How to Church a Joshi

Uwe Mönnich
um@sfs.uni-tuebingen.de
http://tcl.sfs.uni-tuebingen.de/~um

Theoretische Computerlinguistik
Seminar für Sprachwissenschaft
Universität Tübingen

June 7th, 2008

Outline

- Mild Context-Sensitivity
- Context-Sensitive Tree Transductions
- Path Analysis of TAG

Mild Context-Sensitivity

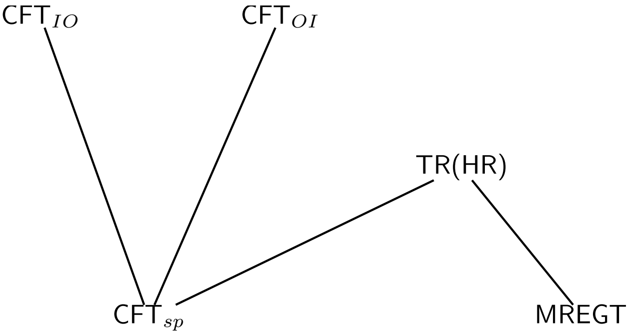
Joshi(1985)

- Inclusion of context-free languages
- Parsing problem solvable in polynomial time
- Constant growth
- Upper bound on cross-serial dependencies

Incompatibility of Current Formalizations

1. $MREGT - CFT_{OI} \neq \emptyset$
   $L = \{a(t, t)\}$
   t an arbitrary tree over $b^{(2)}, e^{(1)}$.
2. $MREGT - CFT_{IO} \neq \emptyset$
   $L = \{a(t, t')\}$
3. $CFT_{sp} - MREGT \neq \emptyset$
   $L = \{a^n(b^n(e))\}$

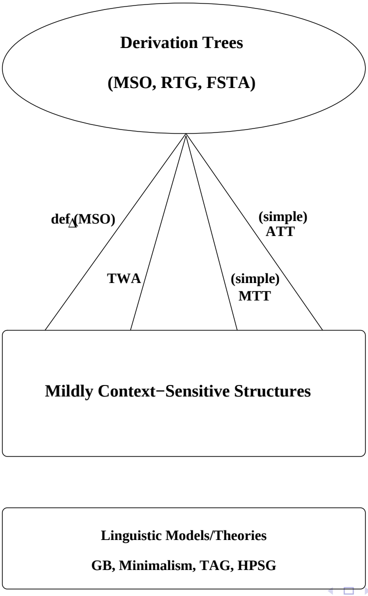
Some Classes of Tree Languages



Basic idea: Feferman

> *"Formal consistency statemens are just one kind of statement of trust in the correctness of a system; more general such statements are called reflection principles, because they result from reflecting on what has led one to accept that system in the first place." (S. Feferman, In the Light of Logics, pp. 17f)*

Linear Bounds on Second-Order Substitution



Weakly equivalent mildly context-sensitive languages

Pregroups & Tupling	languages defined by pregroups with tupling
MCFL	languages generated by multiple context-free grammars
MCTAL	languages generated by multi-component tree adjoining grammars
LCFRL	languages generated by linear context-free rewriting systems
LUSCL	languages generated by local unordered scattered context grammars
$STR(HR)$	languages generated by string generating hyperedge replacement grammars
$OUT(DTWT)$	output languages of deterministic tree-walking tree-to-string transducers
$yDT_{fc}(REGT)$	yields of images of regular tree languages under deterministic finite-copying top-down tree transduction

Models of (tree) transformations

Operational tree transducers

$$\tau \subseteq T_\Sigma \times T_\Omega$$

Tree transducers transform trees over Σ, i.e., elements of T_Σ into trees over Ω, i.e., into elements of T_Ω.
Tree transducers are devices in which the translation of an input tree $t \in T_\Sigma$ may be defined in terms of its subtrees and, in addition, in terms of its context.
Different ways to handle context information lead to different types of tree transducers:

Top-down tree transducer

Top-down tree transducers are finite state devices that transform in a strict recursive top-down manner an input tree into an output tree.

