Parsing Beyond Context-Free Grammars: Data-driven LCFRS Parsing

Laura Kallmeyer & Tatiana Bladier
Heinrich-Heine-Universität Düsseldorf

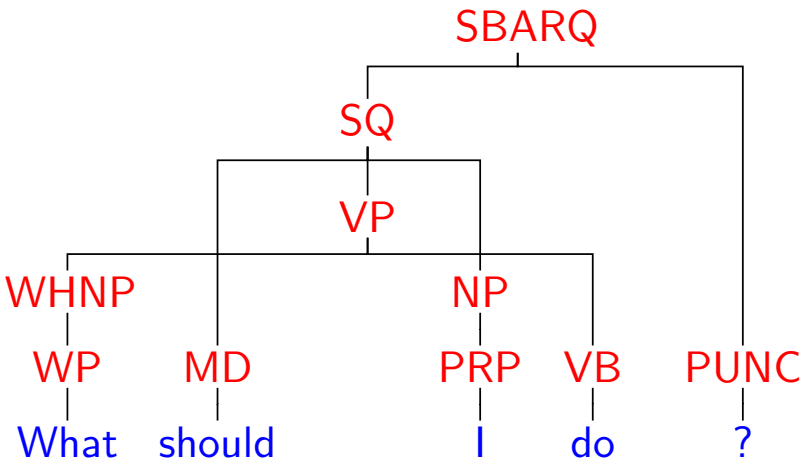
Sommersemester 2018

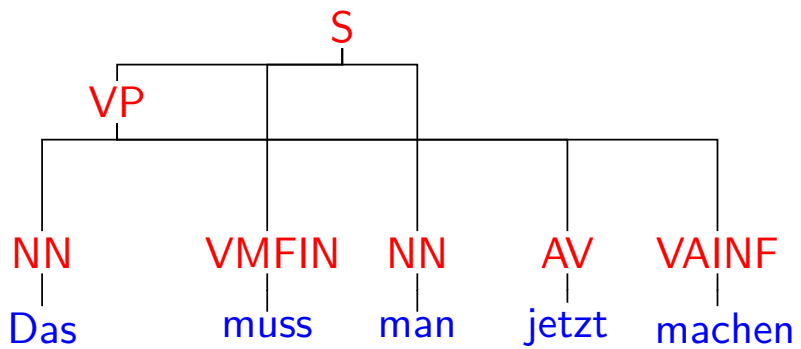
Overview

1. **Induction of LCFRSs from Treebanks**
2. **PLCFRS and weighted deductive CYK parsing**

Discontinuous constituents in natural languages

- Discontinuous constituents are common across natural languages.
- Frequently in languages with relatively free word-order (for example, German or Russian), but also present in other languages.

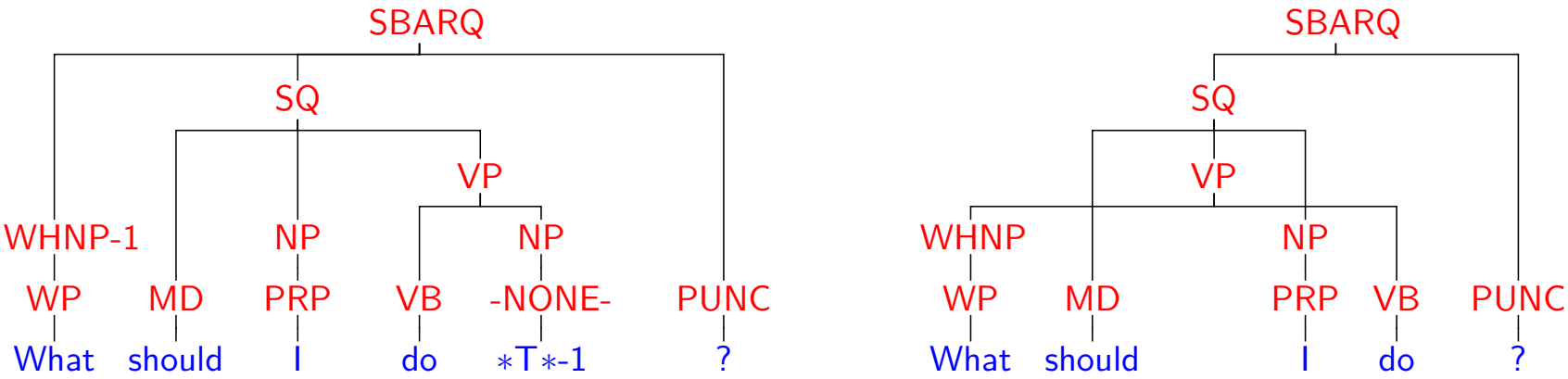




Treebanks with discontinuous fragments (1)

