

## INFORMATION TO USERS

While the most advanced technology has been used to photograph and reproduce this manuscript, the quality of the reproduction is heavily dependent upon the quality of the material submitted. For example:

- Manuscript pages may have indistinct print. In such cases, the best available copy has been filmed.
- Manuscripts may not always be complete. In such cases, a note will indicate that it is not possible to obtain missing pages.
- Copyrighted material may have been removed from the manuscript. In such cases, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, and charts) are photographed by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each oversize page is also filmed as one exposure and is available, for an additional charge, as a standard 35mm slide or as a 17"x 23" black and white photographic print.

Most photographs reproduce acceptably on positive microfilm or microfiche but lack the clarity on xerographic copies made from the microfilm. For an additional charge, 35mm slides of 6"x 9" black and white photographic prints are available for any photographs or illustrations that cannot be reproduced satisfactorily by xerography.

---



8716816

Schoenberger, Annette Virginia Dittmer

OI AND IO MULTIDIMENSIONAL FOREST LANGUAGES

*Iowa State University*

PH.D. 1987

University  
Microfilms  
International 300 N. Zeeb Road, Ann Arbor, MI 48106



**PLEASE NOTE:**

In all cases this material has been filmed in the best possible way from the available copy.  
Problems encountered with this document have been identified here with a check mark ✓.

1. Glossy photographs or pages \_\_\_\_\_
2. Colored illustrations, paper or print \_\_\_\_\_
3. Photographs with dark background \_\_\_\_\_
4. illustrations are poor copy \_\_\_\_\_
5. Pages with black marks, not original copy \_\_\_\_\_
6. Print shows through as there is text on both sides of page \_\_\_\_\_
7. Indistinct, broken or small print on several pages ✓
8. Print exceeds margin requirements \_\_\_\_\_
9. Tightly bound copy with print lost in spine \_\_\_\_\_
10. Computer printout pages with indistinct print \_\_\_\_\_
11. Page(s) \_\_\_\_\_ lacking when material received, and not available from school or author.
12. Page(s) \_\_\_\_\_ seem to be missing in numbering only as text follows.
13. Two pages numbered \_\_\_\_\_. Text follows.
14. Curling and wrinkled pages \_\_\_\_\_
15. Dissertation contains pages with print at a slant, filmed as received ✓
16. Other \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

University  
Microfilms  
International



***OI and IO multidimensional forest languages***

by

**Annette Virginia Dittmer Schoenberger**

**A Dissertation Submitted to the  
Graduate Faculty in Partial Fulfillment of the  
Requirements for the Degree of  
DOCTOR OF PHILOSOPHY  
Major: Computer Science**

**Approved:**

Signature was redacted for privacy.  
\_\_\_\_\_

Signature was redacted for privacy.

**In Charge of Major Work**

Signature was redacted for privacy.

**For the Major Department**

Signature was redacted for privacy.  
\_\_\_\_\_

**For the Graduate College**

**Iowa State University**

**Ames, Iowa**

**1987**

## TABLE OF CONTENTS

<b>1 A COMPARISON OF THE ENGELFRIET AND SCHMIDT HIERARCHY WITH THE MULTI-DIMENSIONAL FOREST HIERARCHY</b>	<b>1</b>
1.1 Trees and Substitution . . . . .	2
1.2 Algebras for Tree Languages . . . . .	8
1.2.1 $\Sigma$ -algebras and $D(\Sigma)$ -algebras . . . . .	8
1.2.2 Continuous $\Sigma$ -algebras . . . . .	12
1.2.3 Context-free tree grammars . . . . .	14
1.3 $n$ -Dimensional Forests, Grammars, and Derivations . . . . .	18
1.3.1 $n$ -Dimensional trees and forests . . . . .	19
1.3.2 The $n$ -ary tree representation for $n$ -dimensional forests of degree $k$ and the depth function . . . . .	22
1.3.3 Paths in $n$ -dimensional forests . . . . .	23
1.3.4 $n$ -Dimensional forest substitution . . . . .	25
1.3.5 The frontier of an $n$ -dimensional forest . . . . .	33
1.3.6 $n$ -Dimensional grammars . . . . .	36
1.3.7 $n$ -Dimensional $OI$ and $IO$ derivations . . . . .	39

---

1.4	Containment of the Engelfriet-Schmidt Hierarchy in the Baldwin-Schoenberger Hierarchy . . . . .	44
1.4.1	Conversion of the projections, $\pi_i^w$ , into path variables, $x_k^p$ . . . .	44
1.4.2	Conversion of Engelfriet-Schmidt trees to Baldwin-Schoenberger forests . . . . .	46
1.4.3	Construction of a Baldwin-Schoenberger forest grammar from an Engelfriet-Schmidt tree grammar . . . . .	51
1.4.4	The string languages are equivalent . . . . .	58
<b>2</b>	<b>THE RELATIONSHIP BETWEEN THE <i>IO</i> AND <i>OI</i> <i>N</i>-DIMENSIONAL LANGUAGE HIERARCHIES</b>	<b>67</b>
2.1	An <i>IO</i> Macro Language Which is Not <i>OI</i> . . . . .	68
2.2	<i>IO</i> Dimension $n$ Is a Subset of <i>OI</i> Dimension $(n + 1)$ . . . . .	71
2.2.1	The Baldwin derivation relation and its relation to the Baldwin-Schoenberger derivation relation . . . . .	71
2.2.2	An <i>OI</i> language which has no <i>IO</i> grammar at any dimension .	75
2.3	<i>OI</i> 3-Dimensional = <i>OI</i> Macro . . . . .	86
<b>3</b>	<b>MORE ABOUT THE NATURE OF THE <i>OI</i> LANGUAGES</b>	<b>97</b>
3.1	<i>OI</i> Completed Grammars and Related Theorems . . . . .	97
3.2	Closure Properties for the <i>OI</i> Languages . . . . .	106
3.3	Removal of Dead Symbols from the <i>OI</i> Grammars . . . . .	115
3.4	A Further Extension of the $n$ -Dimensional Grammars . . . . .	123
<b>4</b>	<b>CONCLUSIONS AND FUTURE WORK</b>	<b>128</b>
<b>5</b>	<b>BIBLIOGRAPHY</b>	<b>130</b>