Definition (Top-Down Tree Transducer)

A top-down tree transducer is a tuple $\mathcal{T} = (Q, \Sigma, \Omega, q_0, R)$ where Σ and Ω are ranked alphabets, called the input and output alphabet, respectively, Q is a unary alphabet of states, q_0 is the initial state and R is a finite set of rules of the following form:

$$q(\sigma(x_1, \ldots, x_m)) \rightarrow \xi$$

where $q \in Q$, $\sigma \in \Sigma_m$ and $\xi \in T_\Omega(\langle Q, X_m \rangle)$, i.e. a tree over $\Omega \cup \langle Q, X_m \rangle$ where each pair of a state q and a variable x_i is considered as an element of rank zero.

Top-down tree transducer

Definition

The relation

$$\tau_{\mathcal{T}} = \{(s,t) | s \in T_{\Sigma}, t \in T_{\Omega}, (q_0, s) \overset{*}{\Rightarrow}_{\mathcal{T}} t\}$$

is the tree transduction realized by the top-down tree transducer $\mathcal{T}$.

Example (Shieber)

Let $T = (Q, \Sigma, \Omega, q_0, R)$ be such that $Q = \{q_0, q_1, q_2\}$, $\Sigma = \Omega = \{a^{(0)}, b^{(0)}, f^{(2)}\}$ and R is the set of the following rules:

$$\begin{array}{rcl} q_0(a) & \to & a \\ q_0(b) & \to & b \\ q_0(f(x,y)) & \to & f(f(q_0(x), q_1(y)), q_2(y)) \\ q_1(f(x,y)) & \to & q_0(x) \\ q_2(f(x,y)) & \to & q_0(y) \end{array}$$

This transducer changes trees over Σ of shape $f(t_1, f(t_2, t_3))$ into trees in left-associative form, i. e., into $f(f(t_1, t_2), t_3)$.

Macro Tree Transducer

Macro tree transducers (MTT) are much more powerful devices. They are a model of tree transformation that transduces, again in a recursive top-down fashion, an input tree into an output tree, handling context information in an implicit way. The elements of context information do not have explicit names, but are passed along as parameters of the states in this kind of translation device.

Macro Tree Transducer

Definition

A macro tree transducer is a tuple $M = (Q, \Sigma, \Omega, q_0, R)$, where Q is a ranked alphabet of states, Σ and Ω are ranked alphabets of input and output symbols, respectively, $q_0 \in Q_0$ is the initial state and R is a set of rules of the following form

$$(q, \sigma(x_1, \ldots, x_m))(y_1, \ldots, y_n) \rightarrow \xi$$

where $q \in Q_n$, $\sigma \in \Sigma_m$ and $\xi \in T_{(Q,X_m) \cup \Omega}(Y_n)$ with $\mathrm{rank}(q, x_i) = \mathrm{rank}(q)$.

The subtrees of the input are represented by input variables $X_m = \{x_1, \ldots, x_m\}$ and the context information is handled by parameters $Y_n = \{y_1, \ldots, y_n\}$.

Macro Tree Transducer

Remark: If every state in Q has rank zero, then M is a top-down transducer. Macro tree transducers can therefore be regarded as a context-sensitive extension of top-down transducers.

Definition

The relation

$$\tau = \{(s, t) | s \in T_\Sigma, t \in T_\Omega, (q_0, s) \stackrel{*}{\Rightarrow}_M t\}$$

is the tree transduction realized by the macro tree transducer M.

Let Σ be a ranked alphabet and let Ω be another ranked alphabet such that $\Omega = \{\sigma' | \sigma \in \Sigma\} \cup \{\text{nil}\}$ where every $\sigma' \in \Omega$ is of rank two and *nil* of rank zero.
With every finite sequence $s = (t_1, \ldots, t_n)$ $(t_i \in T_\Sigma)$ we can associate a corresponding binary tree $bin(s)$ over Ω:

$n \geq 1$: $bin(t_1, \ldots, t_n) = \sigma'(bin(s_1, \ldots, s_m), bin(t_2, \ldots, t_n)$ where $t_1 = \sigma(s_1, \ldots, s_m)$

$n = 0$: $bin(\quad) = \text{nil}$ where $(\quad)$ indicates the empty sequence.

