Automatentheorie und ihre Anwendungen
Teil 2: endliche Automaten auf endlichen Bäumen

Thomas Schneider

28. April – 13. Mai 2015

Überblick

1. *Motivation: semistrukturierte Daten*
2. Grundbegriffe
3. Charakterisierungen erkennbarer Baumsprachen
4. Top-down-Baumautomaten
5. Abschlusseigenschaften
6. Entscheidungsprobleme
7. *Anwendung 2: XML-Schemasprachen*

Und nun ...

1. *Motivation: semistrukturierte Daten*
2. Grundbegriffe
3. Charakterisierungen erkennbarer Baumsprachen
4. Top-down-Baumautomaten
5. Abschlusseigenschaften
6. Entscheidungsprobleme
7. *Anwendung 2: XML-Schemasprachen*

Semistrukturierte Daten sind ...

- ein Datenmodell zur Beschreibung von **Entitäten** und **Attributen**,

  das weniger formale Struktur voraussetzt als z. B. relationale Datenbanken
- ein Vorläufer von XML
- gut geeignet, um
  - Dokumentansichten (z. B. Webseiten) und
  - strukturierte Daten (z. B. Datenbank-Tabellen)

  zu repräsentieren und miteinander zu verbinden

Merkmale semistrukturierter Daten

Daten werden in Entitäten (Einheiten) zusammengefasst

- Markierung von Entitäten durch Tags
- Bildung von Hierarchien
- Gruppieren ähnlicher Entitäten
- Entitäten derselben Gruppe können verschiedene (oder keine) Attribute haben
- Reihenfolge der Attribute *kann* eine Rolle spielen (Mengen oder Listen z. B. von Telefonnummern?)

Beispiel:

```
Person: {Name: {VN: "Thomas", NN: "Schneider"},
         Telnr: 64432,
         Telnr: 43776243,
         Email: "ts@informatik..."}
```

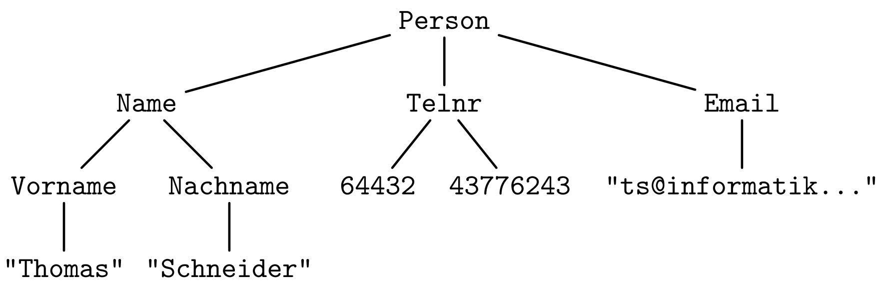

Datenstruktur: Baum

?

Datenstruktur: Baum

```
Person: {Name: {VN: "Thomas", NN: "Schneider"},
         Telnr: 64432,
         Telnr: 43776243,
         Email: "ts@informatik..."}
```

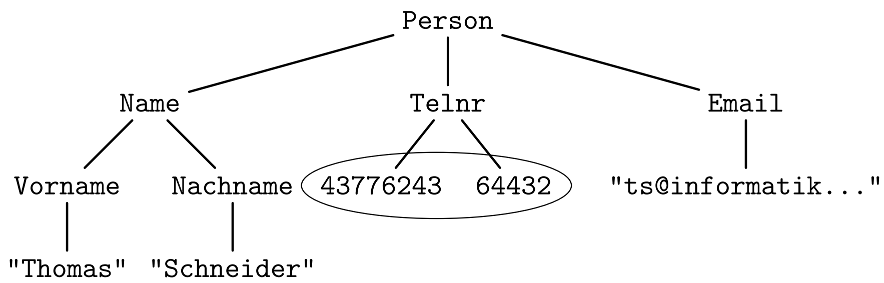
Repräsentation im Baum ist naheliegend:



Datenstruktur: Baum

```
Person: {Name: {VN: "Thomas", NN: "Schneider"},
         Telnr: 64432,
         Telnr: 43776243,
         Email: "ts@informatik..."}
```

Ist das derselbe oder ein anderer Baum?



Automaten auf endlichen Bäumen

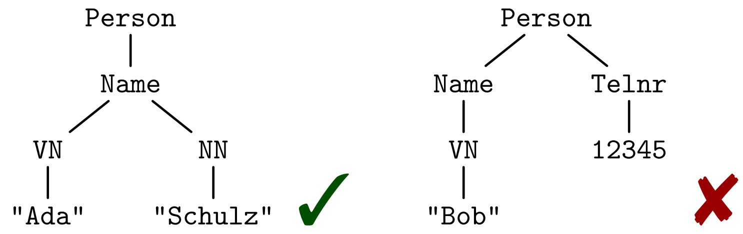
… sind wichtig für semistrukturierte Daten, weil sie …

- XML-Schemasprachen und -validierung zugrunde liegen
- XML-Anfragesprachen auf ihnen aufgebaut sind

XML-Schemasprachen und -validierung

- Schemasprachen zur Beschreibung gültiger Bäume
- Validierung = Erkennen gültiger und ungültiger Bäume
- gängige Schemasprachen benutzen dafür Baumautomaten

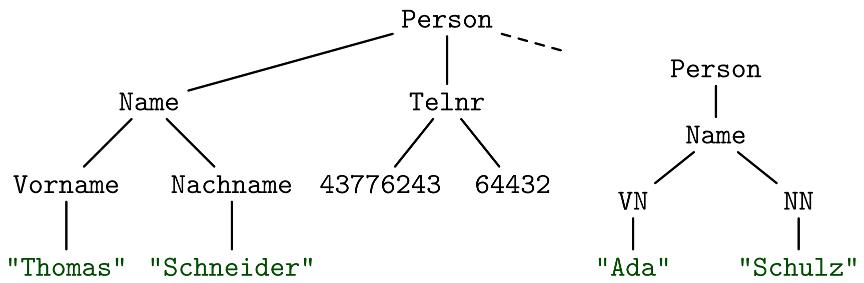
Beispiel: jeder Name muss einen Vor- und Nachnamen haben



XML-Anfragesprachen

- beantworten Anfragen mit Daten aus gegebenen Bäumen

Beispiel: gib alle Namen von Personen zurück



Und nun …

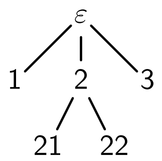
1. *Motivation: semistrukturierte Daten*
2. Grundbegriffe
3. Charakterisierungen erkennbarer Baumsprachen
4. Top-down-Baumautomaten
5. Abschlusseigenschaften
6. Entscheidungsprobleme
7. *Anwendung 2: XML-Schemasprachen*

Positionen im Baum

- **positive** natürliche Zahlen: $\mathbb{N}_+$
- **Position:** Wort $p \in \mathbb{N}_+^*$

Idee: Wurzel ist ε
j-ter Nachfolger von p ist pj

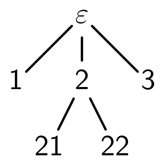
Beispiel:



Alphabet mit Stelligkeit

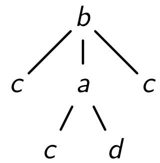
- hier: **r-Alphabet** Σ (auf Englisch: *ranked alphabet*)
- nichtleere endliche Menge von Symbolen; jedem Symbol ist eine Stelligkeit $\in \mathbb{N}$ zugeordnet
- Σ_m = Menge der Symbole mit Stelligkeit m
- Schreibweise: $\Sigma = \{a_1/r_1, \dots, a_n/r_n\}$ heißt: Σ enthält die Symbole a_i mit Stelligkeit r_i, $i = 1, \dots, n$

Beispiel: $\Sigma = \{a/2,\ b/3,\ c/0,\ d/0\}$



a „passt" in Position 2
b „passt" in Position ε
c, d „passen" in Pos. $1, 21, 22, 3$

$\rightsquigarrow$



Baum über Σ

Was genau ist eigentlich ein Baum?

