[AU72] A.V. Aho and J.D. Ullman. *The Theory of Parsing, Translation and Compiling. Vol. I : Parsing.* Prentice-Hall, Englewoof Cliffs, N.J., 1972.

[AU73] A.V. Aho and J.D. Ullman. *The Theory of Parsing, Translation and Compiling. Vol. II : Compiling.* Prentice-Hall, Englewoof Cliffs, N.J., 1973.

[Ber79] Jean Berstel. *Transductions and Context-Free Languages.* Teubner, Stuttgart, 1979. 278 p.

[BGL76] Jean-Paul Boons, Alain Guillet, and Christian Leclère. *La structure des phrases simples en français : 1. Constructions intransitives.* Droz, Genève-Paris, 1976. 377 p.

[Cho56] Noam Chomsky. Three models for the description of language. In *IRE Transactions on Information Theory. Proceedings of the symposium on information theory*, volume IT-2, 1956.

[Ear70] J. Earley. An efficient context-free parsing algorithm. *Communications of the ACM*, 13(2):94–102, 1970.

[Gaz83] Gerald Gazdar. Phrase structure grammars and natural languages. In *Proceedings of the 8th IJCAI*, pages 556–565, Karlsruhe, 1983.

[Gro75] Maurice Gross. *Méthodes en syntaxe.* Hermann, Paris, 1975. 414 p.

[Gro93] Maurice Gross. Lexicon based algorithms for the automatic analysis of natural language. In Posner, Roland und Meggle, Georg, editor, *Theorie und Praxis des Lexicon*, pages 218–236. Walter de Gruyter, Berlin/New York, 1993.

[JS88] Aravind K. Joshi and Yves Schabes. An Earley-type algorithm for tree-adjoining grammars. In *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics*, Buffalo, 1988.

[MT90] Philip Miller and Thérèse Torris. *Formalismes syntaxiques pour le traitement automatique du langage naturel.* Paris : Hermès, 1990. 359 p.

[Roc92] Emmanuel Roche. Looking for syntactic patterns in texts. In Ferenc Kiefer, Gábor Kiss, and Júlia Pajzs, editors, *Papers in Computational Lexicography (COMPLEX)*, pages 279–287, Budapest, 1992. Research Institute for Linguistics, Hungarian Academy of Sciences.

[Roc93] Emmanuel Roche. *Analyse syntaxique transformationnelle du français par transducteurs et lexique-grammaire.* PhD thesis, Université Paris 7, 1993.

[Sag81] Naomi Sager. *Natural Language Information Processing: a computer grammar of English and its applications.* Addison-Wesley, 1981.

[Sal79] Morris Salkoff. *Analyse syntaxique du français. Grammaire en chaîne.* Number 2 in Études en linguistique française et générale. Benjamins, Amsterdam, 1979.

[Tom86] Masaru Tomita. *Efficient Parsing for Natural Language. A fast algorithm for practical systems.* Kluwer, Boston/Dordrecht/Lancaster, 1986. 201 p.

In order to parse a finite number of words at the same time, we compute

$$\begin{aligned} P_0(x, U) &= \bigcup_{u \in U} P_0(x, u) \\ P_i(x, U) &= \theta(P_{i-1}(x, U)) \text{ for } i \geq 1 \end{aligned}$$

The sequence $P_i(x, U)$ reaches a fixed point

$$P(x, U) = \bigcup_{u \in U} P(x, u)$$

The computation of $P_i(x, U)$ consists in applying a finite-state transducer to an acyclic automaton. The fixed point is reached after at most $|u| + 1$ steps, where $|u|$ is the length of the longest word in U, and the fixed point is the set of parsing trees of the strings in U.

The assumption that the grammar is strict is reasonable. Firstly, by Greibach's normal form theorem, any context-free language is generated by a strict grammar. Secondly, in the case of natural-language sentence parsing, this assumption is particularly not costly: as we mentioned in section 1, the bulk of the rules of a realistic, large-coverage, natural-language context-free grammar based on the predicate-argument model must be lexicalized, therefore each rule body contains at least one terminal symbol, the predicative element.

Conclusion

We studied an algorithm designed and implemented by Roche [Roc92, Roc93] for parsing natural language sentences according to context-free grammars. We noted the following facts about this algorithm.

It does not process trees, but strings, and thus it can take advantage of time-saving finite-state algorithms.

Roche successfully applied the algorithm to a context-free grammar with several millions of rules, whereas parsing words according to such large grammars is quite unusual. It should be borne in mind that a context-free grammar with a correct lexical and grammatical coverage is bound to have a very large number of rules.

The parser can be built for any context-free grammar provided that it is strict in the sense of [Ber79], and it outputs the set of parsing trees of the input sequence. The assumption that the grammar is strict is not costly: a realistic, large-coverage context-free grammar for natural-language sentences must be lexicalized, which implies that it naturally comes out as a strict grammar.

Most natural-language words are ambiguous: lexical ambiguity can be dealt with by representing the input sentence by an acyclic automaton recognizing a set of sequences, and by parsing them at the same time without backtracking.

Therefore, the theoretical properties of Roche's algorithm make it particularly adapted to parsing natural-language sentences and it is to be expected that it should significantly improve upon current techniques.

