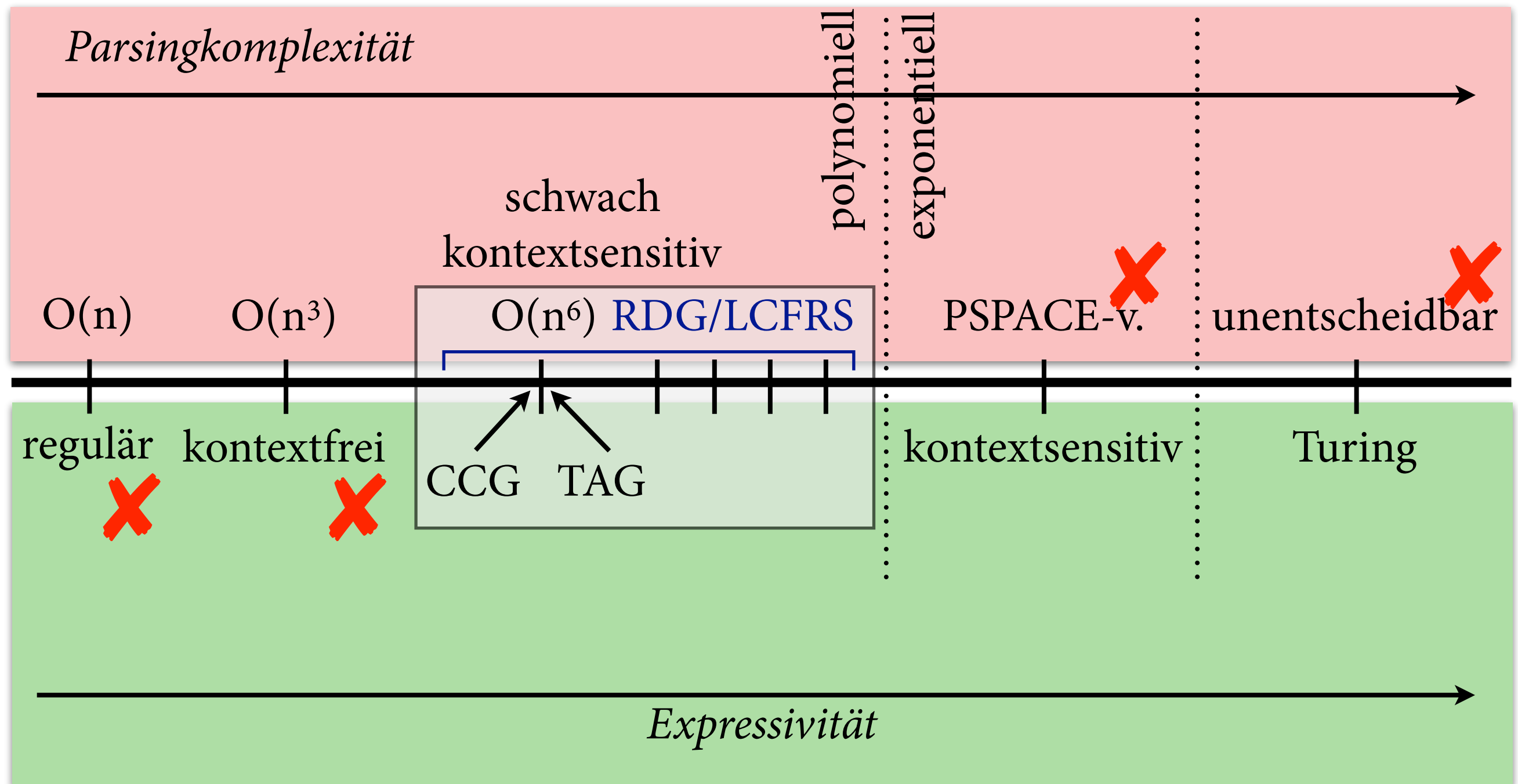


Interpretierte Reguläre Baumgrammatiken

Vorlesung “Grammatikformalismen”
Alexander Koller

11. Juni 2019

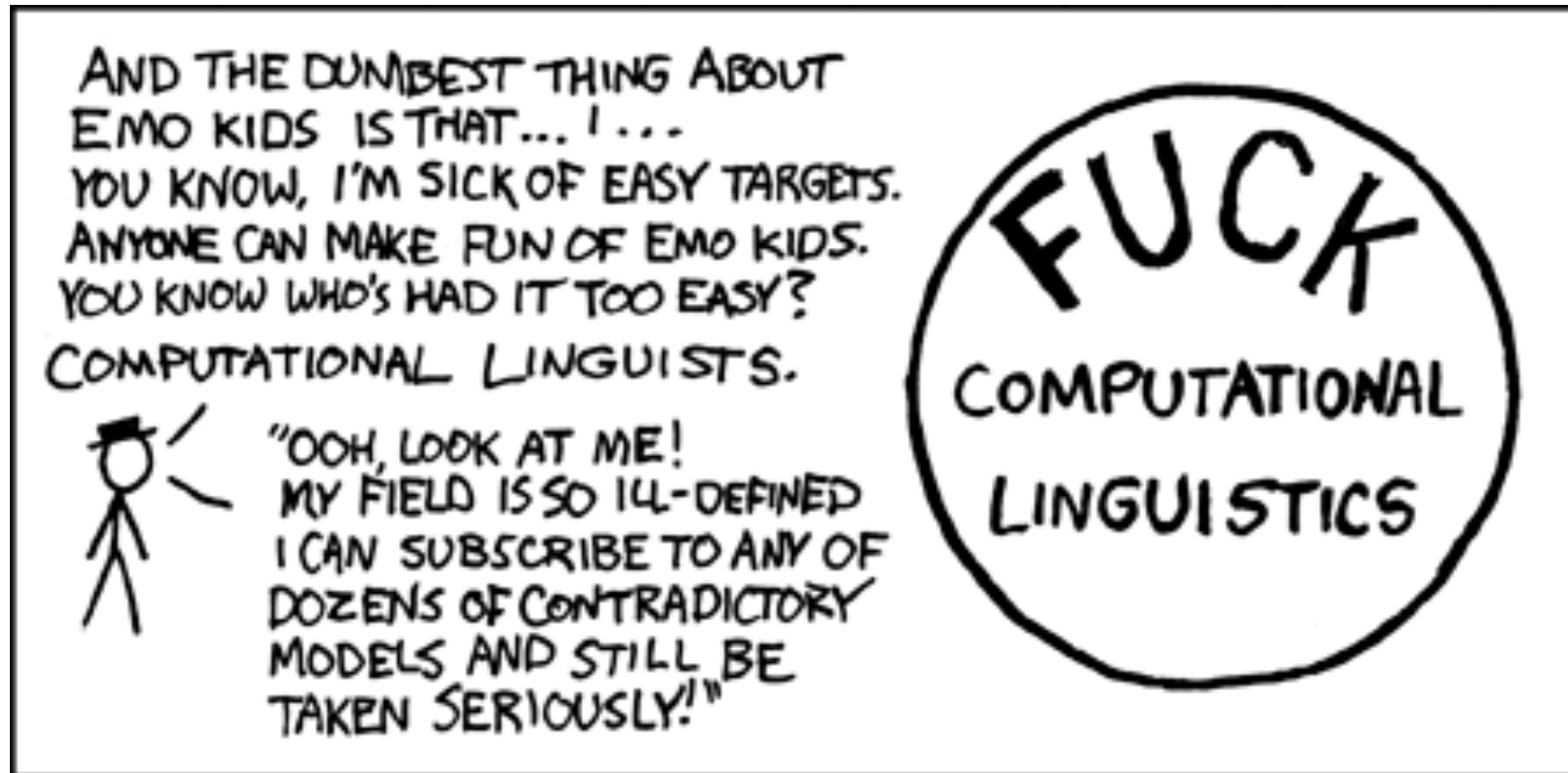
Grammatikformalismen



Das Problem

(xkcd #114)

Das Problem



(xkcd #114)

Bäume

- Statt Alphabeten haben wir jetzt *Signaturen*:
 - ▶ endliche Menge von Zeichen, die man als Knotenlabels in Bäumen verwenden kann
 - ▶ jedes Zeichen f hat eine *Arität/Stelligkeit* n
- *Baum* über einer Signatur Σ :
 - ▶ jeder Knoten hat ein Label $f \in \Sigma$
 - ▶ wenn Label von Knoten u Arität n hat, dann hat u genau n Kinder im Baum
- $T_\Sigma =$ alle Bäume über Σ

Reguläre Baumgrammatiken

- Reguläre Baumgrammatik (RTG) ist ein Tupel $G = (\Sigma, N, S, P)$, wobei
 - ▶ Σ eine Signatur (= Terminalsymbole)
 - ▶ N eine endliche Menge von Nichtterminalsymbolen
 - ▶ $S \in N$ das Startsymbol
 - ▶ P eine Menge von Produktionsregeln von der Form $A \rightarrow f(A_1, \dots, A_n)$, wobei $f \in \Sigma$ und $A, A_1, \dots, A_n \in N$.
- Eine RTG G definiert eine *Baumsprache* $L(G) \subseteq T_\Sigma$.