- The first treebank with discontinuous constituents was the **Negra Treebank** for German [SBKU98].
- Another treebank for German, **TiGer** [Smi03], also uses discontinuities in its annotation.
- Approximately 25 % of sentences in NeGra and TIGER and 20 % of sentences in **Penn Treebank** have discontinuities ([KM13]).
- Discontinuities are presented differently in treebanks:
  - NeGra and TIGER have crossing branches.
  - Penn Treebank uses special labels (called *traces*) to mark discontinuous constituents.
  - French Treebank does not mark discontinuous constituents.

Discontinuous constituents in natural languages (3)



PTB-style annotation and NeGra-style annotation

Discontinuous constituents in natural languages (3)

Discontinuities in NLs are caused by different linguistic phenomena:

Fronting

(1) Drei Papiere will ich heute noch schreiben
three papers want I today still write
'I still want to write three papers today'

Separable prefixes (in German)

(2) Schillen wies dies gestern zurück
Schillen rejected that yesterday VPART
'Schillen rejected that yesterday'

Discontinuous constituents in natural languages (4)

Wh-movement

(3) What should I do?

Long extraction

(4) . . . those chains include Bloomingdale's, which Campeau recently said it will sell

Extraposed nominal modifiers

(5) Prices fell marginally for fuel and electricity

Procedure of LCFRS extraction (1)

- General idea [MS08, KM13]:

  $\rightarrow$ LCFRS productions can be induced from a discontinuous tree provided in the treebank.

  $\rightarrow$ The trees in treebanks (for example, NeGra and TIGER) can be interpreted as LCFRS derivations.

  $\rightarrow$ First, one has to identify the clauses the parse tree is composed of.

  $\rightarrow$ Then, for each tree over a sentence $w_1, \ldots, w_n$, the set of clauses is extracted.

  $\rightarrow$ The clauses are then counted and collected into single (P)LCFRS grammar.

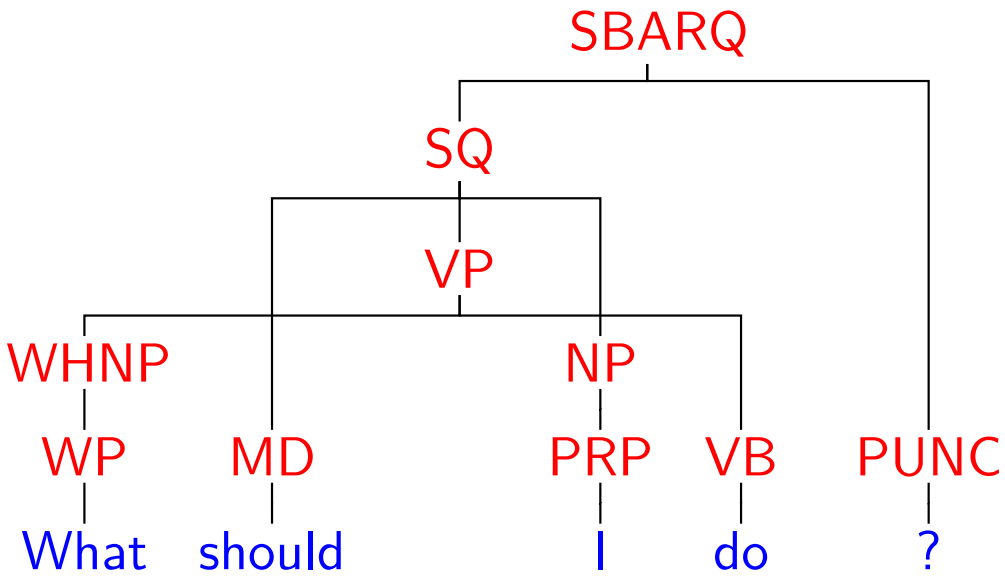
Procedure of LCFRS extraction (2)

