A Unifying Approach to HTML Wrapper Representation and Learning

Gunter Grieser, Klaus P.Jantke,
Steffen Lange, Bernd Thomas

Seminarvortrag - Informationsextraktion (1/2003)

1.Motivation und Zielsetzung

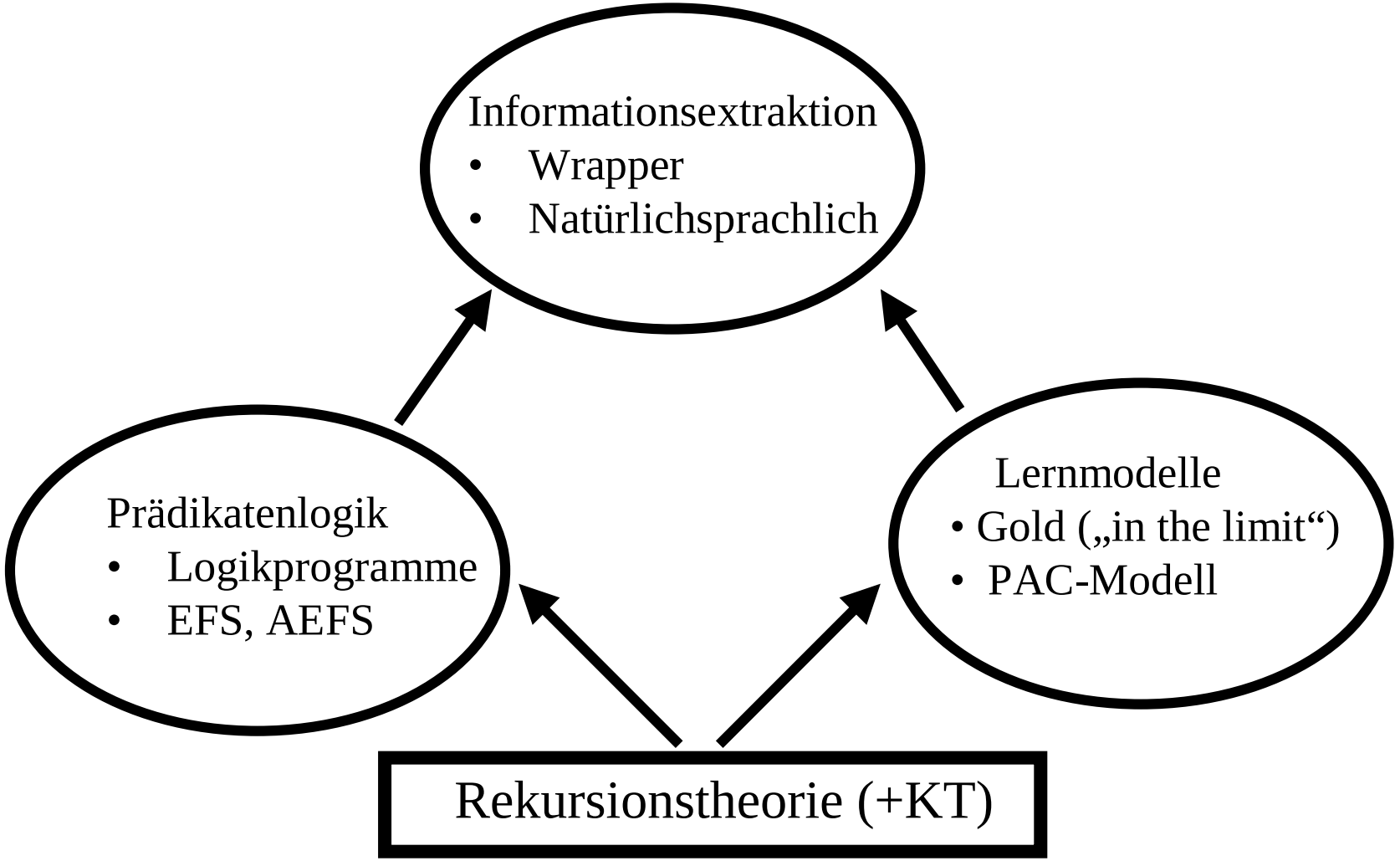
- HTML ist Standardformat für Information aus dem Internet
- Informationsextraktion aus HTML kommerziell lohnend
- Ansatz auf beliebig formatierte Texte übertragbar
- Konstruktiver (algorithmischer) Beitrag zu bestehenden Theorien
- Existenz von Testumgebungen (http://Lexikon.dfki.de)

