An Earley-Style Parser for Multiple Context-Free Grammars

Daniel M. Albro

October 27, 2002

1 Introduction

Harkema (2001) presents an Earley-style parser for Minimalist Grammars. This parser is an adaptation of that algorithm to Multiple Context-Free Grammars, which are loosely equivalent. The proof given here is similar, but more general in scope.

2 Definitions

2.1 Definition of an m-mcfg

We define an m-multiple context-free grammar (m-mcfg) as in Seki *et al.* 1991. A multiple context-free grammar G is a five-tuple $\langle N, O, F, R, S\rangle$, where N is a set of distinguished non-terminal symbols; $O \subseteq \bigcup_{n\in\mathbb{N}^+}(\Sigma^*)^n$ is a set of tuples of strings from the input alphabet Σ; $F \subseteq \bigcup_{n\in\mathbb{N}} F_n$, where F_n is the set of partial functions from O^n to O, is a set of string-transformation functions; $R \subseteq \bigcup_{n\in\mathbb{N}}(N \times N^n \times (F_n \cap F))$ is a set of grammar rules, written $A \rightarrow g[B_1 \dots B_{n(g)}]$ for $A, B_i \in N, 1 \leq i \leq n(g), g \in F_{n(g)}$; and $S \in N$ is a distinguished start symbol. To qualify as an m-mcfg, the grammar must further have the following properties:

1. The dimension of tuples in O is bounded: $O \subseteq \bigcup_{i=1}^{m}(\Sigma^*)^i$
2. For all $g \in F_{n(g)}$,
   (a) The dimension of the result of g and the dimensions of the arguments of g are fixed, that is, there are numbers $r(g) \in \mathbb{N}, d_i(g) \in \mathbb{N}, 1 \leq i \leq n(g)$ such that g is a function from $(\Sigma^*)^{d_i(g)} \times \dots \times (\Sigma^*)^{d_{n(g)}(g)}$ to $(\Sigma^*)^{r(g)}$.
   (b) Let $X = \{x_{ij} | 1 \leq i \leq n(g), 1 \leq j \leq d_i(g)\}$ be a set of pairwise distinct variables and define $x_i = (x_{i1}, \dots, x_{id_i(g)}), 1 \leq i \leq n(g)$. Let $g(\theta) = (g^1(\theta), \dots, g^{r(g)}(\theta))$ for any $\theta = (\theta_1, \dots, \theta_{n(g)}) \in (\Sigma^*)^{d_1(g)} \times \dots \times (\Sigma^*)^{d_{n(g)}(g)}$. If we define g^h as the h^{th} component of $g, 1 \leq h \leq r(g)$, then for each component g^h there is a fixed number $l_h(g) \in \mathbb{N}$ such that g^h is represented by the following concatenation of constant strings in Σ^* and variables in X: $g^h(x_1, \dots, x_{n(g)}) = \alpha_{h0} z_{h1} \alpha_{h1} z_{h2} \dots z_{hl_h(g)} \alpha_{hl_h(g)}$, where $\alpha_{hl} \in \Sigma^*, 0 \leq l \leq l_h(g)$, and $z_{hl} \in X, 1 \leq l \leq l_h(g)$.

(c) For every pair $(i,j), 1 \leq i \leq n(g), 1 \leq j \leq d_i(g)$, there is at most one $h, 1 \leq h \leq r(g)$ and at most one $l, 1 \leq l \leq l_h(g)$ such that z_{hl} in the representation of component g^h of g is the variable $x_{ij} \in X$.

3. $d(A)$ is fixed for all $A \in N$. $\forall A \in N \exists d(A) \in \mathbb{N}$ such that $\forall A \rightarrow g[B_1 \ldots B_{n(g)}] \in R$, $r(g) = d(A)$ and $d_i(g) = d(B_i), 1 \leq i \leq n(g)$.

4. $d(S) = 1$

2.2 Non-terminating versus Terminating Rules

We call rules $A \rightarrow g[B_1 \ldots B_{n(g)}]$ where $n(g) > 0$ *non-terminating rules* and rules $A \rightarrow g[]$ where $n(g) = 0$ *terminating rules*. Where $n(g) = 0$, g is simply a tuple $\alpha_{10}, \ldots, \alpha_{r(g)0}$, so we write $A \rightarrow \alpha_{10}, \ldots, \alpha_{r(g)0}$.

2.3 Normal Form

For the purpose of this paper we will use the following normal form:

1. If $g \in F_{n(g)}$ such that $n(g) > 0$, then for all (h,l), $1 \leq h \leq r(g), 0 \leq l \leq l_h(g)$, $\alpha_{hl} = \epsilon$. That is, in non-terminating rules, no constant elements are introduced by the string function of the rule, and g acts only to rearrange the strings corresponding to the components of the right-hand side.

2. For all terminating rules (rules where $n(g) = 0$) other than the excepted rule $S \rightarrow \epsilon$ we require that $g \subseteq (\Sigma^+)^{r(g)}$ (that is, the right-hand side must not be empty).

3. All nonterminals A in the grammar must be the head of some production $A \rightarrow g[B_1 \ldots B_{n(g)}]$ in the grammar.

An arbitrary MCFG can be transformed into an equivalent normal-form MCFG as follows:

1. For each non-empty α_{hl} in the definition of g for some rule $R = A \rightarrow g[B_1 \ldots B_{n(g)}]$, introduce a new non-terminal Γ with a production $\Gamma \rightarrow \alpha_{hl}$ and replace R with $R' = A \rightarrow g'[B_1 \ldots B_{n(g)}\Gamma]$, where $g = g'$ except that $n(g') = n(g) + 1$ and α_{hl} is replaced in the definition of g' with $x_{n(g')1}$, that is, the variable referring to the first element of the tuple corresponding to Γ.

2. For each terminating rule $A \rightarrow \epsilon$, where $A \neq S$, remove the rule from the grammar, and for each rule $\Gamma \rightarrow g[\ldots A \ldots]$ introduce an additional rule $\Gamma \rightarrow g'[\ldots]$ which no longer refers to A where $g' = g$ with variables referring to A removed.

3. Remove from N any nonterminal that is not the head of a production in the grammar.

2.4 L(G) — the Language Defined by a Grammar

The set of strings defined by an m-mcfg grammar is the same as $L_G(S)$, the set of strings that can be derived from the start symbol S. For $A \in N, k \in \mathbb{N}$, we define $L_G^k(A) \subseteq (\Sigma^*)^{d(A)}$ as follows:

1. If R contains a terminating rule $A \to g$, then $g \in L_G^0(A)$.
2. If $\theta \in L_G^k(A)$, then $\theta \in L_G^{k+1}(A)$.
3. If R contains a non-terminating rule $A \to g[B_1 \dots B_{n(g)}]$ and $\theta_i \in L_G^k(B_i)$, $1 \leq i \leq n$, and $g(\theta_1, \dots, \theta_{n(g)})$ is defined, then $g[\theta_1, \dots, \theta_{n(g)}] \in L_G^{k+1}(A)$.

2.5 Derivation, and Derivation Trees

In the following, α, β, γ, etc., refer to expressions from $(N \cup \Sigma^*)^*$. We say expression α *derives* expression β, that is $\alpha \Rightarrow \beta$, iff $\alpha = \alpha_1 A \alpha_2$ and $\beta = \alpha_1 \gamma \alpha_2$ and a rule $A \to g[\gamma]$ exists in R for some $g \in F$. We write $\alpha \Rightarrow^* \beta$ if $\alpha = \beta$ or if $\alpha \Rightarrow \alpha'$ and $\alpha' \Rightarrow^* \beta$.

