Parsing Beyond Context-Free Grammars: LCFRS Parsing

Laura Kallmeyer & Tatiana Bladier
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2018

Overview

1. **Ranges**
2. **CYK Parsing**
3. **Incremental Earley Parsing**
   - Filters

[Kal10]

Ranges (1)

- During parsing we have to link the terminals and variables in our LCFRS rules to portions of the input string.
- These can be characterized by their start and end positions.
- A *range* is an pair of indices $\langle i, j \rangle$ that characterizes the span of a component within the input and a range vector characterizes a tuple in the yield of a non-terminal.
- The range instantiation of a rule specifies the computation of an element from the lefthand side yield from elements of in the yields of the right-hand side non-terminals based on the corresponding range vectors.

Ranges (2)

Example: Rule $A(aXa, bYb) \rightarrow B(X)C(Y)$ and input string *abababcb*. We assume without loss of generality that our LCFRSs are monotone and ε-free. Furthermore, because of the linearity, the components of a tuple in the yield of an LCFRS non-terminal are necessarily non-overlapping. Then, given this input, we have the following possible instantiations for this rule:

$$
\begin{array}{ll}
A({}_0aba_3, {}_3bab_6) \rightarrow B({}_1b_2)C({}_4a_5) & A({}_0aba_3, {}_3babcb_8) \rightarrow B({}_1b_2)C({}_4abc_7)) \\
A({}_0aba_3, {}_5bcb_8) \rightarrow B({}_1b_2)C({}_6c_7) & A({}_0ababa_5, {}_5bcb_8) \rightarrow B({}_1bab_4)C({}_6c_7) \\
A({}_2aba_5, {}_5bcb_8) \rightarrow B({}_3b_4)C({}_6c_7) &
\end{array}
$$

Ranges (3)

Definition 1 (Range instantiation, [Bou00])

Let $G = (N, T, V, P, S)$ be a LCFRS, $w = t_1 \dots t_n \in T^n$ $(n \geq 0)$ and $r = A(\vec{\alpha}) \to A_1(\vec{\alpha_1}) \cdots A_m(\vec{\alpha_m}) \in P$ $(0 \leq m)$. A range instantiation of r wrt. w is a function $f : V \cup \{Eps_i \,|\, \vec{\alpha}(i) = \varepsilon\} \cup \{t' \,|\, t'$ an occurrence of some $t \in T$ in $\vec{\alpha}\} \to \{\langle i, j\rangle \,|\, 0 \leq i \leq j \leq n\}$ such that

a) for all occurrences t' of a $t \in T$ in $\vec{\alpha}$, $f(t') = \langle i-1, i\rangle$ for some i with $t_i = t$,

b) for all x, y adjacent in one of the $\vec{\alpha}(i)$ there are i, j, k with $f(x) = \langle i, j\rangle, f(y) = \langle j, k\rangle$; we define then $f(xy) = \langle i, k\rangle$,

c) for all $Eps \in \{Eps_i \,|\, \vec{\alpha}(i) = \varepsilon\}$, $f(Eps) = \langle j, j\rangle$ for some j; we define then for every ε-argument $\vec{\alpha}(i)$ that $f(\vec{\alpha}(i)) = f(Eps_i)$.

$A(f(\vec{\alpha})) \to A_1(f(\vec{\alpha_1})) \cdots A_m(f(\vec{\alpha_m}))$ with $f(\langle x_1, \dots, x_k\rangle) = \langle f(x_1), \dots, f(x_k)\rangle$ is then called an instantiated rule.

CYK Parsing (1)

First introduced in [SMFK91]; deduction-based definition in, e.g., [KM10].

Idea: Once all elements in the RHS of an instantiated rule have been found, complete the LHS.