?

Was ist ein Baum?

Definition 1

Ein **endlicher geordneter Baum** über dem r-Alphabet Σ ist ein Paar $T = (P, t)$, wobei

- $P \subseteq \mathbb{N}_+^*$ eine nichtleere endl. präfix-abgeschlossene Menge ist,
- $t : P \to \Sigma$ eine Funktion ist mit den folgenden Eigenschaften.
  1. Wenn $t(p) \in \Sigma_0$, dann $\{j \mid pj \in P\} = \emptyset$.
  2. Wenn $t(p) \in \Sigma_m$, $m \geqslant 1$, dann $\{j \mid pj \in P\} = \{1, \dots, m\}$.

Erklärungen:

- P: Menge der vorhandenen Positionen
- Präfix-Abgeschlossenheit: Baum ist wohlgeformt (z. B.: wenn Position 31 existiert, dann auch Position 3 und ε)

Was ist ein Baum?

Bezeichnungen

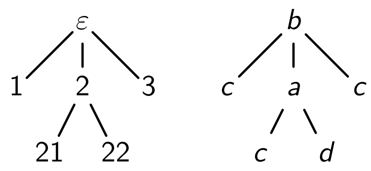
- Position p hat **Kinder** $p1, p2, \ldots$;
  p ist deren **Elternteil**
- jedes Präfix von p ist ein **Vorgänger** von p;
  p ist **Nachfolger** eines jeden Präfixes von p
- **Blatt:** Knoten ohne Kinder
- **Höhe von p in T:**
  Länge des längsten Pfades von p zu einem Blatt
- **Höhe von T:** Höhe von ε in T

Beispiel

$$\Sigma = \{a/2,\ b/3,\ c/0,\ d/0\}$$
$$P = \{\varepsilon, 1, 2, 3, 21, 22\}$$

$$t(\varepsilon) = b,\ t(1) = c,\ t(2) = a,\ t(3) = c,\ t(21) = c,\ t(22) = d$$

Positionen P Baum $T = (P, t)$



andere Schreibweise:

$b(ca(cd)c)$

($\approx$ in-order-Tiefensuche)

- Höhe: 2
- Blätter: 1, 21, 22, 3
- 21 ist Kind von 2 und hat Vorgänger 2, ε

Bottom-up-Baumautomaten

Definition 2

Ein **nichtdet. Bottom-up-Automat auf endl. geord. Bäumen (NEBA)** ist ein Quadrupel $\mathcal{A} = (Q, \Sigma, \Delta, F)$, wobei

- Q eine endliche nichtleere **Zustandsmenge** ist,
- Σ ein r-Alphabet ist,
- Δ eine Menge von **Überführungsregeln** der Form

$$a(q_1, \ldots, q_m) \rightarrow q$$

  ist mit $m \geqslant 0$, $a \in \Sigma_m$, $q, q_1, \ldots, q_m \in Q$, und
- $F \subseteq Q$ die Menge der **Endzustände** ist.

Bottom-up-Baumautomaten: Intuitionen

Bedeutung der Überführungsregeln $a(q_1, \ldots, q_m) \rightarrow q$:

- Wenn $\mathcal{A}$ in Position p Zeichen a liest
- und in p's Kindern Zustände $q_1, \ldots, q_m$ eingenommen hat,

dann darf $\mathcal{A}$ in p Zustand q einnehmen. (Skizze siehe Tafel)

$\rightsquigarrow$ **Andere Betrachtungsweise:**

- $\mathcal{A}$ markiert Eingabebaum T **bottom-up** mit Zuständen
- $\mathcal{A}$ akzeptiert T, wenn $\mathcal{A}$ in der Wurzel einen EZ einnimmt

Was sind dann die Anfangszustände?

- Ü-Regeln $a() \rightarrow q$ deklarieren "zeichenspezifische" AZ:
  $\mathcal{A}$ darf in mit a markierten Blättern in q starten
- Kurzschreibweise: $a \rightarrow q$

Berechnungen (analog zu NEAs)

Definition 3

Sei $\mathcal{A} = (Q, \Sigma, \Delta, F)$ ein NEBA und $T = (P, t)$ ein Σ-Baum.

- **Berechnung (Run)** von $\mathcal{A}$ auf T ist eine Fkt. $r : P \to Q$ mit:
  - Wenn $t(p) = a \in \Sigma_0$ und $r(p) = q$, dann $a \to q \in \Delta$.
  - Wenn $t(p) = b \in \Sigma_m$ $(m \geqslant 1)$ und $r(p) = q$ und wenn $r(p1) = q_1, \quad \dots, \quad r(pm) = q_m$, dann $b(q_1, \dots, q_m) \to q \in \Delta$.

Also gilt:

- Blatt mit a kann q nur zugewiesen kriegen, wenn $a \to q \in \Delta$.
- Nicht-Blatt mit b, dessen Kinder $q_1, \dots, q_m$ haben, kann q nur zugew. kriegen, wenn $b(q_1, \dots, q_m) \to q \in \Delta$.

Berechnungen, Akzeptanz und erkannte Sprache

Definition 3

Sei $\mathcal{A} = (Q, \Sigma, \Delta, F)$ ein NEBA und $T = (P, t)$ ein Σ-Baum.

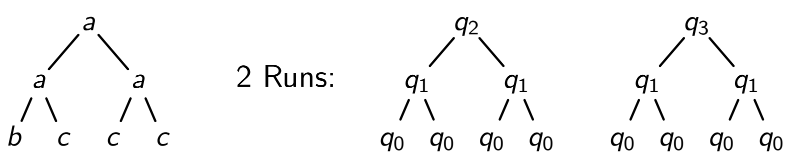
- **Berechnung (Run)** von $\mathcal{A}$ auf T ist eine Fkt. $r : P \to Q$ mit:
  - Wenn $t(p) = a \in \Sigma_0$ und $r(p) = q$, dann $a \to q \in \Delta$.
  - Wenn $t(p) = b \in \Sigma_m$ $(m \geqslant 1)$ und $r(p) = q$ und wenn $r(p1) = q_1, \quad \dots, \quad r(pm) = q_m$, dann $b(q_1, \dots, q_m) \to q \in \Delta$.
- Ein Run r von $\mathcal{A}$ auf T ist **akzeptierend**, wenn $r(\varepsilon) \in F$.
- $\mathcal{A}$ **akzeptiert** T, wenn es einen akz. Run von $\mathcal{A}$ auf T **gibt**.
- Die von $\mathcal{A}$ **erkannte Sprache** ist $L(\mathcal{A}) = \{T \text{ über } \Sigma \mid \mathcal{A} \text{ akzeptiert } T\}$.

Beispiel 1

- Sei $\Sigma = \{a/2, b/0, c/0\}$ und $\mathcal{A} = (\{q_0, \dots, q_3\}, \Sigma, \Delta, \{q_2\})$

$$\text{mit } \Delta = \{\ b \to q_0, \quad c \to q_0, \quad a(q_0, q_0) \to q_1, \quad a(q_1, q_1) \to q_2, \quad a(q_1, q_1) \to q_3\ \}.$$

- Dann gibt es auf dem Baum



- $L(\mathcal{A}) = \{T \text{ über } \Sigma \mid \text{alle Pfade in } T \text{ haben Länge } 2\}$
- **Anmerkung:** Da Σ nur ./2 und ./0 enthält:

  $L(\mathcal{A}) = \{T \text{ über } \Sigma \mid T \text{ ist vollst. Binärbaum der Tiefe } 2\}$

Beispiel 2

Sei $\Sigma = \{a/2,\ b/1,\ c/0,\ d/0\}$. Welcher NEBA erkennt $\{T \text{ über } \Sigma \mid \text{ jedes } c\text{-Blatt hat ein rechtes } d\text{-Geschwister}\}$?