Given a full derivation chain $S \Rightarrow \alpha_1 \Rightarrow \dots \Rightarrow \alpha_m$, where $\alpha_m \in ((\Sigma^+)^p)^q$ for some $p, q \in \mathbb{N}$, formed by applying a sequence of rules $A_1 \to g_1[B_{11} \dots B_{1n(g_1)}]$ $\dots$ $A_m \to g_m[B_{m1} \dots B_{mn(g_m)}]$, where $A_1 = S$, we can compute the string $w = w_1 \dots w_n$ derived by forming the CFG derivation tree T we would get if we treat the non-terminal symbols as non-terminals in a CFG and use the CFG sequence of applications $A_1 \to B_{11} \dots B_{1n(g_1)} \dots A_m \to B_{m1} \dots B_{mn(g_m)}$, with terminal symbols included for the terminal rules[1], and constructing an m-mcfg derivation tree T' by copying over each of the nodes $\nu \in T$ into T', but where the label $l_T(\nu)$ of the node in T is just the terminal or non-terminal used, we label each node in T' with a pair consisting of the string derived at that location plus the non-terminal used, or $\perp$ for a leaf node. That is, a leaf node in the CFG will be labeled with a tuple $w_{i_1}, \dots, w_{i_n}$; the corresponding node in T' is labeled $w_{i_1}, \dots, w_{i_n} : \perp$ to reflect the fact that there's no nonterminal. This is accomplished as follows. First, for each node ν in T that immediately dominates a leaf node labeled $w_{i_1}, \dots, w_{i_n}$ (and it will dominate only one node), we label the corresponding node in T' with $w_{i_1}, \dots, w_{i_n} : l_T(\nu)$. For any node ν in T' immediately dominating nodes whose labels have been filled out already, and formed by a rule $A \to g[B_1 \dots B_{n(g)}]$, $n(g) > 0$, we label ν with $g(\theta_1 \dots \theta_{n(g)}) : l_T(\nu)$, where $\theta_i, 1 \leq i \leq n(g)$ is the string part of the label of node ν's i^{th} child. When derivation tree T' is completed, the string part of the label of the topmost node is the string derived by the chain.

A *partial derivation tree* is the set of nodes above a cut in a derivation tree.

2.6 Position Vectors

A position vector (p, q) is a notation for the subsequence $w_{p+1} \dots w_q$ of a sentence $w_1 \dots w_n$.

[1] Treating the alphabet of the CFG as being comprised of members of O; that is, a tuple from O counts as a single output symbol in the CFG.

3 The Parser

The Earley-style parser to be presented here is a chart parser on the model of .... A chart parser is defined by giving the definition of *items* (indications of some determined fact about the parsing situation), axioms which give the initial items, rules of inference which can be used to derive new items from items already in the *chart*—the closure of the axioms under the inference rules, and a set of goal items. The presence of a goal item in the chart after closing the axioms under the inference rules for a particular sentence indicates, if the parser is sound, that the grammar does in fact generate that sentence.

3.1 Items

Items are three-tuples $\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, \pi \rangle$, where e_A for some $A \in N$ is a *situated expression* $(p_1, q_1), \ldots, (p_{d(A)}, q_{d(A)}) : A$ indicating a node in a possible derivation tree labeled by $w_{P_1+1} \ldots w_{q_1}, \ldots, w_{p_{d(A)}+1} \ldots w_{q_{d(A)}} \colon A$; $\pi \in \mathbb{N}, 0 \leq \pi \leq n$ is a *pointer* into the string $w_1 \ldots w_n$ being parsed; and C is a subset of the situated expressions $e_{B_1}, \ldots, e_{B_{n(g)}}$ on the right hand side of the rule. The pointer π is a device used to ensure that the parser has the *left-to-right* property of moving from w_1 up to w_n and rejecting the string (if the string is going to be rejected) after processing the shortest prefix $w_1 \ldots w_i$ of the string from which it can be determined that $w_1 \ldots w_n \notin L_G(S)$. Pointer π in an item $\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, \pi \rangle$ indicates, informally, that the prefix of the string tuple represented in e_A up to w_π (if $\pi > 0$, otherwise the null prefix) has been established as being a possible prefix of some string in $L_G(A)$, and similarly for $e_{B_i}, 1 \leq i \leq n(g)$. The elements of C are the subset of $\{e_{B_i} | 1 \leq i \leq n(g)\}$ for which it has not yet been established that the prefix up to $w_{\pi+1}$ is a possible prefix of $L_G(B_i)$.

3.1.1 Functions on positions and related definitions

We say that position vector (p, q) *contains* the positions p through $q - 1$, and that the value of the function *positions*$(\{(p, q)\})$ is the set $\{i | p \leq i < q\}$. We extend this function to situated expressions $e_A = (p_1, q_1), \ldots, (p_{d(A)}, q_{d(A)}) : A$ as follows: *positions*$(e_A) = \bigcup_{i=1}^{d(A)}$ *positions*$(\{(p_i, q_i)\})$. For a sequence of situated expressions, *positions*$(e_{B_1} \ldots e_{B_n}) = \bigcup_{i=1}^{n}$ *positions*(e_{B_i}). All derivable items $\Delta = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, \pi \rangle$ are defined such that *positions*$(e_{B_1} \ldots e_{B_{n(g)}}) \subseteq$ *positions*(e_A), so we say *positions*$(\Delta) =$ *positions*(e_A).

We define the function *next*(π) as *next*$(\pi) = \pi + 1$.

If π_l is the leftmost contained position for item Δ, and π_r is the rightmost, then Δ is to the *right* of π_l and all positions left of π_l. Δ is also to the *left* of *next*(π_r) and of all positions right of *next*(π_r). Finally, we say that Δ *crosses* all positions that are both right of π_l and left of *next*(π_r).

3.1.2 Invariant

An item $\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, \pi \rangle$ embodies the following claim, where $w = w_1 \ldots w_n$ is the sentence being parsed[2]:

[2]Clause (1) of the invariant does not apply to items headed by S'.

1. There is a series of situated expressions $e_{E_1}, \ldots, e_{E_m}$ such that:

   (a) $\exists i, 1 \leq i \leq m$ such that $e_{E_i} = e_A$.

   (b) $\forall j, 1 \leq j \leq m, i \neq j$, e_{E_j} does not cross π.

   (c) Expressions $E_1, \ldots, E_m$ are the leaves of a partial derivation tree of w.

   (d) $\forall j, j \neq i, 1 \leq j \leq m$, e_{E_j} left of π implies that e_{E_j} is a lexical expression, *i.e.*, $E_j = \perp$.

2. For every situated expression $e_{B_j} \in C$, $1 \leq j \leq n(g)$, there is a sequence of situated expressions $e_{F_1}, \ldots, e_{F_p}$ such that:

   (a) None of the situated expressions $e_{F_i}, 1 \leq i \leq p$, crosses π.

   (b) Expressions $F_1, \ldots, F_p$ are the leaves of a partial derivation tree with root B_j.

   (c) For every $F_i, 1 \leq i \leq p$, such that e_{F_i} is to the left of π, e_{F_i} is a lexical expression ($F_i = \epsilon$).

3. For every situated expression $e_{B_j} \notin C, 1 \leq j \leq n(g)$, there is a sequence of situated expressions $e_{G_1}, \ldots e_{G_q}$ such that:

   (a) None of the situated expressions $e_{G_i}, 1 \leq i \leq q$, crosses $next(\pi)$.

   (b) $G_1, \ldots, G_q$ are the leaves of a partial derivation tree with root B_j.

   (c) For every $G_i, 1 \leq i \leq q$, such that e_{G_i} is to the left of $next(\pi)$, e_{G_i} is a lexical expression ($G_i = \epsilon$).

3.2 Axioms

For $n \in \mathbb{N}$ the length of the sentence to be parsed, $S \in N$ licenses the following axiom:

$$\langle S' \rightarrow g_I[(0,n) : S], \{(0,n) : S\}, 0 \rangle$$

where g_I is the identity function on Σ^*.

3.3 Goal Items

For a sentence of length $n > 0$, $\langle S' \rightarrow g_I[(0,n) : S], \emptyset, n-1 \rangle$ is a goal item. For a sentence of length 0, $\langle S' \rightarrow g_I[(0,n) : S], \emptyset, 0 \rangle$ is a goal item.

3.4 Rules of Inference

3.4.1 Next

The antecedent is

$$\langle S' \rightarrow g_I[(0,n) : S], \emptyset, \pi \rangle$$

such that $\pi + 1 < n$. The consequent is

$$\langle S' \rightarrow g_I[(0,n) : S], C, \pi \rangle$$

where $(0,n) : S \in C$ iff $\pi + 1 \in \mathit{positions}\{(0,n)\}$. Note that NEXT is a function.