- We start with the terminal symbols: whenever we can find a range instantiation of a rule with rhs ε, we conclude that this rule can be applied (*scan*).
- We parse bottom-up: whenever, for am instantiated rule, all elements in the rhs have been found, we conclude that this rule can be applied and the lhs of the instantiated rule is deduced (complete).
- Our input w is in the language iff S with range vector $\langle\langle 0, n\rangle\rangle$ is in the final set of results that we have deduced.

CYK Parsing (2)

Deduction rules:

Items $[A, \vec{\rho}]$ with $A \in N$, $\vec{\rho}$ is a $dim(A)$-dimensional range vector in w.

Axioms (scan): $\dfrac{}{[A, \vec{\rho}]}$ $A(\vec{\rho}) \rightarrow \varepsilon$ a range instantiated rule

Complete: $\dfrac{[A_1, \vec{\rho_1}], \dots, [A_m, \vec{\rho_m}]}{[A, \vec{\rho}]}$ $A(\vec{\rho}) \rightarrow A_1(\vec{\rho_1}), \dots, A_m(\vec{\rho_m})$ a range instantiated rule

Goal item: $[S, \langle\langle 0, n\rangle\rangle]$

CYK Parsing: Example (1)

Example: MCFG/LCFRS for the double copy language, input word: *ababab*

Rewriting rules:

$$S \rightarrow f_1[A] \qquad A \rightarrow f_2[A] \qquad A \rightarrow f_3[A] \qquad A \rightarrow f_4[\,] \qquad A \rightarrow f_5[\,]$$

Operations:

$$f_1[\langle X, Y, Z\rangle] = \langle XYZ\rangle \qquad f_4[\,] = \langle a, a, a\rangle$$
$$f_2[\langle X, Y, Z\rangle] = \langle aX, aY, aZ\rangle \qquad f_5[\,] = \langle b, b, b\rangle$$
$$f_3[\langle X, Y, Z\rangle] = \langle bX, bY, bZ\rangle$$

CYK Parsing: Example (2)

	Item	Rule
1	$[A, \langle\langle 0,1\rangle, \langle 2,3\rangle, \langle 4,5\rangle\rangle]$	axiom with $A \rightarrow f_4[\,]$
2	$[A, \langle\langle 0,1\rangle, \langle 4,5\rangle, \langle 2,3\rangle\rangle]$	axiom with $A \rightarrow f_4[\,]$
3	$[A, \langle\langle 2,3\rangle, \langle 0,1\rangle, \langle 4,5\rangle\rangle]$	axiom with $A \rightarrow f_4[\,]$
	...	
4	$[A, \langle\langle 1,2\rangle, \langle 3,4\rangle, \langle 5,6\rangle\rangle]$	axiom with $A \rightarrow f_5[\,]$
5	$[A, \langle\langle 1,2\rangle, \langle 5,6\rangle, \langle 3,4\rangle\rangle]$	axiom with $A \rightarrow f_5[\,]$
	...	
6	$[A, \langle\langle 0,2\rangle, \langle 2,4\rangle, \langle 4,6\rangle\rangle]$	complete, with 4 and $A \rightarrow f_2[A]$
7	$[A, \langle\langle 0,2\rangle, \langle 4,6\rangle, \langle 2,4\rangle\rangle]$	complete, with 5 and $A \rightarrow f_2[A]$
	...	
8	$[S, \langle\langle 0,6\rangle\rangle]$	complete, with 6 and $S \rightarrow f_1[A]$

CYK Parsing with binarized LCFRS

Deduction rules for binarized ε-free grammars where, without loss of generality, either the lhs contains a single terminal and the rhs is ε or the rule contains only variables:

Items and goal as before.

Scan: $\dfrac{}{[A, \langle\langle i, i+1\rangle\rangle]}$ $A(w_{i+1}) \to \varepsilon \in P$

Unary: $\dfrac{[B, \vec{\rho}]}{[A, \vec{\rho}]}$ $A(\vec{\alpha}) \to B(\vec{\alpha}) \in P$

Binary: $\dfrac{[B, \vec{\rho_B}], [C, \vec{\rho_C}]}{[A, \vec{\rho_A}]}$ $A(\vec{\rho_A}) \to B(\vec{\rho_B})C(\vec{\rho_C})$ is a range instantiated rule

CYK Parsing: Another example (1)

LCFRS G in sRCG format and the input word *aabbacbbac*.