- For each non-terminal node N (where N is not a preterminal) with n daughters $N'_1, \ldots, N'_m$:
  - introduce a clause with an LHS predicate named N and m RHS predicates named N'_1 to N'_m
- For each w_i, $1 \leq i \leq n$, a new variable X_i is introduced.
- Then for the predicate N, the following conditions must hold:
  - the arguments of N must contain no terminals
  - the concatenation of all variables in all arguments of N must be the concatenation of all $X \in \{X_i \mid N \textit{ dominates } w_i\}$ such that X_i precedes X_j if $i < j$
  - a variable X_i with $1 \leq i < n$, is the right boundary of an argument of the predicate N iff $X_{i+1} \notin \{X_i \mid N \textit{ dominates } w_i\}$, i.e., an argument boundary is introduced at each discontinuity.

Procedure of LCFRS extraction (3)

- The arguments of RHS predicates are determined in the same way.
- Range variables which are adjacent on the LHS and the RHS are collapsed into single variables.
- For each preterminal node N dominating some terminal node w_i, we introduce a clause $N(w_i) \rightarrow \varepsilon$, called a *lexical clause*.
- Discontinuous non-terminals are annotated with a number indicating their fan-out (e.g. $VP_2(x_1, x_3)$).
  - $\rightarrow$ each non-terminal type A can be mapped to a unique fan-out by $dim(A)$.

Procedure of LCFRS extraction (4)



$SBARQ_1(X_1X_2) \rightarrow SQ_1(X_1)PUNC_1(X_2)$

$SQ_1(X_1X_2X_3X_4) \rightarrow VP_2(X_1, X_4)MD_1(X_2)NP_1(X_3)$

$VP_2(X_1, X_2) \rightarrow WHNP_1(X_1)VB_1(X_2)$

$WHNP_1(X_1) \rightarrow WP_1(X_1)$

$NP_1(X_1) \rightarrow PRP_1(X_1)$

$WP_1(What) \rightarrow \varepsilon$

$MD_1(should) \rightarrow \varepsilon$

$PRP_1(I) \rightarrow \varepsilon$

$VB_1(do) \rightarrow \varepsilon$

$PUNC_1(?) \rightarrow \varepsilon$

Dimensions of extracted LCFRSs

	sent	cross	av slen
NeGra	20602	5853 (28.40%)	17.24
TIGER	50474	14114 (27.96%)	17.60

Properties of NeGra and TIGER, [MS08]

	$G_N(NeGra)$	$G_T(TIGER)$
total # of clauses	468,607	1,192,807
total # of different clauses	71,868	127,154
lexical clauses	52,747	92,731
non-lexical clauses	19,121	34,423

Statistics of extracted LCFRSs, [MS08]

Most frequent clauses in extracted LCFRSs

1	733	$S(X_1X_2X_3X_4) \to VP(X_1, X_4)VAFIN(X_2)NP(X_3)$
2	271	$VP(X_1, X_2) \to PP(X_1)VVPP(X_2)$
3	268	$S(X_1X_2X_3X_4X_5) \to VP(X_1, X_3, X_5)VAFIN(X_2)NP(X_4)$
4	236	$S(X_1X_2X_3X_4) \to VP(X_1, X_4)VAFIN(X_2)PPER(X_3)$
5	193	$S(X_1X_2X_3X_4) \to VP(X_1, X_3)NP(X_2)VAFIN(X_4)$
6	149	$NP(X_1X_2, X_3) \to ART(X_1)NN(X_2)S(X_3)$
7	148	$NP(X_1, X_2) \to PPER(X_1)VP(X_2)$
8	142	$S(X_1X_2X_3X_4) \to VP(X_1, X_4)VMFIN(X_2)NP(X_3)$
9	130	$VP(X_1, X_2X_3) \to VP(X_1, X_2)VAINF(X_3)$
10	127	$NP(X_1, X_2) \to PPER(X_1)S(X_2)$

Most frequent clauses with predicated of arity ≥ 2 in G_N (NeGra), [MS08]

Parsing LCFRSs

- Data-driven parsing has largely been dominated by Probabilistic Context-Free Grammar (PCFG).
- LCFRSs are a generalization of CFG
  - $\rightarrow$ different PCFG parsing techniques, such as Best-First Parsing ([CC98]), Weighted Deductive Parsing ([Ned03]) and A* parsing ([KM03]), can be transferred to PLCFRS.