Ableitungen von RTGs

- Ableitungsprozess:
 - ▶ mit Startsymbol anfangen
 - ▶ in jedem Schritt ein Nichtterminalsymbol durch Baum auf der rechten Seite einer Regel ersetzen
 - ▶ wenn der Baum nur noch Terminalsymbole enthält, kommt er in die Sprache.

$S \rightarrow f(A,S)$

$S \rightarrow c$

$A \rightarrow a$

$A \rightarrow b$

RTG G

Ableitungen von RTGs

- Ableitungsprozess:
 - ▶ mit Startsymbol anfangen
 - ▶ in jedem Schritt ein Nichtterminalsymbol durch Baum auf der rechten Seite einer Regel ersetzen
 - ▶ wenn der Baum nur noch Terminalsymbole enthält, kommt er in die Sprache.

$S \rightarrow f(A, S)$

$S \rightarrow c$

$A \rightarrow a$

$A \rightarrow b$

$\bullet S \longrightarrow \bullet c$

RTG G

Ableitungen von RTGs

- Ableitungsprozess:
 - ▶ mit Startsymbol anfangen
 - ▶ in jedem Schritt ein Nichtterminalsymbol durch Baum auf der rechten Seite einer Regel ersetzen
 - ▶ wenn der Baum nur noch Terminalsymbole enthält, kommt er in die Sprache.

$S \rightarrow f(A, S)$

$S \rightarrow c$

$A \rightarrow a$

$A \rightarrow b$

● S



● c

RTG G

Ableitungen von RTGs

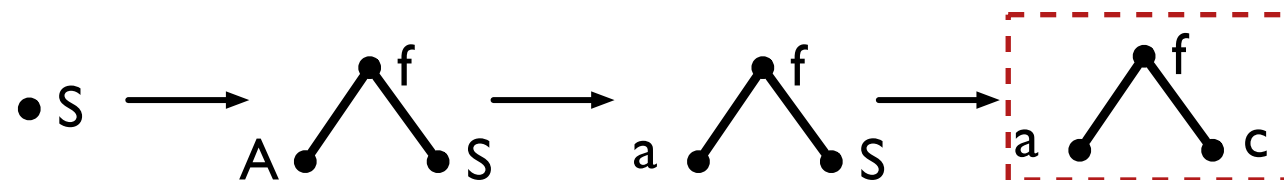
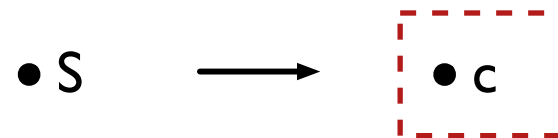
- Ableitungsprozess:
 - ▶ mit Startsymbol anfangen
 - ▶ in jedem Schritt ein Nichtterminalsymbol durch Baum auf der rechten Seite einer Regel ersetzen
 - ▶ wenn der Baum nur noch Terminalsymbole enthält, kommt er in die Sprache.

$S \rightarrow f(A, S)$

$S \rightarrow c$

$A \rightarrow a$

$A \rightarrow b$



RTG G

Ableitungen von RTGs

- Ableitungsprozess:
 - ▶ mit Startsymbol anfangen
 - ▶ in jedem Schritt ein Nichtterminalsymbol durch Baum auf der rechten Seite einer Regel ersetzen
 - ▶ wenn der Baum nur noch Terminalsymbole enthält, kommt er in die Sprache.

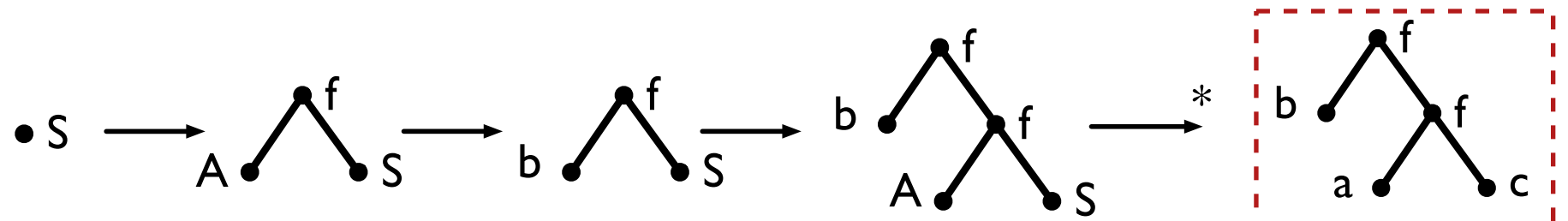
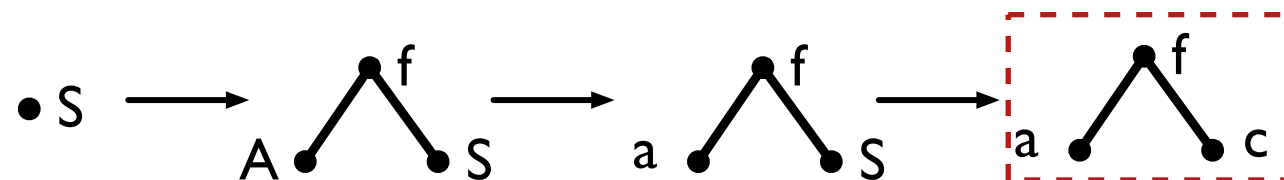
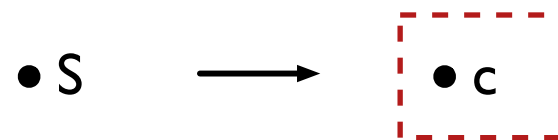
$S \rightarrow f(A, S)$

$S \rightarrow c$

$A \rightarrow a$

$A \rightarrow b$

RTG G



kfG \leftrightarrow RTG

- Für jede kfG G gilt: Die Sprache $T(G)$ der Parsebäume von G ist eine reguläre Baumsprache.

$S \rightarrow NP VP$

$NP \rightarrow \text{Hans}$

$VP \rightarrow \text{schläft}$

kfG

$S \rightarrow s_2(NP, VP)$

$NP \rightarrow np_1(\text{hans}_0)$

$VP \rightarrow vp_1(\text{schläft}_0)$

RTG

- Für jede RTG G gilt: Die Sprache der String-Erträge der Bäume in $L(G)$ ist eine kontextfreie Stringsprache.

Abschlusseigenschaften

- Reguläre Baumsprachen sind abgeschlossen unter
 - ▶ Schnitt
 - ▶ Vereinigung
 - ▶ Komplement
 - ▶ Vorwärts-Anwendung von linearen Baumhomomorphismen
 - ▶ Rückwärts-Anwendung von beliebigen Baumhomomorphismen

Abschluss unter Schnitt

mit Produktkonstruktion (vgl. endliche Stringautomaten)

$S \rightarrow s_2(\text{NP}, \text{VP})$

$\text{VP} \rightarrow \text{vp}_2(\text{V}, \text{NP})$

$\text{NP} \rightarrow \text{hans}_0$

$\text{V} \rightarrow \text{isst}_0$

$\text{NP} \rightarrow \text{kekse}_0$

G_1

$[0,3] \rightarrow s_2([0,1], [1,3])$

$[0,3] \rightarrow s_2([0,2], [2,3])$

$[1,3] \rightarrow \text{vp}_2([1,2], [2,3])$

$[0,1] \rightarrow \text{hans}_0$

$[1,2] \rightarrow \text{isst}_0$

$[2,3] \rightarrow \text{kekse}_0$

G_2

$S[0,3] \rightarrow s_2(\text{NP}[0,1], \text{VP}[1,3])$

$\text{VP}[1,3] \rightarrow \text{vp}_2(\text{V}[1,2], \text{NP}[2,3])$

$\text{NP}[0,1] \rightarrow \text{hans}_0$

$\text{V}[1,2] \rightarrow \text{isst}_0$

$\text{NP}[2,3] \rightarrow \text{kekse}_0$

G_3

mit $L(G_3) = L(G_1) \cap L(G_2)$

IRTGs

- *Interpretierte* reguläre Baumgrammatiken (IRTGs): Koller & Kuhlmann (2011).
- Kann sehr große Klasse von Grammatikformalismen stark äquivalent in IRTGs codieren.
- Verwenden Algorithmen auf Baumautomaten als Grundlage von sehr allgemeinen Parsingalgorithmen.