$G = \langle \{S, A, B, C\}, \{a, b, c\}, \{U, V, W, X, Y, Z\}, P, S \rangle$, where

$$
\begin{aligned}
P = \{ \quad & S(VYWZX) \rightarrow A(V, W, X)B(Y, Z), \\
& A(a, a, a) \rightarrow \epsilon, \\
& A(XU, YV, ZW) \rightarrow A(X, Y, Z)C(U, V, W), \\
& B(b, b) \rightarrow \epsilon \\
& B(XV, WY) \rightarrow B(X, Y)B(V, W) \\
& C(a, c, c) \rightarrow \epsilon \qquad \}
\end{aligned}
$$

CYK Parsing: Another example (2)

	Item	Rule
1	$[A, \langle\langle 0,1\rangle, \langle 4,5\rangle, \langle 8,9\rangle\rangle]$	scan
2	$[A, \langle\langle 0,1\rangle, \langle 1,2\rangle, \langle 4,5\rangle\rangle]$	scan
	...	
3	$[B, \langle\langle 2,3\rangle, \langle 7,8\rangle\rangle]$	scan
4	$[B, \langle\langle 3,4\rangle, \langle 6,7\rangle\rangle]$	scan
5	$[B, \langle\langle 2,3\rangle, \langle 3,4\rangle\rangle]$	scan
	...	
6	$[C, \langle\langle 1,2\rangle, \langle 5,6\rangle, \langle 9,10\rangle\rangle]$	scan
7	$[C, \langle\langle 0,1\rangle, \langle 5,6\rangle, \langle 9,10\rangle\rangle]$	scan
8	$[A, \langle\langle 0,2\rangle, \langle 4,6\rangle, \langle 8,10\rangle\rangle]$	complete, with 1 and 6
9	$[B, \langle\langle 2,4\rangle, \langle 6,8\rangle\rangle]$	complete with 3 and 4
10	$[S, \langle\langle 0,10\rangle\rangle]$	complete with 8 and 9

CYK Parsing: Complexity

Complexity of CYK parsing with binarized LCFRSs:

We have to consider the maximal number of possible applications of the complete rule.

Binary: $\dfrac{[B, \vec{\rho_B}], [C, \vec{\rho_C}]}{[A, \vec{\rho_A}]}$ $A(\vec{\rho_A}) \rightarrow B(\vec{\rho_B})C(\vec{\rho_C})$ is a range instantiated rule

If k is the maximal fan-out in the LCFRS, we have maximal $2k$ range boundaries in each of the antecedent items of this rule. For variables X_1, X_2 being in the same lhs side argument of the rule, X_1 left of X_2 and no other variables in between, the right boundary of X_1 is the left boundary of X_2. In the worst case, A, B, C all have fan-out k and each lhs argument contains two variables. This gives $3k$ independent range boundaries and consequently a time complexity of $\mathcal{O}(n^{3k})$ for the entire algorithm.

Incremental Earley Parsing

Strategy:

- Process LHS arguments incrementally, starting from an S-rule
- Whenever we reach a variable, move into rule of correponding rhs non-terminal (**predict** or **resume**).
- Whenever we reach the end of an argument, **suspend** the rule and move into calling parent rule.
- Whenever we reach the end of the last argument **convert** item into a passive one and **complete** parent item.