3.4.2 Predict

The antecedent is

$$\langle e_A \rightarrow g'[e_{B_1} \ldots e_{B_{n(g')}}], C \cup \{(p_1, q_1), \ldots, (p_{d(B_i)}, q_{d(B_i)}) : B_i\}, \pi \rangle$$

for some $i, 1 \leq i \leq n(g')$, where the antecedent must be such that:

1. $\mathrm{Min}_{p \in \mathit{positions}(e_{B_i})} = \pi$.
2. R contains a rule $B_i \rightarrow g[\Gamma_1 \ldots \Gamma_{n(g)}]$.
3. $\forall \Gamma_j, 1 \leq j \leq n(g)$, $e_{\Gamma_j} = (p_{j1}, q_{j1}), \ldots, (p_{jd(\Gamma_j)}, q_{jd(\Gamma_j)}) : \Gamma_j$ exists such that $\forall k, 1 \leq k \leq d(\Gamma_j)$, $\forall h, 1 \leq h \leq r(g)$, in $g^h(x_1, \ldots, x_{n(g)}) = z_{h1} \ldots z_{hl_h(g)}$:
   (a) if $z_{h1} = x_{jk}$, then $p_{jk} = p_h$.
   (b) If $z_{hl_h(g)} = x_{jk}$, then $q_{jk} = q_h$.
   (c) $\forall l, 1 \leq s < l_h(g)$, if $z_{hs} = x_{jk}$ and $z_{hs+1} = x_{j'k'}$, then $p_{j'k'} = q_{jk}$.
4. If $n(g) = 0$, then $e_{B_i} = (p, q) : B_i$ and $g = w_{p+1} \ldots w_q$.

The consequent is

$$\langle e_{B_i} \rightarrow g[e_{\Gamma_1} \ldots e_{\Gamma_{n(g)}}], C', \pi \rangle$$

where $C' = \{e_{\Gamma_i} | 1 \leq i \leq n(g), \pi \in \mathit{positions}(e_{\Gamma_i})\}$.

This rule of inference is the only one that introduces new rules and new position vectors into the chart. Note that it is defined in such a way that the positions on the right hand side of an item are always subsets of the positions on the left hand side.

PREDICT is not a function. In fact, even for a particular antecendent and a particular rule $B_i \rightarrow g[\Gamma_1 \ldots \Gamma_{n(g)}]$ there can be many consequents.

3.4.3 Up

The antecedent is

$$\begin{gathered}\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, \pi \rangle \\ \langle e_{D_1} \rightarrow g_1[e_{\Gamma_{11}} \ldots e_{\Gamma_{1n(g_1)}}], \emptyset, \pi \rangle \\ \vdots \\ \langle e_{D_m} \rightarrow g_m[e_{\Gamma_{m1}} \ldots e_{\Gamma_{mn(g_m)}}], \emptyset, \pi \rangle\end{gathered}$$

such that

1. $C = \bigcup_{i=1}^{m} \{e_{D_i}\} = C$
2. $\forall e_{D_i}, 1 \leq i \leq m$, $\exists B_j, 1 \leq j \leq n(g)$ such that $e_{D_i} = e_{B_j}$.
3. $\forall e_{D_i}, 1 \leq i \leq m$, $\nexists e_{D_j}, 1 \leq j \neq i \leq m$ such that $e_{D_i} = e_{D_j}$.

The consequent is:

$$\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, \pi \rangle$$

Note that UP is a function.

3.4.4 Down

The antecedent is

$$\begin{gathered}\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C \cup \{e_{B_i}\}, \pi \rangle \\ \langle e_{B_i} \rightarrow g_i[e_{\Gamma_1} \ldots e_{\Gamma_{n(g_i)}}], \emptyset, \rho \rangle\end{gathered}$$

for some $i, 1 \leq i \leq n(g)$ such that:

1. $\text{Max}_{\{p \in positions(e_{B_i}) | p < \pi\}} = \rho$.

The consequent is

$$\langle e_{B_i} \rightarrow g_i[e_{\Gamma_1} \ldots e_{\Gamma_{n(g_i)}}], C', \pi \rangle$$

where $C' = \{e_{\Gamma_j} | 1 \leq j \leq n(g_i), \pi \in positions(e_{\Gamma_j})\}$. Note that DOWN is a function.

4 Proof of Soundness

The recognizer is sound if the invariant holds of every axiom and derivable item. To show this, we show that the invariant holds of all axioms, and that if the invariant holds of the antecedents of a rule, then it also holds of the consequent.

4.1 Axioms

The axiomatic item is $\langle S' \rightarrow g_I[(0,n) : S], \{(0,n) : S\}, 0\rangle$.

1. Take the sequence $e_{E_1} = S'$.

   (a) The only item in the sequence is S', so $i = 1$.

   (b) There are no other items in the sequence, so none of the items crosses π.

   (c) S' is the leaf of the tree consisting of just the node S', which is a partial extended derivation tree for *any* w.

   (d) There are no other items in the sequence, so (1d) trivially satisfied.

2. Since no useless nonterminals exist in the grammar, there must be some rule $S \rightarrow g[A_1 \ldots A_{n(g)}]$ or $S \rightarrow g$. For the first case, consider the sequence $e_{A_1}, \ldots, e_{A_{n(g)}}$, defined to satisfy requirement (3) of the predict rule. For the second case, consider the sequence containing the single item $(0,n) : \epsilon$:

   (a) By requirement (3) of predict, no position in $e_{A_1}, \ldots, e_{A_{n(g)}}$ is left of position 0, so each of the expressions e_{A_i} is to the right of position 0 and thus does not cross it. This is true of $(0,n) : \epsilon$ as well.

   (b) For case (1), the sequence $A_1 \ldots A_{n(g)}$ are the leaves of a partial derivation tree with root S, by virtue of the fact that $S \rightarrow g[A_1 \ldots A_{n(g)}]$ is a rule. For case (2), ϵ is the leaf of a partial derivation tree with root S since $S \rightarrow g$ is a rule.

   (c) No position in $e_{A_1}, \ldots, e_{A_{n(g)}}$ is left of π, so this subclause is trivially satisfied. Same for the sequence $(0,n) : \epsilon$.

3. There is no element not in C, so this clause is trivially satisfied.

4.2 Rules of Inference

4.2.1 Next

Here the antecedent is $\langle S' \rightarrow g_I[(0,n) : S], \emptyset, \pi\rangle$ and the consequent is $\langle S' \rightarrow g_I[(0,n) : S], \{(0,n) : S\}, \pi + 1\rangle$. $\langle S' \rightarrow g_I[(0,n) : S], \emptyset, \pi + 1\rangle$ is not a possible consequent since in this case $\pi + 1$ would have to equal n and the rule would not apply.

1. See clause (1) in §4.1.

2. According to the invariant, it follows from the antecedent that there is a sequence of subitems $e_{G_1}, \ldots, e_{G_q}$ such that (3a) none of the subitems $e_{G_i}, 1 \leq i \leq q$ crosses $\pi + 1$; (3b) $G_1, \ldots, G_q$ are the leaves of a partial derivation tree with root S; (3c) any expression $e_{G_i}, 1 \leq i \leq q$ which appears to the left of $\pi + 1$ is a lexical expression. Note that $(0,n) : S \in C$, so taking the sequence $e_{G_1}, \ldots, e_{G_q}$, we see

   (a) By (3a) in the invariant applied to the antecedent.

   (b) By (3b) in the invariant applied to the antecedent.

(c) By (3c) in the invariant applied to the antecedent.

3. No element of the right hand side is present in C, so this clause trivially applies.

4.2.2 Predict

The antecedent is $\langle e_A \rightarrow g_1[e_{B_1} \ldots e_{B_{n(g_1)}}], C \cup \{(p_1, q_q) \ldots (p_{d(B_i)}, q_{d(B_i)}) : B_i\}, \pi \rangle$. The consequent is $\langle e_{B_i} \rightarrow g[e_{\Gamma_1} \ldots e_{\Gamma_{n(g)}}], C', \pi \rangle$.

