An Earley Parsing Algorithm for Range Concatenation Grammars

Laura Kallmeyer
SFB 441
Universität Tübingen
72074 Tübingen, Germany
lk@sfs.uni-tuebingen.de

Wolfgang Maier
SFB 441
Universität Tübingen
72074 Tübingen, Germany
wo.maier@uni-tuebingen.de

Yannick Parmentier
CNRS - LORIA
Nancy Université
54506 Vandœuvre, France
parmenti@loria.fr

Abstract

We present a CYK and an Earley-style algorithm for parsing Range Concatenation Grammar (RCG), using the deductive parsing framework. The characteristic property of the Earley parser is that we use a technique of range boundary constraint propagation to compute the yields of non-terminals as late as possible. Experiments show that, compared to previous approaches, the constraint propagation helps to considerably decrease the number of items in the chart.

1 Introduction

RCGs (Boullier, 2000) have recently received a growing interest in natural language processing (Søgaard, 2008; Sagot, 2005; Kallmeyer et al., 2008; Maier and Søgaard, 2008). RCGs generate exactly the class of languages parsable in deterministic polynomial time (Bertsch and Nederhof, 2001). They are in particular more powerful than linear context-free rewriting systems (LCFRS) (Vijay-Shanker et al., 1987). LCFRS is unable to describe certain natural language phenomena that RCGs actually can deal with. One example are long-distance scrambling phenomena (Becker et al., 1991; Becker et al., 1992). Other examples are non-semilinear constructions such as case stacking in Old Georgian (Michaelis and Kracht, 1996) and Chinese number names (Radzinski, 1991). Boullier (1999) shows that RCGs can describe the permutations occurring with scrambling and the construction of Chinese number names.

Parsing algorithms for RCG have been introduced by Boullier (2000), who presents a directional top-down parsing algorithm using pseudocode, and Barthélemy et al. (2001), who add an oracle to Boullier's algorithm. The more restricted class of LCFRS has received more attention concerning parsing (Villemonte de la Clergerie, 2002; Burden and Ljunglöf, 2005). This article proposes new CYK and Earley parsers for RCG, formulating them in the framework of parsing as deduction (Shieber et al., 1995). The second section introduces necessary definitions. Section 3 presents a CYK-style algorithm and Section 4 extends this with an Earley-style prediction.

2 Preliminaries

The rules (*clauses*) of RCGs[1] rewrite predicates ranging over parts of the input by other predicates. E.g., a clause $S(aXb) \rightarrow S(X)$ signifies that S is true for a part of the input if this part starts with an a, ends with a b, and if, furthermore, S is also true for the part between a and b.

Definition 1. *A* RCG $G = \langle N, T, V, P, S\rangle$ *consists of a) a finite set of predicates* N *with an arity function* dim*:* $N \rightarrow \mathbb{N} \setminus \{0\}$ *where* $S \in N$ *is the start predicate with* $dim(S) = 1$*, b) disjoint finite sets of terminals* T *and variables* V*, c) a finite set* P *of clauses* $\psi_0 \rightarrow \psi_1 \dots \psi_m$*, where* $m \geq 0$ *and each of the* $\psi_i, 0 \leq i \leq m$*, is a predicate of the form* $A_i(\alpha_1, \dots, \alpha_{dim(A_i)})$ *with* $A_i \in N$ *and* $\alpha_j \in (T \cup V)^*$ *for* $1 \leq j \leq dim(A_i)$*.*

Central to RCGs is the notion of ranges on strings.

Definition 2. *For every* $w = w_1 \dots w_n$ *with* $w_i \in T$ $(1 \leq i \leq n)$*, we define a)* $Pos(w) = \{0, \dots, n\}$*. b)* $\langle l, r\rangle \in Pos(w) \times Pos(w)$ *with* $l \leq r$ *is a* range *in* w*. Its* yield $\langle l, r\rangle(w)$ *is the substring* $w_{l+1} \dots w_r$*. c) For two ranges* $\rho_1 = \langle l_1, r_1\rangle, \rho_2 = \langle l_2, r_2\rangle$*: if* $r_1 = l_2$*, then* $\rho_1 \cdot \rho_2 = \langle l_1, r_2\rangle$*; otherwise* $\rho_1 \cdot \rho_2$ *is undefined. d) A vector* $\phi = (\langle x_1, y_1\rangle, \dots, \langle x_k, y_k\rangle)$ *is a* range vector *of dimension* k *in* w *if* $\langle x_i, y_i\rangle$ *is a range in* w *for* $1 \leq i \leq k$*.* $\phi(i).l$ *(resp.* $\phi(i).r$*) denotes then the*

[1] In this paper, by RCG, we always mean *positive RCG*, see Boullier (2000) for details.

first (resp. second) component of the ith element of ϕ, that is x_i (resp. y_i).