$\mathcal{A} = (\{q_c, q_d, q_f\}, \Sigma, \Delta, \{q_f\})$ mit

$$\Delta = \{ \quad c \to q_c, \quad a(q_c, q_d) \to q_f, \quad d \to q_d, \quad d \to q_f, \quad b(q_f) \to q_f, \quad a(q_f, q_f) \to q_f \ \}$$

Übergang $a(q_d, q_d) \to q_f$ ist überflüssig: $d \to q_f$ und $a(q_f, q_f) \to q_f$.

Beispielbaum und -run: siehe Tafel

Erkennbare Baumsprache

Definition 4

Eine Menge L von (endlichen geordneten) Bäumen über Σ ist eine **erkennbare Baumsprache**,

wenn es einen NEBA $\mathcal{A}$ gibt mit $L(\mathcal{A}) = L$.

Determinismus

Definition 5

Sei $\mathcal{A} = (Q, \Sigma, \Delta, F)$ ein NEBA.

Enthält Δ für jedes jedes $a \in \Sigma_m$ und alle $(q_1, \dots, q_m) \in Q^m$ **höchstens eine**[1] Regel $a(q_1, \dots, q_m) \to q$

dann ist $\mathcal{A}$ ein **deterministischer endlicher Baumautomat (DEBA)**.

$\rightsquigarrow$ Nachfolgezustand für jedes $(m+1)$-Tupel $a(q_1, \dots, q_m)$ ist eindeutig bestimmt (wenn er existiert)

- Jeder DEBA ist ein NEBA, aber nicht umgekehrt (z. B. die vergangenen 2 Beispiele).

Frage

Sind DEBAs und NEBAs gleichmächtig?

[1] „höchstens eine“ ist besser als „genau eine“: vermeidet Papierkorbzustand

Potenzmengenkonstruktion

Antwort

Ja, DEBAs und NEBAs sind gleichmächtig.

Satz 6

Sei L eine erkennbare Baumsprache.

Dann gibt es einen DEBA, der L erkennt.

Beweis: siehe Tafel.

Auch für NEBAs kann die Potenzmengenkonstruktion im schlimmsten Fall zu exponentiell vielen Zuständen führen.

Und nun ...

1. *Motivation: semistrukturierte Daten*
2. Grundbegriffe
3. Charakterisierungen erkennbarer Baumsprachen
4. Top-down-Baumautomaten
5. Abschlusseigenschaften
6. Entscheidungsprobleme
7. *Anwendung 2: XML-Schemasprachen*

Nichterkennbare Baumsprachen: Intuitionen

Beispiel:

- r-Alphabet $\Sigma = \{a/2,\ b/1,\ c/0\}$
- Baumautomat $\mathcal{A} = (\{q_0, q_1\}, \Sigma, \Delta, \{q_0\})$ mit

$$\Delta = \{\ c \to q_0,\quad b(q_0) \to q_1,\quad a(q_0, q_0) \to q_1,$$
$$b(q_1) \to q_0,\quad a(q_1, q_1) \to q_0\ \}.$$

$\rightsquigarrow L(\mathcal{A}) = \{T \mid$ alle Wurzel-Blatt-Pfade in T haben gerade Länge$\}$.
$\neq \{T \mid T$ hat gerade Höhe$\}$ (Bsp. s. Tafel) •

Frage: Sind die folgenden Baumsprachen erkennbar?

$L_1 = \{T \mid T$ hat gerade Höhe$\}$
$L_2 = \{T \mid T$ ist vollständiger Binärbaum$\}$ $(\Sigma = \{a/2,\ c/0\})$ •

Antwort: **Nein.** Idee s. Tafel •

Pumping-Lemma: Hilfsbegriffe

- **Variable**: zusätzliches nullstelliges Symbol $x \notin \Sigma_0$
- **(unärer) Kontext**:
  Baum über $\Sigma \cup \{x\}$, in dem ein Blatt mit x markiert ist •
- **trivialer Kontext** C_0: Kontext der Höhe 0 ($\Rightarrow$ nur Wurzel)
- **Einsetzen** von Bäumen/Kontexten in Kontexte:
  - $C[T]$ = der Baum/Kontext, den man aus C erhält, indem man die Position von x mit Baum/Kontext T ersetzt •
  - C^n induktiv definiert:

$$C^0 = C_0$$
$$C^{n+1} = C^n[C]$$

Pumping-Lemma

Satz 7 (Pumping-Lemma)

Wenn L eine erkennbare Baumsprache über dem r-Alphabet Σ ist,

dann gibt es eine Konstante $k \in \mathbb{N}$,
so dass für alle Bäume $T \in L$ mit $\text{Höhe}(T) \geqslant k$ *gilt:*

Es gibt Kontexte C, D mit $D \neq C_0$ und Baum V mit $T = C[D[V]]$,
so dass $C[D^i[V]] \in L$ für alle $i \in \mathbb{N}$.

Beweis: siehe Tafel. •

□

Anwendung des Pumping-Lemmas

Benutzen Kontraposition (siehe Kapitel „endliche Wörter“):

Wenn es **für alle** Konstanten $k \in \mathbb{N}$
einen Baum $T \in L$ mit $\text{Höhe}(T) \geqslant k$ **gibt**, so dass es
für alle Kontexte C, D mit $D \neq C_0$ und Bäume V mit $T = C[D[V]]$
ein $i \in \mathbb{N}$ **gibt** mit $C[D^i[V]] \notin L$,

dann ist L **keine** erkennbare Baumsprache.

Bsp.: s. Tafel •

Der Satz von Myhill-Nerode für Baumsprachen

Ziel: notwendige **und** hinreichende Bedingung für Erkennbarkeit

Definition 8

Sei L eine Baumsprache über Σ.

Zwei Σ-Bäume T_1, T_2 sind **L-äquivalent** (Schreibw.: $T_1 \sim_L T_2$), wenn für alle Σ-Kontexte C gilt:

$$C[T_1] \in L \quad \text{genau dann, wenn} \quad C[T_2] \in L$$

Satz 9 (Myhill-Nerode)

$L \subseteq \Sigma^*$ is NEBA-erkennbar gdw. $\sim_L$ endlichen Index hat.

Ohne Beweis. Beispiel siehe Tafel.

Auch für Baumsprachen gilt: endlicher Index n von $\sim_L$
= minimale Anzahl von Zuständen in einem DEBA, der L erkennt

Und nun . . .

1 Motivation: semistrukturierte Daten
2 Grundbegriffe
3 Charakterisierungen erkennbarer Baumsprachen
4 Top-down-Baumautomaten
5 Abschlusseigenschaften
6 Entscheidungsprobleme
7 Anwendung 2: XML-Schemasprachen

Drehen wir jetzt alles um? ☺

$\overset{?}{\rightsquigarrow}$

Nein, nur die Runs werden „umgedreht“.

Idee: Top-down-Baumautomaten

- weisen der Wurzel einen Startzustand zu und
- arbeiten sich dann von oben nach unten zu den Blättern durch

Definition 10

Ein **nichtdet. Top-down-Automat auf endl. geord. Bäumen (NETDBA)** ist ein Quadrupel $\mathcal{A} = (Q, \Sigma, \Delta, I)$, wobei

- Q eine endliche nichtleere **Zustandsmenge** ist,
- Σ ein r-Alphabet ist,
- Δ eine Menge von **Überführungsregeln** der Form

  $$(a, q) \to (q_1, \ldots, q_m)$$

  ist mit $m \geqslant 0$, $a \in \Sigma_m$, $q, q_1, \ldots, q_m \in Q$, und
- $I \subseteq Q$ die Menge der **Anfangszustände** ist.