This parser is described in [KM09] and inspired by the Thread Automata in [Vil02]

Incremental Earley Parsing: Items

Passive items: $[A, \vec{\rho}]$ where A is a non-terminal of fan-out k and $\vec{\rho}$ is a range vector of fan-out k

Active items:

$$[A(\vec{\phi}) \to A_1(\vec{\phi_1}) \dots A_m(\vec{\phi_m}), pos, \langle i, j \rangle, \vec{\rho}]$$

where

- $A(\vec{\phi}) \to A_1(\vec{\phi_1}) \dots A_m(\vec{\phi_m}) \in P$;
- $pos \in \{0, \dots, n\}$: We have reached input position pos;
- $\langle i, j \rangle \in \mathbb{N}^2$: We have reached the jth element of ith argument (dot position);
- $\vec{\rho}$ is a range vector containing variable and terminal bindings. All elements are initialized to "?", an initialized vector is called $\vec{\rho}_{init}$.

Incremental Earley Parsing: Example (1)

$$S(X_1X_2) \longrightarrow A(X_1, X_2) \quad A(aX_1, bX_2) \longrightarrow A(X_1, X_2) \quad A(a, b) \longrightarrow \varepsilon$$

Parsing trace for input $w = aabb$:

	pos.	item and dot position	$\vec{\rho}$ (bindings)	rule
1	0	$S(\bullet X_1X_2) \longrightarrow A(X_1, X_2)$	$(?, ?)$	axiom
2	0	$A(\bullet aX_1, bX_2) \longrightarrow A(X_1, X_2)$	$(?, ?, ?, ?)$	predict, 1
3	0	$A(\bullet a, b) \longrightarrow \varepsilon$	$(?, ?)$	predict, 1
4	1	$A(a \bullet X_1, bX_2) \longrightarrow A(X_1, X_2)$	$(\langle 0, 1\rangle, ?, ?, ?)$	scan, 2
5	1	$A(a\bullet, b) \longrightarrow \varepsilon$	$(\langle 0, 1\rangle, ?)$	scan, 3
6	1	$A(\bullet aX_1, bX_2) \longrightarrow A(X_1, X_2)$	$(?, ?, ?, ?)$	predict, 4
7	1	$A(\bullet a, b) \longrightarrow \varepsilon$	$(?, ?)$	predict 4
8	1	$S(X_1 \bullet X_2) \longrightarrow A(X_1, X_2)$	$(\langle 0, 1\rangle, ?)$	susp. 5, 1
9	1	$A(a, \bullet b) \longrightarrow \varepsilon$	$(\langle 0, 1\rangle, ?)$	resume 5, 8

Incremental Earley Parsing: Example (2)

	pos.	item and dot position	$\vec{\rho}$ (bindings)	rule
10	2	$A(a \bullet X_1, bX_2) \longrightarrow A(X_1, X_2)$	$(\langle 1,2\rangle, ?, ?, ?)$	scan 6
11	2	$A(a\bullet, b) \longrightarrow \varepsilon$	$(\langle 1,2\rangle, ?)$	scan 7
12	2	$A(\bullet aX_1, bX_2) \longrightarrow A(X_1, X_2)$	$(?, ?, ?, ?)$	predict 10
13	2	$A(\bullet a, b) \longrightarrow \varepsilon$	$(?, ?)$	predict 10
14	2	$A(aX_1\bullet, bX_2) \longrightarrow A(X_1, X_2)$	$(\langle 0,1\rangle, \langle 1,2\rangle, ?, ?)$	susp. 11, 4
15	2	$S(X_1 \bullet X_2) \longrightarrow A(X_1, X_2)$	$(\langle 0,2\rangle, ?)$	susp. 14, 1
16	2	$A(aX_1, \bullet bX_2) \longrightarrow A(X_1, X_2)$	$(\langle 0,1\rangle, \langle 1,2\rangle, ?, ?)$	resume 14, 15
17	3	$A(aX_1, b \bullet X_2) \longrightarrow A(X_1, X_2)$	$(\langle 0,1\rangle, \langle 1,2\rangle, \langle 2,3\rangle, ?)$	scan 16
18	3	$A(a, \bullet b) \longrightarrow \varepsilon$	$(\langle 1,2\rangle, ?)$	resume 11, 17

