Monadische Prädikatenlogik zweiter Stufe in der Linguistik

Uwe Mönnich and Frank Morawietz

SFB 340, A8
Universität Tübingen

{um,frank}@sfs.nphil.uni-tuebingen.de

http://www.sfs.nphil.uni-tuebingen.de/~frank

CIS München, 04. Februar 1999

1. Motivation and Background
2. What is the Logic–Automaton Connection?
   (a) What is MSO Logic on Trees (WS2S)?
   (b) How is it decidable?
   (c) Complexity?
3. How can we use the technique of the decidability proof for NLP
   (a) in theory?
   (b) in practice?

Model Theoretic Syntax

- well-formedness conditions on structures (e.g. trees, feature structures)
- monostratal grammar formalisms
- "constraint-based"
- formal language with syntax and semantics
- our starting point: Rogers (1994,1998)

Theorem (Doner, Thatcher & Wright)

The weak monadic second-order theory of 2 successor functions (WS2S) is decidable.

- stronger results: Rabin, Rogers
- linguistics needs only finite sets
- n-successors can be compiled to 2 successors (e.g., LISA)

The structure of WS2S

The **structure of 2-successor functions** $\mathcal{N}_2$ is defined as

$$\mathcal{N}_2 = \langle \mathcal{T}_2, \varepsilon, \leq, S_i, \rangle_{i \in \{0,1\}}$$

where

- $\mathcal{T}_2$ is a binary-branching tree domain,
- ε denotes its root,
- $\leq$ is the prefix relation and
- the S_i are the successor functions.

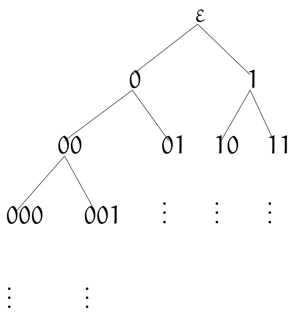
Tree Domains

A tree domain $\mathcal{T}$ is a subset of strings over a linearly ordered set which is closed under prefix and left sister.

$\mathcal{T} \subseteq \mathbb{N}^*$ is a **tree domain** if for all $u, v \in \mathbb{N}^*$ and $i, j \in \mathbb{N}$ the following holds:

1. $uv \in T \Rightarrow u \in T$ (prefix closure) and
2. $ui \in T$ and $j < i \Rightarrow uj \in T$ (left-sister closure).

Here, we use $\{0, 1\}$ instead of $\mathbb{N}$.



The MSO Language of WS2S

The MSO-Language $\mathcal{L}$ of WS2S consists of

- a set of *monadic* second-order variables (X,Y, ...);
- a set of individual variables (x,y, ...);
- boolean connectives: $\neg$, $\bigwedge$, ... ;
- Quantifiers over individuals and sets: $\exists$, $\forall$;
- binary predicates for the successor functions: s_i;
- prefix, equality and membership: $\leq$, $\approx$, $\in$.

Terme und WFF

- **Terme** sind nur die Variablen.[a]
- **Wohlgeformten Formeln** (WFF) sind als die kleinste Menge definiert, die die folgenden Bedingungen erfüllt:
  - Wenn x, y Individuenvariablen sind und s_i ist eine Nachfolgerrelation, dann ist $s_i(x, y)$ eine WFF;
  - Wenn x, y Individuenvariablen sind und X eine Mengenvariable ist, dann sind auch $x \approx y$, $x \leq y$ und $x \in X$ (alternativ $X(x)$) WFF;
  - Wenn φ und ψ WFF sind, dann sind auch $\neg\varphi$, $\varphi \wedge \psi$ und $\varphi \vee \psi$ WFF;
  - Wenn x, X Variablen sind und φ eine Formel ist, dann sind auch $(\exists x)[\varphi]$ bzw. $(\exists X)[\varphi]$ und $(\forall x)[\varphi]$ bzw. $(\forall X)[\varphi]$ WFF;
- **Sätze** sind WFF ohne freie Variablen.