Berechnungen, Akzeptanz und erkannte Sprache

Definition 11

Sei $\mathcal{A} = (Q, \Sigma, \Delta, I)$ ein NETDBA und $T = (P, t)$ ein Σ-Baum.

- **Berechnung (Run)** von $\mathcal{A}$ auf T ist eine Fkt. $r : P \to Q$ mit:
  - $r(\varepsilon) \in I$
  - Wenn $t(p) = a \in \Sigma_m$ $(m \geqslant 1)$ und $r(p) = q$ und wenn $r(p1) = q_1, \quad \dots, \quad r(pm) = q_m$, dann gibt es eine Regel $(a, q) \to (q_1, \dots, q_m) \in \Delta$.
  - Wenn $t(p) = a \in \Sigma_0$ und $r(p) = q$, dann $(a, q) \to () \in \Delta$.
- $\mathcal{A}$ **akzeptiert** T, wenn es einen Run von $\mathcal{A}$ auf T **gibt**.
- Die von $\mathcal{A}$ **erkannte Sprache** ist $L(\mathcal{A}) = \{T \text{ über } \Sigma \mid \mathcal{A} \text{ akzeptiert } T\}$.

Beachte: Keine Endzustände nötig – die Regeln in Δ müssen nur erlauben, von der Wurzel bis zu allen Blättern „durchzukommen".

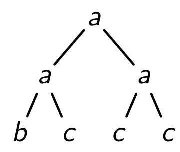
Beispiel 1

- Sei $\Sigma = \{a/2, \ b/0, \ c/0\}$ und

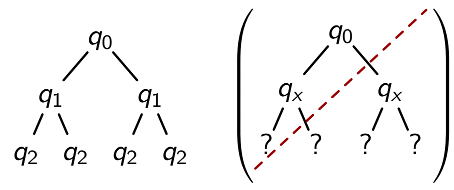
$\mathcal{A} = (\{q_0, q_1, q_2, q_x\}, \ \Sigma, \ \Delta, \ \{q_0\})$ mit

$$\begin{aligned} \Delta = \{ \ & (a, q_0) \to (q_1, q_1), && (b, q_2) \to (), \\ & (a, q_1) \to (q_2, q_2), && (c, q_2) \to (), \\ & (a, q_0) \to (q_x, q_x) && \} \end{aligned}$$

- Dann gibt es auf dem Baum



nur 1 Run:



- $L(\mathcal{A}) = \{T \text{ über } \Sigma \mid \text{alle Pfade in } T \text{ haben Länge } 2\}$

Beispiel 2

Sei $\Sigma = \{a/2, \ b/1, \ c/0, \ d/0\}$. Welcher NETDBA erkennt $L_{\text{cd}} = \{T \text{ über } \Sigma \mid \text{ jedes } c\text{-Blatt hat ein rechtes } d\text{-Geschwister}\}$?

NETDBA $\mathcal{A} = (\{q_0, q_c, q_d\}, \Sigma, \Delta, \{q_0\})$ mit

$$\begin{aligned} \Delta = \{ \ & (a, q_0) \to (q_0, q_0), \quad b(q_0) \to q_0, && (c, q_c) \to (), \\ & (a, q_0) \to (q_c, q_d), && (d, q_d) \to (), \\ & && (d, q_0) \to () \ \} \end{aligned}$$

Vergleiche mit dem NEBA $\mathcal{A} = (\{q_c, q_d, q_f\}, \Sigma, \Delta, \{q_f\})$ mit

$$\begin{aligned} \Delta = \{ \ & a(q_f, q_f) \to q_f, \quad b(q_f) \to q_f, && c \to q_c, \\ & a(q_c, q_d) \to q_f, && d \to q_d, \\ & && d \to q_f \ \} \end{aligned}$$

Was sagt uns das über das Verhältnis NETDBAs : NEBAs?

Sie sind gleichmächtig!

Satz 12

$\{L(\mathcal{A}) \mid \mathcal{A} \text{ ist ein NETDBA}\} \quad = \quad \{L(\mathcal{A}) \mid \mathcal{A} \text{ ist ein NEBA}\}.$

Beweis: siehe Übungsblatt 2

Determinisierung von NETDBAs

Erinnerung an Beispiel 2: Sei $\Sigma = \{a/2,\ b/1,\ c/0,\ d/0\}$ und $L_{\text{cd}} = \{T \text{ über } \Sigma \mid \text{ jedes } c\text{-Blatt hat ein rechtes } d\text{-Geschwister}\}$.

NETDBA $\mathcal{A} = (\{q_0, q_c, q_d\}, \Sigma, \Delta, \{q_0\})$ mit

$$\begin{aligned} \Delta = \{\ & (a, q_0) \to (q_0, q_0), \quad b(q_0) \to q_0, \quad (c, q_c) \to (), \\ & (a, q_0) \to (q_c, q_d), \quad (d, q_d) \to (), \\ & \quad (d, q_0) \to ()\ \} \end{aligned}$$

▲ Nichtdeterminismus!

Wir wissen ja, wie man Nichtdeterminismus „loswird“. **Oder?**

Determinisierung von NETDBAs?

Betrachte

- $\Sigma = \{a/2,\ b/0,\ c/0\}$ und
- die erkennbare Baumsprache $L = \{a(bc), a(cb)\}$.

(denke an die alternative Schreibweise von Folie 18)

Frage: Welcher DETDBA erkennt L?

Anwort: Keiner!

Lemma 13

L wird von keinem DETDBA erkannt.

Beweis: siehe Tafel.

Satz 14

Es gibt erkennbare Baumsprachen, die nicht von einem DETDBA erkannt werden.

Und nun ...

1. Motivation: semistrukturierte Daten
2. Grundbegriffe
3. Charakterisierungen erkennbarer Baumsprachen
4. Top-down-Baumautomaten
5. Abschlusseigenschaften
6. Entscheidungsprobleme
7. Anwendung 2: XML-Schemasprachen

Operationen auf Baumsprachen

Zur Erinnerung: die Menge der erkennbaren Baumsprachen heißt abgeschlossen unter ...

- **Vereinigung**, falls gilt:
  Falls L_1, L_2 erkennbar, so auch $L_1 \cup L_2$.
- **Komplement**, falls gilt:
  Falls L erkennbar, so auch $\overline{L}$.
- **Durchschnitt**, falls gilt:
  Falls L_1, L_2 erkennbar, so auch $L_1 \cap L_2$.

Quiz

Unter welchen Operationen gilt Abgeschlossenheit?

Vereinigung?
Komplement?
Durchschnitt?

Abgeschlossenheit

Satz 15

Die Menge der NEBA-erkennbaren Sprachen ist abgeschlossen unter den Operationen $\cup, \cap, \overline{}$.

Beweis: Direkte Konsequenz aus den folgenden Lemmata. □

Abgeschlossenheit unter Vereinigung

Lemma 16

Seien $\mathcal{A}_1, \mathcal{A}_2$ NEBAs über Σ.
Dann gibt es einen NEBA $\mathcal{A}_3$ mit $L(\mathcal{A}_3) = L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$.

Beweis: Seien $\mathcal{A}_i = (Q_i, \Sigma, \Delta_i, F_i)$ für $i = 1, 2$.
O. B. d. A. gelte $Q_1 \cap Q_2 = \emptyset$.