Incremental Earley Parsing: Example (3)

	pos.	item and dot position	$\vec{\rho}$ (bindings)	rule
19	4	$A(a, b\bullet) \longrightarrow \varepsilon$	$(\langle 1,2\rangle, \langle 3,4\rangle)$	scan 18
20	4	$A(\langle 1,2\rangle, \langle 3,4\rangle)$		convert 19
21	4	$A(aX_1, bX_2\bullet) \longrightarrow A(X_1, X_2)$	$(\langle 0,1\rangle, \langle 1,2\rangle, \langle 2,3\rangle, \langle 3,4\rangle)$	compl. 17, 20
22	4	$A(\langle 0,2\rangle, \langle 2,4\rangle)$		convert 21
23	4	$S(X_1X_2\bullet) \longrightarrow A(X_1, X_2)$	$(\langle 0,2\rangle, \langle 2,4\rangle)$	compl. 15, 22
24	4	$S(\langle 0,4\rangle)$		convert 23

Incremental Earley Parsing: Deduction Rules

- Notation:
  - $\vec{\rho}(X)$: range bound to variable X.
  - $\vec{\rho}(\langle i, j \rangle)$: range bound to jth element of ith argument on LHS.
- Applying a range vector $\vec{\rho}$ containing variable bindings for given rule c to the argument vector of the lefthand side of c means mapping the ith element in the arguments to $\vec{\rho}(i)$ and concatenating adjacent ranges. The result is defined iff every argument is thereby mapped to a range.

Incremental Earley Parsing: Initialize, Goal item

Initialize: $\dfrac{}{[S(\vec{\phi}) \to \vec{\Phi}, 0, \langle 1, 0\rangle, \vec{\rho}_{init}]} \quad S(\vec{\phi}) \to \vec{\Phi} \in P$

Goal Item: $[S(\vec{\phi}) \to \vec{\Phi}, n, \langle 1, j\rangle, \psi]$ with $|\vec{\phi}(1)| = j$ (i.e., dot at the end of lhs argument).

Incremental Earley Parsing: Scan

If next symbol after dot is next terminal in input, scan it.

Scan: $$\frac{[A(\vec{\phi}) \to \vec{\Phi}, pos, \langle i, j \rangle, \vec{\rho}]}{[A(\vec{\phi}) \to \vec{\Phi}, pos + 1, \langle i, j + 1 \rangle, \vec{\rho}']} \quad \vec{\phi}(i, j + 1) = w_{pos+1}$$

where $\vec{\rho}'$ is $\vec{\rho}$ updated with $\vec{\rho}(\langle i, j + 1 \rangle) = \langle pos, pos + 1 \rangle$.

Incremental Earley Parsing: Predict

Whenever our dot is left of a variable that is the first argument of some rhs non-terminal B, we predict new B-rules:

Predict: $$\frac{[A(\vec{\phi}) \rightarrow \dots B(X, \dots) \dots, pos, \langle i, j\rangle, \vec{\rho}_A]}{[B(\vec{\psi}) \rightarrow \vec{\Psi}, pos, \langle 1, 0\rangle, \vec{\rho}_{init}]}$$

where $\vec{\phi}(i, j+1) = X, B(\vec{\psi}) \rightarrow \vec{\Psi} \in P$

Incremental Earley Parsing: Suspend

Suspend:

$$\frac{[B(\vec{\psi}) \rightarrow \vec{\Psi}, pos', \langle i, j \rangle, \vec{\rho}_B], [A(\vec{\phi}) \rightarrow \ldots B(\vec{\xi}) \ldots, pos, \langle k, l \rangle, \vec{\rho}_A]}{[A(\vec{\phi}) \rightarrow \ldots B(\vec{\xi}) \ldots, pos', \langle k, l+1 \rangle, \vec{\rho}]}$$

where

- the dot in the antecedent A-item precedes the variable $\vec{\xi}(i)$,
- $|\vec{\psi}(i)| = j$ (ith argument has length j, i.e., is completely processed),
- $|\vec{\psi}| < i$ (ith argument is not the last argument of B),
- $\vec{\rho}_B(\vec{\psi}(i)) = \langle pos, pos' \rangle$
- and for all $1 \leq m < i$: $\vec{\rho}_B(\vec{\psi}(m)) = \vec{\rho}_A(\vec{\xi}(m))$.