1.1 Information - eingebettet in HTML

```
<ul>
<li>Vortrag über: Line Eikvil (1999) Information extraction from the World Wide Web (Stephan Birkmann) (<a href="birkmann.ppt">ppt</a>/
<a href="birkmann.pdf">pdf</a>)
</li>
<li>Vortrag über: Ralph Grishman (1997) &quot;Information Extraction: Techniques and Challenges&quot; (Felix Jungermann)
 (<a href="jungermann.ppt">ppt</a>/
<a href="jungermann.pdf">pdf</a>)
</li>
<li>
Vortrag über: Roman Yangarber, Ralph Grishman, Pasi Tapanainen, Silja Huttunen (2000)
 &quot;Unsupervised Discovery of Scenario-Level Patterns for Information Extraction&quot; (Bianca Selzam) (<a href="selzam.ppt">ppt</a>/
<a href="selzam.pdf">pdf</a>)
</li>
<li>
Vortrag über: Karsten Winkler, Myra Spiliopoulou &quot; Structuring Domain-Specific
 Text Archives by Deriving a Probabilistic XML DTD&quot;<br>(Andrea Schweer)
 (<a href="AndreaSchweer-SeminarFolien-20021217.pdf">pdf</a>)
</li>
</ul>
```

Automatische Extraktion der Namen der Referenten !?

2. Einordnung dieses Ansatzes



3. Syntax/Semantik formaler Systeme

Grundlegende Definitionen

Σ : Endliche Menge von Zeichen

Π : Endliche Menge von Prädikaten

X : Menge von Variablen

Γ : Endliche Menge von Regeln

Pattern : Elemente aus $(\Sigma \cup X)^+$

Atom : $p(\pi_1, ..., \pi_n)$ (auch : "ground Atom")

Regel : $A \longleftarrow B_1, ..., B_n$ (auch : "ground Instance")

Substitution : $\sigma : X \longrightarrow \Sigma^+$

3.1 Elementary Formal Systems (EFS)

Definition:

$S = (\Sigma, \Pi, \Gamma)$ ist EFS

B(S) : Menge aller ground Atoms (Herbrand Basis)

G(S) : Menge aller ground Instances

Semantik:

$T_S^{\,0}(I) := I \quad (I \subseteq B(S))$

$T_S^{\,n+1}(I) := T_S^{\,n}(I) \cup \{A \mid A \longleftarrow\!\!\!\!- B_1, ..., B_k \in \mathrm{G(S)}, \forall i : B_i \in I\}$

$Somit : T_S^{\,n+1}(I) := T_S(T_S^{\,n}(I))$ und $Sem(S) := \bigcup_{n \in N} T_S^{\,n}(\varnothing)$

Smullyan (1961), Theory of Formal Systems

3.2 Advanced Elementary Formal Systems

Definition:

$S' := (\Sigma, \Pi', \Gamma')$ sei EFS

$S_1 := (\Sigma, \Pi_1, \Gamma_1), S_2 := (\Sigma, \Pi_2, \Gamma_2)$ seien AEFS

Dann:

1. $S = S'$ ist AEFS. $Sem(S) = Sem(S')$
2. Falls $\Pi_1 \cap \Pi_2 = \emptyset$, dann $S = (\Sigma, \Pi_1 \cup \Pi_2, \Gamma_1 \cup \Gamma_2)$ ist AEFS, $Sem(S) = Sem(S_1) \cup Sem(S_2)$
3. Seien $p \notin \Pi_1$ und $q \in \Pi_1$ Prädikate der Stelligkeit n.

   Dann $S = (\Sigma, \Pi_1 \cup \{p\}, \Gamma_1 \cup \{p(x_1,...,x_n) \longleftarrow not\ q(x_1,...,x_n)\})$ ist AEFS

   $Sem(S) = Sem(S_1) \cup \{p(s_1,...,s_n) \mid s_1 \in \Sigma^+,...,s_n \in \Sigma^+, q(s_1,...,s_n) \notin Sem(S_1)\}$
4. Sei $head(\Gamma') \cap \Pi_1 = \emptyset$, dann $S = (\Sigma, \Pi' \cup \Pi_1, \Gamma' \cup \Gamma_1)$ ist AEFS

   $Sem(S) = \bigcup_{n \in N} T_{S'}^n(Sem(S_1))$

Vorteil: Negation as Failure

3.3 Formale Systeme-Formale Spracher

Zusätzliche Definitionen für formale Systeme:

ε : Menge aller EFS

Aε : Menge aller AEFS

$vb - A\varepsilon$: Menge aller "variable - bounded" AEFS

$lb - A\varepsilon$: Menge aller "length - bounded" AEFS

Übergang zu formalen Sprachen (->Wrapper):

1. Sei $S = (\Sigma, \Pi, \Gamma)$ ein AEFS und $p \in \Pi$ einstellig es Prädikat.
   Dann sei $L(S, p) := \{s \mid p(s) \in Sem(S)\}$

2. Sei $M \subseteq A\varepsilon$. Dann ist $L(M)$ die Menge von Sprachen,
   die durch AEFS aus M definiert werden können.

