Vergleich von Ansätzen zur Repräsentation des Sprachbias in der ILP*

Birgit Tausend
Institut für Informatik, Universität Stuttgart, Breitwiesenstr. 20-22, D-70565 Stuttgart
tausend@informatik.uni-stuttgart.de

1 Einführung

Die Verfahren der Induktiven Logischen Programmierung (ILP) [Mug93] haben die Aufgabe, aus einer Menge von positiven Beispielen E^+, einer Menge von negativen Beispielen E^- und dem Hintergrundwissen B ein logisches Programm P zu lernen, das aus einer Menge von definiten Klauseln $C : l_0 \leftarrow l_1, \ldots, l_n$ besteht. Da der Hypothesenraum für Hornlogik unendlich ist, schränken viele Verfahren die Hypothesensprache auf eine endliche ein. Auch wird oft versucht, die Hypothesensprache so einzuschränken, daß nur Programme gelernt werden können, für die die Konsistenz entscheidbar ist. Eine andere Motivation, die Hypothesensprache zu beschränken, ist, daß das Wissen über das Zielprogramm, das schon vorhanden ist, ausgenutzt werden soll. So sind für bestimmte Anwendungen funktionsfreie Hypothesenklauseln ausreichend, oder es ist bekannt, daß das Zielprogramm funktional ist.

2 Repräsentationsansätze und Vergleichskriterien

Die Beschränkungen der Hypothesensprache können sowohl die Signatur betreffen, wie z.B. in den funktionsfreien Hypothesensprachen, als auch die Konstruktion von Termen und Klauseln oder zu modellabhängige Einschränkungen, wie z.B. die Beschränkung auf funktionale Klauseln. Einen Überblick gibt [Tau94a].

Um die Beschränkungen, den sogenannten Sprachbias, zu repräsentieren, werden im wesentlichen vier Repräsentationsformalismen verwendet, nämlich *parameterisierbare Sprachen*, wie in CLINT [DR91], *schemabasierte Ansätze*, wie die Regelmodelle in RDT [KW92] oder die Klauselbeschreibungen in MILES-CTL [Tau94b], *Grammatiken*, wie z.B. in GRENDEL [Coh93] oder GRDT [Kli94], und *Klauselmengen* [Ber93].

In diesem Papier vergleichen wir diese Ansätze anhand verschiedener Kriterien. Ein sehr wichtiger Punkt ist die *Adäquatheit*, d.h. welche der in der ILP benutzten Beschränkungen in einem Ansatz darstellbar sind, da in den meisten Ansätze nicht alle repräsentierbar sind. Deshalb ist auch die *Erweiterbarkeit* des Repräsentationsformalismus zur Darstellung zusätzlicher Beschränkungen ein wichtiges Vergleichskriterium. Ebenfalls wichtig ist, wie gut die *Darstellung gemeinsamer Eigenschaften* der Hypothesen in einem Ansatz möglich ist, z.B. die Verbundenheit der Klauseln. Ein weiteres Kriterium ist, wie kompakt eine Hypothesensprache repräsentiert wird, die aus *Teilmengen von Hypothesen mit*

*teilweise gefördert durch ESPRIT BRA 6020: Inductive Logic Programming

unterschiedlichen Eigenschaften besteht. Außerdem muß beurteilt werden, wie gut aus der Repräsentation erkennbar ist, *welche Klauseln* in der Hypothesensprache sind. Als letztes Kriterium beurteilen wir die *Kompaktheit*, d.h. wie umfangreich die Repräsentation der Hypothesensprache ist.

Neben diesen Kriterien gibt es noch weitere, die wir in diesem Paper aber nicht behandeln, z.B. welche Vorteile ein Repräsentationsansatz für die Suche im Hypothesenraum hat oder wie das Verschieben des Bias durch die Repräsentation unterstützt werden kann.