References

[ASJ88] Anne Abeillé, Yves Schabes, and Aravind K. Joshi. Parsing strategies with 'lexicalised' grammars: Application to tree adjoining grammars. In *Proceedings of COLING'88. 12th International Conference on Computational Linguistics*, Budapest, 1988.

5 Strict grammars

In the following we show that if G is a strict grammar in the sense of [Ber79], i.e.

$$\forall (x \longrightarrow u) \in R \;\; u \neq \epsilon \Longrightarrow u \in (A|X)^{\star} A (A|X)^{\star},$$

then the sequence of $P_i(x,u)$ admits a fixed point which is the set of parsing trees of u. We need to introduce the rational function $r : (A|(\overleftarrow{X}\ A^{\star}\ \overrightarrow{X})|\overset{\leftleftarrows}{X}|\overset{\rightrightarrows}{X})^{\star} \longrightarrow A^{\star}$ such that

$$r(g_0\ \overleftarrow{x_1}\ h_1\ \overrightarrow{x_1}\ g_1 \ldots \overleftarrow{x_n}\ h_n\ \overrightarrow{x_n}\ g_n) = h_1 h_2 \ldots h_n$$

for $g_k \in (A|\overset{\leftleftarrows}{X}|\overset{\rightrightarrows}{X})^{\star}$, $x_k \in X$ and $h_k \in A^{\star}$. If $(x \longrightarrow v) \in R$, then for all $u \in \overset{\leftleftarrows}{x}\ \sigma(v)\ \overset{\rightrightarrows}{x}$, $|u|_A = |v|_A + |r(u)|$.

Proposition 5 *For each word $u \in L$ such that $|u| \geq 3$, $|r(u)| < |r(\psi(u))|$*

Proof. There is a rule $(x \longrightarrow v) \in R$ such that $u \in \overset{\leftleftarrows}{x}\ \sigma(v)\ \overset{\rightrightarrows}{x}$ and $v \neq \epsilon$, therefore $|u| = 2 + |v|_A + 2|v|_X + |r(u)|$ and $|r(\psi(u))| = |u| - 2 - 2|v|_X$. $\bigtriangledown$

Let $(u,v) \in \tau$: if $|r(u)| = 0$, then $|u| = |v| = 2$ and $|r(v)| = 0$; if $|r(u)| \geq 1$, then $|u| \geq 3$ and $|v| \geq 3$, so $|r(v)| < |r(u)|$. In both cases, $|r(v)| \leq |r(u)|$.

Proposition 6 *Let $(u,v) \in \theta$ such that $u \neq v$. Then $|r(v)| \leq |r(u)|$; if $|r(u)| = 0$, then $|u| = |v|$, and if $|r(u)| \geq 1$, then $|r(v)| < |r(u)|$.*

Proof. By induction on $|u|$. $\bigtriangledown$

Proposition 7 *Any sequence $(u_i)_{i \geq 0}$ of words such that $\forall i \geq 0 \; (u_i, u_{i+1}) \in \theta$ is stationary and reaches it stationary part at most when $i = |r(u_0)| + 1$.*

Proof. Assume that $\forall i \leq |r(u_0)| \; u_i \neq u_{i+1}$. Then, $\exists i \leq |r(u_0)| \; |r(u_i)| = 0$: otherwise, in view of proposition (6), $|r(u_i)|$ would be a strictly decreasing sequence of positive integers. By proposition (6) again, $|u_i| = |u_{i+1}|$. It follows that $|u_{i+1}|_{\overleftarrow{X}} = |u_{i+1}|_{\overrightarrow{X}} = 0$, then by proposition (3), $u_{i+2} = u_{i+1}$.

Therefore we have shown that $\exists i \leq |r(u_0)| + 1 \; u_i = u_{i+1}$. By proposition (3), the sequence of words will be stationary from u_i on. $\bigtriangledown$

Thus the sequence of languages $P_i(x,u)$ is stationary as well and it reaches its stationary part at most when $i = |r(u_0)| + 1 = |u| + 1$. Let $P(x,u)$ be the fixed point of that sequence. Then we have the following:

Proposition 8 $\forall x \in X \; \forall u \in A^{\star} \; P(x,u) = L_{\widehat{G}}(x) \cap \varphi^{-1}(u) \cap (A|\ \overset{\leftleftarrows}{X}\ |\ \overset{\rightrightarrows}{X})^{\star}$

Proof.

$\subset$: recall that for each $i \geq 0$, $P_i(x,u) \;\subset\; \sigma(L_{\widehat{G}}(x)) \cap \varphi^{-1}(u)$. Therefore

$$P(x,u) \;\subset\; \sigma(L_{\widehat{G}}(x)) \cap \varphi^{-1}(u) \cap (A|\ \overset{\leftleftarrows}{X}\ |\ \overset{\rightrightarrows}{X})^{\star}$$

but $\sigma(L_{\widehat{G}}(x)) \cap (A|\ \overset{\leftleftarrows}{X}\ |\ \overset{\rightrightarrows}{X})^{\star} \;\subset\; L_{\widehat{G}}(x)$.