[a]Wenn die Sprache Konstanten hätte, wären diese auch Terme.

Erfüllbarkeit

$$\mathcal{N}_n \models \phi(x_1, \ldots, x_n, X_1, \ldots, X_m)[\alpha]$$

wobei α Individuenvariablen zu Knoten und (monadische) Variablen 2.Stufe zu Mengen von Knoten abbildet.

An Alternative (Rogers, 1994): $\mathcal{L}^2_{\mathbf{K},\mathbf{P}}$

- K a set of individual constants
- P a set of predicate constants
- immediate dominance: $\triangleleft$
- reflexive dominance: $\triangleleft^*$
- proper dominance: $\triangleleft^+$
- proper precedence: $\prec$
- connectives, variables, equality, set membership and quantifiers as for WS2S.

Models for $\mathcal{L}^2_{\mathbf{K},\mathbf{P}}$

A model for $\mathcal{L}^2_{\mathsf{K},\mathsf{P}}$ is a tuple $\langle \mathcal{T}, \mathcal{I}, \mathcal{P}, \mathcal{D}, \mathcal{L}, \mathcal{R}_p \rangle_{p \in \mathbf{P}}$ with:

$\mathcal{T}$	a nonempty tree domain
$\mathcal{I}$	a function from K to $\mathcal{T}$
$\mathcal{P}$, $\mathcal{D}$, $\mathcal{L}$	binary relations on $\mathcal{T}$ which interpret $\triangleleft$, $\triangleleft^+$ and $\prec$
$\mathcal{R}_p$	relations of appropriate arity on $\mathcal{T}$ interpreting $p \in \mathsf{P}$.

A model for $\mathcal{L}^2$ is then just a tuple $\langle \mathcal{T}, \mathcal{P}, \mathcal{D}, \mathcal{L} \rangle$.

Definierbarkeit

Wir können die Signatur von WS2S mit beliebigen Prädikaten erweitern, so lange die Relationen

- explizit definierbar;
- monadisch und induktiv-elementar definierbar oder
- durch *tree-walking* Automaten definierbar sind.

Example formulas I

$X \subseteq Y \overset{\text{def}}{\iff} (\forall x)[x \in X \Rightarrow x \in Y]$

$\mathsf{Sing}(X) \overset{\text{def}}{\iff}$

% Alle Teilmengen von X sind gleich X oder leer:

$(\forall Y)[Y \subseteq X \Rightarrow (X \subseteq Y \vee (\forall Z)[Y \subseteq Z]\,)] \wedge$

% X ist nicht die leere Menge:

$(\exists Y)[X \not\subseteq Y]$

Example formulas II

$\mathsf{AC\text{-}Com}(x, y) \overset{\text{def}}{\iff}$

% x c-commandiert y:

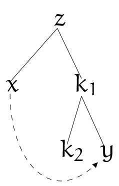
$(\forall z)[z \lhd^+ x \Rightarrow z \lhd^+ y] \wedge \neg(x \lhd^* y) \wedge$

% y c-commandiert x nicht:

$\neg((\forall z)[z \lhd^+ y \Rightarrow z \lhd^+ x] \wedge \neg(y \lhd^* x)) \wedge$

% x kommt vor y:

$x \prec y$



Example formulas III

$\mathsf{LC}(x, y) \overset{\text{def}}{\iff}$

$\mathsf{AC\text{-}Com}(x, y) \wedge$

% Es gibt kein k_1 mit Eigenschaft P:

$(\neg\exists k_1)[k_1 \in P \wedge$

% derart, daß es zwischen x und y interveniert:

$(\exists z)[z \lhd x \wedge z \lhd^+ k_1 \wedge k_1 \lhd^+ y]]$

How is WS2S decidable?

$$\mathcal{N}_2 \models \phi\,[\tau] \text{ iff } t \in T(\mathfrak{A}^{\phi})$$

Interesting Direction:

1. coding of variable assignments into the alphabet of the tree automata (t is the coding of τ)
2. inductive translation of formulas into tree automata
   (a) base case: present automata recognizing the relations from the signature;
   (b) inductive step: use automata constructions to parallel the connectives and quantifiers;
3. emptiness of the languages recognized by tree automata is decidable.