Für den Vergleich repräsentieren wir in den verschiedene Ansätzen die beiden folgenden Hypothesensprachen:

- Die Hypothesensprache L_1 sei die Menge aller verbundenen Klauseln mit eindeutigen Variablen und maximal einer neuen Variable in jedem der Rumpfliterale, die mehrmals in verschiedenen Literalen vorkommen kann.
- Die Hypothesensprache L_2 sei gegeben durch die Regelmodelle

$P(X,Y) :- Q1(Y,X)$ $\quad$ $P(X,Y) :- Q0(X), Q1(X,Z), Q2(Z,Y)$
$P(X,Y) :- Q1(X,Z), Q2(Z,Y)$ $\quad$ $P(X,Y) :- Q0(Y), Q1(X,Z), Q2(Z,Y)$
$P(X,Y) :- Q1(X), Q2(X,Y)$ $\quad$ $P(X,Y) :- Q0(X,Y), Q1(X,Z), Q2(Z,Y)$
$P(X,Y) :- Q1(Y), Q2(X,Y)$ $\quad$ $P(X,Y) :- Q0(Y,X), Q1(X,Z), Q2(Z,Y)$

und das Hintergrundwissen und die Beispielen $B \cup E$ enthalten die Prädikate $p11/1$, $p12/1$ $p21/2$, $p22/2$, $p31/3$ und $p32/3$.

3 Repräsentation mit parameterisierbaren Sprachen

Im Ansatz der parameterisierten Sprachen, den CLINT [DR91] verwendet, wird die Hypothesensprache als eine Menge von Klauseln beschrieben, die bestimmte Bedingungen erfüllen müssen. In CLINT werden die Beschränkung auf funktions- und konstantenfreie, bereichsbeschränkte und verbundene Klauseln mit eindeutigen Variablen und einer maximal Anzahl von neue Variablen repräsentiert. Dabei kann die maximal Anzahl von neue Variablen fest oder variabel in Abhängigkeit von der Literaltiefe sein und der maximalen Tiefe von Variablen beschränkt sein.

Diese Bedingungen reichen aber nicht aus, um die Sprachen L_1 und L_2 zu repräsentieren. Allerdings ist dieser Ansatz ist gut erweiterbar, da nur neue Bedingungen definiert und hinzugefügt werden müssen. Dann kann beispielsweise die Hypothesensprache L_1 folgendermaßen definiert werden:

$L_1 = \{C : l_0 \leftarrow l_1, \ldots, l_n \mid$ (1) C ist verbunden und
(2) alle $l_i \in C$ haben eindeutige Variablen und
(3) $\forall l_i : newvars(l_i) \leq 1$ und $1 \leq i \leq n$
(4) $\forall l_i : l_i = p_i(X_{1_i}, \ldots, X_{n_i})$ u. p_i ist ein Prädikat in $B \cup E\}$

Dieses Beispiel zeigt, daß mit dem Ansatz eine sehr kompakte Repräsentation möglich ist, denn diese vier Zeilen genügen, um die umfangreiche Sprache L_1 zu repräsentieren. Ein anderer, ganz wesentlicher Vorteil ist, daß die gemeinsamen Eigenschaften der Hypothesenklauseln aus der Darstellung gut erkennbar sind, denn diese werden ja gerade durch die Bedingungen repräsentiert.

In der Darstellung mit Bedingungen liegen aber auch die Schwächen des Ansatzes. Gerade weil die Bedingungen normalerweise für alle Klauseln in der Sprache gelten müssen, sind Hypothesensprachen, die aus Teilmengen von Klauseln mit unterschiedlichen Beschränkungen bestehen, nicht kompakt darstellbar, wie das nächste Beispiel zeigt. Zum anderen muß ein Benutzer erst die Bedingungen 'übersetzen', um zu verstehen, welche Klauseln in der Hypothesensprache sind.