Example (Engelfriet, Vogler)

The following macro tree transducer $M_{bin} = (\{q_0, p^{(2)}\}, \Sigma, \Omega, q_0, R)$ with R containing the rules:

$$q_0(\sigma(x_1, \ldots, x_m)) \to \sigma'(p(x_1, p(x_2, \ldots, p(x_m, \text{nil}) \ldots)), \text{nil})$$
$$p(\sigma(x_1, \ldots, x_m), y) \to \sigma'(p(x_1, p(x_2, \ldots, p(x_m, \text{nil}) \ldots)), y)$$

realizes the transduction:

$$\tau_M = \{(s, bin(s)) | s \in \Sigma, bin(s) \in \Omega\}$$

Example (Engelfriet, Vogler)

Remark: The transducer M_{bin} is linear and non-deleting in both the input and the parameter variables. According to a result due to Engelfriet and Maneth the output languages of linear and non-deleting macro tree transducers applied to regular tree languages are exactly the tree languages generated by linear and non-deleting context-free tree grammars. These tree languages may be viewed as straightforward extensions of the family of tree adjoining languages.

Context-Free Tree Grammars

Definition

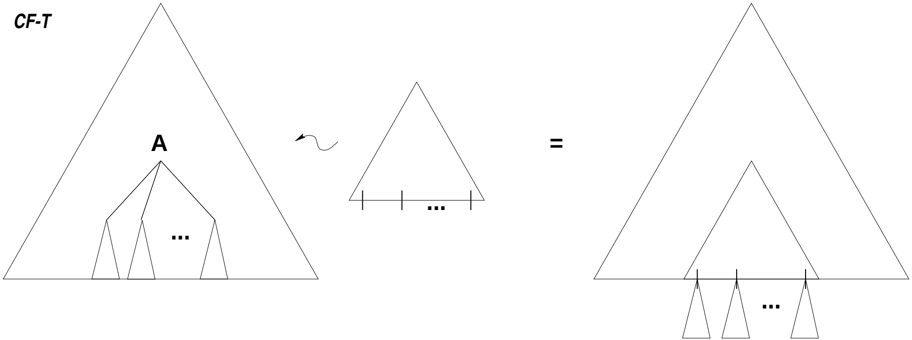
Disregarding the input of a macro tree transducer one obtains a context-free tree (*CFT*) grammar. A *CFT* grammar is a tuple $G = (\mathcal{F}, \Omega, S, P)$ where $\mathcal{F}$ and Ω are ranked alphabets of nonterminals and terminals, respectively, $S \in \mathcal{F}_0$ is the start symbol and P is a finite set of productions of the form

$$F(y_1, \ldots, y_m) \to \xi$$

where $F \in \mathcal{F}$ and ξ is a tree over $\mathcal{F}$, Ω and Y_m.

Context-Free Tree Grammars

Intuitively, the application of a production of the form $F(y_1, \ldots, y_m) \to \xi$ rewrites a (sub-)tree rooted in F as the tree ξ' in which the parameters y_i occurring in ξ are replaced by F's daughters.



CFT_{sp} := Tree families generated by context-free tree grammars which are simple in the parameters, i.e., linear and non-deleting.

Example: Consider the following simple CFT grammar $G = (F, \Omega, S, P)$ where $F = \{S', S\}$, $\Omega = \Omega_{bin}$ and P contains the rules:

$$S' \rightarrow \sigma'(S(S \ldots S(nil) \ldots), nil)$$

$$S(y) \rightarrow \sigma'(S(S \ldots S(nil) \ldots), y)$$

The grammar G is a notational variant of M_{bin}. In the way of illustration, take the case of $\Sigma = \{\sigma^{(2)}, \gamma^{(1)}, \alpha^{(0)}\}$. M_{bin} would transform $\sigma(\gamma(\alpha), \alpha)$ into $\sigma'(\gamma'(\alpha'(nil, nil), \alpha'(nil, nil)), nil)$.
G does not generate the regular language T_{Ω}. E.g., $\alpha'(\alpha'(nil, nil), nil)$ is not a sentence generated by G.