Algebra

- Eine *Algebra* über der Signatur Σ ist eine Struktur $A = (A, I)$, bestehend aus:
 - ▶ einer nichtleeren *Domäne* A (= Objekte der Algebra)
 - ▶ einer *Interpretationsfunktion* I , die jedem Symbol $f \in \Sigma$ mit Arität k eine Funktion $I(f): A^k \rightarrow A$ zuweist.
- Kann *Term* $t \in T_\Sigma$ als Wert von A *evaluieren*:
$$\llbracket f(t_1, \dots, t_n) \rrbracket = I(f) (\llbracket t_1 \rrbracket, \dots, \llbracket t_n \rrbracket).$$

Beispiel: Arithmetik

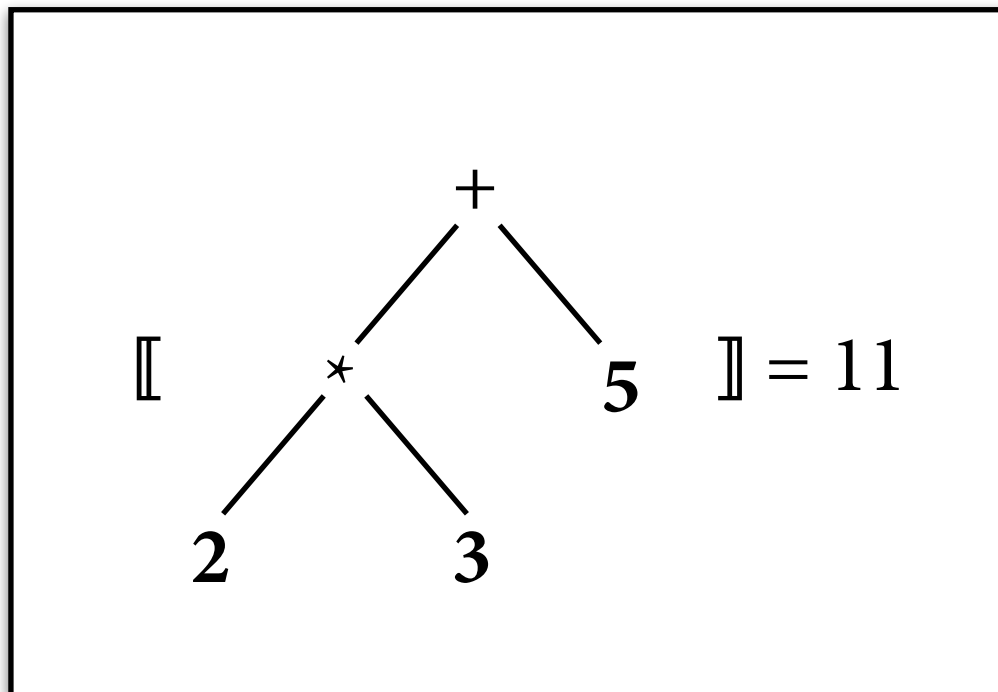
- $\Sigma = \{+|_2, *|_2, \mathbf{1}|_0, \mathbf{2}|_0, \mathbf{3}|_0, \dots\}$

- $\mathbf{N} = (\{1, 2, 3, \dots\}, I_{\mathbf{N}})$ mit

$$I_{\mathbf{N}}(+)(x,y) = x+y$$

$$I_{\mathbf{N}}(*) (x,y) = x*y$$

$$I_{\mathbf{N}}(\mathbf{1}) = 1 \text{ usw.}$$



Beispiel: Arithmetik

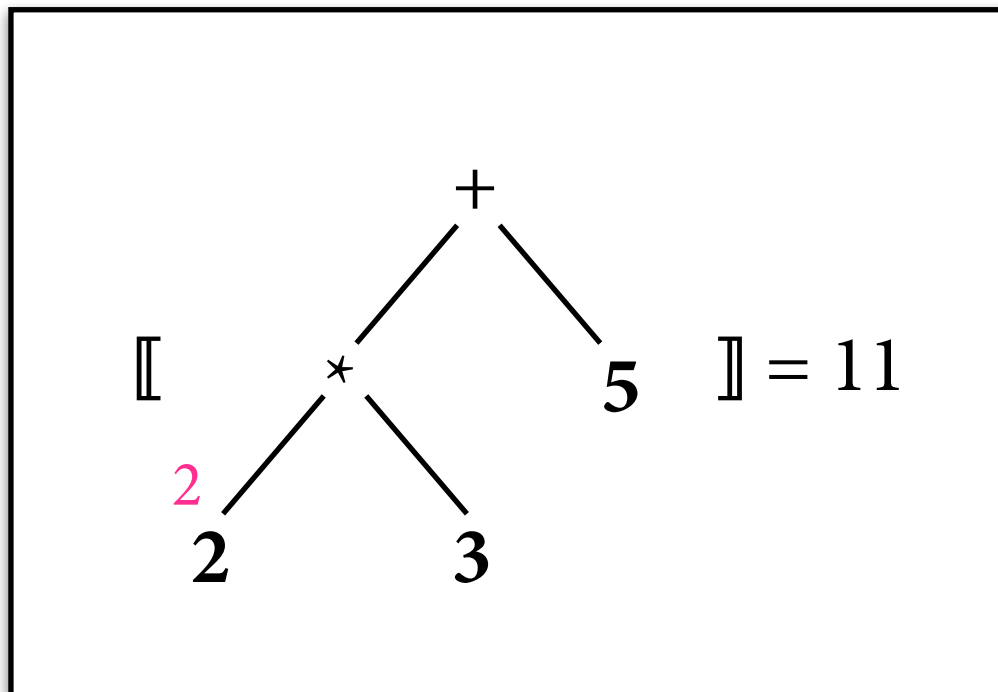
- $\Sigma = \{+|_2, *|_2, \mathbf{1}|_0, \mathbf{2}|_0, \mathbf{3}|_0, \dots\}$

- $\mathbf{N} = (\{1, 2, 3, \dots\}, I_{\mathbf{N}})$ mit

$$I_{\mathbf{N}}(+)(x,y) = x+y$$

$$I_{\mathbf{N}}(*) (x,y) = x*y$$

$$I_{\mathbf{N}}(\mathbf{1}) = 1 \text{ usw.}$$



Beispiel: Arithmetik

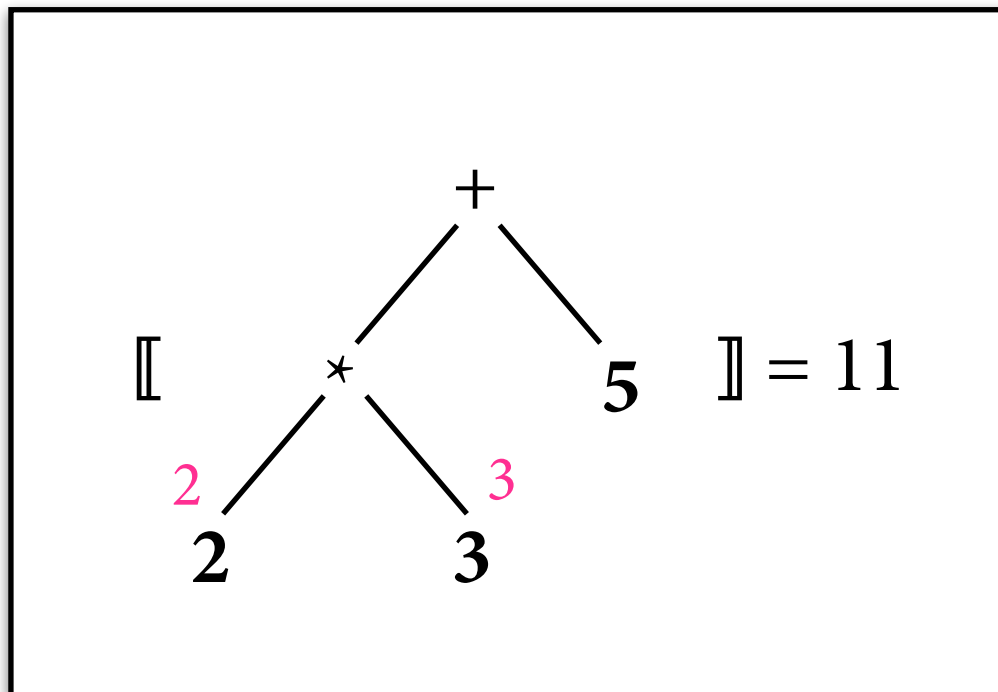
- $\Sigma = \{+|_2, *|_2, \mathbf{1}|_0, \mathbf{2}|_0, \mathbf{3}|_0, \dots\}$

- $\mathbf{N} = (\{1, 2, 3, \dots\}, I_{\mathbf{N}})$ mit

$$I_{\mathbf{N}}(+)(x,y) = x+y$$

$$I_{\mathbf{N}}(*) (x,y) = x*y$$

$$I_{\mathbf{N}}(\mathbf{1}) = 1 \text{ usw.}$$



Beispiel: Arithmetik

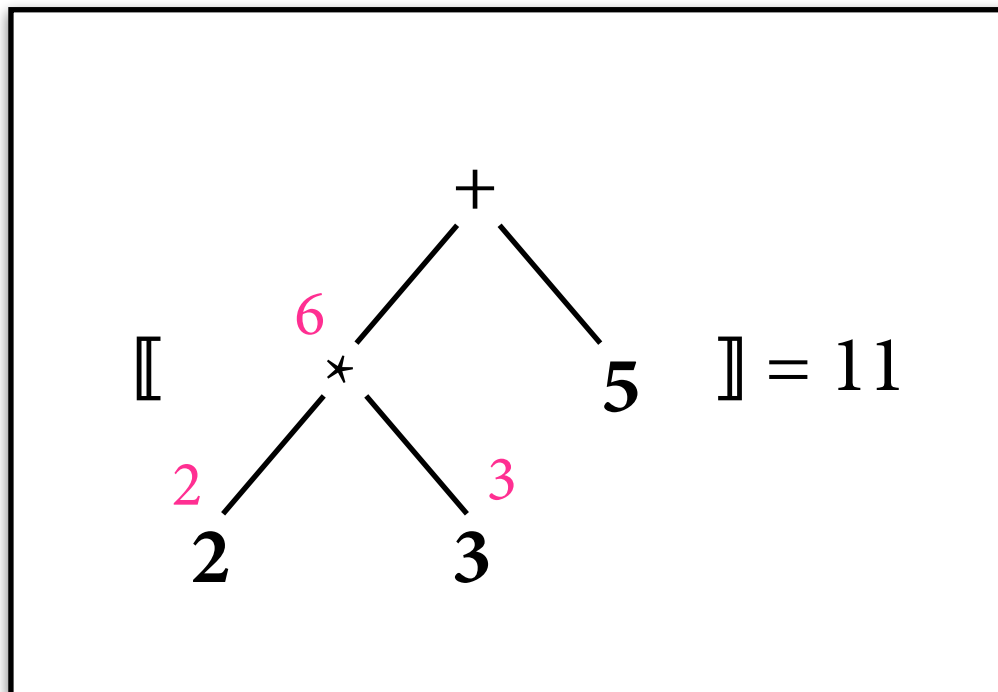
- $\Sigma = \{+|_2, *|_2, \mathbf{1}|_0, \mathbf{2}|_0, \mathbf{3}|_0, \dots\}$