$\vec{\rho}$ is $\vec{\rho}_A$ updated with $\vec{\rho}_A(\vec{\xi}(i)) = \langle pos, pos' \rangle$.

Incremental Earley Parsing: Convert

Whenever we arrive at the end of the last argument, we convert the item into a passive one:

Convert:

$$\frac{[B(\vec{\psi}) \to \vec{\Psi}, pos, \langle i, j\rangle, \vec{\rho}_B]}{[B, \rho]} \quad |\vec{\psi}(i)| = j, |\vec{\psi}| = i, \vec{\rho}_B(\vec{\psi}) = \rho$$

Incremental Earley Parsing: Complete

Whenever we have a passive B item we can use it to move the dot over the variable of the last argument of B in a parent A-rule:

Complete: $$\frac{[B, \vec{\rho}_B], [A(\vec{\phi}) \rightarrow \ldots B(\vec{\xi}) \ldots, pos, \langle k, l \rangle, \vec{\rho}_A]}{[A(\vec{\phi}) \rightarrow \ldots B(\vec{\xi}) \ldots, pos', \langle k, l+1 \rangle, \vec{\rho}]}$$ where

- the dot in the antecedent A-item precedes the variable $\vec{\xi}(|\vec{\rho}_B|)$,
- the last range in $\vec{\rho}_B$ is $\langle pos, pos' \rangle$,
- and for all $1 \leq m < |\vec{\rho}_B|$: $\vec{\rho}_B(m) = \vec{\rho}_A(\vec{\xi}(m))$.

$\vec{\rho}$ is $\vec{\rho}_A$ updated with $\vec{\rho}_A(\vec{\xi}(|\vec{\rho}_B|)) = \langle pos, pos' \rangle$.

Incremental Earley Parsing: Resume

Whenever we are left of a variable that is not the first argument of one of the rhs non-terminals, we resume the rule of the rhs non-terminal.

$$\textbf{Resume}: \frac{\begin{array}{c}[A(\vec{\phi}) \to \dots B(\vec{\xi}) \dots, pos, \langle i, j\rangle, \vec{\rho}_A], \\ [B(\vec{\psi}) \to \vec{\Psi}, pos', \langle k-1, l\rangle, \vec{\rho}_B]\end{array}}{[B(\vec{\psi}) \to \vec{\Psi}, pos, \langle k, 0\rangle, \vec{\rho}_B]}$$

where

- $\vec{\phi}(i, j+1) = \vec{\xi}(k), k > 1$ (the next element is a variable that is the kth element in $\vec{\xi}$, i.e., the kth argument of B),
- $|\vec{\psi}(k-1)| = l$, and
- $\vec{\rho}_A(\vec{\xi}(m)) = \vec{\rho}_B(\vec{\psi}(m))$ for all $1 \leq m \leq k-1$.

Incremental Earley Parsing: Filters

- Filters can be applied to decrease the number of items in the chart
- A filter is an additional condition on the form of items.
- E.g., in a ε-free grammar, the number of variables in the part of the lefthand side arguments of a rule that has not been processed yet must be lower or equal to the length of the remaining input.

We will discuss in the following some filters that are particularly useful when dealing with natural languages.