L_2= $\{C : l_0 \leftarrow l_1 \mid$ (1) $l_0 = p_0(X,Y)$ und $l_1 = p_1(Y,X)$,
(2) p_0 und p_1 sind Prädikate der Wissensbasis $\}$
$\cup \{C : l_0 \leftarrow l_1, l_2 \mid$ (1) $l_0 = p_0(X,Y)$, $l_1 = p_1(X,Z)$ und $l_2 = p_2(Z,Y)$
(2) $\forall l_i : l_i = p_i(X_{1_i}, \ldots, X_{n_i})$ u. p_i ist ein Prädikat in $B \cup E\}$
$\cup \{C : l_0 \leftarrow l_1, l_2 \mid$ (1) $l_0 = p_0(X,Y)$ und $l_2 = p_1(X,Y)$
(2) l_1 hat die Stelligkeit 1 und keine neuen Variablen,
(3) $\forall l_i : l_i = p_i(X_{1_i}, \ldots, X_{n_i})$ u. p_i ist ein Prädikat in $B \cup E\}$
$\cup \{C : l_0 \leftarrow l_1, l_2, l_3 \mid$ (1) $l_0 = p_0(X,Y)$
(2) l_1 hat die Stelligkeit 1 und keine neuen Variablen,
(3) $l_2 = p_2(X,Z)$ und $l_3 = p_3(Z,Y)$
(4) $\forall l_i : l_i = p_i(X_{1_i}, \ldots, X_{n_i})$ u. p_i ist ein Prädikat in $B \cup E\}$
$\cup \{C : l_0 \leftarrow l_1, l_2, l_3 \mid$ (1) $l_0 = p_0(X,Y)$
(2) l_1 hat die Stelligkeit 2 und
hat eindeutige, aber keine neuen Variablen,
(3) $l_2 = p_2(X,Z)$ und $l_3 = p_3(Z,Y)$
(4) $\forall l_i : l_i = p_i(X_{1_i}, \ldots, X_{n_i})$ u. p_i ist ein Prädikat in $B \cup E\}$

4 Repräsentation mit Grammatiken

Grammatiken sind eine andere Möglichkeit, die Hypothesensprache zu beschreiben. In GRENDEL [Coh93] werden Antecedent Description Grammars (ADGs) zur Beschreibung der Antezedenten von Klauseln verwendet. Eine Klausel C gehört zu einer Hypothesensprache, wenn sie vom Startsymbol der Grammatik $body(C)$ mit Hilfe der Grammatikregeln abgeleitet werden kann. Die Anwendung von Regeln kann durch Annotationen eingeschränkt werden [Coh93], die mit dem Schlüsselwort *where* an eine Grammatikregel angehängt werden. Mit einer ADG, wie in [Coh93] beschrieben, können ebenfalls nicht alle Sprachbeschränkungen ausgedrückt werden, so daß beide Sprachen L_1 und L_2 nicht repräsentiert werden können. Aber auch dieser Ansatz ist gut erweiterbar, nämlich durch zusätzliche Annotationen, die die fehlenden Beschränkungen realisieren. Damit kann dann L_1 repräsentiert werden durch:

$body(Head) \rightarrow lits(Vars)$ where $vars\_of(Head, Vars)$.
$lits(Vars) \rightarrow lit(Vars)$ where $true$.
$lits(Vars) \rightarrow lit(Vars), lits(Vars)$ where $true$.
$lit(Vars) \rightarrow [Lit]$ where $predicate(Lit), vars\_of(Lit, Litvars)$,
$vars\_intersect(LitVars, ConnVars)$,
$max\_no\_of\_new\_vars(LitVars, 1)$.
$vars\_of(Lit, Xs)$: $-Xs$ ist eine Liste der Argumente von Lit.
$vars\_intersect(LitVars, ConnVars)$: $-ConnVars$ ist eine Liste der verbundenen Variablen in $LitVars$, wobei $ConnVars$ nicht leer sein darf.
$max\_no\_of\_new\_vars(LitVars, 1)$: $-$ gilt, wenn $LitVars$ nicht mehr als eine neue Variable enthält.
$predicate(p11(X))$. $predicate(p21(X,Y))$.
... ...