- $\mathbf{N} = (\{1, 2, 3, \dots\}, I_{\mathbf{N}})$ mit

$$I_{\mathbf{N}}(+)(x,y) = x+y$$

$$I_{\mathbf{N}}(*) (x,y) = x*y$$

$$I_{\mathbf{N}}(\mathbf{1}) = 1 \text{ usw.}$$



Beispiel: Arithmetik

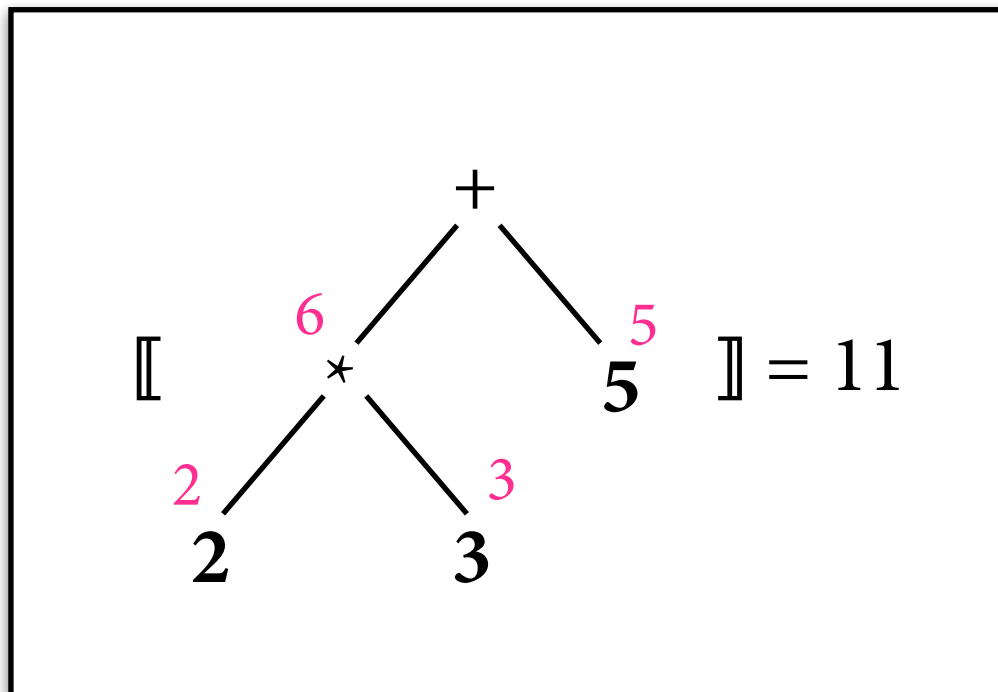
- $\Sigma = \{+|_2, *|_2, \mathbf{1}|_0, \mathbf{2}|_0, \mathbf{3}|_0, \dots\}$

- $\mathbf{N} = (\{1, 2, 3, \dots\}, I_{\mathbf{N}})$ mit

$$I_{\mathbf{N}}(+)(x,y) = x+y$$

$$I_{\mathbf{N}}(*) (x,y) = x*y$$

$$I_{\mathbf{N}}(\mathbf{1}) = 1 \text{ usw.}$$



Beispiel: Arithmetik

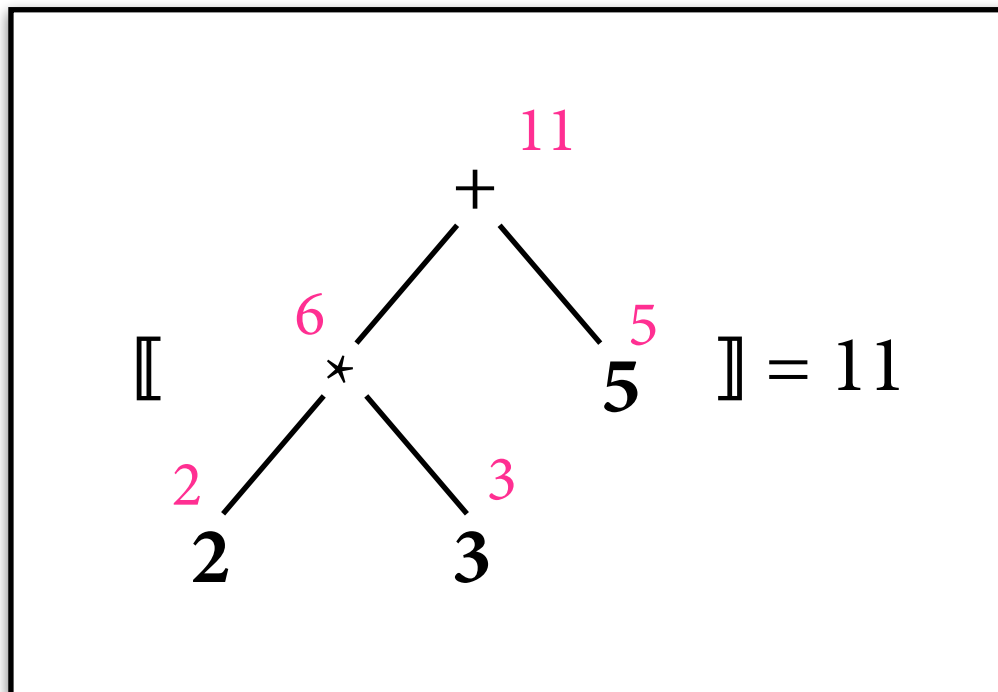
- $\Sigma = \{+|_2, *|_2, \mathbf{1}|_0, \mathbf{2}|_0, \mathbf{3}|_0, \dots\}$