-> Formale Sprachen sind durch formale Systeme definierbar

3.4 Ausdrucksfähigkeit von AEFS

Einordnung von formalen Systemen:

1. $L_{rekursiv\ aufzählbar} \subset L(A\varepsilon)$
2. $L_{kontextsen\ sitiv} = L(lb - A\varepsilon)$
3. $L_{rekursiv\ aufzählbar} = L(vb - \varepsilon)$
4. $L_{kontextsen\ sitiv} = L(lb - \varepsilon)$

Ausserdem:

1. Semantik von allgemeinen EFS nicht entscheidbar
2. Semantik von „length-bounded“ AEFS entscheidbar

4. Lernen und Identifizieren

Abstraktes Modell einer Identifikation (Gold, 1967):

Sei Ω eine Menge von Objekten

1. Definition von Lernbarkeit
2. Methoden zur Informationspräsentation
3. Namensgebung von Objekten

Zusammenhang zwischen Identifikation von Objekten und Lernen von Sprachen (z.b Wrapper)

4.1 Lernbarkeit

Zu jedem Zeitpunkt erfolgt ein Identifikationsversuch:

$$g_t = G(i_1,....,i_t)$$

G : „Guessing function“

g_t : „Guess“ zum Zeitpunkt t

„Language Identification in the Limit“:
Nach endlicher Zeit treten keine falschen „guesses“ mehr auf

-> Existenz zahlreicher Variationen des Lernmodells
Beispiel: zusätzliche Monotonierestriktionen

4.2 Informationspräsentation

1. Text :

   Ausschließlich Positive Information

   Arbitrary: Sequenz wird bestimmt durch beliebige Funktion

   Recursive: Sequenz wird bestimmt durch rekursive Funktion

2. Informant:

   Positive und Negative Information

   Arbitrary: Sequenz wird bestimmt durch beliebige Funktion

   Methodical: Sequenz wird durch Aufzählung bestimmt

   Request: Sequenz wird bestimmt in Abhängigkeit der schon bestehenden Sequenz

4.3 Namensgebung

Namensgebung mittels einer Funktion:

Ω : Menge von Objekten (Sprachen)

N : Menge von Namen

$f : \Omega \longrightarrow N$

Für theoretische Untersuchungen üblich:

1. Tester : Entscheidungsalgorithmus für eine Sprache
2. Generatoren: Aufzählung aller Wörter einer Sprache

4.4 Erkenntnisse zur Lernbarkeit

GOLD

TABLE I

DIVIDING LINES BETWEEN LEARNABILITY AND NONLEARNABILITY OF LANGUAGES

Learnability model	Class of languages
Anomalous text[a] →	Recursively enumerable recursive
Informant →	Primitive recursive Context-sensitive Context-free Regular Superfinite
Text →	Finite cardinality languages

[a] Anomalous text refers to the use of the generator-naming relation and information presentation by means of primitive recursive text.

4.5 Lernbarkeit von AEFS

Theorem 1:

1. $L(vb - A\varepsilon)$ ist nicht lernbar durch "informant"
2. $L(lb - A\varepsilon)$ ist lernbar durch "informant"

Theorem 2:

1. Für $k \geq 2, L(lb - A\varepsilon^k)$ ist nicht lernbar durch "text"
2. $L(lb - A\varepsilon^1)$ ist lernbar durch "text"

4.6 Beweisidee zu Theorem 2 (1)

Definition:

$\Sigma := \{a\}$

$L_0 := \{a\}^+, L_j = L_0 \setminus \{a^j\}$ $\qquad$ $S_0 := (\Sigma, \{p\}, \{p(x) \longleftarrow\})$

$\Omega := \{L_j \mid j \in N\}$ mit $\Omega \subseteq L(lb - A\varepsilon^2)$ $\qquad$ $S_j := (\Sigma, \{p,q\}, \{q(a^j) \longleftarrow, p(x) \longleftarrow notq(x)\})$

Idee: Konstruktion eines Textes für L_0, sodaß Identifizierung unmöglich wird.