$\rightarrow$ in this course: weighted deductive CYK parsing [KM13]

Probabilistic LCFRS [KM13]

- The definition of a probabilistic LCFRS is a straightforward extension of the definition of PCFG.
- A probabilistic LCFRS (PLCFRS) is a tuple $\langle N, T, V, P, S, \rho \rangle$ such that $\langle N, T, V, P, S \rangle$ is a LCFRS and $\rho : P \to [0 \ldots 1]$ is a function such that for all $A \in N : \Sigma_{A(\vec{x}) \to \vec{\Phi} \in P} \rho(A(\vec{x}) \to \vec{\Phi}) = 1$

- Example: A sample PLCFRS with non-terminals $\{S, A, B\}$, terminals $\{a\}$ and start symbol S:

$0.2 : S(X) \to A(X)$ $\quad$ $0.8 : S(XY) \to B(X, Y)$
$0.7 : A(aX) \to A(X)$ $\quad$ $0.3 : A(a) \to \varepsilon$
$0.8 : B(aX, aY) \to B(X, Y)$ $\quad$ $0.2 : B(a, a) \to \varepsilon$

CYK parsing for PLCFRSs: preprocessing [KM13]

- For the CYK-algorithm, the extracted LCFRS should be ordered, binarized, and have no ε-rules.

  $\rightarrow$ if not: make the LCFRS ordered and eliminate ε-rules

- Do binarization and (optionally) incorporate additional context (next two slides)
- The POS-tagging is carried out prior to parsing (POS tags are non-terminals of fan-out 1)
- The rules are then:
  - either of the form $A(a) \rightarrow \varepsilon$ with A being a POS-tag and $a \in T$
  - or of the form $A(\vec{x}) \rightarrow B(\vec{x})$
  - description or $A(\vec{\alpha}) \rightarrow B(\vec{x})C(\vec{y})$
    where $\vec{\alpha} \in (V^+)^{dim(A)}$, $\vec{x} \in V^{dim(B)}$, $\vec{y} \in V^{dim(C)}$

  $\Rightarrow$ only the rules for POS tags contain terminals in their lefthand sides

Head-outward binarization (1)

- Binarization = conversion of the LCFRS to an LCFRS of rank 2 (e.g. bringing LCFRS to Chomsky normal form).

  $\rightarrow$ Minimizing fan-out and the number of variables.

- Head-outward binarization:
  - head is the lowest subtree.
  - the tree is extended by adding first all sisters to its right and then the sisters to its left.

Head-outward binarization (2)

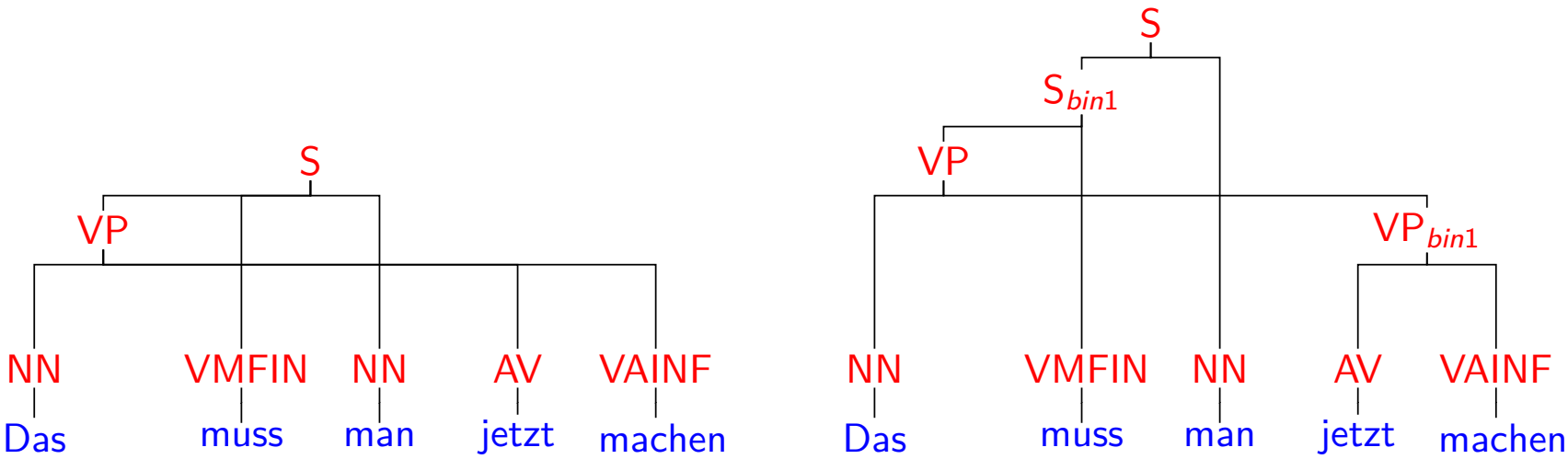
- Rule extracted for the S-node: $S(XYZU) \rightarrow VP(X, U)VMFIN(Y)NN(Z)$