Tree Adjoining Grammars

Theorem (Fujiyoshi and Kasai, Mönnich)

The class of string languages generated by monadic context-free tree grammars coincides with the class of string languages generated by TAGs

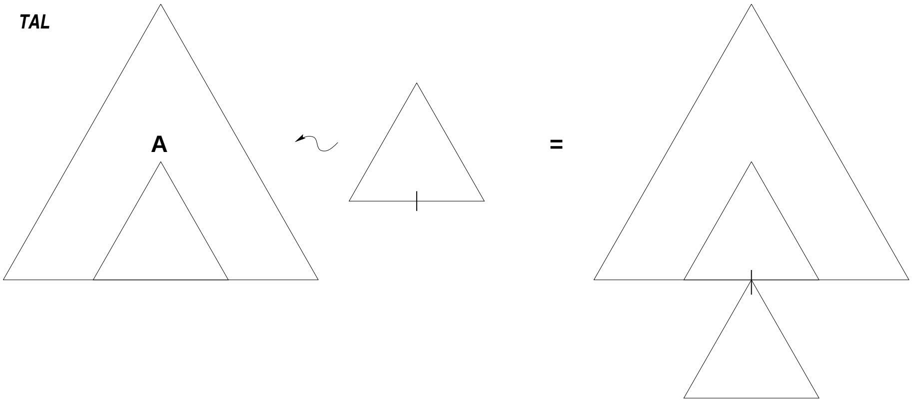
Theorem (Shieber)

The class of string languages generated by TAGs coincides with the yields of output tree languages of simple monadic macro tree transducers.

Theorem (Kepser and Rogers)

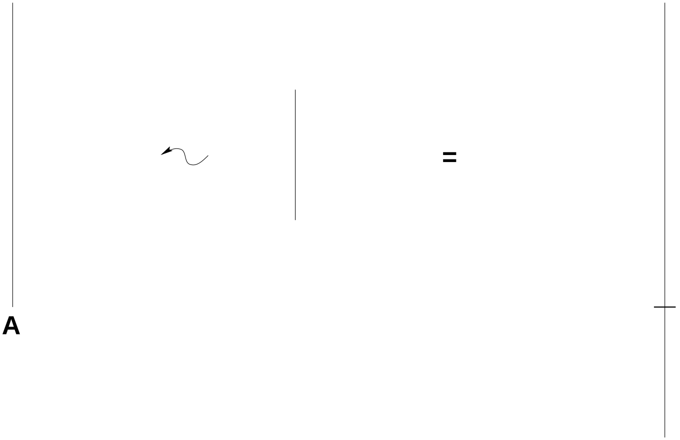
the class of languages definable by non-strict TAGs is exactly the class of tree languages definable by monadic linear context-free tree grammars.