$\supset$: for $x \in X$, $g \in (A|\ \overset{\leftleftarrows}{X}\ |\ \overset{\rightrightarrows}{X})^{\star}$ and $p \geq 0$, we show by induction on p that

$$x \xrightarrow{p,\widehat{G}} g \Longrightarrow g \in P(x, \varphi(g))$$

If we take $g \in L_{\widehat{G}}(x) \cap \varphi^{-1}(u)$, we obtain $g \in P(x,u)$. $\bigtriangledown$

Thus, if the context-free grammar is strict, the computation of $P_i(x,u)$ reaches a fixed point after at most $|u| + 1$ steps, and the fixed point is the set of parsing trees of u.

Let θ be the rational transduction $(\tau \mid \mathrm{Id}_{A|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X}})^\star$, where the star is Kleene's star. The transduction θ is realized by a transducer with one state less and one transition more than in (6). This transducer will be applied iteratively in order to parse input sequences. $\theta \subset (A|(\overleftarrow{X}\ A^\star\ \overrightarrow{X})|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X})^\star \times (A|(\overleftarrow{X}\ A^\star\ \overrightarrow{X})|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X})^\star$. In general, neither θ nor θ^{-1} are functions. If $(u,v)\in\theta$, we have $\varphi(u) = \varphi(v)$, $|u|_A = |v|_A$, $|v|_{\overleftarrow{X}} = |v|_{\overrightarrow{X}}$, $|v|_{\overset{\Rightarrow}{X}} = |u|_{\overset{\Rightarrow}{X}} + |u|_{\overrightarrow{X}}$ and $|v| = |u| + 2|v|_{\overrightarrow{X}} \geq |u|$.

Proposition 3 *For $u\in(A|\overleftarrow{X}|\overrightarrow{X}|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X})^\star$, the following 3 conditions are equivalent:*
(i) $|u|_{\overleftarrow{X}} = |u|_{\overrightarrow{X}} = 0$,
(ii) $\forall v\in(A|\overleftarrow{X}|\overrightarrow{X}|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X})^\star \quad (u,v) \in \theta \iff u = v$,
(iii) $(u,u) \in \theta$.

Proof.
(i) $\Longrightarrow$ *(ii)*: if there were a factor f of u and a $g\in L$ such that $(f,g) \in \tau$, then $|f|_{\overrightarrow{X}} = 1$ and $|u|_{\overrightarrow{X}} > 0$.
(ii) $\Longrightarrow$ *(iii)* is immediate.
(iii) $\Longrightarrow$ *(i)* follows from the fact that $(u,v)\in\theta \Longrightarrow |v|_{\overset{\Rightarrow}{X}} = |u|_{\overset{\Rightarrow}{X}} + |u|_{\overrightarrow{X}}$. $\triangledown$

Proposition 4 *If $(u,v) \in \theta$ and $u \in \sigma(L_{\widehat{G}}(x))$ for some $x\in X$, then $v \in \sigma(L_{\widehat{G}}(x))$.*

Proof. Assume $x \xrightarrow{p,\widehat{G}} f$ and $u \in \sigma(f)$. We have $f = g_0 x_1 g_1 \ldots x_n g_n$ with $x_k\in X$ and $g_k\in(A|\ \overset{\Leftarrow}{X} \mid \overset{\Rightarrow}{X})^\star$, so $u = g_0\ \overleftarrow{x_1}\ h_1\ \overrightarrow{x_1}\ g_1 \ldots\ \overleftarrow{x_n}\ h_n\ \overrightarrow{x_n}\ g_n$ with $h_k\in A^\star$. Since $\tau \subset (\overleftarrow{X}\ A^\star\ \overrightarrow{X}) \times (\overset{\Leftarrow}{X}\ (A|\ \overleftarrow{X} \mid \overrightarrow{X})^\star\ \overset{\Rightarrow}{X})$, $v = g_0\ \overset{\Leftarrow}{x_1}\ h'_1\ \overset{\Rightarrow}{x_1}\ g_1 \ldots\ \overset{\Leftarrow}{x_n}\ h'_n\ \overset{\Rightarrow}{x_n}\ g_n$ with $(\overleftarrow{x_k}\ h_k\ \overrightarrow{x_k}, \overset{\Leftarrow}{x_k}\ h'_k\ \overset{\Rightarrow}{x_k})\in\tau$. But $\overleftarrow{x_k}\ h_k\ \overrightarrow{x_k}\in \sigma(x_k)$, so for each k there is a rule $(x_k \longrightarrow c_k) \in \widehat{R}$ such that $\overset{\Leftarrow}{x_k}\ h'_k\ \overset{\Rightarrow}{x_k}\in \sigma(c_k)$. We can derive $f \xrightarrow{k,\widehat{G}} g_0 c_1 g_1 \ldots c_n g_n$ and $v \in \sigma(g_0 c_1 g_1 \ldots c_n g_n)$. $\triangledown$

Consider for $x\in X$ and $u\in A^\star$ the sequence of languages defined by:

- $P_0(x,u) = \{\overleftarrow{x}\ u\ \overrightarrow{x}\}$;
- $P_i(x,u) = \theta(P_{i-1}(x,u))$ for $i\geq 1$.