In order to instantiate a clause of the grammar, we need to find ranges for all variables in the clause and for all occurrences of terminals. For convenience, we assume the variables in a clause and the occurrences of terminals to be equipped with distinct subscript indices, starting with 1 and ordered from left to right (where for variables, only the first occurrence is relevant for this order). We introduce a function $\Upsilon : P \to \mathbb{N}$ that gives the maximal index in a clause, and we define $\Upsilon(c, x)$ for a given clause c and x a variable or an occurrence of a terminal as the index of x in c.

Definition 3. *An* instantiation *of a $c \in P$ with $\Upsilon(c) = j$ w.r.t. to some string w is given by a range vector ϕ of dimension j. Applying ϕ to a predicate $A(\vec{\alpha})$ in c maps all occurrences of $x \in (T \cup V)$ with $\Upsilon(c, x) = i$ in $\vec{\alpha}$ to $\phi(i)$. If the result is defined (i.e., the images of adjacent variables can be concatenated), it is called an* instantiated predicate *and the result of applying ϕ to all predicates in c, if defined, is called an* instantiated clause.

We also introduce range constraint vectors, vectors of pairs of range boundary variables together with a set of constraints on these variables.

Definition 4. *Let $V_r = \{r_1, r_2, \dots\}$ be a set of range boundary variables. A* range constraint vector *of dimension k is a pair $\langle \vec{\rho}, C \rangle$ where a) $\vec{\rho} \in (V_r^2)^k$; we define $V_r(\vec{\rho})$ as the set of range boundary variables occurring in $\vec{\rho}$. b) C is a set of constraints c_r that have one of the following forms: $r_1 = r_2$, $k = r_1$, $r_1 + k = r_2$, $k \leq r_1$, $r_1 \leq k$, $r_1 \leq r_2$ or $r_1 + k \leq r_2$ for $r_1, r_2 \in V_r(\vec{\rho})$ and $k \in \mathbb{N}$.*

We say that a range vector ϕ satisfies a range constraint vector $\langle \rho, C \rangle$ iff ϕ and ρ are of the same dimension k and there is a function $f : V_r \to \mathbb{N}$ that maps $\rho(i).l$ to $\phi(i).l$ and $\rho(i).r$ to $\phi(i).r$ for all $1 \leq i \leq k$ such that all constraints in C are satisfied. Furthermore, we say that a range constraint vector $\langle \rho, C \rangle$ is satisfiable iff there exists a range vector ϕ that satisfies it.

Definition 5. *For every clause c, we define its* range constraint vector $\langle \rho, C \rangle$ *w.r.t. a w with $|w| = n$ as follows: a) ρ has dimension $\Upsilon(c)$ and all range boundary variables in ρ are pairwise different. b) For all $\langle r_1, r_2 \rangle \in \rho$: $0 \leq r_1$, $r_1 \leq r_2$, $r_2 \leq n \in C$. For all occurrences x of terminals in c with $i = \Upsilon(c, x)$: $\rho(i).l + 1 = \rho(i).r \in C$. For all x, y that are variables or occurrences of terminals in c such that xy is a substring of one of the arguments in c: $\rho(\Upsilon(c, x)).r = \rho(\Upsilon(c, y)).l \in C$. These are all constraints in C.*

The range constraint vector of a clause c captures all information about boundaries forming a range, ranges containing only a single terminal, and adjacent variables/terminal occurrences in c.

An RCG derivation consists of rewriting instantiated predicates applying instantiated clauses, i.e. in every derivation step $\Gamma_1 \Rightarrow_w \Gamma_2$, we replace the lefthand side of an instantiated clause with its righthand side (w.r.t. a word w). The *language* of an RCG G is the set of strings that can be reduced to the empty word: $L(G) = \{w \mid S(\langle 0, |w| \rangle) \stackrel{+}{\Rightarrow}_{G,w} \varepsilon\}$.

The expressive power of RCG lies beyond mild context-sensitivity. As an example, consider the RCG from Fig. 3 that generates a language that is not semilinear.