Monadic Context-Free Tree Grammars

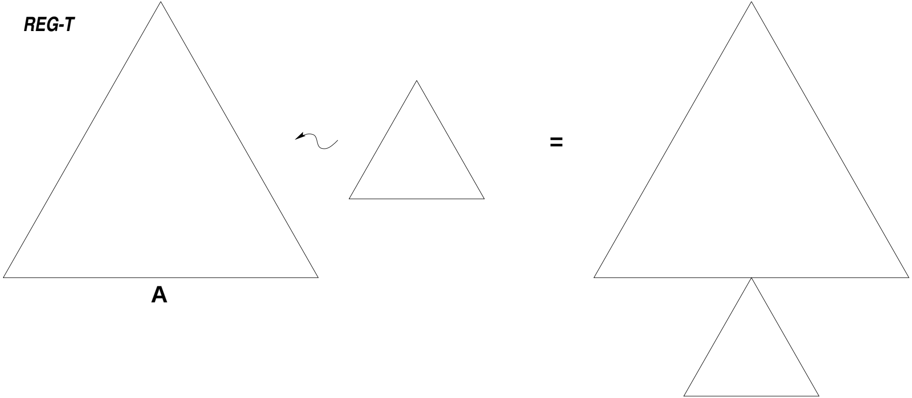


Regular Word Grammars

REG



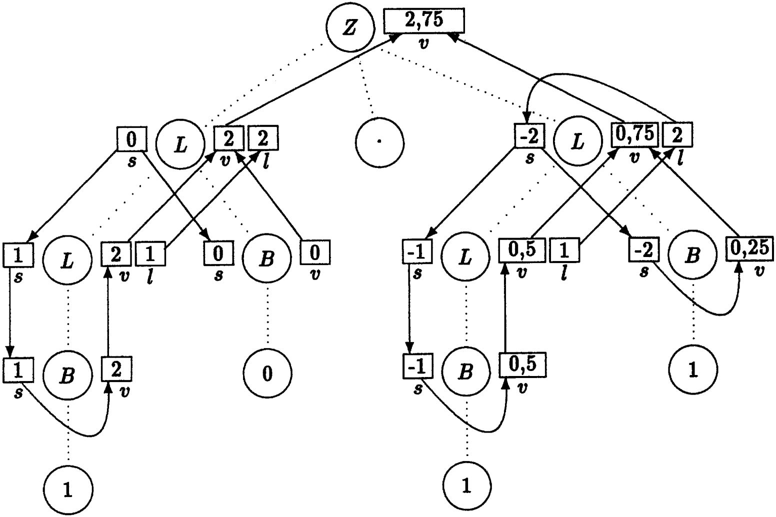
Regular Tree Grammars



Attributed Tree Transducer

Attributed tree transducers are a variant of attribute grammars in which all attribute values are trees. Like macro tree transducers, attributed tree transducers can handle context information. However, in this model of tree transduction context information is treated in an explicit way. Besides meaning names which transmit information in a top-down manner, attributed tree transducers contain explicit context names which allow information to be passed down from a node to its descendants. Consequently, arbitrary tree walks can be realized by attributed tree transducers.

Knuth (Kühnemann/Vogler)



ATT

Definition

An attributed tree transducer (ATT) is a tuple $A = (Syn, Inh, \Sigma, \Omega, \alpha_m, R)$, where *Syn* and *Inh* are disjoint alphabets of synthesized and inherited attributes, respectively, Σ and Ω are ranked alphabets of input and output symbols, respectively, α_m is a synthesized attribute, and R is a finite set of rules of the following form:
For every $\sigma \in \Sigma_m$, for every $(\gamma, \rho) \in ins_\sigma$ (the set of inside attributes of σ), there is exactly one rule in R_σ:

$$(\gamma, \rho) \rightarrow \xi$$

where $\xi \in T_{\Omega \cup out_\sigma}$ and out_σ is the set of outside attributes of σ.

ATT

Definition

For every $\sigma \in \Sigma_m$, the set of inside attributes is the set

$$ins_\sigma = \{(\alpha, \pi) | \alpha \in Syn\} \cup \{(\beta, \pi i) | \beta \in Inh, i \leq m\}$$

and the set of outside attributes is the set

$$out_\sigma = \{(\beta, \pi) | \beta \in Inh\} \cup \{(\alpha, \pi i) | \alpha \in Syn, i \leq m\},$$

where π and ρ are path variables ranging over node occurrences in the input tree.

ATT

The dependencies between attribute occurrences in an input tree s can be represented with the help of R_σ. An instance of an attribute occurrence (α, π) depends on another occurrence (α', π') if σ labels node u in s, R_σ contains the rule $(\alpha', \pi') \to \xi$ and (α, π) labels one of the leaves in ξ. An attributed tree transducer is noncircular if the paths of attribute dependencies are noncircular. It is well known that noncircular ATTs have unique decorations dec., functions which assign each attribute occurrence a tree over $\Omega \cup out_\sigma$ in accordance with the productions R_σ.