It follows from proposition (4) that for $i\geq 0$, $P_i(x,u) \subset \sigma(L_{\widehat{G}}(x)) \cap \varphi^{-1}(u)$. For example, parsing $aacdbb$ according to G_0 involves the languages $P_i(x, aacdbb)$:

$$
\begin{aligned}
P_0(x,aacdbb) &= \{\ \overleftarrow{x}\ aacdbb\ \overrightarrow{x}\ \}\\
P_1(x,aacdbb) &= \{\ \overset{\Leftarrow}{x}\ a\ \overleftarrow{x}\ acdb\ \overrightarrow{x}\ b\ \overset{\Rightarrow}{x}\ \}\\
P_2(x,aacdbb) &= \{\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ a\ \overleftarrow{x}\ cd\ \overrightarrow{x}\ b\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\ \}\\
P_3(x,aacdbb) &= \{\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ c\ \overleftarrow{x}\overrightarrow{x}\ d\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\ \}\\
P_4(x,aacdbb) &= \{\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ c\ \overset{\Leftarrow}{x}\overset{\Rightarrow}{x}\ d\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\ \}\\
P_5(x,aacdbb) &= P_4(x,aacdbb)
\end{aligned}
$$

In general, the languages $P_i(x,u)$ may be empty or contain several elements. Each of these languages can be represented by an acyclic automaton, then the computation of $P_i(x,u)$ from $P_{i-1}(x,u)$ is performed by a quick and simple algorithm.

4 The transducer

Let $\sigma : (A|X|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X})^\star \longrightarrow (A|(\overleftarrow{X}\ A^\star\ \overrightarrow{X})|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X})^\star$ be the rational substitution defined by

$$\begin{array}{rl} \forall a{\in}A & \sigma(a) = a \\ \forall\ \overset{\Leftarrow}{x} \in \overset{\Leftarrow}{X} & \sigma(\overset{\Leftarrow}{x}) = \overset{\Leftarrow}{x} \\ \forall\ \overset{\Rightarrow}{x} \in \overset{\Rightarrow}{X} & \sigma(\overset{\Rightarrow}{x}) = \overset{\Rightarrow}{x} \\ \forall x{\in}X & \sigma(x) = \overleftarrow{x}\ A^\star\ \overrightarrow{x} \end{array}$$

Let $\psi : (A|\overleftarrow{X}|\overrightarrow{X}|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X})^\star \longrightarrow (A|X|\overleftarrow{X}|\overrightarrow{X})^\star$ be the morphism defined by

$$\begin{array}{rl} \forall a{\in}A & \psi(a) = a \\ \forall\ \overleftarrow{x} \in \overleftarrow{X} & \psi(\overleftarrow{x}) = \epsilon \\ \forall\ \overrightarrow{x} \in \overrightarrow{X} & \psi(\overrightarrow{x}) = \epsilon \\ \forall\ \overset{\Leftarrow}{x} \in \overset{\Leftarrow}{X} & \psi(\overset{\Leftarrow}{x}) = \overleftarrow{x} \\ \forall\ \overset{\Rightarrow}{x} \in \overset{\Rightarrow}{X} & \psi(\overset{\Rightarrow}{x}) = \overrightarrow{x} \end{array}$$

We have $\psi\psi = \varphi$ and $|u| = |\psi(u)| + |u|_{\overleftarrow{X}} + |u|_{\overrightarrow{X}}$. For each $(x{\longrightarrow}v){\in}R$ and $u{\in}\ \overset{\Leftarrow}{x}\ \sigma(v)\ \overset{\Rightarrow}{x}$, we have $|\psi(u)|_{\overset{\Rightarrow}{X}} = 0$ and $|\psi(u)|_{\overrightarrow{X}} = |u|_{\overset{\Rightarrow}{X}} = 1$.

Let L be the following rational language:

$$L = \bigcup_{(x\longrightarrow u)\in R} \overset{\Leftarrow}{x}\ \sigma(u)\ \overset{\Rightarrow}{x}$$

For example, with G_0,

$$L = \overset{\Leftarrow}{x}\ a\ \overleftarrow{x}\ A^\star\ \overrightarrow{x}\ b\ \overset{\Rightarrow}{x}\ |\ \overset{\Leftarrow}{x}\ c\ \overleftarrow{x}\ A^\star\ \overrightarrow{x}\ d\ \overset{\Rightarrow}{x}\ |\ \overset{\Leftarrow}{x}\overset{\Rightarrow}{x}$$

L is recognized by a deterministic automaton with at most

$$2\,\mathrm{card}\,X + \sum_{(x\longrightarrow u)\in R} (|u| + 2|u|_X) \tag{6}$$

transitions (this is a computational count: some of the transitions are labelled by the whole terminal alphabet A). For each $u{\in}L$, $|u|_{\overset{\Leftarrow}{X}} = |u|_{\overset{\Rightarrow}{X}} = 1$ and for each $u{\in}\ \overset{\Leftarrow}{x}\ \sigma(v)\ \overset{\Rightarrow}{x}$, $|u|_{\overleftarrow{X}} = |u|_{\overrightarrow{X}} = |v|_X$.