Konstruieren $\mathcal{A}_3 = (Q_3, \Sigma, \Delta_3, F_3)$ wie folgt.

- *Idee wie für Wortautomaten: vereinige $\mathcal{A}_1$ und $\mathcal{A}_2$.*
- $Q_3 = Q_1 \cup Q_2$
- $\Delta_3 = \Delta_1 \cup \Delta_2$
- $F_3 = F_1 \cup F_2$

Dann gilt $L(\mathcal{A}_3) = L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$. □

Abgeschlossenheit unter Komplement

Lemma 17

Sei $\mathcal{A}$ ein NEBA über Σ.
Dann gibt es einen NEBA $\mathcal{A}^c$ mit $L(\mathcal{A}^c) = \overline{L(\mathcal{A})}$.

Beweis:

- *Idee wie für Wortautomaten:*
  - Umwandlung in DEBA
  - Vertauschen von End- und Nicht-Endzuständen

O. B. d. A. sei $\mathcal{A} = (Q, \Sigma, \Delta, F)$ ein DEBA
mit **genau einem** Run pro Eingabebaum (so einer existiert wegen Satz 6).

Dann erkennt $\mathcal{A}^c = (Q, \Sigma, \Delta, Q \setminus F)$ die Sprache $\overline{L(\mathcal{A})}$. □

Abgeschlossenheit unter Durchschnitt

…folgt direkt aus der Abgeschlossenheit unter $\cup$ und $\overline{}$:

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

Alternative: Konstruktion des Produktautomaten wie für NEAs (vermeidet exponentielle "Explosion")

Abgeschlossenheit unter Verkettungsoperationen

Randbemerkung:
Man kann Analoga zu $\cdot$ und $*$ für Baumsprachen definieren:

Seien L, L_1, L_2 Baumsprachen.

Bezeichne $\mathsf{Con}(L)$ die Menge aller Kontexte, die man aus Bäumen in L erhält, indem man ein Blattsymbol durch x ersetzt.

- $L_1L_2 = \{C[T] \mid T \in L_1,\ C \in \mathsf{Con}(L_2)\}$
- $L^* = \{C_1[C_2[\dots[C_n[T]]\dots]] \mid T \in L,\ C_1,\dots,C_n \in \mathsf{Con}(L),\ n \geqslant 0\}$

Abgeschlossenheit unter $\cdot, *$ kann man dann wie für NEAs zeigen, aber mit mehr technischem Aufwand (Eliminierung ε-Kanten ...)

Und nun ...

1. *Motivation: semistrukturierte Daten*
2. Grundbegriffe
3. Charakterisierungen erkennbarer Baumsprachen
4. Top-down-Baumautomaten
5. Abschlusseigenschaften
6. Entscheidungsprobleme
7. *Anwendung 2: XML-Schemasprachen*

Das Leerheitsproblem

Zur Erinnerung:

Gegeben: NEBA $\mathcal{A}$

Frage: Gilt $L(\mathcal{A}) = \emptyset$?

Mengenschreibweise: $\{\text{NEBA } \mathcal{A} \mid L(\mathcal{A}) = \emptyset\}$

Satz 18
Das Leerheitsproblem für NEBAs ist entscheidbar.

Beweis: siehe Tafel. ●

Komplexität: P-vollständig (Wegsuche in Hypergraphen)

Das Zugehörigkeitsproblem

für NEAs: „Wortproblem“

Zur Erinnerung:

Gegeben: NEBA $\mathcal{A}$ und Baum $T = (P, t)$

Frage: Gilt $T \in L(\mathcal{A})$?

Mengenschreibweise: $\{(\mathcal{A}, T) \mid T \in L(\mathcal{A})\}$

Satz 19
Das Zugehörigkeitsproblem für NEBAs ist entscheidbar.

Beweis: Reduktion zum Leerheitsproblem – siehe Tafel. ●

Komplexität: P-vollständig
(folgt aus Reduktion per Produktautomat
alternativ: modifizierte Wegsuche in Hypergraphen)

Das Äquivalenzproblem

Zur Erinnerung:

Gegeben: NEBAs $\mathcal{A}_1, \mathcal{A}_2$

Frage: Gilt $L(\mathcal{A}_1) = L(\mathcal{A}_2)$?

Mengenschreibweise: $\{(\mathcal{A}_1, \mathcal{A}_2) \mid L(\mathcal{A}_1) = L(\mathcal{A}_2)\}$

Satz 20

Das Äquivalenzproblem ist entscheidbar.

Beweis: analog zum Beweis für NEAs. Benutze:

$$L(\mathcal{A}_1) = L(\mathcal{A}_2) \quad \Leftrightarrow \quad \Big(L(\mathcal{A}_1) \cap \overline{L(\mathcal{A}_2)}\Big) \cup \Big(\overline{L(\mathcal{A}_1)} \cap L(\mathcal{A}_2)\Big) = \emptyset$$

Komplexität: für NEBAs ExpTime-vollständig, für DEBAs in P

Das Universalitätsproblem

Zur Erinnerung:

Gegeben: NEBA $\mathcal{A}$

Frage: Gilt $L(\mathcal{A}) = \mathcal{T}(\Sigma)$? $\quad (\mathcal{T}(\Sigma) = \{T \mid T \text{ ist ein } \Sigma\text{-Baum}\})$

Mengenschreibweise: $\{\mathcal{A} \mid L(\mathcal{A}) = \mathcal{T}(\Sigma)\}$

Satz 21

Das Universalitätsproblem ist entscheidbar.

Beweis: analog zum Beweis für NEAs. Benutze:

$$L(\mathcal{A}) = \mathcal{T}(\Sigma) \quad \Leftrightarrow \quad \overline{L(\mathcal{A})} = \emptyset$$

Komplexität: für NEBAs ExpTime-vollständig, für DEBAs in P

Und nun . . .

1. *Motivation: semistrukturierte Daten*
2. Grundbegriffe
3. Charakterisierungen erkennbarer Baumsprachen
4. Top-down-Baumautomaten
5. Abschlusseigenschaften
6. Entscheidungsprobleme
7. *Anwendung 2: XML-Schemasprachen*

Zur Erinnerung: semistrukturierte Daten

Daten werden in Entitäten (Einheiten) zusammengefasst

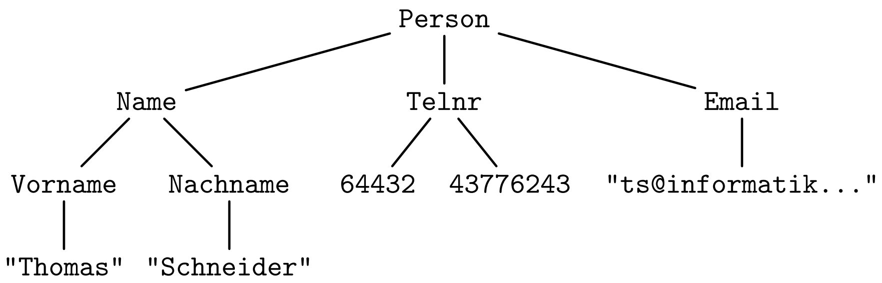
- Markierung von Entitäten durch Tags
- Bildung von Hierarchien
- Gruppieren ähnlicher Entitäten
- Entitäten derselben Gruppe können verschiedene (oder keine) Attribute haben
- Reihenfolge der Attribute *kann* eine Rolle spielen (Mengen oder Listen z. B. von Telefonnummern?)