For simplicity, we assume in the following without loss of generality that empty arguments (ε) occur only in clauses whose righthand sides are empty.[2]

3 Directional Bottom-Up Chart Parsing

In our directional CYK algorithm, we move a dot through the righthand side of a clause. We therefore have *passive* items $[A, \phi]$ where A is a predicate and ϕ a range vector of dimension $dim(A)$ and *active* items. In the latter, while traversing the righthand side of the clause, we keep a record of the left and right boundaries already found for variables and terminal occurrences. This is achieved by subsequently enriching the range constraint vector of the clause. Active items have the form $[A(\vec{x}) \to \Phi \bullet \Psi, \langle \rho, C \rangle]$ with $A(\vec{x}) \to \Phi\Psi$ a clause, $\Phi\Psi \neq \varepsilon$, $\Upsilon(A(\vec{x} \to \Phi\Psi)) = j$ and $\langle \rho, C \rangle$ a range constraint vector of dimension j. We require that $\langle \rho, C \rangle$ be satisfiable.[3]

[2] Any RCG can be easily transformed into an RCG satisfying this condition: Introduce a new unary predicate Eps with a clause $Eps(\varepsilon) \to \varepsilon$. Then, for every clause c with righthand side not ε, replace every argument ε that occurs in c with a new variable X (each time a distinct one) and add the predicate $Eps(X)$ to the righthand side of c.

[3] Items that are distinguished from each other only by a bijection of the range variables are considered equivalent. I.e., if the application of a rule yields a new item such that an equivalent one has already been generated, this new one is not added to the set of partial results.

Scan: $\dfrac{}{[A,\phi]}$ $A(\vec{x}) \to \varepsilon \in P$ with instantiation ψ such that $\psi(A(\vec{x})) = A(\phi)$

Initialize: $\dfrac{}{[A(\vec{x}) \to \bullet\Phi, \langle\rho, C\rangle]}$ $A(\vec{x}) \to \Phi \in P$ with range constraint vector $\langle\rho, C\rangle, \Phi \neq \varepsilon$

Complete:

$$\frac{[B,\phi_B],\ [A(\vec{x}) \to \Phi \bullet B(x_1...y_1,...,x_k...y_k)\Psi, \langle\rho, C\rangle]}{[A(\vec{x}) \to \Phi B(x_1...y_1,...,x_k...y_k) \bullet \Psi, \langle\rho, C'\rangle]}$$

where $C' = C \cup \{\phi_B(j).l = \rho(\Upsilon(x_j)).l,\ \phi_B(j).r = \rho(\Upsilon(y_j)).r \mid 1 \leq j \leq k\}$.

Convert: $\dfrac{[A(\vec{x}) \to \Psi\bullet, \langle\rho, C\rangle]}{[A,\phi]}$ $A(\vec{x}) \to \Psi \in P$ with an instantiation ψ that satisfies $\langle\rho, C\rangle$, $\psi(A(\vec{x})) = A(\phi)$

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

Figure 1: CYK deduction rules

The deduction rules are shown in Fig. 1. The first rule scans the yields of terminating clauses. **Initialize** introduces clauses with the dot on the left of the righthand side. **Complete** moves the dot over a predicate provided a corresponding passive item has been found. **Convert** turns an active item with the dot at the end into a passive item.

4 The Earley Algorithm

We now add top-down prediction to our algorithm. Active items are as above. Passive items have an additional flag p or c depending on whether the item is predicted or completed, i.e., they either have the form $[A, \langle\rho, C\rangle, p]$ where $\langle\rho, C\rangle$ is a range constraint vector of dimension $dim(A)$, or the form $[A, \phi, c]$ where ϕ is a range vector of dimension $dim(A)$.

Initialize: $\dfrac{}{[S, \langle(\langle r_1, r_2\rangle), \{0 = r_1, n = r_2\}\rangle, p]}$

Predict-rule:

$$\frac{[A, \langle\rho, C\rangle, p]}{[A(x_1 \dots y_1, \dots, x_k \dots y_k) \to \bullet\Psi, \langle\rho', C'\rangle]}$$

where $\langle\rho', C'\rangle$ is obtained from the range constraint vector of the clause $A(x_1 \dots y_1, \dots, x_k \dots y_k) \to \Psi$ by taking all constraints from C, mapping all $\rho(i).l$ to $\rho'(\Upsilon(x_i)).l$ and all $\rho(i).r$ to $\rho'(\Upsilon(y_i)).r$, and then adding the resulting constraints to the range constraint vector of the clause.