## LIST OF FIGURES

Figure 1.1	Pictorial Representation of a Forest . . . . .	23
Figure 1.2	Substitution of a Set into a Forest . . . . .	24
Figure 1.3	Substitution of a Forest into a Forest . . . . .	25
Figure 1.4	Substitution of a Set into a Forest . . . . .	27
Figure 1.5	Substitution of a Set into a Set . . . . .	28
Figure 1.6	Nonassociativity of Substitution in General . . . . .	29
Figure 1.7	Computation of a 2-dimensional frontier . . . . .	35
Figure 1.8	Computation of the 1-dimensional frontier . . . . .	37
Figure 1.9	Computation of $f(S)$ using $OI$ substitution . . . . .	41
Figure 1.10	Completion of $f(S)$ using $OI$ substitution . . . . .	42
Figure 1.11	Computation of $f(A _1 x_1^\lambda)$ using $OI$ substitution . . . . .	43
Figure 1.12	$f(F _2 * _2 x_2^\lambda _1 x_2^\lambda _1 x_1^\lambda)     x_1^\lambda$ using $OI$ substitution . . .	45

# 1 A COMPARISON OF THE ENGELFRIET AND SCHMIDT HIERARCHY WITH THE MULTI-DIMENSIONAL FOREST HIERARCHY

Engelfriet and Schmidt defined a hierarchy of tree languages in [24,25] which contains the regular and the context-free tree languages as levels 0 and 1. This hierarchy is actually two hierarchies, the inside-out and the outside-in, and is an extension of the usual 2-dimensional trees over an  $S$ -sorted alphabet  $\Sigma$ . Maibaum [43] first introduced this generalized approach to formal languages but he confused inside-out and outside-in. The inside-out and outside-in modes of derivation were originally described by Fischer [27] for use with his macro grammars. The work of Engelfriet and Schmidt is an attempt to rectify the confusion of Maibaum by extending the general theory of equational subsets of an arbitrary algebra as originally developed by Mezei and Wright [47] and Thatcher and Wright [57]. This theory is further illuminated in Goguen *et al.* [34] where the authors show the existence of initial continuous algebras and use them to directly solve systems of regular equations.

Baldwin [12] describes a hierarchy of forest languages. Levels 2 and 3 of this hierarchy correspond to the regular and inside-out context-free tree languages. The level 1 frontiers of the level 1, 2, and 3 forest languages correspond to the regular, context-free, and inside-out macro string languages. We have extended the outside-in languages in the same way that Baldwin extended the inside-out languages.

Section 1 of this chapter contains the definitions of trees over  $S$ -sorted algebras and tree substitution. In section 2 of this chapter we describe the hierarchy of Engelfriet and Schmidt. Section 3 contains a description of the Baldwin inside-out forest language hierarchy and the corresponding outside-in forest language hierarchy. We call this latter hierarchy the Baldwin-Schoenberger hierarchy. Finally in section 4 we show that the Engelfriet-Schmidt hierarchy is completely contained in the Baldwin-Schoenberger hierarchy.

### 1.1 Trees and Substitution

By an  $S$ -sorted alphabet  $\Sigma$ , we mean a family of sets  $\Sigma = \langle \Sigma_{\langle \omega, s \rangle} \rangle_{\langle \omega, s \rangle \in S^* \times S}$  for  $S$  a set of *sorts*. If  $f \in \Sigma_{\langle \omega, s \rangle}$ , then we say that  $f$  is an *operator* of *type*  $\langle \omega, s \rangle$  with *arity*  $\omega$  and *sort*  $s$ . For the  $S$ -sorted alphabet  $\Sigma$ , the *set of trees over  $\Sigma$  of sort  $s$* ,  $T_{\Sigma, s}$ , is defined to be the smallest set of strings over  $\Sigma \cup \{ (, ) \}$  such that:

1.  $\Sigma_{\langle \lambda, s \rangle} \subseteq T_{\Sigma, s}$ ;
2. if  $f \in \Sigma_{\langle \omega, s \rangle}$ , and  $t_i \in T_{\Sigma, \omega_i}$ , then  $f(t_1 \cdots t_{|\omega|}) \in T_{\Sigma, s}$ .

Then we define  $T_{\Sigma}$ , the *set of all trees over  $\Sigma$  with sorts from  $S$*  as  $\bigcup_{s \in S} T_{\Sigma, s}$ . If  $L$  is a subset of  $T_{\Sigma}$ , then we call it a  $\Sigma$ -tree language.