(Bottom-Up) Tree Automata

$$\mathfrak{A} = \langle A, \Sigma, a_0, F, \alpha \rangle$$

A the set of states

Σ the alphabet

$a_0 \in A$ the initial state

$F \subseteq A$ the set of final states

$\alpha : (A^n \times \Sigma) \to A$ the transition function

Here: $n = 2$

$$h_\alpha(\lambda) = a_0$$
$$h_\alpha(\sigma(t_1, t_2)) = \alpha(h_\alpha(t_1), h_\alpha(t_2), \sigma)$$

$\mathfrak{A}$ accepts a tree t iff $h_\alpha(t) \in F$.

$$T(\mathfrak{A}) = \{t \mid h_\alpha(t) \in F\}$$

An Example Tree Automaton

$$\mathfrak{A} = \langle \{a_0, a_1\}, \{A, B\}, a_0, \{a_0\}, \alpha \rangle$$

$$\alpha(a_0, a_0, A) = a_0$$
$$\alpha(a_0, a_1, A) = a_1$$
$$\alpha(a_1, a_0, A) = a_1$$
$$\alpha(a_1, a_1, A) = a_1$$
$$\alpha(a_0, a_0, B) = a_1$$
$$\alpha(a_0, a_1, B) = a_1$$
$$\alpha(a_1, a_0, B) = a_1$$
$$\alpha(a_1, a_1, B) = a_1$$

Coding the assignments

$$\mathsf{Connected}(P) \stackrel{\text{def}}{\iff} (\forall x, y, z)[(x \in P \land y \in P \land x \lhd^* z \land z \lhd^* y) \Rightarrow z \in P]$$

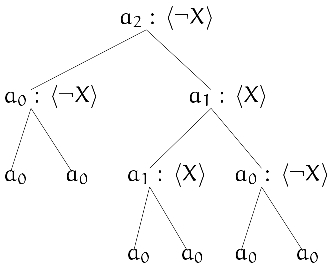
$$\mathsf{Path}(P) \stackrel{\text{def}}{\iff} \mathsf{Connected}(P) \land (\forall x, y \in P)[x \lhd^* y \lor y \lhd^* x]$$

The Tree Automaton for **Path**(X)

$$\mathfrak{A} = \langle \{a_0, a_1, a_2, a_3\}, \{\langle \neg X\rangle, \langle X\rangle\}, a_0, \{a_0, a_1, a_2\}, \alpha\rangle$$

$$\begin{array}{ll}
\alpha(a_0, a_0, \langle \neg X\rangle) = a_0 & \alpha(a_0, a_0, \langle \ X\rangle) = a_1 \\
\alpha(a_0, a_1, \langle \neg X\rangle) = a_2 & \alpha(a_0, a_1, \langle \ X\rangle) = a_1 \\
\alpha(a_1, a_0, \langle \neg X\rangle) = a_2 & \alpha(a_1, a_0, \langle \ X\rangle) = a_1 \\
\alpha(a_0, a_2, \langle \neg X\rangle) = a_2 & \alpha(a_2, a_0, \langle \neg X\rangle) = a_2
\end{array}$$

all other transitions are to a_3



Some facts about Tree Automata

Definition: Mengen von Bäumen, die von Baumautomaten erkannt werden, nennt man **erkennbar**.

Lemma: Erkennbare Mengen sind abgeschlossen unter Konjunktion, Disjunktion, Negation (Komplement), Projektion und Zylindrifizierung, da es für sie entsprechende bezeugende Automatenkonstruktionen gibt. Weitere wohldefinierte Konstruktionen sind Determinisierung und Minimierung.

