEA für Adjektiv-Endungen, Übergangstabelle

δ :	e	m	n	r	s	t
{1}	{1a,3,4}	\emptyset	\emptyset	\emptyset	{1b}	\emptyset
{1a,3,4}	\emptyset	{4}	{4}	{2,4}	{4}	\emptyset
{1b}	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	{2,4}
{2,4}	{3,4}	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
{3,4}	\emptyset	{4}	{4}	{4}	{4}	\emptyset
{4}	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset

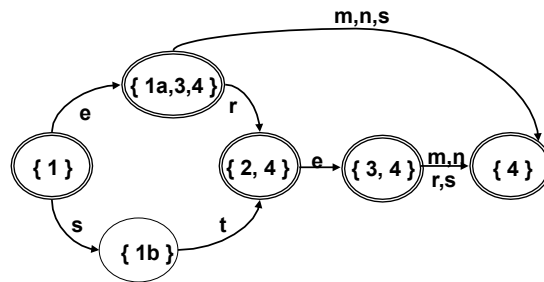
► 51

Das Diagramm



► 52

Das Diagramm, vereinfacht

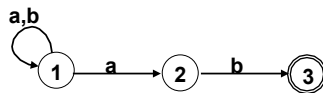


► 53

Potenzautomatenkonstruktion, ein weiteres Beispiel

NEA $A = \langle \{1,2,3\}, \{a,b\}, \Delta, 1, \{3\} \rangle$

Δ gegeben durch:



DEA

$A' = \langle \wp(\{1,2,3\}), \{a,b\}, \delta, \{1\}, F' \rangle$

$F' = \{\{3\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$

► 54

Potenzautomatenkonstruktion, Beispiel 2:
Die Übergangstabelle

q	$\delta(q, a)$	$\delta(q, b)$
{1}	{1,2}	{1}
{2}	\emptyset	{3}
{3}	\emptyset	\emptyset
{1,2}	{1,2}	{1,3}
{1,3}	{1,2}	{1}
{2,3}	\emptyset	{3}
{1,2,3}	{1,2}	{1,3}
\emptyset	\emptyset	\emptyset

► 55

Potenzautomatenkonstruktion, Beispiel 2:
Die Übergangstabelle

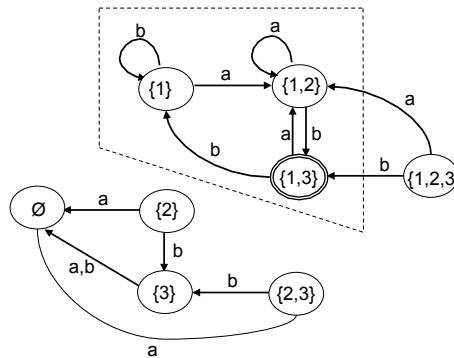
q	$\delta(q, a)$	$\delta(q, b)$
{1}	{1,2}	{1}
{2}	\emptyset	{3}
{3}	\emptyset	\emptyset
{1,2}	{1,2}	{1,3}
{1,3}	{1,2}	{1}
{2,3}	\emptyset	{3}
{1,2,3}	{1,2}	{1,3}
\emptyset	\emptyset	\emptyset

► 56

Potenzautomatenkonstruktion, Beispiel 2: Das Zustandsdiagramm

Nur ein Teil der
Zustände ist vom
Startzustand aus
erreichbar.

Die übrigen
Zustände sind
funktionslos.



► 57

Schlußbemerkungen über Morphologie

- ▶ Flexionsmorphologie: Lemmatisierung
 - ▶ *veranstalt+et, Veranstaltung+en*
- ▶ Ableitungs-/Derivationsmorphologie
 - ▶ *Veranstalt+ung, un+glaubwürdig*
- ▶ Komposita-Zerlegung
 - ▶ *Fach+veranstaltung, glaub+würdig*

► 58

Lemmatisierung/Stemming

- ▶ Methode: Endliche Automaten + Flexionsklassen-Information im Lexikon + Regeln für vorhersagbare morphophonologische Prozesse
- ▶ Morphologische Analyse (und damit auch deren Umkehrung, die Wortformengenerierung) ist für Lemmatisierung gelöst.
 - ▶ Ausnahmen werden im Lexikon kodiert
- ▶ Verbleibendes Problem für das Deutsche: Trennbare Präfixverben.
- ▶ Kommerzielle Systeme haben eine gute Abdeckung (bis ca. 98%). Abdeckungsprobleme sind meist auf Kompositazerlegung zurückzuführen.

▶ 59

Wortbildungsmorphologie: Ableitung/ Derivationsmorphologie

- ▶ Analyse liefert über den Wortstamm Information zur Ableitung der Bedeutung und über das Suffix Wortart- und Flexionsklasseninformation: Vervielfältig+ung bezeichnet den Vorgang des Vervielfältigens, ist feminines Substantiv und wird schwach flektiert.
- ▶ Zwei Probleme, die zusammenhängen:
 - ▶ viele Ableitungspräfixe und -suffixe sind semiproduktiv
 - ▶ viele Ableitungen sind semantisch "nicht transparent": Sie haben eine konventionelle, lexikalisierte Bedeutung, die sich nicht aus den Bestandteilen erschließen lässt.
- ▶ Derivationsanalyse führt aus diesen Gründen zur Übergenerierung:
 - ▶ die *Lesung* bezeichnet den Akt des Vorlesens,
 - ▶ die *Vorlesung* aber nicht den Akt des Vorlesens,
 - ▶ die *Schreibung* nicht den Akt des Schreibens, und
 - ▶ die *Singung* ist ganz unmöglich

▶ 60

Wortbildungsmorphologie: Ableitung/ Derivationsmorphologie

- ▶ Die Analyse von Derivativen ist in Morphologiesystemen meist mehr oder weniger ausführlich mit integriert. Sie kann grundsätzlich mit endlichen Automaten realisiert werden.
- ▶ Wegen Semiproduktivität und fehlender semantischer Transparenz ist es sinnvoll, Wissen über Ableitungen im großen Umfang im Lexikon zu kodieren.

▶ 61

Kompositazerlegung

- ▶ Die Analyse von Komposita ist besonders in Sprachen wie dem Deutschen unerlässlich (potenziell beliebige Länge von Komposita und deshalb unbegrenzt großer Wortschatz).
- ▶ Rechtschreibprüfung ohne oder mit begrenzter Kompositabehandlung führt zu vielen korrekten, aber unerkannten Wörtern (MS Word bis vor einigen Jahren).

▶ 62

Kompositazerlegung: Probleme

- ▶ **Fugenelemente:**
 - ▶ sind keine Flexionssuffixe, vgl.: "Absichts" in Absichtserklärung
 - ▶ sind nicht vollständig aus der phonologischen/ orthografischen Umgebung vorhersagbar
 - ▶ gehören nicht zu der Information, die in Einträgen konventioneller Wörterbücher mit aufgeführt wird.
- ▶ **Übergenerierung:**
 - ▶ Beispiel zur Konversion alte-neue Rechtschreibung (falsch angewandte Drei-Konsonanten-Regel)
 - ▶ Hufeisenniere → Huf|e|isenn|niere
 - ▶ Gute Morphologien haben eine Kompositazerlegung mit guter Abdeckung und akzeptablem Übergenerierungsverhalten.
 - ▶ Zur Vermeidung von Übergenerierung werden Blockierungslisten mit kurzen und seltenen Wörtern angelegt. Komposita, die diese Wörter enthalten, werden explizit im Lexikon aufgeführt.