From the invariant we see that there is a sequence of situated expressions e_{E_1},..., e_{E_m} such that (1a) for some $j, 1 \leq j \leq m$, $e_A = e_{E_j}$; (1b) none of the subitems $e_{E_k} \neq A, 1 \leq k \leq m$, crosses π; (1c) expressions $E_1, \ldots, E_m$ are the leaves of a partial derivation tree of w; (1d) any expression E_k,$1 \leq k \leq m$, such that $e_{E_k} \neq e_A$ is to the left of π, is a lexical expression.

For any $e_{B_c} = (p_{c1}, q_{c1}), \ldots, (p_{cd(B_c)}, q_{cd(B_c)}) : B_c \in C$, there is a sequence of situated expressions $e_{F_{c1}}, \ldots, e_{F_{cr_c}}$ such that: (2a) none of the subitems $e_{F_{ci}}, 1 \leq i \leq r_c$, crosses π; (2b) expressions $F_{c1}, \ldots, F_{cr_c}$ are the leaves of a partial derivation tree with root B_c; (2c) any expression F_{ci}, such that $e_{F_{ci}}$ is to the left of π, is a lexical expression.

We know that at least $e_{B_i} \in C$.

For any $e_{B_{c'}} = (p_{c'1}, q_{c'1}), \ldots, (p_{c'd(B_{c'})}, q_{c'd(B_{c'})}) : B_{c'} \notin C$, there is a sequence of subitems $e_{G_{c'1}}, \ldots, e_{G_{c's_{c'}}}$ such that: (3a) none of the subitems $e_{G_{c'i}}, 1 \leq i \leq s_{c'}$, crosses $next(\pi)$; (3b) expressions $G_{c'1}, \ldots, G_{c's_{c'}}$ are the leaves of a partial derivation with root $B_{c'}$; (3c) any expression $G_{c'i}, 1 \leq i \leq s_{c'}$, such that $e_{G_{c'i}}$ is left of $next(\pi)$ is a lexical expression.

1. Consider the sequence $e_{E_1}, \ldots, e_{E_{j-1}}, e_{\gamma_{B_1}}, \ldots, e_{\gamma_{B_{i-1}}}, e_{B_i}, e_{\gamma_{B_{i+1}}}, \ldots, e_{\gamma_{B_{n(g_1)}}}$, $e_{E_{j+1}}$, ..., e_{E_m}, where $e_{\gamma_{B_k}} = e_{F_{k1}}, \ldots, e_{F_{kr_k}}$ for $e_{B_k} \in C$, and $e_{\gamma_{B_k}} = e_{G_{k1}}$, ..., $e_{G_{ks_k}}$ for $e_{B_k} \notin C$.

   (a) Clearly e_{B_i} is one of the subitems of the sequence.

   (b) It's given above that none of the subitems $e_{G_{kl}}, 1 \leq k \leq n(g_1), 1 \leq l \leq r_k$ crosses $\pi + 1$. Thus, for a subitem $e_{G_{kl}}$ to cross π, it has to be to the immediate left of $\pi + 1$. According to the invariant, for any subitem $e_{G_{kl}}$ to the left of $\pi + 1$, $e_{G_{kl}}$ is lexical, and lexical items cross no positions. Furthermore, by the invariant none of the subitems $e_{F_{kl}}$ crosses π, nor do any of the subitems e_{E_i} other than e_{E_j}, which is not included. Therefore none of the subitems in the sequence cross π.

   (c) It can be seen from the rules of inference that $A \rightarrow g_1[B_1 \ldots B_{n(g_1)}]$ must be a production from the grammar (since items with new rule types are introduced only by the predict rule, and the predict rule only introduces rule types from the grammar) or $A = S'$, in which case $n(g_1) = 1$ and $B_1 = S$. For each $F_{k1}, \ldots, F_{kr_k}$ or $G_{k1}, \ldots, G_{ks_k}$, the invariant states them as leaves of a derivation tree with head B_k, so since $E_1 \ldots E_m$ are leaves of a partial derivation tree of w, $E_1, \ldots, E_{j-1}$, $\gamma_{B_1}, \ldots, \gamma_{B_{i-1}}$, B_i, $\gamma_{B_{i+1}}, \ldots, \gamma_{B_{n(g_1)}}$, $E_{j+1}, \ldots, E_m$ are the leaves of such a partial derivation tree, where $B_1 \ldots B_{n(g_1)}$ is immediately dominated by A and γ_{B_k} is immediately dominated by B_k for $k \neq i$.

(d) By the antecedent, E_k for any $k \neq j$ is a lexical expression if e_{E_k} is to the left of π. This is also true for all F_{kl}. For all G_{kl}, G_{kl} is lexical if it is to the left of $\pi + 1$. This is a stronger requirement, so the lesser requirement that G_{kl} be lexical if $e_{G_{kl}}$ is left of π is also true. Therefore, any of $e_{E_1}, \ldots, e_{E_{j-1}}, e_{\gamma_{B_1}}, \ldots, e_{\gamma_{B_{i-1}}}, e_{B_i}, e_{\gamma_{B_{i+1}}}, \ldots, e_{\gamma_{B_{n(g_1)}}}, e_{E_{j+1}}, \ldots, e_{E_m}$ whose subitem is to the left of π is a lexical expression.

2. Since all useless nonterminals have been removed from the grammar, for each nonterminal in the right hand side of the consequent, there must be at least one production headed by that nonterminal. For any e_{Γ_i} in which π is the leftmost position, that is, for any $e_{\Gamma_i} \in C'$, since π is the leftmost position in the consequent, take the situated expressions that would be formed by applying a modified form of predict (predict without requirement (4), since without this requirement predict will always apply if there is an appropriate rule in the grammar) to e_{Γ_i} as $e_{F_{i1}}, \ldots, e_{F_{ip_i}}$.

   (a) The sequence $e_{F_{i1}}, \ldots, e_{F_{ip_i}}$ cannot cross π, since the sequence was produced to obey condition (3) of predict, which means that the positions in it are a subset of the positions in its head e_{Γ_i} and thus do not contain anything left of π.

   (b) Since $F_{i1}, \ldots, F_{ip_i}$ are the right hand side of a production headed by Γ_i, they are the leaves of a partial derivation tree with root Γ_i.

   (c) Since no position left of π exists in $e_{F_{i1}}, \ldots, e_{F_{ip_i}}$, the requirement that all positions left of π are lexical trivially applies.

3. For any sequence $e_{\Gamma_i} \notin C'$, there must be a rule $\Gamma_i \rightarrow g_i[G_{i1} \ldots G_{in(g_i)}]$ in the grammar, for $n(g_i) \geq 0$, by the above argument. We can use a modified form of predict (removing requirement (4)) to produce an item $\langle e_{\Gamma_i} \rightarrow g_i[e_{G_{i1}} \ldots e_{G_{in(g_i)}}], C_i, \pi_i \rangle$ for $\pi_i > \pi$, where π_i is the leftmost position in e_{Γ_i} (since $e_{\Gamma_i} \notin C'$, the lowest position in e_{Γ_i} must be greater than π). Take from this item the sequence $e_{G_{i1}} \ldots e_{G_{in(g_i)}}$.

   (a) Since $next(\pi)$ is the leftmost position conceivably contained within the sequence, it cannot cross $next(\pi)$.

   (b) By virtue of the way the sequence was formed, $G_{i1} \ldots G_{in(g_i)}$ must be the leaves of a partial derivation tree headed by Γ_i.

   (c) Since no position left of $next(\pi)$ could be contained in the sequence, the condition that all such positions be lexical applies by default.

4.2.3 Up

Here the antecedent is $\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, \pi \rangle$ plus $\langle e_{B_i} \rightarrow g_i[e_{\Gamma_{i1}} \ldots e_{\Gamma_{in(g_i)}}], C_i = \emptyset, \pi \rangle$ for each $e_{B_i} \in C$. The consequent is $\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C' = \emptyset, \pi \rangle$.

1. Since condition (1) of the invariant does not involve the C component of the items, and the first antecedent is otherwise identical to the consequent, the sequence $e_{E_1}, \ldots, e_{E_m}$ given for the first antecedent will suffice for the consequent as well.