“Linguistic” Definitions

- A constraint is **satisfiable** iff it results in a non-empty automaton.
- A constraint is **valid** iff it results in the trivial non-empty automaton.
- A (maximally) binary branching **tree** is a subset T of the domain of $\mathcal{N}_2$. A **labeled tree** is a tuple $\langle T, F_1, \ldots, F_k\rangle$.
- A **grammar** in this setting is a definition of a $k+1$-ary relation on $\mathcal{N}_2$ designating all and only the well-formed labeled trees.
- The **recognition problem** is the determinization of the emptiness of the conjunction of the grammar formula/automaton with the input formula/automaton.
- The **parsing problem** is the conjunction of the grammar formula/automaton with the input formula/automaton.

A Mini Grammar I

$$(\forall x)[\mathsf{Lex}(x) \lor \mathsf{Rule}(x)]$$

$$\begin{array}{lllll}
\mathsf{Lex}(x) \overset{\text{def}}{\Longleftrightarrow} & & (C(x) & \Rightarrow & \text{weil}) \\
& \lor & (V(x) & \Rightarrow & \text{schlug}) \\
& \lor & (N(x) & \Rightarrow & (\text{Schuft} \lor \text{Hund})) \\
& \lor & (D(x) & \Rightarrow & (\text{der} \lor \text{den}))
\end{array}$$

$$\mathsf{Rule}(x) \overset{\text{def}}{\Longleftrightarrow}$$

$$\begin{array}{l}
(\exists y, z)[(C(x) \Rightarrow x \lhd y \land x \lhd z \land C(y) \land V(z) \land y \prec z) \\
\quad \lor (V(x) \Rightarrow x \lhd y \land x \lhd z \land D(y) \land V(z) \land y \prec z) \\
\quad \lor (D(x) \Rightarrow x \lhd y \land x \lhd z \land D(y) \land N(z) \land y \prec z)]
\end{array}$$

$$\begin{array}{lllll}
\mathsf{Cat}(x, y) \overset{\text{def}}{\Longleftrightarrow} & & (C(x) & \Leftrightarrow & C(y)) \\
& \lor & (V(x) & \Leftrightarrow & V(y)) \\
& \lor & (N(x) & \Leftrightarrow & N(y)) \\
& \lor & (D(x) & \Leftrightarrow & D(y))
\end{array}$$

A Mini Grammar II

Binary Branching:

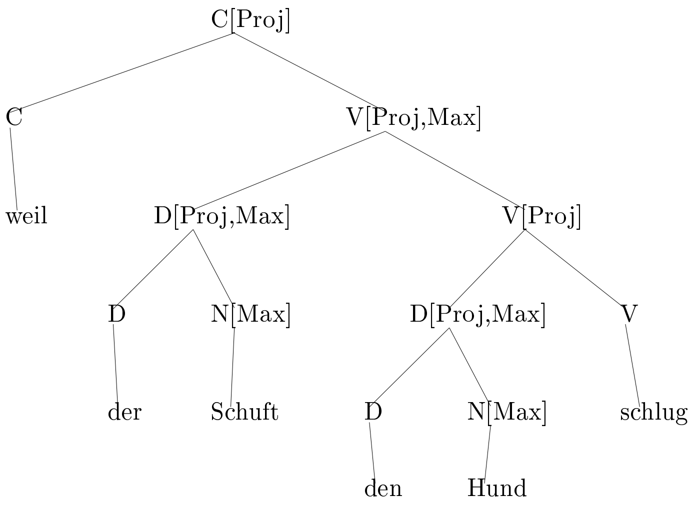
$$(\forall x)[(\exists y,z)[x \lhd y \wedge x \lhd z \wedge \neg(y \approx z) \wedge (\forall w)[x \lhd w \Rightarrow (w \approx y \vee w \approx z)] \wedge \neg \mathsf{Lex}(x)] \vee \neg(\exists y)[x \lhd y]\ ]$$

Distribution of Proj:

$$(\forall x)[(\exists y)[x \lhd y \wedge \mathsf{Cat}(x,y)] \Leftrightarrow \mathsf{Proj}(x)]$$

Distribution of Max:

$$(\forall x)[(\exists y)[y \lhd x \wedge \neg \mathsf{Cat}(x,y)] \Leftrightarrow Max(x)]$$



Some more Results for MSO Logic on Trees

Lemma: Die Klasse von String-Sprachen, die die Yields von erkennbaren Mengen (Bäumen) formen, entspricht genau der der kontextfreien Sprachen.