Predict-pred:

$$\frac{[A(...) \to \Phi \bullet B(x_1...y_1,...,x_k...y_k)\Psi, \langle\rho, C\rangle]}{[B, \langle\rho', C'\rangle, p]}$$

where $\rho'(i).l = \rho(\Upsilon(x_i)).l$, $\rho'(i).r = \rho(\Upsilon(y_i)).r$ for all $1 \leq i \leq k$ and $C' = \{c \mid c \in C, c$ contains only range variables from $\rho'\}$.

Scan: $\dfrac{[A, \langle\rho, C\rangle, p]}{[A, \phi, c]}$ $A(\vec{x}) \to \varepsilon \in P$ with an instantiation ψ satisfying $\langle\rho, C\rangle$ such that $\psi(A(\vec{x})) = A(\phi)$

Figure 2: Earley deduction rules

The deduction rules are listed in Fig. 2. The axiom is the prediction of an S ranging over the entire input (**initialize**). We have two predict operations: **Predict-rule** predicts active items with the dot on the left of the righthand side, for a given predicted passive item. **Predict-pred** predicts a passive item for the predicate following the dot in an active item. **Scan** is applied whenever a predicted predicate can be derived by an ε-clause. The rules **complete** and **convert** are the ones from the CYK algorithm except that we add flags c to the passive items occurring in these rules. The **goal** is again $[S, (\langle 0, n\rangle), c]$.

To understand how this algorithm works, consider the example in Fig. 3. The crucial property of this algorithm, in contrast to previous approaches, is the dynamic updating of a set of constraints on range boundaries. We can leave range boundaries unspecified and compute their values in a more incremental fashion instead of guessing all ranges of a clause at once at prediction.[4]

For evaluation, we have implemented a directional top-down algorithm where range boundaries are guessed at prediction (this is essentially the algorithm described in Boullier (2000)), and the new Earley-style algorithm. The algorithms were tested on different words of the language $L = \{a^{2^n} \mid n \leq 0\}$. Table 1 shows the number of generated items.

Word	Earley	TD	Word	Earley	TD
a^2	15	21	a^{16}	100	539
a^4	30	55	a^{30}	155	1666
a^8	55	164	a^{32}	185	1894
a^9	59	199	a^{64}	350	6969

Table 1: Items generated by both algorithms

Clearly, range boundary constraint propagation increases the amount of information transported in single items and thereby decreases considerably the number of generated items.

5 Conclusion and future work

We have presented a new CYK and Earley parsing algorithms for the full class of RCG. The crucial difference between previously proposed top-down RCG parsers and the new Earley-style algorithm is that while the former compute all clause instantiations during **predict** operations, the latter

[4] Of course, the use of constraints makes comparisons between items more complex and more expensive which means that for an efficient implementation, an integer-based representation of the constraints and adequate techniques for constraint solving are required.

Grammar for $\{a^{2^n} \mid n > 0\}$: $S(XY) \rightarrow S(X)eq(X,Y)$, $S(a_1) \rightarrow \varepsilon$, $eq(a_1X, a_2Y) \rightarrow eq(X,Y)$, $eq(a_1, a_2) \rightarrow \varepsilon$
Parsing trace for $w = aa$:

	Item	Rule
1	$[S, \langle(\langle r_1, r_2\rangle), \{0 = r_1, r_1 \leq r_2, 2 = r_2\}\rangle, p]$	initialize
2	$[S(XY) \rightarrow \bullet S(X)eq(X,Y), \{X.l \leq X.r, X.r = Y.l, Y.l \leq Y.r, 0 = X.l, 2 = Y.r\}]$	predict-rule from 1
3	$[S, \langle(\langle r_1, r_2\rangle), \{0 = r_1, r_1 \leq r_2\}\rangle, p]$	predict-pred from 2
4	$[S, (\langle 0, 1\rangle), c]$	scan from 3
5	$[S(XY) \rightarrow \bullet S(X)eq(X,Y), \{X.l \leq X.r, X.r = Y.l, Y.l \leq Y.r, 0 = X.l, \}]$	predict-rule from 3
6	$[S(XY) \rightarrow S(X) \bullet eq(X,Y), \{\ldots, 0 = X.l, 2 = Y.r, 1 = X.r\}]$	complete 2 with 4
7	$[S(XY) \rightarrow S(X) \bullet eq(X,Y), \{X.l \leq X.r, X.r = Y.l, Y.l \leq Y.r, 0 = X.l, 1 = X.r\}]$	complete 5 with 4
8	$[eq, \langle(\langle r_1, r_2\rangle, \langle r_3, r_4\rangle), \{r_1 \leq r_2, r_2 = r_3, r_3 \leq r_4, 0 = r_1, 2 = r_4, 1 = r_2\}\rangle]$	predict-pred from 6
9	$[eq(a_1X, a_2Y) \rightarrow \bullet eq(X,Y), \{a_1.l + 1 = a_1.r, a_1.r = X.l, X.l \leq X.r,$ $a_2.l + 1 = a_2.r, a_2.r = Y.l, Y.l \leq Y.r, X.r = a_2.l, 0 = a_1.l, 1 = X.r, 2 = Y.r\}]$	predict-rule from 8
	...	
10	$[eq, (\langle 0, 1\rangle, \langle 1, 2\rangle), c]$	scan 8
11	$[S(XY) \rightarrow S(X)eq(X,Y) \bullet, \{\ldots, 0 = X.l, 2 = Y.r, 1 = X.r, 1 = Y.l\}]$	complete 6 with 10
12	$[S, (\langle 0, 2\rangle), c]$	convert 11