2. Irrelevant, since $C' = \emptyset$.

3. For each $e_{B_i} \notin C$, the sequence $e_{G_{i1}}, \ldots, e_{G_{iq_i}}$ by which clause (3) of the invariant is satisfied for the first antecedent also satisfies the invariant for the consequent. For the others, the invariant gives us a sequence $e_{G_{ij1}}, \ldots, e_{G_{ijq_{ij}}}$ satisfying condition (3) for each $e_{\Gamma_{ij}}, 1 \leq j \leq n(g_i)$ in the right hand side of the antecedent headed by e_{B_i}. We then take the sequence $e_{G_{i11}}$, ..., $e_{G_{i1q_{i1}}}$, ..., $e_{G_{in(g_i)1}}$, ..., $e_{G_{in(g_i)q_{in(g_i)}}}$.

   (a) None of the elements of the sequence crosses $next(\pi)$, since all of its elements are elements of sequence for which that is true.

   (b) We can form a partial derivation tree with B_i as its top layer, Γ_{i1}, ..., $\Gamma_{in(g_i)}$ as its middle layer, and G_{i11}, ..., $G_{i1q_{i1}}$, ..., $G_{in(g_i)1}$, ..., $G_{in(g_i)q_{in(g_i)}}$ as the leaves.

   (c) All of the elements of the sequence which are left of $next(\pi)$ are lexical, since all of the elements of the sequence are also elements of the sequence that satisfies this requirement for some antecedent.

4.2.4 Down

The antecedent here is $\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, \pi\rangle$ and $\langle e_{B_i} \rightarrow g_i[e_{\Gamma_1} \ldots e_{\Gamma_{n(g_i)}}], C_i = \emptyset, \rho\rangle$, where $e_{B_i} \in C$. The consequent is $\langle e_{B_i} \rightarrow g_i[e_{\Gamma_1}, \ldots, e_{\Gamma_{n(g_i)}}], C', \pi\rangle$. From the invariant there must be a sequence $e_{E_1}, \ldots, e_{E_m}$satisfying clause (1) of the invariant for the first antecedent, plus a sequence $e_{F_{j1}}, \ldots, e_{F_{jp_j}}$ satisfying clause (2) for each $e_{B_j} \in C$, and finally a sequence $e_{F_{j1}}, \ldots, e_{F_{jp_j}}$ satisfying clause (3) for each $e_{B_j} \notin C$. Furthermore, there must be a sequence $e_{G_{j1}}, \ldots, e_{G_{jq_j}}$ that satisfies clause (3) for each e_{Γ_j} in the second antecedent.

1. Take the sequence e_{E_1}, ..., $e_{E_{a-1}}$, e_{γ_1}, ..., $e_{\gamma_{i-1}}$, e_{B_i}, $e_{\gamma_{i+1}}$, ..., $e_{\gamma_{n(g)}}$, $e_{E_{a+1}}$, ..., e_{E_m}, where $e_{E_a} = e_A$ and $e_{\gamma_j} = e_{F_{j1}}, \ldots, e_{F_{jp_j}}, 1 \leq j \neq i \leq n(g)$.

   (a) This sequence contains e_{B_i} .

   (b) For each e_{γ_j} sequence where $e_{B_j} \in C$, we know none of the elements cross π because they satisfy clause (2a) for the antecedent. For each e_{γ_j} sequence where $e_{B_j} \notin C$, we know that none of the elements cross $next(\pi)$. The only way they could cross π and not $next(\pi)$ is by having π as their rightmost contained positions, but in that case they would contain π and thus e_{B_j} would have to have been a members of C. Thus none of the elements cross π.

   (c) Take the partial derivation tree of which the members of the sequence $e_{E_1}, \ldots, e_{E_m}$ are leaves and replace e_A the tree headed by e_A, with e_{B_1}, ..., $e_{B_{n(g)}}$ as the next level and $e_{F_{j1}}, \ldots, e_{F_{jp_j}}$ immediately dominated by each e_{B_j} where $j \neq i$.

   (d) For $e_{E_1}, \ldots, e_{E_{a-1}}, e_{E_{a+1}}, \ldots, e_{E_m}$ and $e_{F_{jk}}$ where $e_{B_j} \in C$, we are given that any of these expressions left of π is lexical, and for $e_{F_{jk}}$ where $e_{B_j} \notin C$, we are given that any $e_{F_{jk}}$ left of $next(\pi)$ is lexical, which is a stronger requirement.

2. For each element $e_{\Gamma_j} \in C'$, take the sequence $e_{G_{j1}}, \ldots, e_{G_{jq_j}}$ which satisfies clause (3) of the invariant for the second antecedent.

   (a) None of the elements of this sequence crosses ρ, so any subitem $e_{G_{jk}}$ is either to the left of ρ or to the right of ρ. If $e_{G_{jk}}$ is to the left of ρ, it must be to the left of π as well, since $\rho < \pi$, so it can't cross π. If $e_{G_{jk}}$ is to the right of ρ, then its leftmost position must be $\geq \pi$, since ρ is the rightmost position in e_{Γ_j} (and hence in $e_{G_{jk}}$) to the left of π. If this is so, it can't cross π.

   (b) $G_{j1}, \ldots, G_{jq_j}$ are the leaves of a partial derivation tree headed by Γ_j, since they satisfy (3b) for the second antecedent.

   (c) Any $e_{G_{jk}}$ left of ρ is lexical by the invariant, and no $e_{G_{jk}}$ may exist between ρ and π (by the same reasoning as for (a) above), so all $e_{G_{jk}}$ left of π are lexical.

3. For each element $e_{\Gamma_j} \notin C'$, take the sequence $e_{G_{j1}}, \ldots, e_{G_{jq_j}}$ which satisfies clause (3) of the invariant for the second antecedent.

   (a) The elements left of ρ are also left of $\mathit{next}(\pi)$, so they don't cross it. Those elements right of ρ have as their leftmost position nothing left of $\mathit{next}(\pi)$, since ρ is the rightmost contained position in e_{B_i} left of π, and if π were contained in e_{Γ_j}, then e_{Γ_j} would be in C'.

   (b) $G_{j1}, \ldots, G_{jq_j}$ are the leaves of a partial derivation tree headed by Γ_j since they satisfy (3b) for the second antecedent.

   (c) Any $e_{G_{jk}}$ to the left of ρ is lexical by the invariant, and no $e_{G_{jk}}$ may exist between ρ and $\mathit{next}(\pi)$, by the same reasoning as for (3a) above, so all $e_{G_{jk}}$ left of $\mathit{next}(\pi)$ are lexical.

Because the axioms and rules of inference are sound, the recognizer is sound.

5 Proof of Completeness

The recognizer is complete for an m-mcfg G if it generates a goal item for every sentence $w = w_1 \ldots w_n \in L(G)$. Consider any pair $\langle w, T \rangle$ such that $w \in L(G)$ and T is a derivation tree of w in G. Assume first that T has just one node. In this case w consists of either one or zero words, and in either case the lexicon of G must contain the production $S \to w$, in which case the node is $w : S$. If $w = \epsilon$, the derivation of the goal item is as follows:

$\langle S' \to g_I[(0,0) : S], \{(0,0) : S\}, 0 \rangle$	axiom
$\langle (0,0) : S \to \epsilon, \emptyset, 0 \rangle$	predict
$\langle S' \to g_I[(0,0) : S], \emptyset, 0 \rangle$	up,goal

For $w \neq \epsilon$, the derivation is as follows:

$\langle S' \to g_I[(0,1) : S], \{(0,1) : S\}, 0 \rangle$	axiom
$\langle (0,1) : S \to w, \emptyset, 0 \rangle$	predict
$\langle S' \to g_I[(0,1) : S], \emptyset, 0 \rangle$	up,goal

To deal with a derivation tree T that consists of more than one node, we need to define the following notions. A derivation tree consists of nodes labeled with expressions $e = w_{i_1}, \ldots, w_{i_n} : A$ in precedence and dominance relations with each other, where $A \in N \cup \{\perp\}$. For any expression e as above, define $\Delta(e) = (x_1, y_1), \ldots, (x_{d(A)}, y_{d(A)}) : A$. Note that the use of a terminal production $C \rightarrow w_i$ is denoted by a leaf node labeled $w_i : \perp$ dominated by an interior node $w_i : C$, for which $\Delta(w_i : C) = (i-1, i) : C$.