Beispiel:

```
Person: {Name: {VN: "Thomas", NN: "Schneider"},
         Telnr: 64432,
         Telnr: 43776243,
         Email: "ts@informatik..."}
```

Repräsentation im Baum

```
Person: {Name: {VN: "Thomas", NN: "Schneider"},
         Telnr: 64432,
         Telnr: 43776243,
         Email: "ts@informatik..."}
```

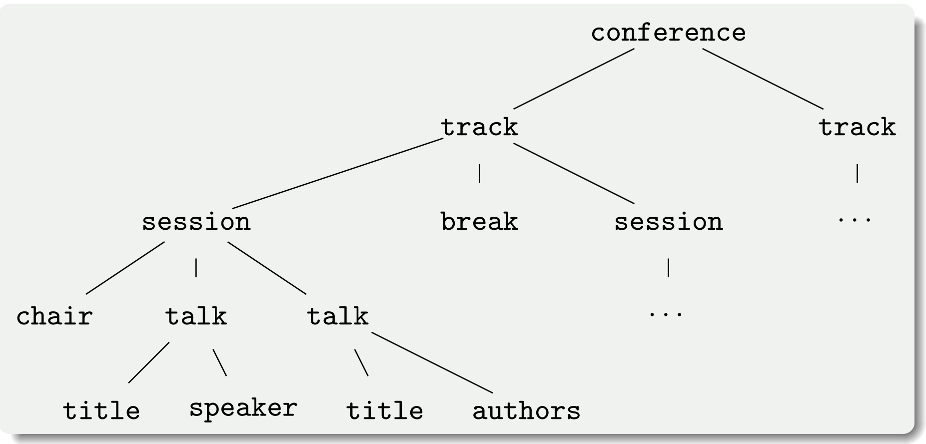


Größeres Bsp.: XML-Dokument für Konferenzprogramm

```
<conference>
  <track>
    <session>
       <chair> F. Angorn </chair>
       <talk>
         <title> The Pushdown Hierarchy </title>
         <speaker> D.J. Gaugal </speaker>
       </talk>
       <talk>
         <title> Trees Everywhere </title>
         <authors> B. Aum, T. Rees  </authors>
       </talk>
    </session>
    <break> Coffee </break>
    <session>
       ....
    </session>
  </track>
  <track>
    ....
  </track>
</conference>
```

aus *Tree Automata Techniques and Applications*, S. 230

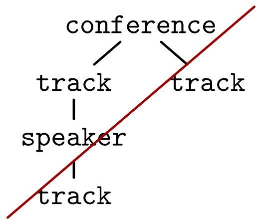
Zugehöriger Baum

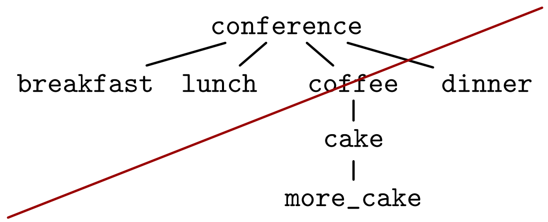


aus *Tree Automata Techniques and Applications*, S. 230

- **Ab jetzt:** wir beschreiben nur die Struktur, ignorieren die Daten

Was ist ein gültiges Konferenzdokument?





Mögliche Anforderungen an gültige Konferenzdokumente

- Eine Konferenz kann in mehrere Blöcke (Tracks) geteilt sein.
- Jeder Block (oder die Konf. selbst, wenn sie keine Blöcke hat) ist in Sitzungen aufgeteilt.
- Jede Sitzung hat einen oder mehrere Vorträge.
- Jede Sitzung hat einen Vorsitzenden (Chair).
- Jeder Vortrag hat einen Titel und
  - Autoren (falls es sich um einen Konferenzbeitrag handelt)
  - oder Vortragenden (falls es ein eingeladener Vortrag ist).
- Zwischen den Sitzungen kann es Pausen geben.

Gültige Dokumente als Baumsprachen!

Die gelisteten Anforderungen beschreiben eine **Baumsprache** über dem Alphabet $\{\texttt{conference}, \texttt{track}, \dots\}$.

Eine solche Beschreibung wird auch **Schema** genannt.

Ein Dokument ist **gültig** für ein Schema, wenn sein Baum zur Baumsprache des Schemas gehört.

Ziele dieses Abschnitts

- Vorstellen von XML-Schemasprachen
- Diskutieren von Verbindungen zur Automatentheorie
- Untersuchen der Ausdrucksstärke von Schemasprachen
- und ihre **Entscheidungsprobleme**

Entscheidungsprobleme für XML-Schemasprachen

Die bekannten Entscheidungsprobleme entsprechen natürlichen Fragen für XML-Dokumente und -Schemasprachen:

Zugehörigkeitsproblem

Ist ein gegebenes Dokument gültig für ein gegebenes Schema?
(im Bsp.: erfüllt ein gegebenes Konf.-dokument die Anforderungen?)

Leerheitsproblem

Gibt es für ein gegebenes Schema gültige Dokumente?
(Enthält das gegebene Schema keinen „Widerspruch“?)

Äquivalenzproblem

Haben zwei Schemata dieselben gültigen Dokumente?
(Wichtig bei der Vereinfachung von Schemata: Ist das Resultat der Vereinfachung äquivalent zum ursprünglichen Schema?)

Dokumenttypdefinitionen (DTDs)

DTDs sind ein Standard zur Beschreibung gültiger Dokumente

Eine DTD ist eine kontextfreie Grammatik (kfG), deren rechte Regelseiten reguläre Ausdrücke enthalten können

Ableitungsbäume der kfG bilden die Baumsprache, die durch die DTD bestimmt wird

Beispiel-DTD für Konferenzdokumente

```
<!DOCTYPE CONFERENCE [
  <!ELEMENT conference (track+|(session,break?)+)>
  <!ELEMENT track      ((session,break?)+)>
  <!ELEMENT session    (chair,talk+)>
  <!ELEMENT talk       ((title,authors)|(title,speaker))>
  <!ELEMENT chair      (#PCDATA)>
  <!ELEMENT break      (#PCDATA)>
  <!ELEMENT title      (#PCDATA)>
  <!ELEMENT authors    (#PCDATA)>
  <!ELEMENT speaker    (#PCDATA)>
]>
```
, ≙ Verkettung

Beschreibt Bäume, in denen z. B. jeder `conference`-Knoten

- ein oder mehrere `track`-Kinder hat oder
- ein oder mehrere `session`-Kinder hat, zwischen denen einzelne `break`-Geschwister stehen dürfen

$\rightsquigarrow$ **beliebige Stelligkeit!**

Wir brauchen Symbole mit beliebiger Stelligkeit!

Erweitern unser r-Alphabet:

- U: Menge von **Symbolen ohne Stelligkeit**
- $\Sigma = U \cup \bigcup_{i \geqslant 0} \Sigma_i$

Endlicher geordneter Baum $T = (P, t)$ über Σ:

- $P \subseteq \mathbb{N}_+^*$ nichtleere endl. präfix-abgeschlossene Menge
- $t : P \to \Sigma$ Funktion mit
  1. Wenn $t(p) \in \Sigma_m$, dann $\{j \mid pj \in P\} = \{1, \dots, m\}$.
  2. Wenn $t(p) \in U$, dann $\{j \mid pj \in P\} = \{1, \dots, k\}$ für ein $k \geqslant 0$.

Beschränken uns auf den Fall ohne Stelligkeit (o. S.): $\Sigma = U$

Weitere Begriffe

- **Höhe, Tiefe, Teilbaum**: wie für Bäume mit Stelligkeit
- $a(T_1 \cdots T_n)$: Baum mit a in Wurzel und Teilbäumen $T_1, \dots, T_n$ direkt darunter
- **Hecke (Hedge)**: Folge $T_1 \cdots T_n$ von Bäumen o. S.