Two functions of interest to us are the *yield*, or *frontier* function, and the *depth* function. The *yield* function  $y: T_{\Sigma} \rightarrow \Sigma^*$  is defined recursively by:

1. if  $t \in \Sigma_{\langle \lambda, s \rangle}$ , then  $y(t) = t$ ;
2. if  $f(t_1 \cdots t_{|\omega|}) \in T_{\Sigma}$ , then  $y(f(t_1 \cdots t_{|\omega|})) = y(t_1) \cdots y(t_{|\omega|})$ .

The *depth* function  $d: T_{\Sigma} \rightarrow \mathcal{N}$  is defined recursively by:

1. if  $t \in \Sigma_{\langle \lambda, s \rangle}$ , then  $d(t) = 1$ ;
2. if  $f(t_1 \cdots t_{|\omega|}) \in T_\Sigma$ , then  $d(f(t_1 \cdots t_{|\omega|})) = 1 + \max\{d(t_i) \mid 1 \leq i \leq |\omega|\}$ .

If  $Y = \langle Y_s \rangle_{s \in S}$  is a family of disjoint sets of *variables* of sort  $s$ , we can define the family of trees over  $\Sigma$  with variables from  $Y$ ,  $T_{\Sigma(Y)}$ , where  $\Sigma(Y)$  is the  $S$ -sorted alphabet with  $\Sigma(Y)_{\langle \lambda, s \rangle} = \Sigma_{\langle \lambda, s \rangle} \cup Y_s$  and  $\Sigma(Y)_{\langle \omega, s \rangle} = \Sigma_{\langle \omega, s \rangle}$  for  $\omega \in S^+$ . In other words, the variables from  $Y$  are treated as constants of sort  $S$  in the construction of the trees.  $T_{\Sigma(Y)}$  will also be denoted by  $T_\Sigma(Y)$ . Let  $X_S = \{x_{i,s} \mid i \geq 1 \text{ and } s \in S\}$  be a set of “sorted” variables. For  $\omega \in S^*$ , define  $X_\omega = \{x_{i,\omega_i} \mid 1 \leq i \leq |\omega|\}$ .  $X_\omega$  is the subset of  $X_S$  which describes  $\omega$  by specifying exactly those objects from  $S$  comprising  $\omega$  and their positions in  $\omega$ . Finally, define  $(X_\omega)_s = \{x_{i,\omega_i} \mid \omega_i = s\}$ . For example, if  $S = \{a, b, c\}$  then  $X_{abccbb} = \{x_{1,a}, x_{2,b}, x_{3,c}, x_{4,c}, x_{5,b}, x_{6,b}\}$  and  $(X_{abccbb})_b = \{x_{2,b}, x_{5,b}, x_{6,b}\}$ . So if we let  $Y_s = (X_\omega)_s$ , then  $Y = \langle Y_s \rangle_{s \in S}$  is a disjoint family of sets and we can obtain  $T_\Sigma(X_\omega)$ , the set of *trees over  $\Sigma$  with variables from  $X_\omega$* .  $T_\Sigma(X_\omega)_s$  is then defined as the set of *trees over  $\Sigma$  of sort  $s$  with variables from  $X_\omega$* .

Using these variables, we define substitution of trees into a tree as follows:

**Definition 1** For  $\omega, \nu \in S^*$ ,  $t \in T_\Sigma(X_\omega)_s$ , and  $t_i \in T_\Sigma(X_\nu)_{\omega_i}$ , then  $t[t_1, \dots, t_{|\omega|}]$  denotes the result of substituting  $t_i$  for  $x_{i,\omega_i}$  in  $t$ .

□

Note that  $t[t_1, \dots, t_{|\omega|}]$  is in  $T_\Sigma(X_\nu)_s$  and that for  $\omega = \lambda$ ,  $t[t_1, \dots, t_{|\omega|}] = t[\ ] = t$ . The example below demonstrates the use of this definition with three simple trees. Notice that the tree being substituted into has two occurrences of the variable  $x_{1,s}$ .

**Example 1** Let  $S \in \{s\}$ ,  $\Sigma_{\langle \lambda, s \rangle} = \{b, c\}$ ,  $\Sigma_{\langle ss, s \rangle} = \{a\}$ ,  $\Sigma_{\langle sss, s \rangle} = \{d\}$ ,  
 $X_{ss} = \{x_{1,s}, x_{2,s}\}$ ,  $t = d(x_{1,s} x_{1,s} x_{2,s})$ ,  $t_1 = a(bb)$ , and  $t_2 = a(cc)$ , then

$$\begin{aligned} t[t_1, t_2] &= d(x_{1,s} x_{1,s} x_{2,s})[a(bb), a(cc)] \\ &= d(a(bb) a(bb) a(cc)) \end{aligned}$$

□

For the substitution of *sets* of trees into a tree, there are two different definitions. These definitions reflect the two different strategies which may be followed if the tree being substituted into contains more than one occurrence of any variable  $(x_{i,\omega_i})$ . Since the objects which replace the variables are now to be chosen from a set, the question is whether or not each  $x_{i,\omega_i}$  is to be replaced by the same tree. If we require that they be replaced by the same tree, then the substitution is called *Inside-Out (IO) substitution*. On the other hand, if they may be replaced by different trees then it is called *Outside-In (OI) substitution*.

**Definition 2** Let  $\omega, \nu \in S^*$ ,  $t \in T_\Sigma(X_\omega)_s$ , and  $L_i \subseteq T_\Sigma(X_\nu)_{\omega_i}$ .

The IO Substitution of  $L_1, \dots, L_{|\omega|}$  into  $t$ , denoted by  $t \stackrel{\text{IO}}{=} (L_1, \dots, L_{|\omega|})$ , is defined to be the tree language  $\{t[t_1, \dots, t_{|\omega|}] \mid t_i \in L_i \text{ for } 1 \leq i \leq |\omega|\}$ .