Let $\tau \subset\ (\overleftarrow{X}\ A^\star\ \overrightarrow{X}) \times (\overset{\Leftarrow}{X}\ (A|\ \overleftarrow{X}\ |\ \overrightarrow{X})^\star\ \overset{\Rightarrow}{X})$ be the rational transduction defined by

$$\tau = \{\ (\psi(u), u)\ \ |\ \ u{\in}L\ \}$$

It is realized by a transducer with the same number of transitions as in (6). With G_0, τ can be written as the union of 3 rational transductions:

$$\begin{array}{rcl} \tau & = & \{\ (\overleftarrow{x}\ afb\ \overrightarrow{x}, \overset{\Leftarrow}{x}\ a\ \overleftarrow{x}\ f\ \overrightarrow{x}\ b\ \overset{\Rightarrow}{x})\ \ |\ \ f{\in}A^\star\ \} \quad \cup \\ & & \{\ (\overleftarrow{x}\ cfd\ \overrightarrow{x}, \overset{\Leftarrow}{x}\ c\ \overleftarrow{x}\ f\ \overrightarrow{x}\ d\ \overset{\Rightarrow}{x})\ \ |\ \ f{\in}A^\star\ \} \quad \cup \\ & & \{\ (\overleftarrow{x}\overrightarrow{x}, \overset{\Leftarrow}{x}\overset{\Rightarrow}{x})\ \} \end{array}$$

If $(u,v){\in}\tau$, we have $\varphi(u) = \varphi(v)$, $|u|_A = |v|_A$, $|v|_{\overleftarrow{X}} = |v|_{\overrightarrow{X}}$, $|u|_{\overrightarrow{X}} = |v|_{\overset{\Rightarrow}{X}} = 1$, $|u|_{\overset{\Rightarrow}{X}} = 0$, and therefore $|v| = |u| + 2|v|_{\overrightarrow{X}} \geq |u|$; moreover, if $u \in \sigma(x)$ for some $x{\in}X$, then there exists a rule $(x{\longrightarrow}c) \in \widehat{R}$ such that $v \in \sigma(c)$.

3 Coding trees as strings

Let $G=(A,X,R)$ be the context-free grammar to be used by the parser. We will use four copies of X: $\overleftarrow{X},\overrightarrow{X},\overset{\Leftarrow}{X},\overset{\Rightarrow}{X}$, disjoint from A and X and pairwise disjoint. Let $\widehat{G}=(A|\overset{\Leftarrow}{X}\mid\overset{\Rightarrow}{X},X,\widehat{R})$ where

$$\widehat{R}=\{\ (x\longrightarrow\overset{\Leftarrow}{x}\ u\ \overset{\Rightarrow}{x})\ \mid\ (x\longrightarrow u)\in R\ \}$$

For example, if G is the grammar G_0 defined in the preceding section:

$$x\longrightarrow axb|cxd|\epsilon$$

then $\widehat{G_0}$ is defined as:

$$x\longrightarrow\overset{\Leftarrow}{x}\ axb\ \overset{\Rightarrow}{x}\mid\overset{\Leftarrow}{x}\ cxd\ \overset{\Rightarrow}{x}\mid\overset{\Leftarrow}{x}\overset{\Rightarrow}{x}$$

Let $\varphi:A|X|\overleftarrow{X}|\overrightarrow{X}|\overset{\Leftarrow}{X}\mid\overset{\Rightarrow}{X}\longrightarrow A|X$ be the projection that erases the symbols of $\overleftarrow{X}|\overrightarrow{X}|\overset{\Leftarrow}{X}\mid\overset{\Rightarrow}{X}$, and extend it to a morphism

$$\varphi:(A|X|\overleftarrow{X}|\overrightarrow{X}|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X})^{\star}\longrightarrow(A|X)^{\star}$$

Parsing trees according to G will be coded in the form of words of $(A|\overset{\Leftarrow}{X}\mid\overset{\Rightarrow}{X})^{\star}$. In order to show it we need a technical lemma about $\widehat{G}$:

Proposition 1 *For $x\in X$, no derived sequence of $D_{\widehat{G}}(x)$ takes the form*

$$fyg\ \overset{\Leftarrow}{z}\ h\ \overset{\Rightarrow}{z}\ k$$

with $y,z\in X$, $h\in(X|A)^{\star}$ and $f,g,k\in(X|A|\overset{\Leftarrow}{X}\mid\overset{\Rightarrow}{X})^{\star}$.

By induction on the length n of the derivation. $\triangledown$

Proposition 2 *For $x\in X$, $D_G(x)$ is in bijection with $D_{\widehat{G}}(x)$ and $L_{\widehat{G}}(x)$.*

Proof.