Lemma: Prinzipien bzw. Constraints, die man in WS2S definieren kann, lizensieren genau die schwach kontextfreien Sprachen.

Advantages

- Concise Representation in a Powerful and Flexible Logic
- Descriptive Complexity Results
- Decidable
- Compilation: Formulas to Tree Automata

Shortcomings

- Non-Elementary Complexity of the Decidability Algorithm:

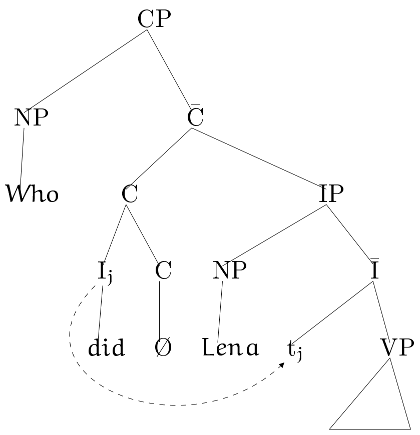
$$|A| = \left.2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}}\right\}m$$

- Descriptive Complexity: Only Context-Free String Languages

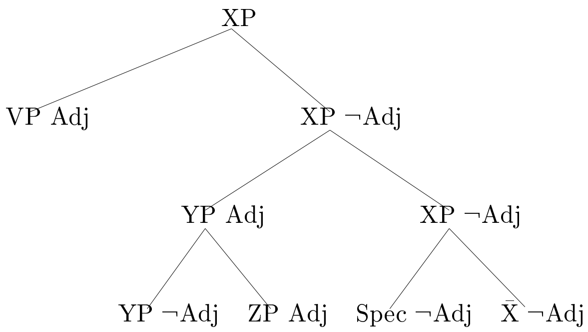
How can we use it for NLP

1. Naivly: Compilation of a grammar automaton
2. Compilation of modules
3. Verification of properties
4. Compilation for particular inputs
5. CP extension
6. MSO as an automaton specification language
7. *Query*-Sprachen: Neven and Schwentick (xxxx)
8. Feature Trees (Records): Niehren and Podelski (1992), Ayari et al. (1998)
9. Hardware Verification: Basin and Klarlund (1995)
10. TAGs: Rogers and Vijay-Shanker (1992,1994), Rogers (1994), Mönnich (1998), Michaelis, Mönnich and Morawietz (2000)
11. ...

Adjunction Structures



Who did Lena invite to her birthday party?



Categories

$$\begin{aligned}
\mathsf{F.Eq}(x,y) &\overset{\text{def}}{\iff} \\
&\quad (\mathsf{N}(x) \Leftrightarrow \mathsf{N}(y)) \land (\mathsf{V}(x) \Leftrightarrow \mathsf{V}(y)) \land \ldots \\
\mathsf{Comp}(X) &\overset{\text{def}}{\iff} \\
&\qquad \mathsf{Path}(X) \\
&\quad \land (\forall x,y)[(x \in X \land y \in X) \Rightarrow \mathsf{F.eq}(x,y)] \\
&\quad \land (\forall x,x')(\exists y)(\forall z)[(x \in X \land x' \in X \land x \lhd x') \Rightarrow \\
&\qquad\qquad (x' \notin \mathrm{Adj} \land x \lhd y \land y \not\approx x' \land y \in \mathrm{Adj} \land \\
&\qquad\qquad\quad (x \lhd z \Rightarrow (z \approx x' \lor z \approx y)))] \\
\mathrm{Cat}(X) &\overset{\text{def}}{\iff} \\
&\qquad \mathsf{Comp}(X) \\
&\quad \land (\forall Y)[(X \subseteq Y \land Y \not\subseteq X) \Rightarrow \neg\mathsf{Comp}(Y)]
\end{aligned}$$

$$\begin{aligned}
&(\forall x)[\mathrm{Adj}(x) \Rightarrow \\
&\qquad (\exists y,z,Y)[y \lhd x \land y \lhd z \land \\
&\qquad\qquad \mathsf{Cat}(Y) \land Y(y) \land Y(z)]]
\end{aligned}$$