- $\mathbf{N} = (\{1, 2, 3, \dots\}, I_{\mathbf{N}})$ mit

$$I_{\mathbf{N}}(+)(x,y) = x+y$$

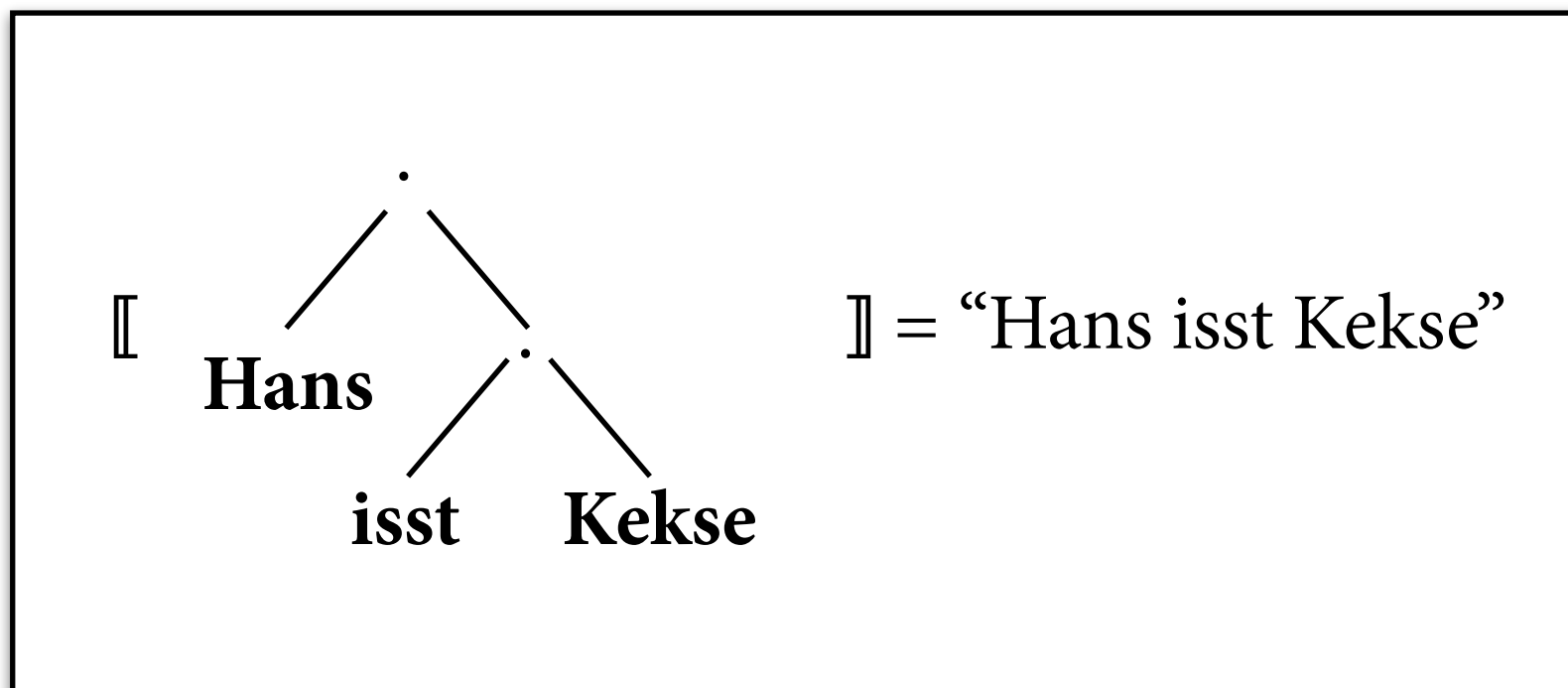
$$I_{\mathbf{N}}(*) (x,y) = x*y$$

$$I_{\mathbf{N}}(\mathbf{1}) = 1 \text{ usw.}$$



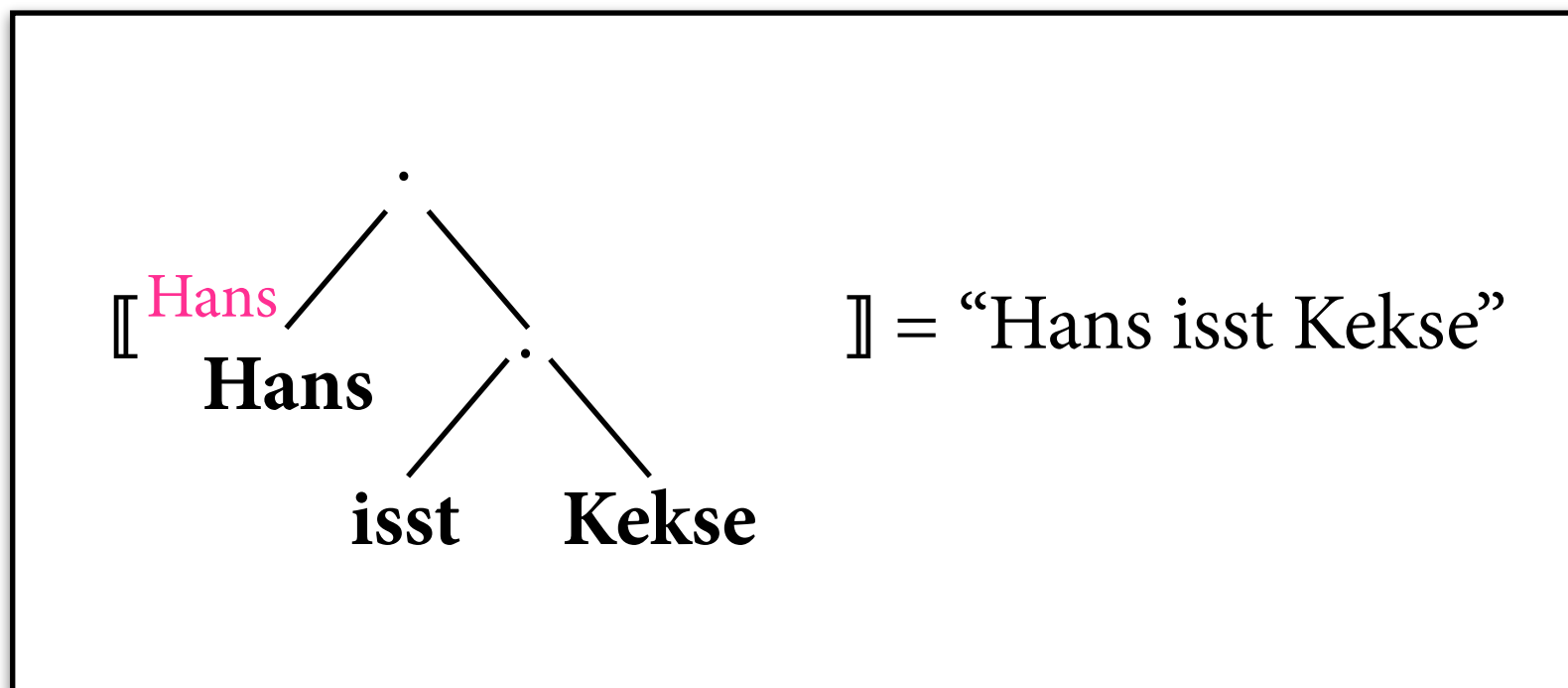
Beispiel: Strings

- $\Sigma = \{\cdot|_2, \mathbf{Hans}|_0, \mathbf{isst}|_0, \mathbf{Kekse}|_0, \dots\}$
- $\mathbf{A} = (\{\mathbf{Hans}, \mathbf{isst}, \mathbf{Hans\ isst}, \mathbf{isst\ Kekse}, \dots\}, I_A)$ mit z.B.
 $I_A(\cdot)(x,y) = x\ y$ (String-Konkatenation)
 $I_A(\mathbf{Hans}) = \mathbf{Hans}$ usw.



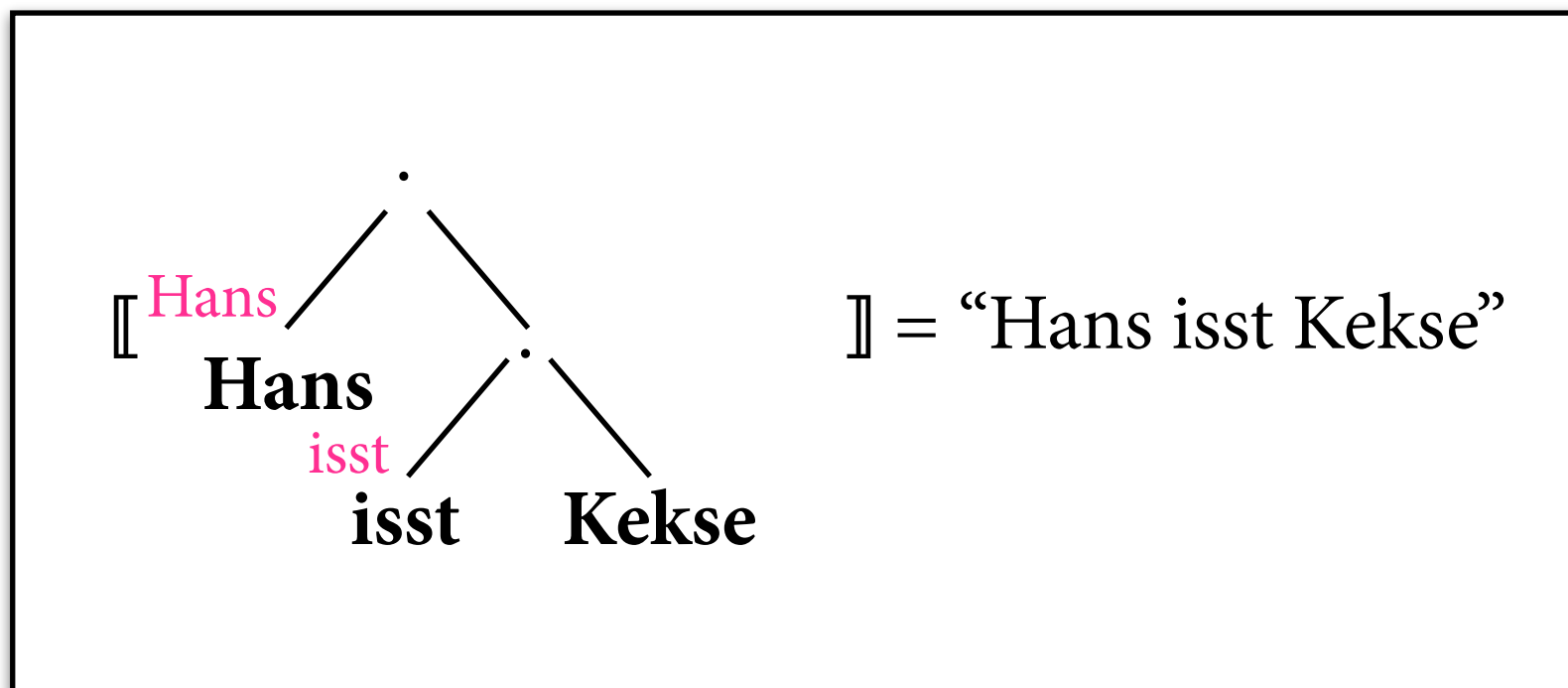
Beispiel: Strings

- $\Sigma = \{\cdot|_2, \mathbf{Hans}|_0, \mathbf{isst}|_0, \mathbf{Kekse}|_0, \dots\}$
- $\mathbf{A} = (\{\mathbf{Hans}, \mathbf{isst}, \mathbf{Hans\ isst}, \mathbf{isst\ Kekse}, \dots\}, I_A)$ mit z.B.
 $I_A(\cdot)(x,y) = x\ y$ (String-Konkatenation)
 $I_A(\mathbf{Hans}) = \mathbf{Hans}$ usw.