(i) $D_G(x)$ is in bijection with $D_{\widehat{G}}(x)$: each derivation of $D_G(x)$ or $D_{\widehat{G}}(x)$ applies a sequence of rules. Thus we have two injective mappings $\hat{\imath}:D_{\widehat{G}}(x)\longrightarrow\widehat{R}^{\star}$ and $i:D_G(x)\longrightarrow R^{\star}$. By induction on n, we show that for each $(x,u_1\ldots u_n)\in D_{\widehat{G}}(x)$, where the rules $\widehat{r_1}\ldots\widehat{r_n}$ are applied, then $(x,\varphi(u_1)\ldots\varphi(u_n))\in D_G(x)$, where the rules $r_1\ldots r_n$ are applied. Therefore, φ induces a mapping $\alpha:D_{\widehat{G}}(x)\longrightarrow D_G(x)$ and a bijection $\beta:\widehat{R}^{\star}\longrightarrow R^{\star}$, with $\hat{\imath}\beta=\alpha i$; thus $\alpha=\hat{\imath}\beta i^{-1}$ is bijective.

(ii) We only need to show that $\widehat{G}$ is unambiguous, i.e. the mapping

$$\left\{\begin{array}{lcl} D_{\widehat{G}}(x) & \longrightarrow & L_{\widehat{G}}(x)\\ (x,u_1\ldots u_n) & \longmapsto & u_n\end{array}\right.$$

is injective. By induction on n, if $(x,u_1\ldots u_n),(x,v_1\ldots v_m)\in D_{\widehat{G}}(x)$, $u_n=v_m$ and $n\leq m$, then $n=m$ and $\forall i\in[1,n]\ u_i=v_i$. $\triangledown$

The set of parsing trees of a word $u\in A^{\star}$ will thus be represented by $L_{\widehat{G}}(x)\cap\varphi^{-1}(u)\cap(A|\overset{\Leftarrow}{X}\mid\overset{\Rightarrow}{X})^{\star}$. For example, each parsing tree according to G_0 may be coded either in the form of a leftmost derivation:

$$x\longrightarrow axb\longrightarrow acxdb\longrightarrow acdb$$

or in the form of a word $\overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ c\ \overset{\Leftarrow}{x}\overset{\Rightarrow}{x}\ d\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\in L_{\widehat{G_0}}(x)$.

Each parsing tree according to G_0 may be coded either in the form of a leftmost derivation:

$$x \longrightarrow axb \longrightarrow acxdb \longrightarrow acdb$$

or in the form of a word $\overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ c\ \overset{\Leftarrow}{x}\overset{\Rightarrow}{x}\ d\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x} \in L_{\widehat{G_0}}(x)$.

The algorithm makes use of a second set of labelled parentheses, which we will note $\overleftarrow{x}$ and $\overrightarrow{x}$ in order to keep them distinct from $\overset{\Leftarrow}{x}$ and $\overset{\Rightarrow}{x}$. The algorithm will parenthesize a string $f \in A^\star$ with $\overleftarrow{x}$ and $\overrightarrow{x}$ whenever it attempts to find a derivation $x \overset{n,G_0}{\longrightarrow} f$. For example, il the algorithm is run to determine whether a string $f \in A^\star$ is in $L_{G_0}(x)$, its input should be $\overleftarrow{x}\ f\ \overrightarrow{x}$. An elementary step in the parsing process consists in:

1. substituting $\overset{\Leftarrow}{x}$ and $\overset{\Rightarrow}{x}$ for $\overleftarrow{x}$ and $\overrightarrow{x}$ in each $\overleftarrow{x}\ g\ \overrightarrow{x} \in \overleftarrow{x}\ A^\star\ \overrightarrow{x}$, and
2. inserting new pairs of labelled parentheses into g in such a way that if, in the result, one replaces each parenthesized string $\overleftarrow{x}\ h\ \overrightarrow{x}$ with x, then one obtains a right-hand side of a rule of $\widehat{G_0}$.

For example, when this elementary step is applied to $\overleftarrow{x}\ acdb\ \overrightarrow{x}$, the result is

$$\overset{\Leftarrow}{x}\ a\ \overleftarrow{x}\ cd\ \overrightarrow{x}\ b\ \overset{\Rightarrow}{x}$$

because $(x \longrightarrow \overset{\Leftarrow}{x}\ axb\ \overset{\Rightarrow}{x})$ is a rule of $\widehat{G_0}$. The two operations of substitution and insertion are performed at the same time by applying a finite-state transducer. In the case of G_0, this transducer is a 9-state transducer that realizes the transduction θ defined by:

$$\begin{aligned} \tau \ = \ & \{\ (\overleftarrow{x}\ afb\ \overrightarrow{x}, \overset{\Leftarrow}{x}\ a\ \overleftarrow{x}\ f\ \overrightarrow{x}\ b\ \overset{\Rightarrow}{x})\ \mid\ f \in A^\star\ \} \quad \cup \\ & \{\ (\overleftarrow{x}\ cfd\ \overrightarrow{x}, \overset{\Leftarrow}{x}\ c\ \overleftarrow{x}\ f\ \overrightarrow{x}\ d\ \overset{\Rightarrow}{x})\ \mid\ f \in A^\star\ \} \quad \cup \\ & \qquad\qquad \{\ (\overleftarrow{x}\overrightarrow{x}, \overset{\Leftarrow}{x}\overset{\Rightarrow}{x})\ \} \end{aligned}$$