$\rightsquigarrow$ **induktive Charakterisierung von Bäumen o. S.:**

- Jede Folge von Bäumen o. S. ist eine Hecke. (Das schließt die leere Folge ein.)
- Wenn h eine Hecke und $a \in \Sigma$ ein Symbol ist, dann ist $a(h)$ ein Baum o. S. ◀ Bsp.: s. Tafel

- $h = \varepsilon$ $\rightsquigarrow$ schreiben a statt $a(\varepsilon)$
- $H(\Sigma)$: Menge aller Hecken über Σ

Heckenautomaten

- . . . sind Bottom-up-Automaten auf Bäumen o. S.
- Können sie analog zu NEBAs definiert werden? **Nein:**
  - Weil Stelligkeit von $a \in \Sigma$ nicht festgelegt ist, brauchten wir 1 Regel $a(q_1, \dots, q_m) \to q$ **pro** $m \geqslant 0$
  - **Abhilfe:** Nutzen reguläre Ausdr. über Q in linken Regelseiten

Definition 22

Ein **nichtdeterministischer endlicher Heckenautomat (NEHA)** ist ein Quadrupel $\mathcal{A} = (Q, \Sigma, \Delta, F)$, wobei

- Q, Σ, F wie für NEBAs definiert sind und
- Δ eine Menge von Überführungsregeln der Form

$$a(R) \to q$$

ist, wobei $a \in \Sigma$ und $R \subseteq Q^*$ eine reg. Sprache über Q ist.

Berechnungen, Akzeptanz und erkannte Sprache

Definition 23

Sei $\mathcal{A} = (Q, \Sigma, \Delta, F)$ ein NEHA und $T = (P, t)$ ein Σ-Baum o. S.

- **Berechnung (Run)** von $\mathcal{A}$ auf T ist eine Fkt. $r : P \to Q$ mit:

  Wenn $t(p) = a$, $r(p) = q$ und $m =$ Anzahl von p's Kindern,
  dann gibt es $a(R) \to q$ in Δ mit $r(p1) \cdots r(pm) \in R$.

Anmerkungen

- Wenn p Blattposition mit Markierung a (d. h. $t(p) = a$), dann darf $a(R) \to q$ nur angewendet werden, wenn $\varepsilon \in R$.
- Repräsentation der reg. Sprache $R \subseteq Q^*$: NEAs, DEAs oder reg. Ausdrücke
  - das ist egal für die Mächtigkeit von NEHAs,
  - aber nicht für Entscheidungsverfahren und deren Komplexität!

Berechnungen, Akzeptanz und erkannte Sprache

Definition 23 (Fortsetzung)

Sei $\mathcal{A} = (Q, \Sigma, \Delta, F)$ ein NEHA und $T = (P, t)$ ein Σ-Baum o. S.

- **Berechnung (Run)** von $\mathcal{A}$ auf T ist eine Fkt. $r : P \to Q$ mit:

  Wenn $t(p) = a$, $r(p) = q$ und $m =$ Anzahl von p's Kindern,
  dann gibt es $a(R) \to q$ in Δ mit $r(p1) \cdots r(pm) \in R$.

- Ein Run r von $\mathcal{A}$ auf T ist **akzeptierend**, wenn $r(\varepsilon) \in F$.
- $\mathcal{A}$ **akzeptiert** T, wenn es einen akz. Run von $\mathcal{A}$ auf T **gibt**.
- Die von $\mathcal{A}$ **erkannte Sprache** ist
  $L(\mathcal{A}) = \{T \text{ über } \Sigma \mid \mathcal{A} \text{ akzeptiert } T\}$.

Beispiel: siehe Tafel

Umwandlung der Beispiel-DTD in einen NEHA (1)

```
<!DOCTYPE CONFERENCE [
  <!ELEMENT conference (track+|(session,break?)+)>
  <!ELEMENT track      ((session,break?)+)>
  <!ELEMENT session    (chair,talk+)>
  <!ELEMENT talk       ((title,authors)|(title,speaker))>
  <!ELEMENT chair      (#PCDATA)>
  ...
  <!ELEMENT speaker    (#PCDATA)>
]>
```

Zugehörige erweiterte kontextfreie Grammatik:

$$
\begin{array}{lcl}
\texttt{conference} & \to & \texttt{track}^+ + (\texttt{session}\ (\texttt{break} + \varepsilon))^+ \\
\texttt{track} & \to & (\texttt{session}\ (\texttt{break} + \varepsilon))^+ \\
\texttt{session} & \to & \texttt{chair}\ \texttt{talk}^+ \\
\texttt{talk} & \to & (\texttt{title}\ \texttt{authors}) + (\texttt{title}\ \texttt{speaker}) \\
\texttt{chair} & \to & \texttt{DATA} \\
\ldots & & \\
\texttt{speaker} & \to & \texttt{DATA}
\end{array}
$$

Umwandlung der Beispiel-DTD in einen NEHA (2)

Erweiterte kontextfreie Grammatik (Wdhlg.):

$$
\begin{array}{lcl}
\texttt{conference} & \to & \texttt{track}^+ + (\texttt{session}\ (\texttt{break} + \varepsilon))^+ \\
\texttt{track} & \to & (\texttt{session}\ (\texttt{break} + \varepsilon))^+ \\
\texttt{session} & \to & \texttt{chair}\ \texttt{talk}^+ \\
\texttt{talk} & \to & (\texttt{title}\ \texttt{authors}) + (\texttt{title}\ \texttt{speaker}) \\
\texttt{chair} & \to & \texttt{DATA} \\
\ldots & & \\
\texttt{title} & \to & \texttt{DATA}
\end{array}
$$

Startsymbol: hier `conference`

Ableitungsschritt:

- Wähle mit ℓ beschriftetes Blatt, $\ell \in \Sigma$
- Wähle Regel $\ell \to R$ (R heißt **Inhaltsmodell**)
- Wähle $a_1 \cdots a_n \in R$ und füge Kinder $a_1, \ldots, a_n$ zu ℓ hinzu

Beispielableitung: siehe Tafel

Umwandlung der Beispiel-DTD in einen NEHA (3)

Erweiterte kontextfreie Grammatik (Wdhlg.):

$$\begin{array}{lll} \texttt{conference} & \to & \texttt{track}^+ + (\texttt{session}\ (\texttt{break} + \varepsilon))^+ \\ \texttt{track} & \to & (\texttt{session}\ (\texttt{break} + \varepsilon))^+ \\ \vdots & & \\ \texttt{title} & \to & \texttt{DATA} \end{array}$$

Zugehöriger NEHA: $\mathcal{A} = (Q, \Sigma, \Delta, F)$ mit

$$\begin{array}{lll} \Sigma & = & \{\texttt{conference}, \texttt{track}, \texttt{session}, \texttt{talk}, \texttt{chair}, \ldots, \texttt{DATA}\} \\ Q & = & \Sigma \\ F & = & \{\texttt{conference}\} \\ \Delta & = & \{\ \texttt{conf}\big(\mathit{track}^+ + (\mathit{session}\ (\mathit{break} + \varepsilon))^+\big) \to \mathit{conf}, \\ & & \quad \texttt{track}\big((\mathit{session}\ (\mathit{break} + \varepsilon))^+\big) \to \mathit{track}, \\ & & \quad \vdots \\ & & \quad \texttt{title}\big(\mathit{DATA}\big) \to \mathit{title}, \\ & & \quad \texttt{DATA}\big(\big) \to \mathit{DATA}\ \} \end{array}$$

Formalisierung von DTDs und den zugehörigen NEHAs

Definition 24

Eine **Dokumenttypdefinition (DTD)** ist ein Tupel $D = (\Sigma, s, \Delta)$ mit

- einem Alphabet o. S. Σ,
- einem **Startsymbol** $s \in \Sigma$ und
- einer Abbildung $\Delta : \Sigma \longrightarrow$ reguläre Ausdrücke über Σ:

  (Δ entspricht einer Menge von Regeln – die Folge der Symbole in den Kindern jedes Knotens mit $a \in \Sigma$ muss in $L(\Delta(a))$ sein.)