Start mit $x := 1$, Sequenz (Text) wird gebildet durch:

1. Lexikographische Reihenfolge startend mit a^{x+1}
2. Nach Erkennung von L_x, Präsentation von a^x
3. Fortsetzung der Lexikographischen Reihenfolge
3. Nach Erkennung von L_0 (zum Zeitpunkt t),
   setze $x := t + 1$ und weiter bei 1

-> „Subset-Principle“

5. Wrapper für HTML-Dokumente

Semantik und Interpretation von Dokumenten:

Sei $D \in \Sigma^+$ ein Dokument.

$S : \Sigma^+ \longrightarrow \wp((\Sigma^+)^n)$ heißt Semantik unter Interpretation I, gdw :

Für alle Dokumente D, erfüllt $(s_1,...,s_n) \in S(D)$ die Bedingungen :

1. $(p_1,..., p_n) = I((s_1,..., s_n), D)$ gibt die Startpositionen der (s_i) an
2. s_{i+1} beginnt in D nach Ende von s_i

Semantik eines Dokumentes ist Tupel aus Wörtern des Dokumentes

5.1 „Marked Text“ (1.Erweiterung)

Definition:

Eine Sequenz $t = (D_i, P_i)$ heißt "marked text", gdw :

1. D_i ist Dokument aus Σ^+
2. $P_i = (s, p)$, wobei $s \in S(D)$ und $p = I(s, D)$ (Interpretation)
3. Sequenz enthält für jedes Dokument D jegliche Tupel $s \in S(D)$

Lernen von Wrappern durch Vorlage einer Sequenz von markierten Dokumenten

5.2 Island Wrapper

Island Wrapper <-> Kombination von mehreren AEFS

$$w(V_1,...,V_n, X_1L_1V_1R_1...X_nL_nV_nR_nX_{n+1}) \longleftarrow$$
$$p_{\ell_1}(L_1), nc - p_{r_1}(V_1), p_{r_1}(R_1),$$
$$nc - p_{\ell_2}(X_2), p_{\ell_2}(L_2), nc - p_{r_2}(V_2), p_{r_2}(R_2),....,$$
$$nc - p_{\ell_n}(X_n), p_{\ell_n}(L_n), nc - p_{r_n}(V_n), p_{r_n}(R_n).$$

$$nc - p_{\ell_i}(X) \longleftarrow not\ c - p_{\ell_i}(X)$$
$$c - p_{\ell_i}(X) \longleftarrow p_{\ell_i}(X).\quad c - p_{\ell_i}(XY) \longleftarrow p_{\ell_i}(X).\quad c - p_{\ell_i}(XY) \longleftarrow p_{\ell_i}(Y).$$

$$nc - p_{r_i}(X) \longleftarrow not\ c - p_{r_i}(X)$$
$$c - p_{r_i}(X) \longleftarrow p_{r_i}(X).\quad c - p_{r_i}(XY) \longleftarrow p_{r_i}(X).\quad c - p_{r_i}(XY) \longleftarrow p_{r_i}(Y).$$

Gesucht sind Ankersprachen: $L_{\ell_i} = L(S_{\ell_i}, p_{\ell_i}),\ L_{r_i} = L(S_{r_i}, p_{r_i})$

5.3 Wrapper Learner (2.Erweiterung)

Anwendung des Wrappers auf ein Dokument:

Sei S_{iw} ein Island Wrapper (AEFS)

$$view(S_{iw}, D) := \{(s_1,...,s_n) \mid w(s_1,...,s_n, D) \in Sem(S_{iw})\}$$

Ein Wrapper (AEFS S_{iw}) beschreibt Semantik S, gdw:
view(S_{iw},D) = S(D)

1. Ein Wrapper-Learner (WIM) lernt Semantik S aus „marked Text“, falls nach endlicher Zeit der Wrapper diese Semantik beschreibt

2. Ein Wrapper-Learner (WIM) lernt eine Klasse C von Wrappern, falls er jegliche Semantiken lernen kann, die durch Wrapper aus C beschrieben werden können

5.4 Lernen mit Island Wrappern

Definition:

$A\varepsilon_{iw}$: Menge aller "length - bounded" Island Wrapper (AEFS)

$A\varepsilon_{\mathrm{iw}}^{\mathrm{k}}$: Menge aller "length - bounded" Island Wrapper mit $Card(L) \leq k$ für Ankersprachen L

Theorem 3:

Die Klasse $A\varepsilon_{iw}$ ist nicht lernbar durch "marked text"

Theorem 4:

Die Klasse $A\varepsilon_{iw}^{k}$ ist lernbar durch "marked text" für alle $k \geq 1$

Lernalgorithmus ?