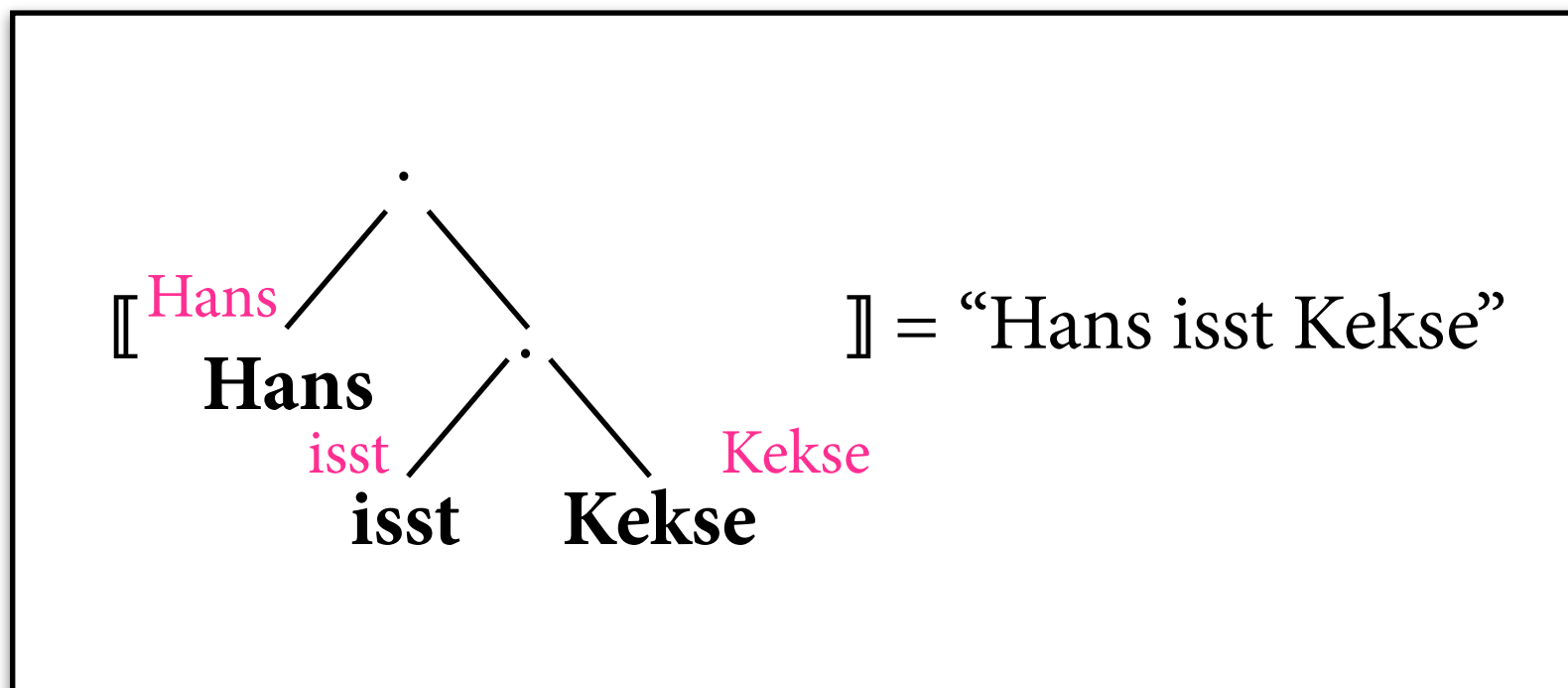
Beispiel: Strings

- $\Sigma = \{\cdot|_2, \mathbf{Hans}|_0, \mathbf{isst}|_0, \mathbf{Kekse}|_0, \dots\}$
- $\mathbf{A} = (\{\mathbf{Hans}, \mathbf{isst}, \mathbf{Hans\ isst}, \mathbf{isst\ Kekse}, \dots\}, I_A)$ mit z.B.
 $I_A(\cdot)(x,y) = x\ y$ (String-Konkatenation)
 $I_A(\mathbf{Hans}) = \mathbf{Hans}$ usw.



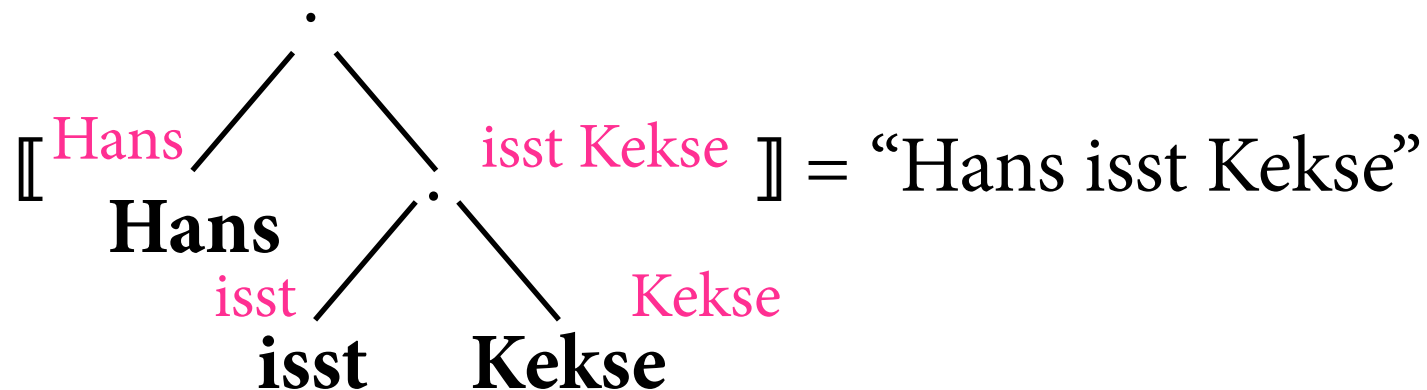
Beispiel: Strings

- $\Sigma = \{\cdot|_2, \mathbf{Hans}|_0, \mathbf{isst}|_0, \mathbf{Kekse}|_0, \dots\}$
- $\mathbf{A} = (\{\mathbf{Hans}, \mathbf{isst}, \mathbf{Hans\ isst}, \mathbf{isst\ Kekse}, \dots\}, I_A)$ mit z.B.
 $I_A(\cdot)(x,y) = x\ y$ (String-Konkatenation)
 $I_A(\mathbf{Hans}) = \mathbf{Hans}$ usw.