Definition

The transduction realized by a noncircular attributed tree transducer A is the function

$$\tau_A = \{(s,t) | s \in T_\sigma, t \in T_\Omega, t = dec_s(\alpha_m, \epsilon)\}$$

Remark: ATTs which have synthesized attributes only are very close to top-down tree tranducers. Their rules

$$(\gamma, \rho) \to \xi \text{ in } R_\sigma$$

correspond to rules

$$\alpha(\sigma(x_1, \ldots, x_m)) \to \xi'$$

where $\sigma \in \Sigma_m$ and ξ' is obtained from ξ by substituting every $(\alpha, \pi i)$ by $\alpha(x_i)$.

Monadic Context-Free Tree Grammar (Normal Form)

Definition (Monadic Context-Free Tree Grammar)

A monadic context-free tree grammar (MCFTG) is a 5-tuple $\Gamma = \langle F_0 \cup F_1, \Sigma, S, P \rangle$, i.e., a CFTG, where all the rules in P are of one of the following "unary" types ($A, B, C, B_i \in F_1 \cup F_0$, $1 \leq i \leq n$, $a \in \Sigma$):

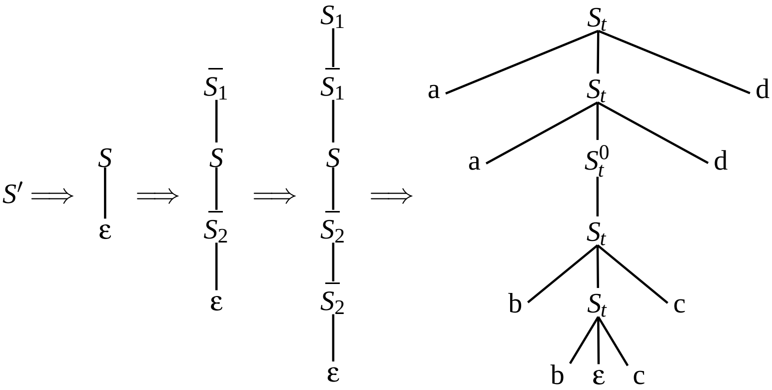
$$\begin{aligned} A &\longrightarrow a \\ A &\longrightarrow B(C) \\ A(x) &\longrightarrow a(B_1, \ldots, B_{i-1}, x_i, B_{i+1}, \ldots, B_n) \\ A(x) &\longrightarrow B_1(B_2(\ldots B_n(x) \ldots)) \end{aligned}$$

Classical Example of TAG as MCFTG

Consider the MCFTG
$\Gamma_{\mathrm{TAG}} = \langle \{S, S', \overline{S}_1, \overline{S}_2, \overline{a}, \overline{b}, \overline{c}, \overline{d}\}, \{a, b, c, d, \varepsilon, S_t, S_t^0\}, S', P\rangle$
corresponding to the classical TAG example with P given as follows

$$\begin{aligned} S' &\longrightarrow S(\varepsilon) \\ S(x) &\longrightarrow \overline{S}_1(S(\overline{S}_2(x))) \\ S(x) &\longrightarrow S_t^0(x) \\ \overline{S}_1(x) &\longrightarrow S_t(\overline{a}, x, \overline{d}) \\ \overline{S}_2(x) &\longrightarrow S_t(\overline{b}, x, \overline{c}) \end{aligned} \qquad \begin{aligned} \overline{a} &\longrightarrow a \\ \overline{b} &\longrightarrow b \\ \overline{c} &\longrightarrow c \\ \overline{d} &\longrightarrow d \end{aligned}$$

An example derivation of the MCFTG Γ_{TAG}



The construction of a macro tree transdcuer that outputs the same tree language that is generated by a given monadic context-free tree grammar proceeds as follows: as input one takes the derivation trees and as states a finite set of natural numbers, where the arity of the input symbols correspond to the number of nonterminals in the right-hand side of the corresponding production from the given grammar and the arity of the states is provided by the number of free (parameter) variables in the right-hand sides of the productions associated with a certain nonterminal, which is either zero or one in the simple case of monadic context-free tree grammars.