6.1 Lernalgorithmus (Teil I)

WIM M: On input $(D_1, P_1), \ldots, (D_m, P_m)$ do the following.

Set $t^0, \ldots, t^n$ to be the empty sequences.

For each $j = 1, \ldots, m$, do the following:

For each pair $((s_1, \ldots, s_n), (p_1, \ldots, p_n))$ in P_j, compute (identified by the given starting positions $p_1, \ldots, p_n$ of $s_1, \ldots, s_n$) the uniquely determined substrings $w_0, w_1, \ldots, w_n$ of D_j such that $D_j = w_0 s_1 w_1 s_2 \ldots w_{n-1} s_n w_n$. Append w_0 to t^0, w_1 to t^1, ..., and w_n to t^n.

Let Γ be the rules depicted in Figure 3 and Π be the set of all predicates used to formulate the rules in Γ. Do in parallel:

- On input t^0, run M_A – an IIM for problems of type A. Fix $\Gamma' = M_A(t^0)$. Γ_0 is built by replacing, everywhere in Γ', the predicate p by p_{l_1}.
- For $i = 1, \ldots, n-1$, compute in parallel:
  On input t^i, run M_B – an IIM for problems of type B. Fix $\Gamma' = M_B(t^i)$. Γ_i is built by replacing, everywhere in Γ', the predicates p and q by p_{r_i} and $p_{l_{i+1}}$, respectively.
- On input t^n, run M_C – an IIM for problems of type C. Fix $\Gamma' = M_C(t^n)$. Γ_i is built by replacing, everywhere in Γ', the predicate p by p_{r_n}.

Output the AEFS (Σ, Π, Γ') with $\Gamma' = \Gamma \cup \Gamma_0 \cup \Gamma_1 \cup \ldots \cup \Gamma_n$.

6.2 Lernalgorithmus (Teil II)

IIM M_A: On input $S = u_0, \ldots, u_k$ do the following:

Set $\Gamma' = \emptyset$. Determine the set E of all non-empty suffixes of strings in S. For all strings $e \in E$ check whether or not, for all $a \in \Sigma$, $u = a \circ e$ for some $u \in S$. Let T be the set of all strings e passing this test. Goto $(\alpha 1)$.

$(\alpha 1)$ If $T = \emptyset$, output Γ'. Otherwise, goto $(\alpha 2)$.

$(\alpha 2)$ Determine a shortest string e in T. Set $\Gamma' = \Gamma' \cup \{p(e) \leftarrow\}$ and $T = T \setminus T_e$, where T_e contains all strings in T with the suffix e. Goto $(\alpha 1)$.

The IIM M_C can be obtained from M_A by replacing everywhere the term suffix by prefix. It is not hard to see that M_A and M_C behave as required.

6.3 Lernalgorithmus (Teil III)

It remains to define an IIM for learning problems of type B. In the definition of M_B, we let $\Gamma'' = \{nc\text{-}q(X) \leftarrow c\text{-}q(X), c\text{-}q(X) \leftarrow q(X), c\text{-}q(XY) \leftarrow c\text{-}q(X), c\text{-}q(XY) \leftarrow c\text{-}q(Y), r(XYZ) \leftarrow p(X), nc\text{-}q(Y), q(Z)\}$ and $\Pi'' = \{p, q, nc\text{-}q, c\text{-}q, r\}$.

IIM M_B: On input $S = u_0, \ldots, u_k$ do the following:

Let B and E be the set of all non-empty prefixes and suffixes of strings in S.

Moreover, let $B' = \{p(b) \leftarrow \mid b \in B\}$ and $E' = \{q(e) \leftarrow \mid e \in E\}$.

Let H be the collection of all sets $h \subseteq B' \cup E'$ such that h contains at most k rules from B' and at most k rules from E'.

Search for an $h \in H$ such that, for every $u \in S$, $r(u) \in Sem(S)$ holds, where S is the EFS $(\Sigma, \Pi'', \Gamma'' \cup h)$. If such an h is found, let Γ' be the lexicographically first of them. Otherwise, set $\Gamma' = \emptyset$. Output Γ'.

7. Praxisrelevanz und Bewertung

- Algorithmus wurde bereits implementiert und an HTML getestet
- http://LExIKON.dfki.de
- Basisprinzip der Wrapper-Induction auch in diesem Ansatz vorhanden
- Theorie zeigt das Potential des gewählten Ansatzes
- Vorgestellte Wrapper-Induction bisher nur in der Praxis evaluiert