Da durch die Annotationen fast alle Beschränkungen ausgedrückt werden, können die Hypothesensprachen meist kompakt durch wenige Grammatikregeln beschrieben und die gemeinsamen Eigenschaften das Hypothesenklauseln gut erkannt werden. Dagegen ist es für einen Benutzer schwierig, zu verstehen, welche Hypothesenklauseln in der Sprache sind, da erst alle Ableitungen gebildet werden müssen. Auch ist die Darstellung von Hypothesensprachen aufwendig, die Teilmengen mit unterschiedlichen Beschränkungen haben, wie die Repräsentation von L_2 zeigt.

$body(Head) \rightarrow lits(Vars)$ where $vars\_of(Head, [X, Y])$.
$lits(Vars) \rightarrow [Lit]$ where $predicate(Lit), Vars = [X, Y]$ und $LitVars = [Y, X]$.
$lits(Vars) \rightarrow lit1(Vars), [Lit]$ where $predicate(Lit), vars\_of(Lit, LitVars)$
$Vars = [X, Y] = LitVars$.
$lits(Vars) \rightarrow lits1(Vars)$ where $true$.
$lits(Vars) \rightarrow lit1(Vars), lits1(Vars)$ where $true$.
$lits(Vars) \rightarrow lit2(Vars), lits1(Vars)$ where $true$.
$lits1(Vars) \rightarrow [Lit1], [Lit2]$ where $predicate(Lit1), vars\_of(Lit1, LitVars1)$,
$predicate(Lit2), vars\_of(Lit2, LitVars2)$,
$Vars = [X, Y]$, $LitVars1 = [X, Z]$ u. $LitVars2 = [Z, Y]$.
$lit1(Vars) \rightarrow [Lit]$ where $predicate(Lit), vars\_of(Lit, LitVars)$,
$length(LitVars, 1)$ und $members(LitVars, Vars)$.
$lit2(Vars) \rightarrow [Lit]$ where $predicate(Lit), vars\_of(Lit, LVars), perm(Vars, LVars)$.
$vars\_of(Lit, Xs)$: $-Xs$ ist eine Liste der Argumente von Lit.
$members(Xs, Ys)$: –jedes X in der Liste Xs ist in der Liste Ys enthalten.
$perm(Xs, Ys)$: $-Ys$ ist eine Permuation der Liste Ys.
$length(Xs, N)$: $-N$ ist die Länge der Liste Xs.
$predicate(p11(X))$. $predicate(p21(X, Y))$.
... ...

5 Repräsentation mit Schemata

Im schemabasierten Ansatz betrachten wir nur die Repräsentation mit Regelmodellen, wie in RDT [KW92], und mit Klauselbeschreibungen, wie in MILES-CTL [Tau94b], da andere Ansätzen, wie die Abhängigkeitsgraphen in SIERES [WO91], zur Repräsentation der meisten Beschränkungen nicht geeignet sind.

Regelmodelle sind Hornklauseln zweiter Ordnung, die Prädikatvariablen enthalten, d.h. sie abstrahieren nur von den Prädikatnamen. Sie sind gut geeignet zur Darstellung funktionfreier Hypothesensprachen, selbst wennn diese aus Teilmengen von Hypothesen mit unterschiedlichen Beschränkungen bestehen, wie die Hypothesensprache L_2 zeigt, die ja mit Regelmodellen spezifiziert wurde. Jedoch ist dieser Ansatz nicht adäquat, da z.B. nur funktionsfreie Hypothesensprachen darstellbar sind. Problematisch ist ebenfalls das Erkennen von gemeinsamen Eigenschaften. Ein weiterer Nachteil der Regelmodelle ist, daß es nicht möglich ist, eine große Hypothesensprache kompakt darzustellen, da nur von den Prädikatnamen abstrahiert wird und für jede Verteilung der Terme ein eigenes Regelmodell definiert werden muß, wie die Repräsentation von L_1 zeigt:

$P(X) : -Q1(X), Q2(X)$	$P(X) : -Q1(X, Y), Q2(X)$
$P(X) : -Q1(Y, X), Q2(X)$	$P(X) : -Q1(X), Q2(X, Y)$
$P(X) : -Q1(X), Q2(Y, X)$	$P(X) : -Q1(X), Q2(X, Y)$
$P(X, Y) : -Q1(X), Q2(X)$	$P(X, Y) : -Q1(Y), Q2(Y)$
$P(X, Y) : -Q1(X), Q2(Y)$	$P(X, Y) : -Q1(Y), Q2(X)$
$P(X, Y) : -Q1(Y, X), Q2(X)$	$P(X, Y) : -Q1(Y, X), Q2(Y)$
$P(X, Y) : -Q1(X, Y), Q2(X)$	$P(X, Y) : -Q1(X, Y), Q2(Y)$
$P(X, Y) : -Q1(Y, X), Q2(X, Y)$	$P(X, Y) : -Q1(Y, X), Q2(Y, X)$
$P(X, Y) : -Q1(X, Y), Q2(Y, X)$	$P(X, Y) : -Q1(X, Y), Q2(X, Y)$
$P(X, Y) : -Q1(Z, X), Q2(X)$	$P(X, Y) : -Q1(Z, Y), Q2(Y)$
$P(X, Y) : -Q1(Z, X), Q2(Y)$	$P(X, Y) : -Q1(Z, Y), Q2(X)$
...	

Dieses Problem soll mit den Klauselbeschreibungen, einer Erweiterung von Regelmodellen in MILES-CTL [Tau94b] vermieden werden. Eine Klauselbeschreibung besteht aus Literalbeschreibungen für die Kopf- und Rumpfliterale. Jede

Literalbeschreibung besteht aus einer Menge von Informationen, die die Menge der abgedeckten Literale spezifizieren. Jede Information wird durch einen eindeutigen Identifier gekennzeichnet, z.B. *predicate* für die Information über das Prädikat. In MILES-CTL sind folgenden Identifiers erlaubt:

predicate:	das Prädikat,
arguments:	die Menge der Argumente mit ihrer Position,
terms:	die Menge der Terme in den Argumenten, d.h. $terms(l_i)$
determinate_terms:	die Menge der determinierten Terme in den Argumenten,
variables:	die Menge der Variablen in den Argumenten, d.h. $vars(l_i)$
new_variables:	die Menge der neuen Variablen in den Argumenten,
unique_variables:	Flag, dessen Wert *yes*, falls die Variablen in den Argumenten eindeutig sind,
generative:	Flag, dessen Wert *yes* ist, wenn das Kopfliteral nur Variablen enthält, die auch im im Klauselrumpf vorkommen,
arity:	die Stelligkeit des Prädikats in *predicate*,
argument_types:	die Menge der Argumenttypen mit ihrer Position,
depth:	die Tiefe des Literals,
mode:	die Modusdeklaration,
predicate_type:	der Typ des Prädikats in *predicate*,
commutative:	gesetzt, falls kommutative Varianten des Literals erlaubt sind,
solutions:	Anzahl der Lösungen eines Literals.

Nach dem Identifier folgt, getrennt durch einen Doppelpunkt, eine Schemakonstante oder eine Schemavariable. Darüberhinaus ist es möglich, daß nach einer Schemavariable eine Bedingung steht. Dies bedeutet, daß eine solche Literalbeschreibung nur die Literale abdeckt, die diese Bedingung erfüllt. So beschränkt z.B. die Information $arity : A||(A \leq 3)$ die Stelligkeit der abgedeckten Literale auf höchstens 3.