  Reordering for head-outward binarization: $S(XYZU) \rightarrow NN(Z)VP(X, U)VMFIN(Y)$

  New rules after binarization: $S(XYZ) \rightarrow S_{bin1}(X, Z)NN(Y)$ and $S_{bin1}(XY, Z) \rightarrow VP(X, Z)VMFIN(Y)$

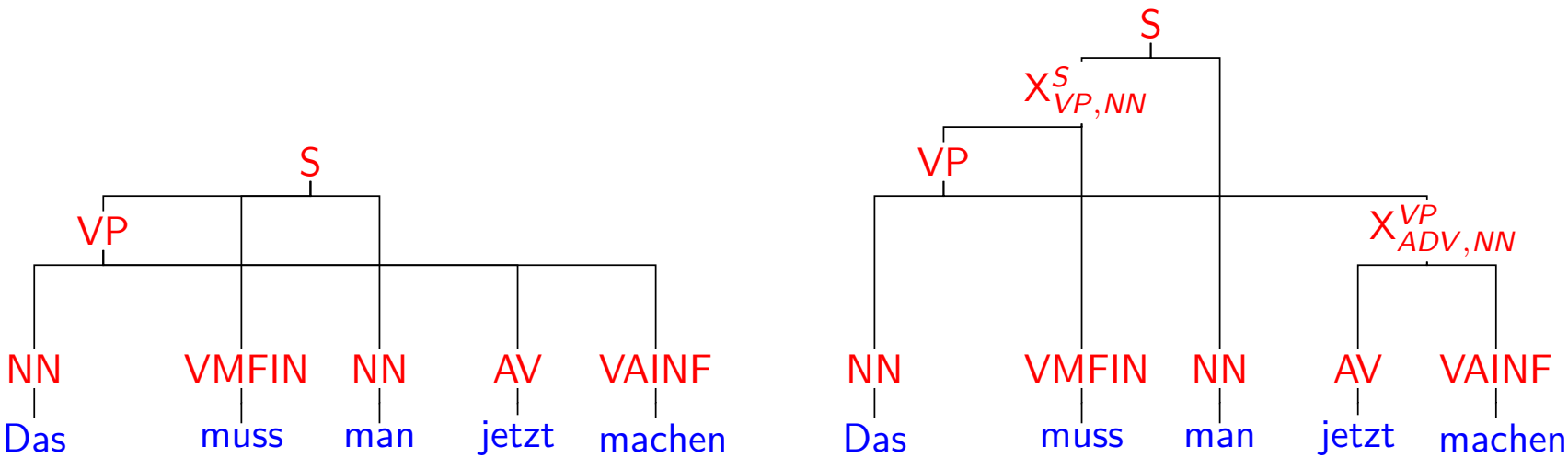
- Rule extracted for the VP-node: $VP(X, YZ) \rightarrow NN(X)AV(Y)VAINF(Z)$

  New rules after binarization: $VP(X, Y) \rightarrow NN(X)VP_{bin1}(Z)$ and $VP_{bin1}(XY) \rightarrow AV(X)VAINF(Y)$



Markovization

- In order to capture certain generalizations during binarization, **markovization** can be applied.
- *Markovization* is achieved by introducing only a single new non-terminal for the new rules and adding *vertical* and *horizontal context* from the original tree.
  - Vertical context is collected during grammar extraction
    $\Rightarrow$ we add the first v labels on the path from the root node of the tree that we want to binarize to the root of the entire tree.
  - Horizontal context is collected during binarization:
    Taken some rule $\Rightarrow A(\vec{\alpha}) \rightarrow A_0(\vec{\alpha_0}) \dots A_m(\vec{\alpha_m})$, for the new non-terminal that comprises the right-hand side elements $A_i \dots A_m$ for some $(1 \leq i \leq m)$, we add the first h elements of $A_i, A_{i-1}, \dots, A_0$



Parsing PLCFRSs

- A probabilistic version of the CYK parser extended with weights (weighted deductive parsing).
- Idea of weighted deductive parsing [Ned03]:
  - Give a deductive definition of the probability of a parse tree.
  - Use Knuth's algorithm to compute the best parse tree for category S and a given input w.