The OI Substitution of  $L_1, \dots, L_{|\omega|}$  into  $t$ , denoted by  $t \stackrel{\text{OI}}{=} (L_1, \dots, L_{|\omega|})$ , is defined inductively as follows:

1. for  $t \in \Sigma_{\langle \lambda, s \rangle}$ ,  $t \stackrel{\text{OI}}{=} (L_1, \dots, L_{|\omega|}) = \{t\}$ ;
2. for  $t = x_{i,\omega_i}$ ,  $1 \leq i \leq |\omega|$ ,  $t \stackrel{\text{OI}}{=} (L_1, \dots, L_{|\omega|}) = L_i$ ;
3. for  $t = f(t_1 \cdots t_{|\nu|})$ ,  $f \in \Sigma_{\langle \nu, s \rangle}$ ,  $\nu \in S^*$ ,  $t_i \in T_\Sigma(X_\omega)_{\nu_i}$ ,  $t \stackrel{\text{OI}}{=} (L_1, \dots, L_{|\omega|}) = \{f(s_1 \cdots s_{|\nu|}) \mid 1 \leq i \leq |\nu|, s_i \in (t_i \stackrel{\text{OI}}{=} (L_1, \dots, L_{|\omega|}))\}$ .

□

The following example shows that these two types of substitution are indeed different.

**Example 2** Let  $S = \{s\}$ ,  $\Sigma_{\langle \lambda, s \rangle} = \{b, c\}$ ,  $\Sigma_{\langle ss, s \rangle} = \{a\}$ ,  $X_s = \{x_{1,s}\}$ ,  $t = a(x_{1,s}x_{1,s})$ , and  $L_1 = \{b, c\}$ .

Then the IO substitution of  $L_1$  into  $t$  is  $\{a(bb), a(cc)\}$  since each  $x_{1,s}$  must be replaced by the same item from the set.

The OI substitution of  $L_1$  into  $t$  is  $\{a(bb), a(bc), a(cc), a(cb)\}$ , since each  $x_{1,s}$  may be replaced by a different item from the set.

□

Sets of trees can also be substituted into sets of trees. If  $L$  is a set of trees with variables from  $X_\omega$  and  $L_1, \dots, L_{|\omega|}$  are sets of trees with each tree from the set  $L_i$  having sort  $\omega_i$ , then we can substitute  $L_1, \dots, L_{|\omega|}$  into  $L$  by substituting into each tree from  $L$  individually. Formally, we define the substitution of  $L_1, \dots, L_{|\omega|}$  into  $L$  as follows:

**Definition 3** Let  $\omega, \nu \in S^*$ ,  $L \subseteq T_\Sigma(X_\omega)$ , and  $L_i \subseteq T_\Sigma(X_\nu)_{\omega_i}$ .

The IO Substitution of  $L_1, \dots, L_{|\omega|}$  into  $L$ , denoted by  $L \stackrel{\Leftarrow}{\underset{IO}{\hookrightarrow}} (L_1, \dots, L_{|\omega|})$ , is defined to be the tree language

$$\bigcup_{t \in L} \left( t \stackrel{\Leftarrow}{\underset{IO}{\hookrightarrow}} (L_1, \dots, L_{|\omega|}) \right).$$

The OI Substitution of  $L_1, \dots, L_{|\omega|}$  into  $L$ , denoted by  $L \stackrel{\Leftarrow}{\underset{OI}{\hookrightarrow}} (L_1, \dots, L_{|\omega|})$ , is defined to be the tree language

$$\bigcup_{t \in L} \left( t \stackrel{\Leftarrow}{\underset{OI}{\hookrightarrow}} (L_1, \dots, L_{|\omega|}) \right).$$

□

**Example 3** Let  $S = \{s, u\}$ ,  $\Sigma_{\langle \lambda, s \rangle} = \{b, c\}$ ,  $\Sigma_{\langle ss, u \rangle} = \{a\}$ ,  $\Sigma_{\langle us, s \rangle} = \{d\}$ ,  $X_{ss} = \{x_{1,s}, x_{2,s}\}$ ,  $X_{us} = \{x_{1,u}, x_{2,s}\}$ , then

$$\begin{aligned}
& \{d(a(x_{1,s} x_{2,s}) x_{1,s}), a(x_{2,s} x_{2,s})\} \xrightarrow{IO} (\{b, c\}, \{c\}) \\
&= \left\{ d(a(x_{1,s} x_{2,s}) x_{1,s}) \xrightarrow{IO} (\{b, c\}, \{c\}), a(x_{2,s} x_{2,s}) \xrightarrow{IO} (\{b, c\}, \{c\}) \right\} \\
&= \{d(a(x_{1,s} x_{2,s}) x_{1,s})|c, c|, d(a(x_{1,s} x_{2,s}) x_{1,s})|b, c|, \\
&\quad a(x_{2,s} x_{2,s})|c, c|, a(x_{2,s} x_{2,s})|b, c|\} \\
&= \{d(a(bc)b), d(a(cc)c), a(cc)\}
\end{aligned}$$