Zugehöriger NEHA: $\mathcal{A}_D = (Q_D, \Sigma, \Delta_D, F_D)$ mit

- $Q_D = \Sigma$
- $F_D = \{s\}$
- $\Delta_D = \{a(\Delta(a)) \to a \mid a \in \Sigma\}$

Lokale Sprachen

Definition 25

- Die von einer DTD D **erzeugte Sprache** ist $L(\mathcal{A}_D)$.
- Eine Baumsprache über Σ heißt **lokal**, wenn sie von einer DTD über Σ erzeugt wird.

Frage:
Wie verhalten sich lokale Sprachen zu NEHA-erkennbaren Spr.?
(Gibt es NEHAs, die keine DTD beschreiben?)

Anwort:

Nicht jede NEHA-erkennbare Sprache ist lokal. (Ja.)

Weil DTDs „immer nur eine Ebene nach unten schauen“ •

(Nicht ausdrückbar:
„alle Sitzungen jeder Konf. haben zusammen $\geqslant 5$ Vortragende“)

Deterministische Inhaltsmodelle

Die W3C[a]-Empfehlung für XML fordert,
dass Inhaltsmodelle **deterministische reguläre Ausdrücke** sind.

[a] *World Wide Web Consortium*, int. Agentur für WWW-Standards

Regulärer Ausdruck r über Σ ist **deterministisch**, falls

- für jedes Wort $w \in \Sigma^*$ und jeden Buchstaben a in w höchstens ein Vorkommen von a in r existiert, auf das a passt.
- Dann lässt sich in Poly-zeit ein äquivalenter DEA konstruieren.

$\leadsto$ Stellt sicher, dass das Zugehörigkeitsproblem für DTDs in Poly-zeit lösbar ist.

Deterministische Inhaltsmodelle – Beispiel

Betrachte die Zeile

```
<!ELEMENT talk        ((title,authors)|(title,speaker))>
```

und die zugehörige Regel

$$\texttt{talk} \;\rightarrow\; (\texttt{title authors}) + (\texttt{title speaker})$$

Für Wörter über Σ, die mit dem Buchstaben `title` beginnen, ist nicht klar, welchem Vorkommen von `title` im Inhaltsmodell dieser Buchstabe entspricht!

Jetzt genau: deterministische reguläre Ausdrücke

Idee: Sei r ein RA über Σ.

- Markiere das i-te Vorkommen jedes Buchstaben a in r mit a_i.
- Bsp.: $(a+b)^*b(ab)^* \quad \rightsquigarrow \quad (a_1+b_1)^*b_2(a_2b_3)^* =: r'$.
- r ist deterministisch, wenn $L(r')$ keine zwei Wörter ua_iv und ua_jw mit $i \neq j$ enhält.

Etwas Notation:

- RA r über $\Sigma \quad \rightsquigarrow \quad$ markierter RA r' über Σ'
- wie üblich: $L(r) \subseteq \Sigma^*$ und $L(r') \subseteq \Sigma'^*$

Definition 26

Ein **deterministischer RA (DRA)** ist ein RA r über Σ, so dass für alle Wörter $u, v, w \in \Sigma'^*$ und Zeichen $a \in \Sigma$ mit $ua_iv,\ ua_jw \in L(r')$ gilt: $i = j$. Bsp.: siehe Tafel.

Was nützen uns nun DRAs?

Satz 27

Zu jedem DRA r kann man in Polynomialzeit einen DEA $\mathcal{A}$ mit $L(\mathcal{A}) = L(r)$ konstruieren.

(Ohne Beweis.)

Folgerung 28

Zu jeder deterministischen DTD kann man in Polynomialzeit einen äquivalenten NEHA(DEA) konstruieren.

NEHA(DEA): $\mathcal{A} = (Q, \Sigma, \Delta, F)$, bei dem für alle $a(R) \rightarrow q \in \Delta$ R als DEA gegeben ist.

Und dieses Resultat garantiert nun ...?

Deterministische DTDs sind effizient!

Satz 29

Für deterministische DTDs sind in Polynomialzeit lösbar:

- *das Zugehörigkeitsproblem*
- *das Leerheitsproblem*
- *das Äquivalenzproblem*

(Ohne Beweis.)

Zur Erinnerung:

- **Zugehörigkeitsproblem** (Gültigkeit)
  Ist ein gegebenes Dokument gültig für ein gegebenes Schema?
- **Leerheitsproblem** (Widerspruchsfreiheit)
  Gibt es für ein gegebenes Schema gültige Dokumente?
- **Äquivalenzproblem**
  Haben zwei Schemata dieselben gültigen Dokumente?

Sind deterministische DTDs schwächer als allgemeine?

1. Im Allgemeinen **ja**,
2. **aber** es ist entscheidbar, ob eine gegebene DTD äquivalent zu einer deterministischen DTD ist:

Satz 30

1. *Nicht jede reg. Sprache wird durch einen DRA beschrieben:*

$$\{L(r) \mid r \text{ ist DRA}\} \subset \{L(r) \mid r \text{ ist RA}\}$$

2. *Das folgende Problem ist in Polynomialzeit entscheidbar.*

   Gegeben: DEA $\mathcal{A}$
   Frage: Gibt es einen DRA r mit $L(r) = L(\mathcal{A})$?

   Wenn ein solcher DRA existiert,
   dann kann er in Exponentialzeit konstruiert werden.

Zusammenfassung für deterministische DTDs

Deterministische DTDs . . .

- sind **echt schwächer als NEHAs**, weil sie
  - nur **lokale Sprachen** beschreiben
    (sie können keine Bedingungen über Knoten ausdrücken, die durch einen Pfad der Länge > 1 getrennt sind);
  - nur **DRAs** auf rechten Regelseiten erlauben.
- Dafür sind **die wichtigen Entscheidungsprobleme effizient lösbar**.

Ausblick: Lockern der Einschränkungen

Extended DTDs (EDTDs)

- führen durch eine einfache syntaktische Erweiterung aus den lokalen Sprachen heraus
- sind fast äquivalent zu NEHAs
  (beschränkt auf Sprachen, in denen alle Bäume dasselbe Wurzelsymbol haben)
- haben ein in Polynomialzeit lösbares Zugehörigkeits- und Leerheitsproblem

Weitere Einschränkung von EDTDs

- garantiert auch ein in Polynomialzeit lösbares Äquivalenzproblem
- liegt **XML Schema** zugrunde

Damit sind wir am Ende dieses Kapitels.

Vielen Dank.

Literatur für diesen Teil (1)

Hubert Comon, Max Dauchet, Rémi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, Marc Tommasi.
Tree Automata Techniques and Applications.
http://tata.gforge.inria.fr Nov. 2008.
Kapitel 1
Abschnitt 2.4 (Verbindung zu kontextfreien Wortsprachen)
Abschnitte 8.2.1, 8.2.2, 8.7 (Heckenaut. und XML-Schemasprachen)

Meghyn Bienvenu.
Automata on Infinite Words and Trees.
Vorlesungsskript, Uni Bremen, WS 2009/10.
http://www.informatik.uni-bremen.de/tdki/lehre/ws09/automata/automata-notes.pdf
Kapitel 3

Literatur für diesen Teil (2)

Anne Brüggemann-Klein, Derick Wood.
One-Unambiguous Regular Languages.
Information and Computation, 142:1998, S. 182–206.
http://dx.doi.org/10.1006/inco.1997.2695
Grundlegende Resultate für deterministische reguläre Ausdrücke.