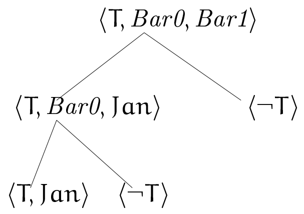
$$\begin{aligned}
\mathsf{D}(x,y) &\overset{\text{def}}{\iff} (\forall x')[\mathsf{Cat}(x,x') \Rightarrow x' \lhd^{+} y] \\
\mathsf{Excl}(x,y) &\overset{\text{def}}{\iff} (\forall x')[\mathsf{Cat}(x,x') \Rightarrow \neg(x' \lhd^{*} y)] \\
\mathsf{Incl}(x,y) &\overset{\text{def}}{\iff} \neg\mathsf{Excl}(x,y) \\
\mathsf{Prec}(x,y) &\overset{\text{def}}{\iff} \mathsf{Excl}(x,y) \land \mathsf{Excl}(y,x) \land x \prec y \\
\mathsf{ID}(x,y) &\overset{\text{def}}{\iff} \mathsf{D}(x,y) \\
&\qquad \land \neg(\exists z)[(\mathsf{Excl}(z,x) \land \mathsf{D}(z,y)) \\
&\qquad\qquad \lor (\mathrm{Adj}(z) \land \mathsf{Excl}(z,x) \land \mathsf{Incl}(z,y))]
\end{aligned}$$

$$\begin{aligned}
\mathsf{C\text{-}Com}(x,y) &\overset{\text{def}}{\iff} \\
&\qquad \neg\mathsf{D}(x,y) \land \neg\mathsf{D}(y,x) \\
&\quad \land \quad (\forall z)[\mathsf{D}(z,x) \Rightarrow \mathsf{D}(z,y)] \\
\mathsf{M\text{-}Com}(x,y) &\overset{\text{def}}{\iff} \\
&\qquad \neg\mathsf{D}(x,y) \land \neg\mathsf{D}(y,x) \\
&\quad \land \quad (\forall z)[(\mathit{Bar2}(z) \land \mathsf{D}(z,x)) \Rightarrow \mathsf{D}(z,y)] \\
\mathsf{AC\text{-}Com}(x,y) &\overset{\text{def}}{\iff} \\
&\qquad \mathsf{C\text{-}Com}(x,y) \\
&\quad \land \neg\mathsf{C\text{-}Com}(y,x) \\
&\quad \land \mathsf{Prec}(x,y)
\end{aligned}$$

X-Bar Theory

$$\begin{aligned}
&(\forall x)[\mathit{Bar0}(x) \lor \mathit{Bar1}(x) \lor \mathit{Bar2}(x) \lor \mathsf{Lexicon}(x)] \\
&(\forall x)[\mathit{Bar2}(x) \Rightarrow \\
&\qquad \mathrm{Trace}(x) \land (\forall y)[\neg\mathsf{D}(x,y)] \\
&\quad \lor (\exists y)[\mathsf{HdXP}(y,x) \land \mathit{Bar1}(y) \land \mathsf{Proj}(x,y)] \land \\
&\qquad (\forall y)[(\mathsf{ID}(x,y) \land \neg\mathsf{HdXP}(y,x)) \Rightarrow \mathit{Bar2}(y)]] \\
&(\forall x)[\mathit{Bar1}(x) \Rightarrow \\
&\qquad (\exists y)[\mathsf{Hd}\overline{\mathsf{X}}(y,x) \land \mathit{Bar0}(y) \land \mathsf{Proj}(x,y)] \\
&\quad \land (\forall y)[(\mathsf{ID}(x,y) \land \neg\mathsf{Hd}\overline{\mathsf{X}}(y,x)) \Rightarrow \mathit{Bar2}(y)]] \\
&(\forall x)[\mathit{Bar0}(x) \Rightarrow \\
&\qquad (\exists! y)[\mathsf{ID}(x,y)] \\
&\quad \land (\forall y)[\mathsf{ID}(x,y) \Rightarrow (\mathsf{Lexicon}(y) \land \mathsf{Proj}(x,y) \\
&\qquad\qquad\qquad (\mathrm{Base}(x) \Leftrightarrow \mathrm{Base}(y)))]]
\end{aligned}$$