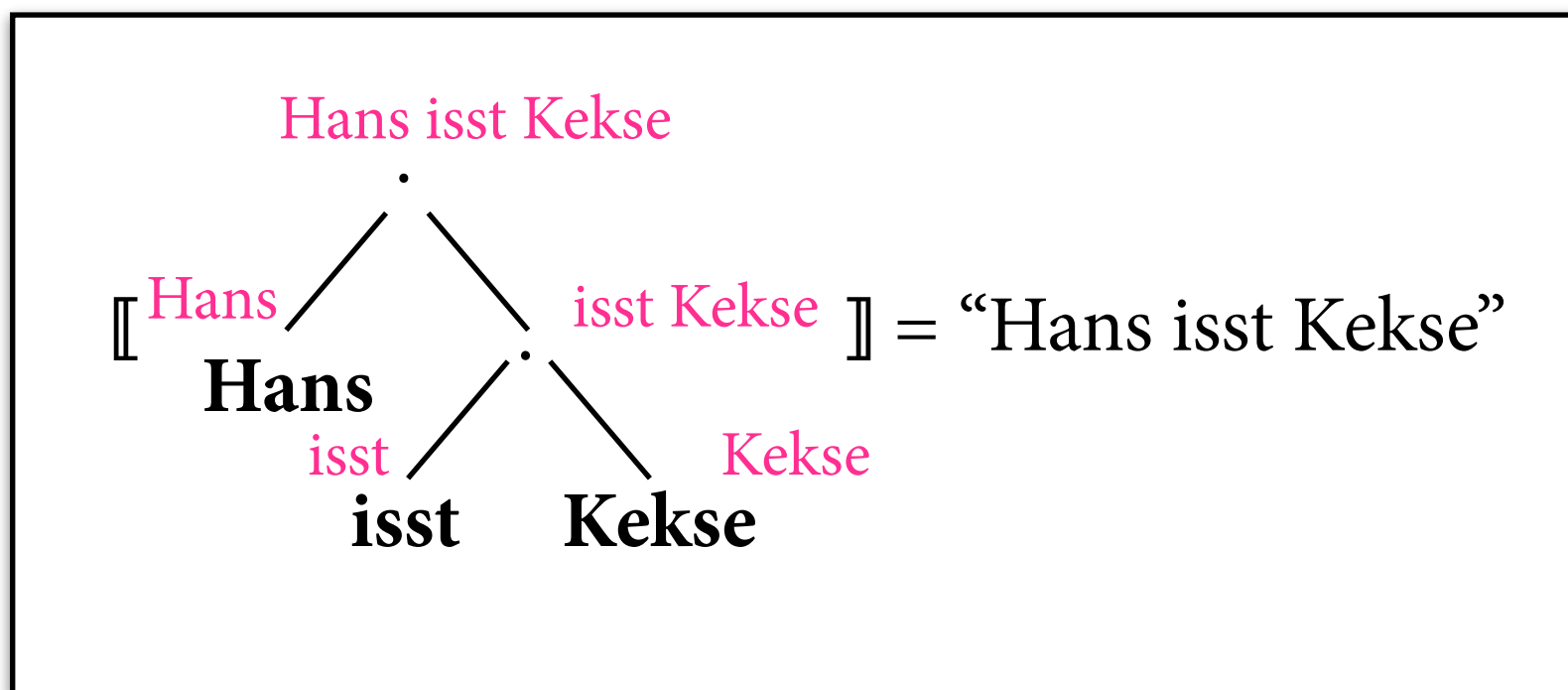
Beispiel: Strings

- $\Sigma = \{\cdot|_2, \mathbf{Hans}|_0, \mathbf{isst}|_0, \mathbf{Kekse}|_0, \dots\}$
- $\mathbf{A} = (\{\mathbf{Hans}, \mathbf{isst}, \mathbf{Hans\ isst}, \mathbf{isst\ Kekse}, \dots\}, I_A)$ mit z.B.
 $I_A(\cdot)(x,y) = x\ y$ (String-Konkatenation)
 $I_A(\mathbf{Hans}) = \mathbf{Hans}$ usw.



Beispiel: Strings

- $\Sigma = \{\cdot|_2, \mathbf{Hans}|_0, \mathbf{isst}|_0, \mathbf{Kekse}|_0, \dots\}$
- $\mathbf{A} = (\{\mathbf{Hans}, \mathbf{isst}, \mathbf{Hans\ isst}, \mathbf{isst\ Kekse}, \dots\}, I_A)$ mit z.B.
 $I_A(\cdot)(x,y) = x\ y$ (String-Konkatenation)
 $I_A(\mathbf{Hans}) = \mathbf{Hans}$ usw.



IRTGs (vereinfacht)

- Eine interpretierte reguläre Baumgrammatik (IRTG) ist ein Paar $\mathbf{G} = (G, (\mathbf{A}_1, \dots, \mathbf{A}_k))$ aus:
 - ▶ einer RTG G über einer Signatur Σ
 - ▶ Algebren $\mathbf{A}_1, \dots, \mathbf{A}_k$ über der Signatur Σ
- RTG beschreibt eine Sprache $L(G) \subseteq T_\Sigma$ von *Ableitungsbäumen*.
- IRTG beschreibt die Sprache
$$L(\mathbf{G}) = \{ (\llbracket t \rrbracket_{\mathbf{A}_1}, \dots, \llbracket t \rrbracket_{\mathbf{A}_k}) \mid t \in L(G) \}$$
$$\subseteq A_1 \times \dots \times A_k$$

Beispiel

$G = (G, A)$ mit $A = \text{Stringalgebra}$

G

$S \rightarrow \cdot(\text{NP}, \text{VP})$

$\text{VP} \rightarrow \cdot(\text{V}, \text{NP})$

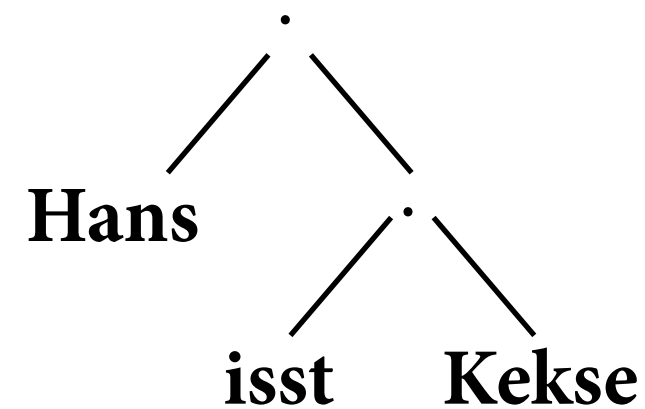
$\text{NP} \rightarrow \mathbf{Hans}$

$\text{V} \rightarrow \mathbf{isst}$

$\text{NP} \rightarrow \mathbf{Kekse}$



Abl.baum $t \in L(G)$



“Hans isst Kekse”

Wert $\llbracket t \rrbracket \in A$

$\Rightarrow \text{“Hans isst Kekse”} \in L(G)$

Parsing mit IRTGs

- Sei $\mathbf{G} = (G, (\mathbf{A}_1, \dots, \mathbf{A}_k))$ IRTG und $a \in A_1$ Eingabe.
- Gesucht: $\text{parses}(a) = \{ t \in L(G) \mid \llbracket t \rrbracket_{A_1} = a \}$
 $= L(G) \cap \text{terms}(a)$
mit $\text{terms}(a) = \{ t \in T_\Sigma \mid \llbracket t \rrbracket_{A_1} = a \}$.
- Nenne \mathbf{A}_1 *regulär zerlegbar*, wenn für jedes $a \in A_1$ eine RTG D_a existiert mit $L(D_a) = \text{terms}(a)$
(*Zerlegungsgrammatik*).
- Dann gilt: $\text{parses}(a) = L(G) \cap L(D_a)$.

Beispiel

$S \rightarrow \cdot(\text{NP}, \text{VP})$

$\text{VP} \rightarrow \cdot(\text{V}, \text{NP})$

$\text{NP} \rightarrow \mathbf{Hans}$

$\text{V} \rightarrow \mathbf{isst}$

$\text{NP} \rightarrow \mathbf{Kekse}$

G

$S[0,3] \rightarrow \cdot(\text{NP}[0,1], \text{VP}[1,3])$

$\text{VP}[1,3] \rightarrow \cdot(\text{V}[1,2], \text{NP}[2,3])$

$\text{NP}[0,1] \rightarrow \mathbf{Hans}$

$\text{V}[1,2] \rightarrow \mathbf{isst}$

$\text{NP}[2,3] \rightarrow \mathbf{Kekse}$

$G \cap D_a$

Eingabe $a \in A$

“Hans isst Kekse”

Zerlegungsgrammatik D_a

$[0,3] \rightarrow \cdot([0,1], [1,3])$

$[0,3] \rightarrow \cdot([0,2], [2,3])$

$[1,3] \rightarrow \cdot([1,2], [2,3])$

$[0,1] \rightarrow \mathbf{Hans}$

$[1,2] \rightarrow \mathbf{isst}$

$[2,3] \rightarrow \mathbf{Kekse}$

Baum-Homomorphismen

- Einfache Abbildungen von Bäumen über Signatur Σ in Bäume über Signatur Δ .
- Definiere Bild für jedes Symbol $f|n \in \Sigma$ als Term über Δ mit Variablen x_1, \dots, x_n : $h_f \in T_\Delta[x_1, \dots, x_n]$.
- Dann rekursiv auf Funktion $h: T_\Sigma \rightarrow T_\Delta$ fortsetzen:
$$h(f(t_1, \dots, t_n)) = h_f [h(t_1)/x_1, \dots, h(t_n)/x_n]$$

Homomorphism: Beispiel

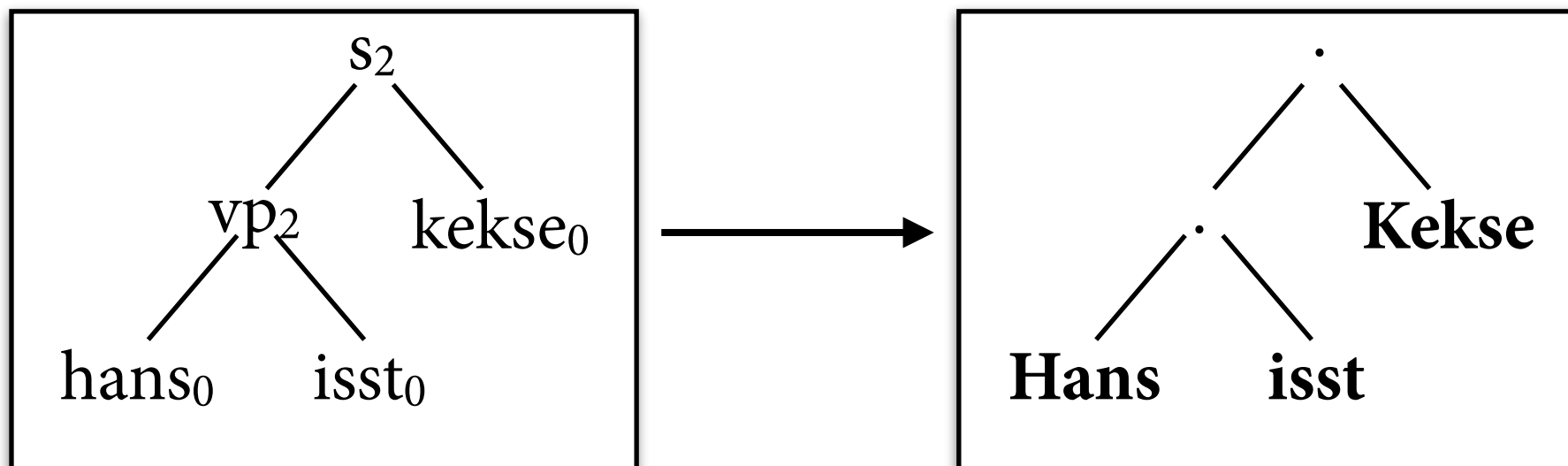
$$h(s_2) = \cdot(x_1, x_2)$$

$$h(vp_2) = \cdot(x_1, x_2)$$

$$h(hans_0) = \mathbf{Hans}$$

$$h(isst_0) = \mathbf{isst}$$

$$h(kekse_0) = \mathbf{Kekse}$$



IRTGs (vollständige Version)