and by $\theta = (\tau \mid \mathrm{Id}_{A|\overset{\Leftarrow}{X}|\overset{\Rightarrow}{X}})^\star$, where the star is Kleene's star. When θ is applied to a string, one step of top-down parsing is performed by further analyzing each of the strings g that had been parenthesized by $\overleftarrow{x}$ and $\overrightarrow{x}$ in the preceding step. For example, if we want to parse $aacdbb$, the iterative application of θ to $\overleftarrow{x}\ aacdbb\ \overrightarrow{x}$ yields the following strings:

$$\begin{gathered} \overleftarrow{x}\ aacdbb\ \overrightarrow{x} \\ \overset{\Leftarrow}{x}\ a\ \overleftarrow{x}\ acdb\ \overrightarrow{x}\ b\ \overset{\Rightarrow}{x} \\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ a\ \overleftarrow{x}\ cd\ \overrightarrow{x}\ b\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x} \\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ c\ \overleftarrow{x}\overrightarrow{x}\ d\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x} \\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ c\ \overset{\Leftarrow}{x}\overset{\Rightarrow}{x}\ d\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x} \\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ a\ \overset{\Leftarrow}{x}\ c\ \overset{\Leftarrow}{x}\overset{\Rightarrow}{x}\ d\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x}\ b\ \overset{\Rightarrow}{x} \end{gathered}$$

The iterative application of θ reaches a fixed point, which is a coding of the parsing tree of $aacdbb$. In general, θ is not a function: when applied to a string, it can yield several strings. In this case, the transducer is applied to the finite set of strings obtained from the preceding application. Each of these sets of strings can be represented by an acyclic automaton. We will see that in the assumption that the grammar is strict, the iterative application of θ to the set of strings reaches a fixed point and that the elements of the fixed point are the strings that code the parsing trees of the input sequence.

i.e. a total of $n+p$ rules. However, that method can be applied to n predicative elements and p sentence structures only if the np combinations are acceptable. Unfortunately, the various sentence structures associated with a predicative element correspond to numerous and diverse syntactic transformations: reversals, omissions, pronominalizations, passive transformations, symmetric transformations... that never apply to all other predicative elements and that are distributed among them in an irregular way. Specialists estimate that no two simple verbs in French enter exactly in the same sentence structures, except a few series like that of verbs denoting animal shouts [Gro75, BGL76].

Due to these difficulties, the bulk of the rules of a realistic natural-language context-free grammar must be lexicalized. Several authors preferred shifting from context-free parsing to more powerful grammars and parsers: string grammars [Sal79, Sag81], generalized phrase-structure grammars [Gaz83], tree-adjoining grammars [JS88, ASJ88]...

However, in comparison with context-free grammars, the structural and computational complexity of such formalisms is clearly a drawback. Thus, context-free parsing is worth revisiting with the following additional constraint: the number of rules being a parameter of paramount importance, the time complexity with respect to this number must be reduced to a minimum.

The program of [Roc93] parses natural language sentences according to a context-free grammar with more than 8.3 million rules [Roc92]. The parser is probably the first that was ever applied to a several-million-rule grammar, but it parsed 30-word sentences in 2 seconds. One of its advantages is that it does not process trees but strings. The rules are compacted into an acyclic finite-state automaton. The data of the parser consists of a finite-state transducer built from the automaton. The parser applies that transducer to the input sequence, then it iteratively applies it to the result, until a fixed point is reached. The theoretical time complexity of this process has not been studied yet, but in practice it seems to depend essentially on the height of the parsing trees and on the length and degree of ambiguity of the input sequence, but little or not on the number of rules. In the next section, we exemplify its operation on a small, non-linguistic grammar.

2 A non-linguistic example

We will use the following notations. The empty word is noted ϵ. If $u \in A^\star$ and $B \subset A$, $|u|_B$ is the number of positions in u that belong to B. A context-free grammar $G=(A,X,R)$ is defined by an alphabet A of terminal symbols, an alphabet X of non-terminal symbols and a set of rules, $R \subset X \times (A|X)^\star$. A rule is noted $(x \longrightarrow u)$ with $x \in X$ and $u \in (A|X)^\star$. If there is a derivation in n paths from $u \in (A|X)^\star$ to $v \in (A|X)^\star$, we write $u \xrightarrow{n,G} v$. The set of words of $(A|X)^\star$ derived from $u \in (A|X)^\star$ according to G is noted $L_G(u)$. The set of leftmost derivations from u according to G is noted $D_G(u)$.

Let G_0 be the following context-free grammar:

$$x \longrightarrow axb|cxd|\epsilon$$

and let us see how Roche's algorithm parses strings according to G_0. The computation does not process trees but strings. Parsing trees are coded as strings by using a grammar $\widehat{G_0}$, a variant of G_0 where for each rule $(x \longrightarrow u)$, the left-hand side x appears in the right-hand side in the form of a pair of "labelled parentheses" $\overleftarrow{x}$ and $\overrightarrow{x}$:

$$x \longrightarrow \overleftarrow{x}\ axb\ \overrightarrow{x} \mid \overleftarrow{x}\ cxd\ \overrightarrow{x} \mid \overleftarrow{x}\overrightarrow{x}$$

1 Size of realistic natural-language grammars

Context-free grammars were first designed and mathematically defined in order to model the syntax of natural languages [Cho56]; since then, they have been considered as the standard formalism to parse natural-language sentences [Ear70, Tom86, MT90]. However, a close examination of the problem shows that parsing natural language requires making an intensive use of lexical information [Gro93]. In particular, the syntax of natural languages is highly dependent on lexical information: as [Gro75, Sal79, Gaz83] first showed, a context-free grammar for natural-language sentences with a correct lexical and grammatical coverage would have at least hundreds of millions of rules; [Sal79] describes the actual construction of such a grammar.