Let T' be an extended derivation tree obtained from derivation tree T by adding a node labeled S', immediately dominating the root of T.

Given a position π in sentence w, define the set of nodes $N_\pi = \{a | a$ is a non-terminal expression in T', $\pi \in$ *positions*$(e_A = \Delta(a))$; if a immediately dominates b in T', then b is lexical (terminal) or $\pi \notin$ *positions*$(e_B = \Delta(b))\}$. Informally, set N_π consists of the non-terminal nodes in T' whose corresponding subitems contain position π and that are furthest removed from the root S' of the extended derivation tree T'. By the normal form we're using, the effect of this is that N_π consists only of nodes with only a single child node, which is terminal. Let the π-depth d_π of tree T' be $\text{Max}_{l \in N_\pi} d(S', l)$, that is, the length of the longest path from the root of T' to an element of N_π.

For an extended derivation tree T', input sentence $w = w_1 \ldots w_n$, and a value for pointer π, $0 \leq \pi < n$, we inductively define the sets of items $I^{\downarrow}_{k,\pi}$ and $I^{\uparrow}_{k,\pi}$ for $0 \leq k \leq d_\pi$:

1. $I^{\downarrow}_{0,\pi} = \{\langle S' \rightarrow g_I[\Delta(w : S)], \{\Delta(w : S)\}, \pi\rangle\}$.

2. $I^{\downarrow}_{k+1,\pi} = I^{\downarrow}_{k,\pi} \cup \{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, \pi\rangle |$ there are expressions $a, b_i \in T'$ such that $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n(g)$: $d(S', a) = k + 1$; a immediately dominates $b_1, \ldots, b_n$; e_A contains π; $e_{B_i} \in C$ iff e_{B_i} contains π, $1 \leq i \leq n(g)\}$.

and

1. $I^{\uparrow}_{0,\pi} = \{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, \pi\rangle |$ there are expressions $a, b_i \in T'$ such that $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n(g)$: $a \in N_\pi$; a immediately dominates $b_1, \ldots, b_n$; e_A contains $\pi\}$.

2. $I^{\uparrow}_{k+1,\pi} = I^{\uparrow}_{k,\pi} \cup \{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, \pi\rangle |$ there are expressions $a, b_i \in T'$ such that $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n(g)$: $\text{Max}_{l \in N_\pi} d(a, l) = k + 1$; a immediately dominates $b_1, \ldots, b_n$; e_A contains $\pi\} \cup \{\langle S' \rightarrow g_I[\Delta(w : S)], \emptyset, \pi\rangle\}$.

Theorem 1

$\forall \pi, 0 \leq \pi < n$, the recognizer will generate all items in the sets $I^{\downarrow}_{d_\pi,\pi}$ and $I^{\uparrow}_{d_\pi,\pi}$.

Proof

The proof is by induction on π, $0 \leq \pi < n$, and within that, by induction on k, $0 \leq k \leq d_\pi$.

Proof by Induction that the Elements of $I^{\downarrow}_{d_\pi,\pi}$ And $I^{\uparrow}_{d_\pi,\pi}$ Are Generated

Base Case: $I^{\downarrow}_{k,0}$ Generated by induction on k:

Base case The base case here is $k = 0$. Here, the single element of $I^{\downarrow}_{0,0} = \{\langle S' \rightarrow g_I[\Delta(w : S)], \{\Delta(w : S)\}, 0\rangle\}$ is axiomatic. □

Induction Step Assume the contents of the set $I^{\downarrow}_{k-1,0}$ have been generated. The task is then to show that the items of $I^{\downarrow}_{k,0} = I^{\downarrow}_{k-1,0} \cup \{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, 0\rangle$ | $\exists a, b \in T'$ such that for $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n(g)$: $d(S', a) = k$; a immediately dominates $b_1, \ldots, b_{n(g)}$; e_A contains 0; $e_{B_i} \in C$ iff e_{B_i} contains 0, $1 \leq i \leq n(g)\}$ will generated. Pick an arbitrary item $\iota = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, 0\rangle$ from the set $I^{\downarrow}_{k,0}$. If $\iota \in I^{\downarrow}_{k-1,0}$, we are given that ι is generated. For $\iota \notin I^{\downarrow}_{k-1,0}$, there must be an expression $a \in T'$ such that $e_A = \Delta(a)$, for which $d(S', a) = k$, from the definition of the set $I^{\downarrow}_{k,0}$. Also, subitem e_A contains 0. Since $d(S', a) = k > 0$, there must be a node x in T' which immediately dominates expression a. Either x is another expression a', or x is the root node S'. In either case, $d(S', x) = k - 1$. Assume the immediate daughters of x in T' are $b'_1 \ldots b'_m$, and suppose that $a = b'_j$ for some $j, 1 \leq j \leq m$. In the case where $x = a'$, let $e_{A'}$, $e_{B'_i}$ be arbitrary subitems such that $e_{A'} = \Delta(a')$, $e_{B'_i} = \Delta(b'_i)$, $1 \leq i \leq m$, $i \neq j$, and $e_{B'_j} = e_A$. Since expression a' immediately dominates expression a in a complete derivation tree and subitem $e_A = \Delta(a)$ contains position 0, subitem $e_{A'} = \Delta(a')$ must also contain position 0. Then, by the induction hypothesis, the item $\iota' = \langle e_{A'} \rightarrow g'[e_{B'_1} \ldots e_{B_{m=n(g')}}], C', 0\rangle \in I'_{k-1,0}$ will have been generated. Since $e_{B'_j} = e_A$ and e_A contains position 0, $e_{B'_j} = e_A \in C'$. Since there are no positions to the left of 0, position 0 will be leftmost in $e_{B'_j} = e_A$. Therefore the predict rule will apply to ι', producing, among other items, item ι. For $x = S'$, e_A must be $(0, n) : S$, and ι' here is the axiomatic item $\langle S' \rightarrow g_I[(0, n) : S], \{(0, n) : S\}, 0\rangle$, to which predict must apply to produce ι. Since ι was arbitrary, all items in the set $I^{\downarrow}_{k,0}$ will be generated. □

Base Case: $I^{\uparrow}_{k,0}$ Generated by induction on k.

Base Case Here we must show that the items in the set $I^{\uparrow}_{0,0} = \{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, 0\rangle$ | $\exists a, b \in \mathrm{T}'$ such that for $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n$: $a \in N_0$; a immediately dominates $b_1 \ldots b_n$; e_A contains $0\}$ are generated. Pick an arbitrary item $\iota = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, 0\rangle$ out of this set. Expression $a \neq S'$, since $S' \notin N_0$. Since $a \in N_0$, $d(S', a) \leq d_0$. Hence, as shown above, the item $\iota' = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, 0\rangle \in I^{\downarrow}_{d_0,0}$ has been generated, where $e_{B_i} \in C$ iff $0 \in \mathit{positions}(e_{B_i})$, $1 \leq i \leq n(g)$. Since expression a is an element of N_0, $n(g) = 0$, and therefore $C = \emptyset$. Thus, the item ι' is actually indentical to ι. Since ι was arbitrary, all elements of $I^{\uparrow}_{0,0}$ will be generated. □

Induction Step Here we must show that all elements of set $I^{\uparrow}_{k,0}$ will be generated, for $k > 0$, assuming that the contents of the set $I^{\uparrow}_{k-1.0}$ have been generated. Pick an arbitrary item $\iota = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, 0\rangle$ from the set $I^{\uparrow}_{k,0} = I^{\uparrow}_{k-1,0} \cup$