Damit kann L_1 in MILES-CTL repräsentiert werden durch Klauselbeschreibungen der Form:

$$T : \begin{bmatrix} predicate : P0, \\ arguments : A0 \end{bmatrix} \leftarrow \begin{bmatrix} predicate : P1, \\ arguments : A1, \\ new\_vars : N1||(N1 \leq 1) \\ depth : D1||(D1 \geq 1) \\ unique : yes \end{bmatrix}, \ldots \begin{bmatrix} predicate : PN, \\ arguments : AN, \\ new\_vars : NN||(NN \leq 1), \\ depth : DN||(DN \geq 1), \\ unique : yes \end{bmatrix}.$$

Die Hypothesensprache L_2 kann mit den vier Klauselbeschreibungen $T1$, $T2$, $T3$ und $T4$ dargestellt werden:

$$T1 : \begin{bmatrix} predicate : P0, \\ arguments : (X, Y) \end{bmatrix} \leftarrow \begin{bmatrix} predicate : P1, \\ arguments : (Y, X) \end{bmatrix}.$$

$$T2: \begin{bmatrix} predicate: P0, \\ arguments: (X,Y) \end{bmatrix} \leftarrow \begin{bmatrix} predicate: P1, \\ arguments: (X,Z) \end{bmatrix}, \begin{bmatrix} predicate: P1, \\ arguments: (Z,Y) \end{bmatrix}.$$

$$T3: \begin{bmatrix} predicate: P0, \\ arguments: (X,Y) \end{bmatrix} \leftarrow \begin{bmatrix} predicate: P1, \\ arguments: A1 || (A1 \subseteq A0), \\ arity: 1 \end{bmatrix}, \begin{bmatrix} predicate: P2, \\ arguments: (X,Y) \end{bmatrix}.$$

$$T4: \begin{bmatrix} predicate: P0, \\ arguments: (X,Y) \end{bmatrix} \leftarrow \begin{bmatrix} predicate: P1, \\ arguments: A1 || (A1 \subseteq A0), \\ arity: Ar1 || (Ar1 \leq 2), \\ new\_vars: \emptyset \\ unique: yes \end{bmatrix}, \begin{bmatrix} predicate: P3, \\ arguments: (X,Z) \end{bmatrix}, \begin{bmatrix} predicate: P3, \\ arguments: (Z,Y) \end{bmatrix}.$$

6 Repräsentation mit Klauselmengen

Der vierte Repräsentationsansatz sind die Klauselmengen. Die Hypothesensprache wird repräsentiert, indem Klauseln, Literale und Terme, die in den Hypothesen erlaubt sind, in Mengen zusammengefaßt werden, um eine kompaktere Darstellung der Hypothesen zu erhalten. Aus diesen Mengen können für eine Hypothese Elemente ausgewählt werden. Ein Vorteil dieser Repräsentation ist, daß schnell erkennbar ist, welche Klauseln in der Hypothesensprache sind. Dagegen können nur sehr wenige der Sprachbeschränkungen repräsentiert werden und die Repräsentation von größeren Hypothesensprachen oder Sprachen aus Hypothesenmengen mit unterschiedlichen Beschränkungen ist nicht kompakt. Soll beispielweise L_2 repräsentiert werden, muß fast für jede Hypothesenklausel eine Klausel in der Klauselmenge direkt vorhanden sein:

$$\begin{array}{ll}
\{\ p21(X,Y) :- p12(Y,X), & p21(X,Y) :- p11(\{X,Y\}), p21(X,Y) \\
\ \ p21(X,Y) :- p22(Y,X), & p21(X,Y) :- p12(\{X,Y\}), p21(X,Y) \\
\ \ p22(X,Y) :- p21(Y,X), & p21(X,Y) :- p11(\{X,Y\}), p22(X,Y) \\
\ \ p22(X,Y) :- p22(Y,X), & p21(X,Y) :- p12(\{X,Y\}), p22(X,Y) \\
\ \ p21(X,Y) :- p21(X,Z), p21(Z,Y), & p22(X,Y) :- p11(\{X,Y\}), p21(X,Y) \\
\ \ p21(X,Y) :- p22(X,Z), p21(Z,Y), & p22(X,Y) :- p12(\{X,Y\}), p21(X,Y) \\
\ \ \ldots & \ldots \qquad\qquad \}
\end{array}$$