- Advantage:
  - Yields the best parse without exhaustive parsing.
  - Can be used to parse any grammar formalism as long as an appropriate weighted deductive system can be defined.

CYK Weighted deductive parsing for LCFRSs

- CYK parser for LCFRSs
- Each item has an additional weight. Intuition: weight = costs to build an item.
- CYK deduction rules specify how to compute the weight of the consequent item from the weights of the antecedent items.

- For a given input w, our items have the form [A, $\vec{\rho}$]
  - where $A \in N$
  - and $\vec{\rho}$ is a range vector that characterizes the span of A.
- Each item has a weight in that encodes the Viterbi inside score of its best parse tree.
  - More precisely, we use the *log* probability $log(\rho)$ where ρ is the probability.

Weighted CYK LCFRS parsing

Scan: $\dfrac{}{0 : [A, \langle\langle i, i+1 \rangle\rangle]}$ A is the POS tag of w_{i+1}

Unary: $\dfrac{in : [B, \vec{\rho}]}{in + log(p) : [A, \vec{\rho}]}$ $p : A(\vec{\alpha}) \rightarrow B(\vec{\alpha}) \in P$

Binary: $\dfrac{in_B : [B, \vec{\rho_B}], in_C : [C, \vec{\rho_C}]}{in_B + in_C + log(p) : [A, \vec{\rho_A}]}$ p : $A(\vec{\rho_A}) \rightarrow B(\vec{\rho_B})C(\vec{\rho_C}) \in P$ is an instantiated rule

Weighted CYK LCFRS parsing: Knuth's algorithm

add $SCAN$ results to $\mathcal{A}$
while $\mathcal{A} \neq \emptyset$
    remove best item $x : \mathcal{I}$ from $\mathcal{A}$
    add $x : \mathcal{I}$ to $\mathcal{C}$
    if $\mathcal{I}$ goal item
    then stop and output true
    else
        for all $y : \mathcal{I}'$ deduced from $x : \mathcal{I}$ and items in $\mathcal{C}$:
        if there is no z with $z : \mathcal{I}' \in \mathcal{C} \cup \mathcal{A}$
        then add $y : \mathcal{I}'$ to $\mathcal{A}$
        else if $z : \mathcal{I}' \in \mathcal{A}$ for some z
            then update weight of $\mathcal{I}'$ in A to $max(y, z)$

Weighted CYK LCFRS parsing: Example

A sample PLCFRS: $\{S, A, B\}$, terminals $\{a\}$, start symbol S and the input word aa (the number in brackets is the *log*):

$0.2 : S(X) \to A(X)$ (-0.7)
$0.7 : A(XY) \to T_a(X)A(Y)$ (-0.15)
$0.8 : B(ZX, Y) \to T_a(Z)B'(X, Y)$ (-0.1)
$0.2 : B(X, Y) \to T_a(X)T_a(Y)$ (-0.7)
$0.8 : S(XY) \to B(X, Y)$ (-0.1)
$0.3 : A(X) \to T_a(X)$ (-0.5)
$1 : B'(X, UY) \to B(X, Y)T_a(U)$ (0)
$1 : T_a(a) \to \varepsilon$ (0)