and

$$\begin{aligned}
& \{d(a(x_{1,s} x_{2,s}) x_{1,s}), a(x_{2,s} x_{2,s})\} \xrightarrow{OI} (\{b, c\}, \{c\}) \\
&= \left\{ d(a(x_{1,s} x_{2,s}) x_{1,s}) \xrightarrow{OI} (\{b, c\}, \{c\}) \right\} \cup \left\{ a(x_{2,s} x_{2,s}) \xrightarrow{OI} (\{b, c\}, \{c\}) \right\} \\
&= \left\{ d(a(s_1 s_2) s_3) \mid s_1 \in x_{1,s} \xrightarrow{OI} (\{b, c\}, \{c\}), s_2 \in x_{2,s} \xrightarrow{OI} (\{b, c\}, \{c\}), \right. \\
&\quad \left. s_3 \in x_{1,s} \xrightarrow{OI} (\{b, c\}, \{c\}) \right\} \\
&\quad \cup \left\{ a(s_1 s_2) \mid s_1 \in x_{2,s} \xrightarrow{OI} (\{b, c\}, \{c\}), s_2 \in x_{2,s} \xrightarrow{OI} (\{b, c\}, \{c\}) \right\} \\
&= \{d(a(s_1 s_2) s_3) \mid s_1 \in \{b, c\}, s_2 \in \{c\}, s_3 \in \{b, c\}\} \\
&\quad \cup \{a(s_1 s_2) \mid s_1 \in \{c\}, s_2 \in \{c\}\} \\
&= \{d(a(bc)b), d(a(bc)c), d(a(cc)c), d(a(cc)b), a(cc)\}.
\end{aligned}$$

□

Note that if each  $L_i$  has only one item in it, then  $L \xrightarrow{OI} (L_1, \dots, L_{|w|})$  is the same as  $L \xrightarrow{IO} (L_1, \dots, L_{|w|})$  since, if a variable occurs more than once in a tree in  $L$ , then all occurrences of a given variable will be replaced by the same tree. The following example illustrates that  $IO$  substitution is not associative while  $OI$  substitution is.

**Example 4**  $S = \{s\}$ ,  $\Sigma_{(\lambda,s)} = \{b, c\}$ ,  $\Sigma_{(ss,s)} = \{a\}$ ,  $X_{ss} = \{x_{1,s}, x_{2,s}\}$ . Then the *IO* substitution gives:

$$\begin{aligned} (a(x_{1,s}, x_{2,s}) \xleftarrow{IO} (\{x_{1,s}\}, \{x_{1,s}\})) &\xleftarrow{IO} (\{b, c\}) \\ &= \{a(x_{1,s}, x_{1,s})\} \xleftarrow{IO} (\{b, c\}) \\ &= \{a(bb), a(cc)\} \end{aligned}$$

while

$$\begin{aligned} a(x_{1,s}, x_{2,s}) &\xleftarrow{IO} (\{x_{1,s}\} \xleftarrow{IO} (\{b, c\}), \{x_{1,s}\} \xleftarrow{IO} (\{b, c\})) \\ &= a(x_{1,s}, x_{2,s}) \xleftarrow{IO} (\{b, c\}, \{b, c\}) \\ &= \{a(bb), a(bc), a(cc), a(cb)\} \end{aligned}$$

and the *OI* substitution gives:

$$\begin{aligned} (a(x_{1,s}, x_{2,s}) \xleftarrow{OI} (\{x_{1,s}\}, \{x_{1,s}\})) &\xleftarrow{OI} (\{b, c\}) \\ &= \{a(x_{1,s}, x_{1,s})\} \xleftarrow{OI} (\{b, c\}) \\ &= \{a(s_1 s_2) \mid s_1 \in (x_{1,s} \xleftarrow{OI} \{b, c\}), s_2 \in (x_{1,s} \xleftarrow{OI} \{b, c\})\} \\ &= \{a(bb), a(bc), a(cc), a(cb)\} \end{aligned}$$

while

$$\begin{aligned} a(x_{1,s}, x_{2,s}) &\xleftarrow{OI} (\{x_{1,s}\} \xleftarrow{OI} (\{b, c\}), \{x_{1,s}\} \xleftarrow{OI} (\{b, c\})) \\ &= a(x_{1,s}, x_{2,s}) \xleftarrow{OI} (\{b, c\}, \{b, c\}) \\ &= \{a(bb), a(bc), a(cc), a(cb)\} \end{aligned}$$

□

If we require that each  $x_{i,\omega_i}$  appears only once in each tree of  $L$ , *IO* substitution is associative. In fact, under this condition *IO* substitution is equal to *OI* substitution.

## 1.2 Algebras for Tree Languages

In this section we describe the role of  $\Sigma$ -algebras in the hierarchy of Engelfriet and Schmidt [24,25]. They give both equational and fixed-point characterizations for the *IO* and *OI* language hierarchies and show that these two characterizations are equivalent. We are interested in the fixed point characterization so in this section we describe that characterization. Finally, the language hierarchies are defined in terms of least fixed points of a function over certain algebras called substitution algebras.

### 1.2.1 $\Sigma$ -algebras and $D(\Sigma)$ -algebras

A  $\Sigma$ -algebra  $A$  consists of a family  $\langle A_s \rangle_{s \in S}$  of sets called the carriers of  $A$  with  $A_s$  the carrier of sort  $s \in S$ , together with an operator  $f_A$  of type  $\langle \omega, s \rangle$ , for each  $\langle \omega, s \rangle \in S^* \times S$  and each  $f \in \Sigma_{\langle \omega, s \rangle}$ , that is,  $f_A$  is a function such that  $f_A: A_{\omega_1} \times \cdots \times A_{\omega_{|\omega|}} \rightarrow A_s$ . The operators  $f_A$  of type  $\langle \lambda, s \rangle$  are called *constants* of sort  $s$ : that is,  $f_A \in A_s$ . A  $\Sigma$ -algebra  $A$  is *nondeterministic* if at least one  $f_A$  is a relation rather than a function. Engelfriet and Schmidt use nondeterministic  $\Sigma$ -algebras to construct substitution algebras for their hierarchy.