The most common principle in the construction of context-free grammars for natural languages consists in modelling sentences into predicate and arguments. Many rules can take the form $S \longrightarrow u$, where S is a non-terminal symbol standing for a sentence type and u is a sequence containing

- a predicative element, in the form of one or several terminal symbols,
- one or several arguments in the form of non-terminals,
- and possibly grammatical words, in the form of terminal symbols or non-terminals.

The following examples model English sentence structures with 1, 2 and 3 arguments:

$$S \longrightarrow \textit{Nconc works} \tag{1}$$
$$S \longrightarrow \textit{Nabs is complementary to Nabs} \tag{2}$$
$$S \longrightarrow \textit{Nhum performs the installation of Nconc in Nconc} \tag{3}$$

where S stands for *sentence*, *Nconc* for *concrete noun*, *Nabs* for *abstract noun* and *Nhum* for *human noun*.

The predicate-argument description is an imperfect model, but it has been successfully applied to dozens of languages and it is close enough to linguistic reality for nearly any tentative formal description to be founded on one of its variants. However, it needs numerous rules to be built for it. Each predicative element can appear in several sentence structures, due to syntactic transformations, e.g.:

$$S \longrightarrow \textit{Nabs is complementary to Nabs} \tag{4}$$
$$S \longrightarrow \textit{Nabs-plur are complementary} \tag{5}$$

If the language has n predicative elements as in (1)–(3) and if each of them admits on average p sentence structures as in (4) and (5), they can be represented by np rules. Rules like (1)–(3) are "lexicalized" in the sense that the rule body contains at least one terminal symbol, the predicative element (*works, complementary, installation*).

It is natural to try to reduce the number of rules by associating predicative elements which admit the same sentence structures. One could thus represent np forms by combining p structural rules, e.g.:

$$S \longrightarrow \textit{Nabs Pred to Nabs}$$
$$S \longrightarrow \textit{Nabs-plur Pred}$$

with n lexical rules:

$$\textit{Pred} \longrightarrow \textit{are complementary}$$
$$\textit{Pred} \longrightarrow \textit{relate}$$
$$\textit{Pred} \longrightarrow \textit{have some relevance}$$

Context-free parsing with finite-state transducers

Éric Laporte*

Abstract

This article is a study of an algorithm designed and implemented by Roche [Roc92, Roc93] for parsing natural language sentences according to a context-free grammar. This algorithm is based on the construction and use of a finite-state transducer. Roche successfully applied it to a context-free grammar with very numerous rules. In contrast, the complexity of parsing words according to context-free grammars is usually considered in practice as a function of one parameter: the length of the input sequence; the size of the grammar is generally taken to be a constant of a reasonable value. In this article, we first explain why a context-free grammar with a correct lexical and grammatical coverage is bound to have a very large number of rules and we review work related with this problem. Then we exemplify the principle of Roche's algorithm on a small grammar. We provide formal definitions of the construction of the parser and of the operation of the algorithm and we prove that the parser can be built for a large class of context-free grammars, and that it outputs the set of parsing trees of the input sequence.

The complexity of parsing words according to context-free grammars is usually considered as having theoretically two parameters: the length of the input sequence and the size of the grammar, namely the number of rules or the sum of the lengths of rule bodies (see e.g. [AU72, AU73] for a survey). In practice, one tends to consider the size of the grammar as a constant. This approximation is acceptable if the size of the grammar is of a reasonable order of magnitude. As a matter of fact, existing parsers are usually applied to context-free grammars with at most a few thousand rules.

However, in the particular case of natural-language text parsing, we will see that a context-free grammar with a correct lexical and grammatical coverage is bound to have a very large number of rules. This problem has been addressed in various ways. Several authors preferred shifting from context-free parsing to more powerful grammatical and parsing formalisms. In two original contributions [Roc92, Roc93], Roche designs, implements and tests a top-down, breadth-first algorithm for parsing natural language sentences according to a context-free grammar with very numerous rules. His algorithm is based on the construction and use of a finite-state transducer. He describes how he constructed an actual, large-coverage context-free grammar and a parser for that grammar. He successfully applied it to French sentences. In this article, we exemplify the principle of Roche's algorithm on a small, non-linguistic grammar. Then, we give formal definitions of the construction of the parser and of the operation of the algorithm. We prove that the parser can be built for a large class of context-free grammars, and that it outputs the set of parsing trees of the input sequence.

*Laboratoire d'études et recherches en informatique, Université de Reims-Champagne-Ardenne, Moulin de la Housse, B.P.1039, F-51687 Reims CEDEX 2, France.