Example MTT

According to these stipulations we arrive at the following macro tree transducer $M = (Q, \Sigma, \Omega, q_0, R)$, where $Q = \{0, 1\}, \Sigma = \{p_0, \ldots, p_8\}$ (each p_i corresponding to a production in the previous example), $q_0 = \{0\}$ and R consisting of the rules:

$$
\begin{aligned}
\langle 0, p_0(x)\rangle &\longrightarrow \langle 1, x\rangle(\varepsilon) \\
\langle 1, p_1(x_1, x_2, x_3)\rangle(y) &\longrightarrow \langle 1, x_1\rangle(\langle 1, x_2\rangle(\langle 1, x_3\rangle(y))) \\
\langle 1, p_2\rangle(y) &\longrightarrow S_t^0(y) \\
\langle 1, p_3(x_1, x_2)\rangle(y) &\longrightarrow S_t(\langle 0, x_1\rangle, y, \langle 0, x_2\rangle) \\
\langle 1, p_4(x_1, x_2)\rangle(y) &\longrightarrow S_t(\langle 0, x_1\rangle, y, \langle 0, x_2\rangle) \\
\langle 0, p_5\rangle &\longrightarrow a \\
\langle 0, p_6\rangle &\longrightarrow b \\
\langle 0, p_7\rangle &\longrightarrow c \\
\langle 0, p_8\rangle &\longrightarrow d
\end{aligned}
$$

From MTT to ATT

For a given simple macro tree transducer $M = (Q, \Sigma, \Omega, q_0, R)$ an attributed tree transducer $A = (Syn, Inh, \Sigma, \Omega, q_0, R')$ that realizes the same translation is defined in the following way.

- $Syn = Q$
- $Inh = \{y_j | 1 \leq j \leq m,$ with m the maximal arity of a state $q \in Q\}$
- If $\langle q_1, \sigma(x_1, \ldots x_n)\rangle(y_1, \ldots, y_m) \rightarrow \xi$ is an element of R then R'_σ is specified for both the synthesized and inherited attributes by structural induction on the right-hand side ξ:

$$(q, \pi) \rightarrow \vartheta(\xi),$$

  where ϑ substitutes $\langle q, x_i\rangle$ in ξ by (q, π_i) and y_j by (y_j, π)

$$(y_j, \pi_i) \rightarrow \vartheta(\xi_j),$$

  where ξ_j occurs in the jth parameter position of some $\langle q, x_i\rangle$ in ξ.

Example ATT

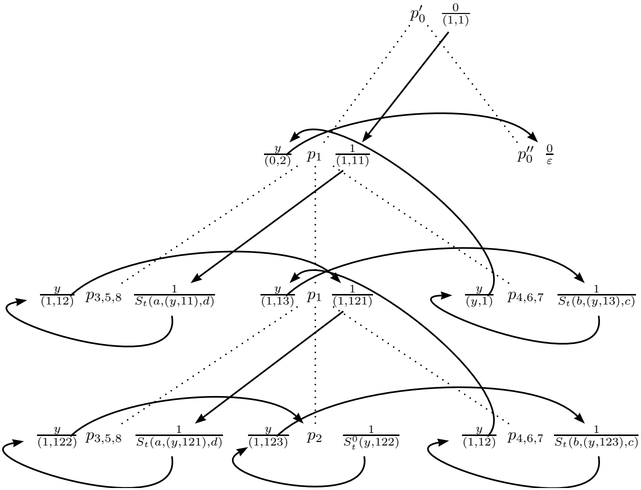
Applying the construction just outlined to the macro tree transducer of the last example we obtain the following attributed tree transducer $A = (Syn, Inh, \Sigma, \Omega, q_0, R')$:

- $Syn = \{0, 1\}$
- $Inh = \{y\}$
- $\Sigma = \{p_0, \ldots, p_8\}$
- $\Omega = \{a, b, c, d, \varepsilon, S_t, S_t^0\}$
- $q_0 = 0$
- $R' = \bigcup_{p_i} R'_{p_i}$