Chart	Agenda
	$0 : [T_a, \langle 0,1\rangle]$, $0 : [T_a, \langle 1,2\rangle]$
$0 : [T_a, \langle 0,1\rangle]$	$0 : [T_a, \langle 1,2\rangle]$, $-0.5 : [A, \langle 0,1\rangle]$
$0 : [T_a, \langle 0,1\rangle]$, $0 : [T_a, \langle 1,2\rangle]$	$-0.5 : [A, \langle 0,1\rangle]$, $-0.5 : [A, \langle 1,2\rangle]$, $-0.7 : [B, \langle 0,1\rangle, \langle 1,2\rangle]$
$0 : [T_a, \langle 0,1\rangle]$, $0 : [T_a, \langle 1,2\rangle]$ $-0.5 : [A, \langle 0,1\rangle]$	$-0.5 : [A, \langle 1,2\rangle]$, $-0.7 : [B, \langle 0,1\rangle, \langle 1,2\rangle]$, $-1.2 : [S, \langle 0,1\rangle]$
$0 : [T_a, \langle 0,1\rangle]$, $0 : [T_a, \langle 1,2\rangle]$ $-0.5 : [A, \langle 0,1\rangle]$, $-0.5 : [A, \langle 1,2\rangle]$	$-0.65 : [A, \langle 0,2\rangle]$, $-0.7 : [B, \langle 0,1\rangle, \langle 1,2\rangle]$, $-1.2 : [S, \langle 0,1\rangle]$, $-1.2 : [S, \langle 1,2\rangle]$
$0 : [T_a, \langle 0,1\rangle]$, $0 : [T_a, \langle 1,2\rangle]$ $-0.5 : [A, \langle 0,1\rangle]$, $-0.5 : [A, \langle 1,2\rangle]$ $-0.65 : [A, \langle 0,2\rangle]$	$-0.7 : [B, \langle 0,1\rangle, \langle 1,2\rangle]$, $-1.2 : [S, \langle 0,1\rangle]$, $-1.2 : [S, \langle 1,2\rangle]$, $-1.35 : [S, \langle 0,2\rangle]$
$0 : [T_a, \langle 0,1\rangle]$, $0 : [T_a, \langle 1,2\rangle]$ $-0.5 : [A, \langle 0,1\rangle]$, $-0.5 : [A, \langle 1,2\rangle]$ $-0.65 : [A, \langle 0,2\rangle]$, $-0.7 : [B, \langle 0,1\rangle, \langle 1,2\rangle]$	$-0.8 : [S, \langle 0,2\rangle]$, $-1.2 : [S, \langle 0,1\rangle]$, $-1.2 : [S, \langle 1,2\rangle]$

Weighted CYK LCFRS parsing: Complexity

Parsing a binarized LCFRS has complexity

$$\mathcal{O}(|w|^{3\phi})$$

where ϕ is the maximum number of spans covered by a non-terminal (fan-out).

Weighted CYK LCFRS parsing: Some issues

- The CYK-like parser for probabilistic LCFRS requires binarized grammar.
- Binarizing an LCFRS may increase the fan-out of PLCFRS
  $\Rightarrow$ increase in asymptotic complexity [vCB13]

References I

[CC98] Sharon A Caraballo and Eugene Charniak.
New figures of merit for best-first probabilistic chart parsing.
Computational Linguistics, 24(2):275–298, 1998.

[KM03] Dan Klein and Christopher D Manning.
A parsing: fast exact viterbi parse selection.
In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 40–47. Association for Computational Linguistics, 2003.

[KM13] Laura Kallmeyer and Wolfgang Maier.
Data-driven parsing using probabilistic linear context-free rewriting systems.
Computational Linguistics, 39(1):87–119, 2013.

[MS08] Wolfgang Maier and Anders Søgaard.
Treebanks and mild context-sensitivity.
Proceedings of Formal Grammar 2008, pages 61–76, 2008.

[Ned03] Mark-Jan Nederhof.
Weighted deductive parsing and knuth's algorithm.
Comput. Linguist., 29(1):135–143, March 2003.

[SBKU98] Wojciech Skut, Thorsten Brants, Brigitte Krenn, and Hans Uszkoreit.
A linguistically interpreted corpus of german newspaper text.
arXiv preprint cmp-lg/9807008, 1998.

[Sch90] R Scha.
Language theory and language technology; competence and performance (in dutch). in de kort, q. and leerdam, g., editors.
Computertoepassingen in de Neerlandistiek, 1990.

References II

[Smi03] George Smith.
A brief introduction to the tiger treebank.
Ms. Universität Potsdam, 2003.

[vCB13] Andreas van Cranenburgh and Rens Bod.
Discontinuous parsing with an efficient and accurate dop model.
In *Proceedings of the International Conference on Parsing Technologies (IWPT 2013)*. Citeseer, 2013.

[VCSS11] Andreas Van Cranenburgh, Remko Scha, and Federico Sangati.
Discontinuous data-oriented parsing: A mildly context-sensitive all-fragments grammar.
In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 34–44. Association for Computational Linguistics, 2011.