$$\mathsf{XBar}(\mathsf{T}) \Rightarrow \mathsf{Disjoint}(\mathit{Bar0}, \mathit{Bar1})$$



$$(\neg\exists x, y \in \mathsf{T})[\mathsf{Lexicon}(x) \wedge \mathsf{ID}(x, y)]$$

Approximation of P&P grammars

Theorem (Rogers 1994): Free Indexation is undecidable.

Index Estimation

$$n = \sum_{i=0}^{m} f(w_i) + g(w_i) + 1$$

input string: $w = w_1 \ldots w_m$,

f computes the functional elements in the theory and

g computes the number of argument positions.

Defining Indices

$$\begin{aligned}
\mathsf{Co_Idx}(x, y, \vec{I}_n) &\overset{\text{def}}{\Longleftrightarrow} \\
&\quad (x \in I_1 \Leftrightarrow y \in I_1) \\
&\wedge (x \in I_2 \Leftrightarrow y \in I_2) \\
&\wedge \ldots \\
&\wedge (x \in I_n \Leftrightarrow y \in I_n) \\
\mathsf{C\text{-}Com}(x, y) &\overset{\text{def}}{\Longleftrightarrow} \\
&\quad (\forall z)[z \lhd^{+} x \Rightarrow z \lhd^{+} y] \\
&\wedge \neg(x \lhd^{*} y) \\
\mathsf{TBind}(\mathsf{P}, \mathit{Trace}, \vec{I}_n) &\overset{\text{def}}{\Longleftrightarrow} \\
&(\forall x \in \mathsf{P})[x \in \mathit{Trace} \Rightarrow \\
&\quad (\exists y \in \mathsf{P})[\mathsf{C\text{-}Com}(y, x) \wedge \\
&\qquad \mathsf{Co_Idx}(x, y, \vec{I}_n)]]
\end{aligned}$$

Statistics for various predicates

Predicate/N	**Total Time**	$\lvert\mathbf{A}\rvert$	$\lvert\alpha\rvert$	$\lvert\mathbf{BDD}\rvert$
Connected	00:00:00.090	4	25	7
Path	00:00:00.130	4	21	9
C-Com	00:00:00.170	5	40	19
TBind/2	00:00:00.230	12	832	52
TBind/4	00:00:01.610	80	76288	422
TBind/8	00:55:15.000	2304	–	13881
TBind/16	core dump after approx. 10 min			
TBind′/4	00:00:07.480	87	145963	759
TBind′/6	00:04:58.830	457	–	4901
TBind′/7	core dump after approx. 30 min			
XBar	00:04:51.080	66	18133	982

MSO Logic and CP

- Tree automata function as constraint store.
- The compiler serves as constraint solver.
- Solutions to a search branch are tree automata.
- WS2S is a decidable constraint language closed under renaming and intersection (in the sense of Höhfeld and Smolka, 1988).

⇒ MSO Definite Clause Programs

MSO Logic as an Automaton Specification Language

- comparable to FSA-Tools (van Noord) or the Xerox calculus (Karttunen)
- context-free (tree automata) or regular (FSA)
- but what about transductions?

Outlook

- Which formulas are efficiently compilable?
- Can we compile an entire grammar?
- Higher-level languages on top of WS2S
- Definability of inductive relations
- Adequate Generative Capacity
- Finite Model Theory