$\{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, 0\rangle \mid \exists a, b_i \in T'$ for $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n(g)$: such that $\text{Max}_{l \in N_0} d(a, l) = k$; a immediately dominates $b_1 \ldots b_{n(g)}$; e_A contains $0\}$ $\cup$ $\{\langle S' \rightarrow g_I[(0, n) : S], \emptyset, 0\rangle\}$, ignoring elements from $I^{\uparrow}_{k-1,0}$. Assume first $e_A \neq S'$. Since $\text{Max}_{l \in N_0} d(a, l) = k > 0$, $d(S', a) < d_0$. Hence, as shown above, there is an item $\iota' = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, 0\rangle \in I^{\downarrow}_{d_0,0}$, where $e_{B_i} \in C$ iff $0 \in \textit{positions}(e_{B_i})$, $1 \leq i \leq n(g)$. If $n(g) = 0$, then $C = \emptyset$, and $\iota = \iota'$. Otherwise, for each $e_{B_i} \in C$, $1 \leq i \leq n(g)$, since b_i is immediately dominated by a, $\text{Max}_{l \in N_0} d(b, l) = k - 1$. By the induction hypothesis, then, an item $\langle e_{B_i} \rightarrow g_i[e_{\Gamma_1} \ldots e_{\Gamma_{n(g_i)}}], \emptyset, 0\rangle \in I^{\uparrow}_{k-1,0}$ has been generated. Thus all conditions for the rule Up to apply to item ι' have been met, and the result will be the item ι. A similar argument will suffice for $e_A = S'$. Since ι was an arbitrary member of set $I^{\uparrow}_{k,0}$, all items of this set will be generated.□

Induction Step, $I^{\downarrow}_{d_k,k}$ Assume that the elements of sets $I^{\downarrow}_{d_p,p}$ and $I^{\uparrow}_{d_p,p}$ have been generated, for $0 \leq p \leq k - 1 < n - 1$, and show that the elements of set $I^{\downarrow}_{d_k,k}$ are generated. This will be done by induction on j, showing that the contents of all sets $I^{\downarrow}_{j,k}$ will be generated, for $0 \leq j \leq d_k$.

Base Case To prove the base case, we must show that the single element of set $I^{\downarrow}_{0,k} = \{\langle S' \rightarrow g_I[(0, n) : S], \{(0, n) : S\}, k\rangle\}$ is generated. Let $\iota = \langle S' \rightarrow g_I[(0, n) : S], \{(0, n) : S\}, k\rangle$. From the induction hypothesis we know that the item $\iota' = \langle S' \rightarrow g_I[(0, n) : S], \emptyset, k\rangle \in I^{\uparrow}_{d_{k-1},k-1}$ has been generated. Since $k - 1 < n - 1$, the rule Next will apply to item ι' to produce item ι, the only element in the set $I^{\downarrow}_{0,k}$. □

Induction Step For the induction step, assume that the elements of $I^{\downarrow}_{j-1,k}$ have been generated for arbitrary j, $1 \leq j \leq d_k$. Then we must show that all elements of the set $I^{\downarrow}_{j,k} = I^{\downarrow}_{j-1,k} \cup \{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, k\rangle \mid \exists a, b_i \in T'$ for $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n(g)$: such that $d(S', a) = j$; a immediately dominates $b_1 \ldots b_{n(g)}$; $k \in \textit{positions}(e_A)$; $e_{B_i} \in C$ iff $k \in \textit{positions}(e_{B_i})\}$ will be generated. Let $\iota = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, k\rangle$ be an arbitrary item in this set such that $\iota \notin I^{\downarrow}_{j-1,k}$. Since $d(S', a) = j > 0$, there must be a node x in T' which immediately dominates expression a. Either x is another expression a', or x is the root node S'. In either case, $d(e_{A'} = \Delta(a'), x) = j - 1$. Assume the immediate daughters of x in T' are $b'_1 \ldots b'_{n(g')}$, and suppose that $a = b'_l$, $1 \leq l \leq n(g')$. In the case where $x = a'$, let $e_{A'}$, $e_{B'_i}$ be arbitrary subitems such that $e_{A'} = \Delta(a')$, $e_{B'_i} = \Delta(b'_i)$, $1 \leq i \leq n(g')$, $i \neq j$, and $e_{B'_l} = e_A$. Since expression a' immediately dominates expression a in a complete derivation tree and $k \in \textit{positions}(e_A)$, subitem $e_{A'}$ must also contain position k. Then, by the induction hypothesis, the item $\iota' = \langle e_{A'} \rightarrow g'[e_{B'_1} \ldots e_{B'_{n(g')}}], C', k\rangle \in I^{\downarrow}_{j-1,k}$ will have been generated. Since $e_{B'_l} = e_A$ and e_A contains position k, $e_{B'_l} = e_A \in C'$. If position k is leftmost in $e_{B'_l}$, then the Predict rule will apply, and among the results will be item ι, as was shown in the proof that the items in $I^{\downarrow}_{k,0}$ are generated. If k is not leftmost in $e_{B'_l}$, then let ρ be the rightmost position to the left of k contained in the position vectors of $e_{B'_l}$. Obviously, $\rho < k$. Ergo, by the induction hypothesis, the item ι''= $\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, q\rangle \in I^{\uparrow}_{d_q,q}$ has been generated. The Down rule will apply to items i' and i'' and produce item ι. For

the case that expression a is immediately dominated by node S', a similar argument can be constructed, involving items $\iota' = \langle S' \rightarrow g_I[e_A = (0,n) : S], \{e_A\}, k\rangle \in I^{\downarrow}_{0,k}$ and $\iota'' = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, q\rangle \in I^{\uparrow}_{d_q,q}$. Item ι was arbitrary, so all items of set $I^{\downarrow}_{j,k}$ will be generated. □

Induction Step, $I^{\uparrow}_{d_k,k}$ Assume that the elements of sets $I^{\downarrow}_{d_p,p}$ and $I^{\uparrow}_{d_p,p}$ have been generated, for $0 \leq p \leq k-1 < n-1$, and show that the elements of set $I^{\uparrow}_{d_k,k}$ are generated. This will be done by induction on j, showing that the contents of all sets $I^{\uparrow}_{j,k}$ will be generated, for $0 \leq j \leq d_k$.

Base Case Here the relevant elements are those of the set $I^{\uparrow}_{0.k} = \{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, k\rangle \mid \exists a, b_i \in T'$ for $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n(g)$: such that $a \in N_k$; a immediately dominates $b_1, \ldots, b_{n(g)}$; e_A contains $k\}$. Pick an arbitrary item $\iota = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, k\rangle$ out of this set. Since $a \in N_k$, $a \neq S'$, and $d(S', a) \leq d_k$. Hence, as shown above, the item $\iota' = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, k\rangle \in I^{\downarrow}_{d_k,k}$ has been generated, where $e_{B_i} \in C$ iff $0 \in \mathit{positions}(e_{B_i})$, $1 \leq i \leq n(g)$. Since expression a is an element of N_k, $n(g) = 0$, and therefore $C = \emptyset$. Thus, the item ι' is actually indentical to ι. Since ι was arbitrary, all elements of $I^{\uparrow}_{0,k}$ will be generated. □

Induction Step Here we must show that all elements of set $I^{\uparrow}_{j,k}$ will be generated, for $j > 0$, assuming that the contents of the set $I^{\uparrow}_{j-1.k}$ have been generated. Pick an arbitrary item $\iota = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, k\rangle$ from the set $I^{\uparrow}_{j,k} = I^{\uparrow}_{k-1,k} \cup \{\langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], \emptyset, k\rangle \mid \exists a, b_i \in T'$ for $e_A = \Delta(a)$, $e_{B_i} = \Delta(b_i)$, $1 \leq i \leq n(g)$: such that $\mathrm{Max}_{l \in N_k} d(a,l) = j$; a immediately dominates $b_1 \ldots b_{n(g)}$; e_A contains $k\}$ $\cup \{\langle S' \rightarrow g_I[(0,n) : S], \emptyset, k\rangle\}$, ignoring elements from $I^{\uparrow}_{j-1,k}$. Assume first $e_A \neq S'$. Since $\mathrm{Max}_{l \in N_k} d(a,l) = j > 0$, $d(S', a) < d_k$. Hence, as shown above, there is an item $\iota' = \langle e_A \rightarrow g[e_{B_1} \ldots e_{B_{n(g)}}], C, k\rangle \in I^{\downarrow}_{d_k,k}$, where $e_{B_i} \in C$ iff $0 \in \mathit{positions}(e_{B_i})$, $1 \leq i \leq n(g)$. If $n(g) = 0$, then $C = \emptyset$, and $\iota = \iota'$. Otherwise, for each $e_{B_i} \in C$, $1 \leq i \leq n(g)$, since b_i is immediately dominated by a, $\mathrm{Max}_{l \in N_k} d(b,l) = j - 1$. By the induction hypothesis, then, an item $\langle e_{B_i} \rightarrow g_i[e_{\Gamma_1} \ldots e_{\Gamma_{n(g_i)}}], \emptyset, k\rangle \in I^{\uparrow}_{j-1,k}$ has been generated. Thus all conditions for the rule UP to apply to item ι' have been met, and the result will be the item ι. A similar argument will suffice for $e_A = S'$. Since ι was an arbitrary member of set $I^{\uparrow}_{j,k}$, all items of this set will be generated.□