Example ATT

$$
\begin{aligned}
R'_{p_0} &= \{(0,\pi) \to (1,\pi 1) \\
&\qquad (y,\pi 1) \to \varepsilon\} \\
R'_{p_1} &= \{(1,\pi) \to (1,\pi 1) \\
&\qquad (y,\pi 1) \to (1,\pi 2) \\
&\qquad (y,\pi 2) \to (1,\pi 3) \\
&\qquad (y,\pi 3) \to (y,\pi)\} \\
R'_{p_2} &= \{(1,\pi) \to S_t^0(y,\pi)\} \\
R'_{p_3} = R'_{p_4} &= \{(1,\pi) \to S_t((0,\pi 1),(y,\pi),(0,\pi 2))\} \\
R'_{p_5} &= \{(0,\pi) \to a\} \\
R'_{p_6} &= \{(0,\pi) \to b\} \\
R'_{p_7} &= \{(0,\pi) \to c\} \\
R'_{p_8} &= \{(0,\pi) \to d\}
\end{aligned}
$$

Attributed derivation tree with semantic dependency relations



Final Result

Lemma

For any monadic context-free tree grammar G, there is a monadic context-free tree grammar G' in Greibach normal form such that

$$L(G') = L(G)$$

We are now in a position to state the main result of this talk:

Theorem

For every monadic context-free tree grammar G, there is an equivalent single use restricted attributed tree transducer with one synthesized attribute only which is direction preserving/reversing (i.e., dependency relations correspond to direction preserving or to direction reversing paths in the input tree).

Envoi

- TAG $\simeq$ Direction preserting/direction reversing $ATT_{SUR,\ I\text{-}S}$
- MG $\simeq$ Direction preserving ATT_{SUR}
- HPSG $\simeq$?

Further Reading I

Joost Engelfriet and Sebastian Maneth. Tree Languages Generated by Context-Free Graph Grammars. In Hartmut Ehrig et al., editor, *Graph Transformation*, number 1764 in LNCS, pages 15–29. Springer, 2000.

Joost Engelfriet and Sebastian Maneth. Macro Tree Transducers, Attribute Grammars, and MSO Definable Tree Translations. *Information and Computation*, 154:34–91, 1999.

Michael J. Fischer. Grammars with Macro-Like Productions. In *Proc. 9th IEEE Symp. on Switching and Automata Theory*, pages 131–142, 1968.

Akio Fujiyoshi and Takumi Kasai. Spinal-formed context-free tree grammars. *Theory of Computing Systems*, 33(1):59–83, 2000.

Zoltán Fülöp and Heiko Vogler. *Syntax-Directed Semantics - Formal Models Based on Tree Transducers*. Springer, New York and Berlin, 1998.

Further Reading II

Aravind Joshi. Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions. In David Dowty, Lauri Karttunen, and Arnold Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, 1985.

Stephan Kepser and Uwe Mönnich. (Un-)Decidability results for head-driven phrase structure grammar. In Giuseppe Scollo and Anton Nijholt, editors, *Proceedings Algebraic Methods in Language Processing (AMiLP-3)*, pages 141–152, 2003.

Stephan Kepser and James Rogers. The equivalence of tree adjoining grammars and monadic linear context-free tree grammars. In Marcus Kracht, Gerald Pen, and Ed Stabler, editors, *Mathematics of Language 10*, 2007.

Uwe Mönnich. Minimalist syntax, multiple regular tree grammars and direction preserving transductions. In *Model Theoretic Syntax @ 10*, 2007.

Further Reading III

Uwe Mönnich. Adjunction as substitution. In G-J. M. Kruijff, G. Morill, and R. Oehrle, editors, *Formal Grammar '97*, pages 169–178, 1997.

Frank Morawietz and Uwe Mönnich. A model-theoretic description of tree adjoining grammars. In Geert-Jan Kruiff, Larry Moss, and Richard Oehrle, editors, *Proceedings FG-MOL 2001*, volume 53 of *ENTCS*. Kluwer, 2001.

Jean-Claude Raoult. Rational Tree Relations. *Bull. Belg. Math. Soc.*, 4: 149–176, 1997.

William C. Rounds. Mappings and grammars on trees. *Mathematical Systems Theory*, 4(3):257–287, 1970.

Stuart M. Shieber. Synchronous grammars as tree transducers. In *TAG+7*, 2004.