Figure 3: Trace of a sample Earley parse

avoids this using a technique of dynamic updating of a set of constraints on range boundaries. Experiments show that this significantly decreases the number of generated items, which confirms that range boundary constraint propagation is a viable method for a lazy computation of ranges.

The Earley parser could be improved by allowing to process the predicates of the righthand sides of clauses in any order, not necessarily from left to right. This way, one could process predicates whose range boundaries are better known first. We plan to include this strategy in future work.

References

François Barthélemy, Pierre Boullier, Philippe Deschamp, and Éric de la Clergerie. 2001. Guided parsing of Range Concatenation Languages. In *Proceedings of ACL*, pages 42–49.

Tilman Becker, Aravind K. Joshi, and Owen Rambow. 1991. Long-distance scrambling and tree adjoining grammars. In *Proceedings of EACL*.

Tilman Becker, Owen Rambow, and Michael Niv. 1992. The Derivationel Generative Power of Formal Systems or Scrambling is Beyond LCFRS. Technical Report IRCS-92-38, Institute for Research in Cognitive Science, University of Pennsylvania.

E. Bertsch and M.-J. Nederhof. 2001. On the complexity of some extensions of RCG parsing. In *Proceedings of IWPT 2001*, pages 66–77, Beijing, China.

Pierre Boullier. 1999. Chinese numbers, mix, scrambling, and range concatenation grammars. In *Proceedings of EACL*, pages 53–60, Bergen, Norway.

Pierre Boullier. 2000. Range concatenation grammars. In *Proceedings of IWPT 2000*, pages 53–64, Trento.

Håkan Burden and Peter Ljunglöf. 2005. Parsing linear context-free rewriting systems. In *Proceedings of IWPT 2005*, pages 11–17, Vancouver.

Laura Kallmeyer, Timm Lichte, Wolfgang Maier, Yannick Parmentier, and Johannes Dellert. 2008. Developing an MCTAG for German with an RCG-based parser. In *Proceedings of LREC-2008*, Marrakech, Morocco.

Wolfgang Maier and Anders Søgaard. 2008. Treebanks and mild context-sensitivity. In *Proceedings of the 13th Conference on Formal Grammar 2008*, Hamburg, Germany.

Jens Michaelis and Marcus Kracht. 1996. Semilinearity as a Syntactic Invariant. In *Logical Aspects of Computational Linguistics*, Nancy.

Daniel Radzinski. 1991. Chinese number-names, tree adjoining languages, and mild context-sensitivity. *Computational Linguistics*, 17:277–299.

Benoît Sagot. 2005. Linguistic facts as predicates over ranges of the sentence. In *Proceedings of LACL 05*, number 3492 in Lecture Notes in Computer Science, pages 271–286, Bordeaux, France. Springer.

Stuart M. Shieber, Yves Schabes, and Fernando C. N. Pereira. 1995. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24(1& 2):3–36.

Anders Søgaard. 2008. Range concatenation grammars for translation. In *Proceedings of COLING*, Manchester, England.

K. Vijay-Shanker, David Weir, and Aravind Joshi. 1987. Characterising structural descriptions used by various formalisms. In *Proceedings of ACL*.

Eric Villemonte de la Clergerie. 2002. Parsing mildly context-sensitive languages with thread automata. In *Proceedings of COLING*, Taipei, Taiwan.