- Eine interpretierte reguläre Baumgrammatik (IRTG) ist ein Paar $G = (G, ((h_1, \mathbf{A}_1), \dots, (h_k, \mathbf{A}_k)))$ aus:
 - ▶ einer RTG G über einer Signatur Σ
 - ▶ Algebren $\mathbf{A}_1, \dots, \mathbf{A}_k$ über Signaturen $\Delta_1, \dots, \Delta_k$
 - ▶ Baumhomomorphismen $h_i: T_\Sigma \rightarrow T_{\Delta_i}$
- IRTG beschreibt die Sprache
$$L(\mathbf{G}) = \{ (\llbracket h_1(t) \rrbracket_{\mathbf{A}_1}, \dots, \llbracket h_k(t) \rrbracket_{\mathbf{A}_k}) \mid t \in L(G) \}$$
$$\subseteq A_1 \times \dots \times A_k$$

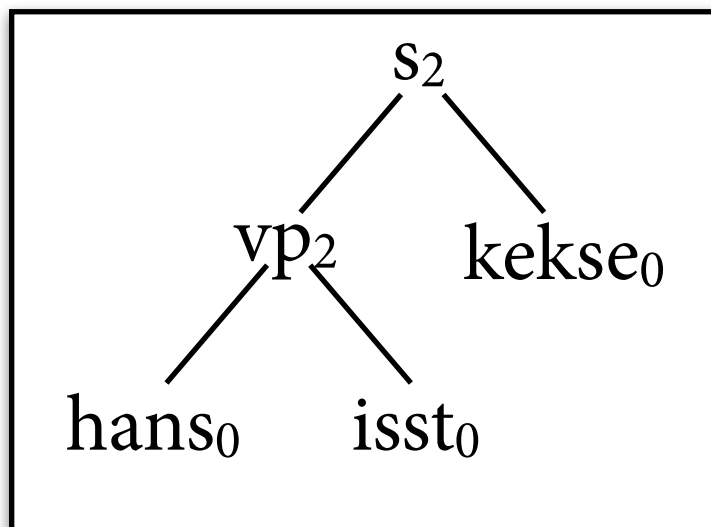
Beispiel

$$S \rightarrow s_2(\text{NP}, \text{VP})$$
$$\text{VP} \rightarrow \text{vp}_2(\text{V}, \text{NP})$$
$$\text{NP} \rightarrow \text{hans}_0$$
$$\text{V} \rightarrow \text{isst}_0$$
$$\text{NP} \rightarrow \text{kekse}_0$$

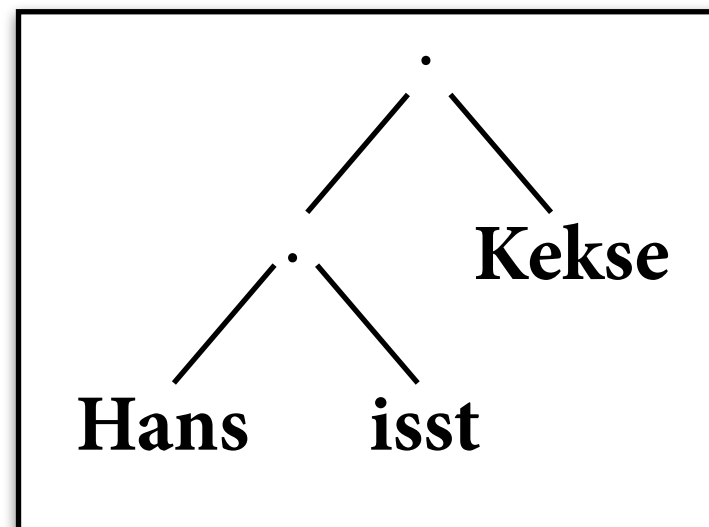
G

$$h(s_2) = \cdot(x_1, x_2)$$
$$h(\text{vp}_2) = \cdot(x_1, x_2)$$
$$h(\text{hans}_0) = \mathbf{Hans}$$
$$h(\text{isst}_0) = \mathbf{isst}$$
$$h(\text{kekse}_0) = \mathbf{Kekse}$$

Abl.baum $t \in L(G)$



Term $h(t) \in T_\Delta$



Wert $\llbracket h(t) \rrbracket \in A$

“Hans isst Kekse”

RDG als IRTG

- Definiere für max. Blockgrad k Stringtupelmengen über dem Alphabet L : $S_i = (L^*)^i$
 - ▶ Domäne = $S_1 \cup \dots \cup S_k$
- Jede ordnungsannotiert Wort $w:\omega$ wird als Stringtupel-Konkatenation interpretiert:
 $I(\omega): S_{i1} \times \dots \times S_{ir} \rightarrow S_j$
 - ▶ $\llbracket \langle \text{gegeben:1,20} \rangle (\text{Hans, ein KB}) \rrbracket = (\text{“Hans”, “ein KB gegeben”})$
 - ▶ $\llbracket \langle \text{wird:101} \rangle ((\text{Hans, ein KB gegeben})) \rrbracket$
= “Hans wird ein KB gegeben”
- *Partielle* Algebra, nicht alle Terme haben einen Wert.

RDG als IRTG

$S \rightarrow \langle \text{wird: 101} \rangle (\text{VP})$
 $\text{VP} \rightarrow \langle \text{geg: 1,20} \rangle (\text{NP}, \text{NP})$
 $\text{NP} \rightarrow \langle \text{Hans: 0} \rangle$
 $\text{NP} \rightarrow \langle \text{KB: 10} \rangle (\text{Det})$
 $\text{Det} \rightarrow \langle \text{ein: 0} \rangle$

RDG-Grammatik

$S \rightarrow r_1(\text{VP})$ $h(r_1) = \text{"wird:101"}(x_1)$
 $\text{VP} \rightarrow r_2(\text{NP}, \text{NP})$ $h(r_2) = \text{"gegeben:1,20"}(x_1, x_2)$
 $\text{NP} \rightarrow r_3$ $h(r_3) = \text{"Hans:0"}$
 $\text{NP} \rightarrow r_4(\text{Det})$ $h(r_4) = \text{"KB:10"}(x_1)$
 $\text{Det} \rightarrow r_5$ $h(r_5) = \text{"ein:0"}$

IRTG-Grammatik

Baumhomomorphismus in
RDG-Stringtupel-Algebra

Parsing = Zerlegungsgrammatiken für die Algebra berechnen;
 $\text{pareses}(a) = L(G) \cap h^{-1}(L(D_a))$

IRTG-Formalismen

	Ableitungsbäume	Algebra
kontextfreie Grammatiken	RTL	Stringalgebra
TSG: Strings	RTL	Stringalgebra
TSG: Abgeleitete Bäume	RTL	Baumalgebra
TAG: Strings	RTL	TAG-Stringpaar-Algebra
TAG: Abgeleitete Bäume	RTL	TAG-Baumalgebra
LCFRS/RDG	RTL	komplexe Stringalgebra
etc. etc. etc.	RTL	etc. etc. etc.

IRTG-Formalismen

	Abl.bäume	Algebra 1	Algebra 2
SCFG	RTL	Stringalgebra	Stringalgebra
STSG: Strings	RTL	Stringalgebra	Stringalgebra
STSG: abg. Bäume	RTL	Baumalgebra	Baumalgebra
STAG: Strings	RTL	TAG Stringpaare	TAG Stringpaare
STAG: abg. Bäume	RTL	TAG-Baumalgebra	TAG-Baumalgebra
Bottom-up Baumtransducer	RTL	Baumalgebra (Tiefe 1)	Baumalgebra
Tree-to-String- Transducer	RTL	Baumalgebra	Stringalgebra
etc. etc. etc.	RTL	etc. etc. etc.	etc. etc. etc.

IRTGs: Ausblick

- Mehrere Interpretationen ($k > 1$): Relationen zwischen Strings, Bäumen usw.
 - ▶ maschinelle Übersetzung
 - ▶ Abbildung von String in Parsebäume oder sem. Repräsentationen, oder umgekehrt
- PCFG-artige und log-lineare W.modelle sehr einfach zu übertragen.

Zusammenfassung

- Reguläre Baumgrammatiken / -sprachen und endliche Baumautomaten.
 - ▶ Abschlusseigenschaften!
- Interpretierte RTGs:
 - ▶ Beschreibe Sprache von Objekten in beliebiger Algebra.
 - ▶ Regulär zerlegbare Algebra \Rightarrow Parsing mit RTGs.
 - ▶ Implementierung: <http://github.com/coli-saar/alto>