The set,  $T_\Sigma$ , of trees over the alphabet  $\Sigma$ , as defined in the previous section, forms a  $\Sigma$ -algebra  $T$  with carriers  $\langle T_{\Sigma, s} \rangle_{s \in S}$  and the following operations:

1. for  $f \in \Sigma_{\langle \lambda, s \rangle}$ ,  $f_T = f \in T_{\Sigma, s}$ ;
2. for  $f \in \Sigma_{\langle \omega, s \rangle}$ ,  $t_i \in T_{\Sigma, \omega_i}$ ,  $f_T(t_1, \dots, t_{|\omega|}) = f(t_1 \cdots t_{|\omega|}) \in T_{\Sigma, s}$ .

For two  $\Sigma$ -algebras  $A, B$  we may have a  $\Sigma$ -homomorphism  $h: A \rightarrow B$  which is a family  $\langle h_s \rangle_{s \in S}$  of mappings  $h_s: A_s \rightarrow B_s$  satisfying the following two conditions:

1. if  $f \in \Sigma_{\langle \lambda, s \rangle}$ , then  $h_s(f_A) = f_B$ ;
2. if  $f \in \Sigma_{\langle \omega, s \rangle}$ , and  $a_i \in A_{\omega_i}$ , then  $h_s(f_A(a_1, \dots, a_{|\omega|})) = f_B(h_{\omega_1}(a_1), \dots, h_{\omega_{|\omega|}}(a_{|\omega|}))$ .

If  $Y = \langle Y_s \rangle_{s \in S}$  is a family of disjoint sets, then  $T_\Sigma(Y)$  formed as in Section 1.1 is the *absolutely free  $\Sigma$ -algebra generated by  $Y$* . It can be shown that if  $A$  is a  $\Sigma$ -algebra and  $h = \langle h_s \rangle_{s \in S}$  is a family of functions such that  $h_s: Y_s \rightarrow A_s$ , then there is a unique  $\Sigma$ -homomorphism  $\bar{h}: T_\Sigma(Y) \rightarrow A$  such that  $\bar{h}(y) = h(y)$  for all  $y \in Y_s$  and all  $s \in S$ . If  $X_S$  and  $X_\omega$  such that  $\Sigma_{\langle \omega, s \rangle} \neq \emptyset$  are as defined in Section 1.1, then  $\langle T_\Sigma(X_\omega) \rangle_{\langle \omega, s \rangle \in S^+ \times S}$  is the absolutely free  $\Sigma$ -Algebra generated by  $X_S$ . For simplicity, we will use  $T_\Sigma(X)$  to denote this algebra.

Given  $S$ , a set of sorts, and  $\Sigma = \langle \Sigma_{\langle \omega, s \rangle} \rangle_{s \in S}$ , an  $S$ -sorted alphabet, the *derived  $(S^+ \times S)$ -sorted alphabet of  $\Sigma$* , denoted by  $D(\Sigma)$  is obtained from  $\Sigma$  as follows:

**Definition 4** *The symbols in  $D(\Sigma)$  have types from  $(S^+ \times S)^+ \times (S^+ \times S)$ . The elements of  $D(\Sigma)$  are:*

1. for  $f \in \Sigma_{\langle \omega, s \rangle}$ ,  $f$  is in  $D(\Sigma)_{\langle \lambda, \langle \omega, s \rangle \rangle}$ ;
2. for each  $\omega \in S^+$  such that  $\Sigma_{\langle \omega, s \rangle} \neq \emptyset$  and each  $i$ ,  $1 \leq i \leq |\omega|$ ,  $\pi_i^\omega$  is in  $D(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}$ ;
3. for each  $\omega, \nu \in S^+$  such that  $\Sigma_{\langle \omega, s \rangle} \neq \emptyset$ ,  $\Sigma_{\langle \nu, s \rangle} \neq \emptyset$ , and  $s \in S$ ,  $c_{\omega, \nu, s}$  is in  $D(\Sigma)_{\langle \langle \omega, s \rangle \langle \nu, \omega_1 \rangle \dots \langle \nu, \omega_{|\omega|} \rangle, \langle \nu, s \rangle \rangle}$ .

□

The *derived alphabet of order  $n$* ,  $(D^n(\Sigma))$ , is defined by  $D^0(\Sigma) = \Sigma$ ,  $D^{n+1}(\Sigma) = D(D^n(\Sigma))$ .

Using  $D(\Sigma)$  we can form the set,  $T_{D(\Sigma)}$ , of trees over the alphabet  $D(\Sigma)$ . And if  $Y = \langle Y_\varrho \rangle_{\varrho \in S^* \times S}$  is a disjoint family of sets, we can form  $T_{D(\Sigma)}(Y)$ , the set of trees over  $D(\Sigma)$  with variables from  $Y$ , the absolutely free  $D(\Sigma)$ -algebra generated by  $Y$ . If we let  $X_{S^* \times S} = \{x_{i, \langle \omega, s \rangle} \mid i \geq 1 \text{ and } \langle \omega, s \rangle \in S^* \times S \text{ and } \Sigma_{\langle \omega, s \rangle} \neq \emptyset\}$ , then we can form  $X_\varpi = \{x_{i, \varpi_i} \mid 1 \leq |\varpi|\}$  for  $\varpi \in (S^* \times S)^*$ , and  $(X_\varpi)_\varrho = \{x_{i, \varpi_i} \mid \varpi_i = \varrho\}$ . Then, as in Section 1.1 we can form  $T_{D(\Sigma)}(X_\varpi)_\varrho$ , the set of trees over  $D(\Sigma)$  with variables from  $X_\varpi$  of type  $\varrho$ . This process can be iterated for each  $n$  to form the hierarchy of  $D^n(\Sigma)$ -algebras  $T_{D^n(\Sigma)}(X)$ .