Incr. Earley Parsing: Remaining Input Length Filter

- In ε-free grammars each variable must cover at least one input symbol.
- i input symbols left implies no prediction of a clause with more than i variables or terminals on LHS since no instantiation is possible
- Condition on active items, can be applied with predict, resume, suspend and complete

The length of the remaining input must be $\geq$ the number of variables and terminal occurrences to the right of the dot in the lefthand side of the clause, i.e.
An item $[A(\vec{\phi}) \rightarrow A_1(\vec{\phi_1}) \dots A_m(\vec{\phi_m}), pos, \langle i, j \rangle, \vec{\rho}]$ satisfies the **length filter** iff

$$(n - pos) \geq (|\vec{\phi}(i)| - j) + \Sigma_{k=i+1}^{dim(A)} |\vec{\phi}(k)|$$

Incremental Earley Parsing: Preterminal Filter (1)

- Check for the presence of (pre)terminals in the predicted part of a clause (the part to the right of the dot) in the remaining input, and
- check that terminals appear in the predicted order and that distance between two of them is at least the number of variables/terminals in between.

continued...

Incremental Earley Parsing: Preterminal Filter (2)

In other words, an active item $[A(\vec{\phi}) \rightarrow A_1(\vec{\phi_1}) \dots A_m(\vec{\phi_m}), pos, \langle i, j \rangle, \vec{\rho}]$ satisfies the **preterminal filter** iff we can find an injective mapping $f_T : \mathrm{Term} = \{\langle k, l \rangle \,|\, \vec{\phi}(k, l) \in T$ and either $k > i$ or $(k = i$ and $l > j)\} \rightarrow \{pos + 1, \dots, n\}$ such that

1. $w_{f_T(\langle k,l \rangle)} = \vec{\phi}(k, l)$ for all $\langle k, l \rangle \in Term$;
2. for all $\langle k_1, l_1 \rangle, \langle k_2, l_2 \rangle \in Term$ with $k_1 = k_2$ and $l_1 < l_2$: $f_T(\langle k_2, l_2 \rangle) \geq f_T(\langle k_1, l_1 \rangle) + (l_2 - l_1)$;
3. for all $\langle k_1, l_1 \rangle, \langle k_2, l_2 \rangle \in Term$ with $k_1 < k_2$: $f_T(\langle k_2, l_2 \rangle) \geq f_T(\langle k_1, l_1 \rangle) + (|\vec{\phi}(k_1)| - l_1) + \Sigma_{k=k_1+1}^{k_2-1} |\vec{\phi}(k)| + l_2$.

Checking this filtering condition amounts to a linear traversal of the part of the lefthand side of the clause that is to the right of the dot. We start with index $i = pos + 1$, for every variable or gap we increment i by 1. For every terminal a, we search the next a in the input, starting at position i. If it occurs at position j, then we set $i = j$ and continue our traversal of the remaining parts of the lefthand side of the clause.

References I

[Bou00] Pierre Boullier.
Range Concatenation Grammars.
In *Proceedings of the Sixth International Workshop on Parsing Technologies (IWPT2000)*, pages 53–64, Trento, Italy, February 2000.

[Kal10] Laura Kallmeyer.
Parsing Beyond Context-Free Grammars.
Cognitive Technologies. Springer, Heidelberg, 2010.

[KM09] Laura Kallmeyer and Wolfgang Maier.
An incremental Earley parser for simple Range Concatenation Grammar.
In *Proceedings of IWPT 2009*, 2009.

[KM10] Laura Kallmeyer and Wolfgang Maier.
Data-driven parsing with probabilistic Linear Context-Free Rewriting Systems.
In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, Beijing, China, 2010.

[SMFK91] Hiroyuki Seki, Takahashi Matsumura, Mamoru Fujii, and Tadao Kasami.
On multiple context-free grammars.
Theoretical Computer Science, 88(2):191–229, 1991.

[Vil02] Éric Villemonte de La Clergerie.
Parsing mildly context-sensitive languages with thread automata.
In *Proc. of COLING'02*, August 2002.