This concludes the inductive step of the proof of Theorem 1 and so completes the proof of this theorem. □

Completeness now follows as a corollary from Theorem 1: by the definition of $I^{\uparrow}_{d_{n-1},n-1}$, if $w \in L(G)$, $w = w_1 \ldots w_n$, then a goal item $\langle S' \rightarrow g_I[(0,n) : S], \emptyset, n-1\rangle$ is an element of set $I^{\uparrow}_{d_{n-1},n-1}$.

6 Complexity

For this section we will assume the normal form given in §2.3, with the following additional restriction:

For a nonterminating rule, the right-hand side can contain no more than two elements.

The normal form used is thus equivalent to that given in Nakanishi *et al.* 1997. The restriction is not necessary for the definition of the algorithm above, but it greatly adds to the efficiency of the algorithm.

6.1 Steps Overall

Overall, the algorithm takes each item, as it is added to the chart, and checks whether the rules NEXT, PREDICT, UP, and DOWN are triggered by it. In particular, for each item with $C = \emptyset$, we check whether it is an argument to NEXT, or UP. For items with $C \neq \emptyset$, we check for applicability of PREDICT and DOWN. Thus, for each item in the chart we check applicability of rules of inference, apply those rules, and check whether the output of those rules is in the chart. Our complexity measure will be based on the variables n, the length of the input sentence, and m, the maximum arity of the grammar.

First, we need to estimate the number of items in the chart. An item is maximally of the form

$$\langle (p_1, q_1) \dots (p_j, q_j) : A \rightarrow g[(p_{11}, q_{11}) \dots (p_{1k}, q_{1k}) : B_1, (p_{21}, q_{21}) \dots (p_{2l}, q_{2l}) : B_2], C, \pi \rangle$$

and the primary metric for the number of items is the assignment of p, q values. The maximally complex p, q value assignment is in the case where B_1 and B_2 both exist, and the concatenation represented by g^h for each h where $1 \leq h \leq d(A)$ is comprised of exactly two elements. In this case, for each h the p, q assignment is a choice of 3 items from a set of n positions, unordered, without repetition. There are at most m values for h, so the number of possible p, q assignments is maximally $\binom{n}{3}^m$. Given a particular value for π (of which there are n possible values, there are only two possible values for C, so the number of chart items is $O(n\binom{n}{3}^m)$, which is $O(n^{3m+1})$, very roughly.

6.2 Steps per item

6.2.1 Next

When NEXT applies it is not necessary to check whether the resulting item is already in the chart. The check and application of NEXT are all $O(1)$, so NEXT as a whole is $O(1)$.

6.2.2 Predict

PREDICT can create on the order of $O(n^m)$ new items (although in practice the number is considerably smaller), each of which must be checked for addition to the chart. The check is $O(\log\binom{n}{3}^m)$, so overall predict is $O(n^m \log\binom{n}{3}^m)$. In some grammars it may be the case that items will not need to be checked for addition; in these cases, predict is simply $O(n^m)$.

6.2.3 Up

UP must look up at most two elements in the chart, a procedure which takes $O(\log(\frac{n}{3})^m)$ steps. The result must then be checked for addition to the chart, which takes $O(\log(\frac{n}{3})^m)$ steps, so UP is $O(\log(\frac{n}{3})^m)$.

6.2.4 Down

DOWN needs to do at most two lookups of $O(\log(\frac{n}{3})^m)$ steps, which may return up to $O(n^m)$ results. As with PREDICT, these items will need to be checked for addition to the chart. Thus DOWN is $O(\log(\frac{n}{3})^m + n^m \log(\frac{n}{3})^m)$, which is $O(n^m \log(\frac{n}{3})^m)$, for non-degenerate grammars.

6.3 Combined complexity

Since there are $O(n(\frac{n}{3})^m)$ items in the chart and the most expensive operation is $O(n^m \log(\frac{n}{3})^m)$, recognition is $O(n^{m+1}(\frac{n}{3})^m \log(\frac{n}{3})^m)$, which is strictly less than $O(n^{4m+1} \log n^{3m})$. Note that these numbers were obtained under the assumption that the chart is implemented as some set of balanced binary trees (this is why lookup is taken to be logarithmic with the number of elements being searched). The chart could instead be implemented as a hash table, in which case the worst-case time would be $O(n^{m+1}(\frac{n}{3})^{2m})$, but the average-case time would be $O(n^{m+1}(\frac{n}{3})^m)$.

6.4 Optimizations

The real-world speed of this algorithm can be improved in a number of (possibly grammar-specific) ways. First, grammars can be analyzed to reveal the maximum and mimimum spread between p and q values for some categories. For example, if the only rule headed by A is $A \rightarrow b$, where b is a terminal element, then for any item mentioning A, the p and q values used in conjunction with A must be exactly one position apart. In practice, the effect of this optimization is to take PREDICT and NEXT from $O(n^m \log(\frac{n}{3})^m)$ to $O(\log(\frac{n}{3})^m)$ in the usual case, making the average time $O(n(\frac{n}{3})^m \log(\frac{n}{3})^m)$ (with a hash-table implementation, the average time would be $O(n(\frac{n}{3})^m)$).

It should also be possible to determine for most categories whether the result of a PREDICT or DOWN application with the category as a head of the resulting item will need to be checked for prior inclusion in the chart or not. This should lead to an average time of $O(n^m)$ for PREDICT and DOWN, or even $O(1)$, if the previous optimization is used. This would not affect the overall timing if the min/max p, q spread optimization is used, but would probably improve actual efficiency.

6.5 Experimental Results

The algorithm (using a hashed chart and specified minimum and maximum $p - q$ spreads where possible) was coded in the programming language Python and applied to parse randomly generated sentences of increasing lengths (from 8 words to 392 words, by 8) following a particular grammar (a 2-mcfg designed to model phonological prefix reduplication). The run times, on a 1GHz AMD Duron machine (bearing in mind that Python is the world's least efficient language), were as follows, in seconds: 0.05, 0.13, 0.2, 0.33, 0.48, 0.63, 0.88, 1.13, 1.49, 1.85, 2.41, 2.87, 3.47, 4.45,

5.01, 5.84, 6.96, 8.04, 9.19, 10.57, 12.0, 13.39, 15.04, 17.13, 18.16, 21.35, 23.53, 25.83, 29.06, 31.24, 34.73, 37.78, 41.43, 44.71, 48.75, 52.64, 56.7, 61.08, 66.05, 70.9, 76.14, 80.14, 87.38, 93.27, 100.37, 106.34, 113.23, 119.65, 127.18. This appears to represent an order of growth somewhere between $O(n^2)$ and $O(n^3)$, possibly around $O(n^{2.2})$.

References

Harkema, Hendrik, 2001. *Parsing Minimalist Languages*. UCLA dissertation.

Nakanishi, Ryuichi, Keita Takada, and Hiroyuki Seki. 1997. An efficient recognition algorithm for multiple context-free languages. In *Proceedings of the Fifth Meeting on Mathematics of Language*, 119–123, Saarbrücken, Germany.

Seki, Hiroyuki, Takashi Masumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science* 88.