Ähnliches gilt L_1, denn insbesondere die Beschränkung auf verbundene Klauseln mit eindeutigen Variablen läßt sich nur exakt ausdrücken, wenn für jede Hypothese eine Klausel in der Klauselmenge formuliert wird. Andere Abstraktionen, wie z.B. durch Termmengen, würden dazu führen, daß die Menge der abgedeckten Hypothesen auch Klauseln mit nicht eindeutigen Variablen in den Literalen enthält. Der Grund ist, daß dieser Ansatz sehr wenige Abstaktionsmöglichkeiten bietet, nämlich nur die durch das Bilden der verschiedenen Mengen. Deshalb und weil bereits alle Möglichkeiten zur Mengenbildung ausgeschöpft sind, ist diese Repräsentation schlecht erweiterbar.

7 Zusammenfassung

Parameterisierten Sprachen und Grammatiken schneiden bezüglich der Kompaktheit, der Erweiterbarkeit und der Darstellung gemeinsamer Eigenschaften am besten ab. Die Nachteile beider Ansätze liegen vor allem darin, daß sich in beiden Repräsentationen oft nicht direkt erkennen läßt, welche Klauseln in

Hypothesensprache sind und auch Hypothesensprachen aus Gruppen von Hypothesen mit unterschiedlichen Beschränkungen meist nicht kompakt darstellbar sind. Die Stärken von Regelmodelle und Klauselbeschreibungen liegen dagegen genau in diesen beiden Punkten. Der Nachteil von Regelmodellen ist im wesentlichen, daß viele der Beschränkungen nicht repräsentiert werden können und gemeinsame Eigenschaften von Hypothesen nur schwer erkannt werden. Klauselbeschreibungen sind eine Verbesserung gegenüber Regelmodellen, denn sie ermöglichen die Darstellung der meisten Beschränkungen und eine kompaktere Darstellung der Hypothesensprache, da sie über größere Abstraktionsmöglichkeiten als die Regelmodelle verfügen. Für beide Ansätze gilt jedoch, daß eine kompakte Darstellung bei großen Sprachen nicht immer möglich ist, da die Anzahl der Literale in einem Schema festgelegt ist. Klauselmengen schneiden noch schlechter ab, den außer beim Erkennen, welche Klauseln in der Hypothesenspache sind, versagt diese Repräsentation in allen Punkten.

Literatur

[Ber93] F. Bergadano. Towards an inductive logic programming language. Deliverable TO1, ESPRIT Project No. 6020 ILP, 1993.

[Coh93] W.W. Cohen. Rapid prototyping of ILP systems using explicit bias. In *Proc. of IJCAI-93 Workshop on ILP.* Morgan Kaufmann, 1993.

[DR91] L. De Raedt. *Interactive Concept-Learning.* PhD thesis, Katholieke Universiteit Leuven, 1991.

[Kli94] V. Klingspor. GRDT: Enhancing model-based learning for its application in robot navigation. LS-8 report 5, FB Informatik, Univers. Dortmund, 1994.

[KW92] J.U. Kietz and S. Wrobel. Controlling the complexity of learning in logic through syntactic and task-oriented models. In S. Muggleton, editor, *Inductive Logic Programming.* Academic Press, 1992.

[Mug93] S. Muggleton. Inductive Logic Programming: Derivations, successes and shortcoming. In *Machine Learning: ECML-93, Wien.* Springer, 1993.

[Tau94a] B. Tausend. Biases and their effects in Inductive Logic Programming. In *Machine Learning: ECML-94, Catania, Italy.* Springer, 1994.

[Tau94b] B. Tausend. Representing biases for Inductive Logic Programming. In *Machine Learning: ECML-94, Catania, Italy.* Springer, 1994.

[WO91] R. Wirth and P. O'Rorke. Constraints on predicate invention. In *Eighth Int. Conference on Machine Learning.* Morgan Kaufmann, 1991.