Associated with  $T_\Sigma(X)$  is the *tree substitution  $D(\Sigma)$ -algebra*  $A$ , constructed by  $D(T_\Sigma(X))$ , and denoted by  $DT_\Sigma(X)$ . The carrier of sort  $\langle \omega, s \rangle$  is  $T_\Sigma(X_\omega)_s$ , and the operators are given by:

1. if  $a \in D(\Sigma)_{\langle \lambda, \langle \omega, s \rangle \rangle}$ , then  $a$  is either an  $f$ , or a  $\pi_i^\omega$  in which case  $s = \omega_i$ ,
  - (a) for  $a = f$ ,  $a_A = f(x_{1, \omega_1} \cdots x_{|\omega|, \omega_{|\omega|}})$ ;
  - (b) for  $a = \pi_i^\omega$ ,  $a_A = x_{i, \omega_i}$ ;
2. if  $a = c_{\omega, \nu, s} \in D(\Sigma)_{\langle \langle \omega, s \rangle \langle \nu, \omega_1 \rangle \dots \langle \nu, \omega_{|\omega|} \rangle, \langle \nu, s \rangle \rangle}$ , then  $a_A(t, t_1, \dots, t_{|\omega|}) = t[t_1, \dots, t_{|\omega|}]$ , the result of substituting each  $t_i$  for  $x_{i, \omega_i}$  in  $t$ .

The unique  $D(\Sigma)$ -homomorphism  $T_{D(\Sigma)} \rightarrow DT_\Sigma(X)$  is called *YIELD*.

*YIELD*:  $T_{D(\Sigma)} \rightarrow DT_\Sigma(X)$  is defined recursively by:

1.  $YIELD(f) = f(x_{1, \omega_1} \cdots x_{|\omega|, \omega_{|\omega|}})$  for  $f \in D(\Sigma)_{\langle \lambda, \langle \omega, s \rangle \rangle}$ ;
2.  $YIELD(\pi_i^\omega) = x_{i, \omega_i}$  for  $\pi_i^\omega \in D(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}$ ; and
3.  $YIELD(c_{\omega, \nu, s}(t, t_1, \dots, t_{|\omega|})) = YIELD(t)[YIELD(t_1), \dots, YIELD(t_{|\omega|})]$   
for  $c_{\omega, \nu, s} \in D(\Sigma)_{\langle \langle \omega, s \rangle \langle \nu, \omega_1 \rangle \dots \langle \nu, \omega_{|\omega|} \rangle, \langle \nu, s \rangle \rangle}$ .

The *YIELD* function takes an object from  $T_{D(\Sigma)}$  and produces an object in  $DT_{\Sigma}(X)$  by substituting for the variables on the frontier. The  $\mathcal{Y}$  function from Section 1.1 takes an object from  $T_{\Sigma}$  and produces objects in  $\Sigma^*$ . If we consider the right end of a string to be the frontier, then we see that  $\mathcal{Y}$  also makes substitutions on the frontier. No variables are necessary in this case because there is only one position on the frontier of the strings. The example which follows takes the *YIELD* of a tree from  $T_{D(\Sigma)}$  and then takes the frontier,  $\mathcal{Y}$ , of that tree.

**Example 5**  $\Sigma_{\langle \lambda, s \rangle} = \{a\}$ ,  $\Sigma_{\langle ss, s \rangle} = \{g\}$ ,  $D(\Sigma)_{\langle \lambda, \langle \lambda, s \rangle \rangle} = \{a\}$ ,  $D(\Sigma)_{\langle \lambda, \langle ss, s \rangle \rangle} = \{g, \pi_1^{ss}, \pi_2^{ss}\}$ ,  $D(\Sigma)_{\langle \langle \lambda, s \rangle, \langle \lambda, s \rangle \rangle} = \{c_{\lambda, \lambda, s}\}$ ,  $D(\Sigma)_{\langle \langle ss, s \rangle, \langle \lambda, s \rangle, \langle \lambda, s \rangle \rangle} = \{c_{s, \lambda, s}\}$ ,  $D(\Sigma)_{\langle \langle ss, s \rangle, \langle ss, s \rangle, \langle ss, s \rangle \rangle} = \{c_{ss, s, s}\}$ . Then we compute the *YIELD* of the derived tree below to get the tree:

$$\begin{aligned}
& \text{YIELD} \left( \begin{array}{c} c_{s, \lambda, s} \\ \swarrow \quad \searrow \\ c_{s, s, s} \quad c_{\lambda, \lambda, s} \\ \swarrow \quad \downarrow \quad \searrow \quad | \\ c_{ss, s, s} \quad c_{ss, s, s} \quad a \\ \swarrow \quad \downarrow \quad \searrow \quad \swarrow \quad \downarrow \quad \searrow \\ g \quad \pi_1^{ss} \quad \pi_1^{ss} \quad g \quad \pi_1^{ss} \quad \pi_1^{ss} \end{array} \right) \\
&= \text{YIELD} \left( \begin{array}{c} c_{s, s, s} \\ \swarrow \quad \searrow \\ c_{ss, s, s} \quad c_{ss, s, s} \\ \swarrow \quad \downarrow \quad \searrow \quad \swarrow \quad \downarrow \quad \searrow \\ g \quad \pi_1^{ss} \quad \pi_1^{ss} \quad g \quad \pi_1^{ss} \quad \pi_1^{ss} \end{array} \right) \left[ \text{YIELD} \left( \begin{array}{c} c_{\lambda, \lambda, s} \\ | \\ a \end{array} \right) \right] \\
&= \text{YIELD} \left( \begin{array}{c} c_{ss, s, s} \\ \swarrow \quad \downarrow \quad \searrow \\ g \quad \pi_1^{ss} \quad \pi_1^{ss} \end{array} \right) \left[ \text{YIELD} \left( \begin{array}{c} c_{ss, s, s} \\ \swarrow \quad \downarrow \quad \searrow \\ g \quad \pi_1^{ss} \quad \pi_1^{ss} \end{array} \right) \right] [a]
\end{aligned}$$

$$= YIELD(g) [YIELD(\pi_1^{ss}), YIELD(\pi_1^{ss})] [YIELD(g) [YIELD(\pi_1^{ss}), YIELD(\pi_1^{ss})]] [a]$$

$$= \begin{array}{c} g \\ \swarrow \quad \searrow \\ x_{1,s} \quad x_{2,s} \end{array} [x_{1,s}, x_{1,s}] \left[ \begin{array}{c} g \\ \swarrow \quad \searrow \\ x_{1,s} \quad x_{2,s} \end{array} [x_{1,s}, x_{1,s}] \right] [a]$$

$$= \begin{array}{c} g \\ \swarrow \quad \searrow \\ g \quad g \\ \swarrow \quad \searrow \quad \swarrow \quad \searrow \\ x_{1,s} \quad x_{1,s} \quad x_{1,s} \quad x_{1,s} \end{array} [a] = \begin{array}{c} g \\ \swarrow \quad \searrow \\ g \quad g \\ \swarrow \quad \searrow \quad \swarrow \quad \searrow \\ a \quad a \quad a \quad a \end{array}$$

and the frontier of this tree is the string here computed:

$$\begin{aligned} \mathcal{Y}(g(g(aa) g(aa))) &= \mathcal{Y}(g(aa) g(aa)) \\ &= \mathcal{Y}(aa) \mathcal{Y}(aa) \\ &= aaaa \end{aligned}$$

□

### 1.2.2 Continuous $\Sigma$ -algebras

Given a partially ordered set  $A$  with partial order  $\sqsubseteq$  and minimal element  $\perp$ , we say that  $A_1 \subseteq A$  is *directed* if any two elements of  $A_1$  have an upper bound in  $A_1$ . If every subset of  $A$  has a least upper upper bound ( $\sqcup$ ) in  $A$ , then  $A$  is  $\sqcup$ -complete and if every directed subset of  $A$  has a least upper bound in  $A$  then  $A$  is  $\Delta$ -complete. A function  $f: A \rightarrow B$  where  $B$  is another partially ordered set is  $\sqcup(\Delta)$ -continuous if  $f(\sqcup A_1) = \sqcup f(A_1)$  for all subsets (all directed subsets) of  $A$  for which  $\sqcup$  exists.

If each carrier of a  $\Sigma$ -algebra  $A$  is  $\sqcup(\Delta)$ -complete and its operations are  $\sqcup(\Delta)$ -continuous in each of their arguments, then  $A$  is called a  $\sqcup(\Delta)$ -continuous  $\Sigma$ -algebra. In any  $\sqcup$ -continuous  $\Sigma$ -algebra  $A$ ,  $f_A(a_1, \dots, a_{k-1}, \perp, a_{k+1}, \dots, a_n) = \perp$ .

A  $\sqcup$ -continuous deterministic  $\Sigma$ -algebra can be constructed from a nondeterministic  $\Sigma$ -algebra  $A$ . This algebra is called the *subset algebra* of  $A$ . The carrier of sort  $s$  is  $\mathcal{P}(A_s)$  and for  $f \in \Sigma_{\langle \omega, s \rangle}$ ,  $A_i \in \mathcal{P}(A_{\omega_i})$ ,  $f_{\mathcal{P}(A)}(A_1, \dots, A_{|\omega|}) = \bigcup \{f_A(a_1, \dots, a_{|\omega|}) \mid a_i \in A_i \text{ for } 1 \leq i \leq |\omega|\}$ . Taking  $\sqcup$  as set union and  $\sqsubseteq$  to be  $\subseteq$ ,  $\mathcal{P}(A)$  forms a  $\sqcup$ -continuous  $\Sigma$ -algebra. Engelfriet and Schmidt give the following facts as Lemmas and Theorems about subset algebras.

1. For each nondeterministic  $\Sigma$ -algebra  $A$ ,  $\mathcal{P}(A)$  is a  $\sqcup$ -continuous  $\Sigma$ -algebra.
2. Let  $A$  be a  $\Sigma$ -algebra and  $B$  a  $\sqcup$ -continuous  $\Sigma$ -algebra with  $h: A \rightarrow B$  a  $\Sigma$ -homomorphism, then  $h$  is uniquely extendible to a  $\sqcup$ -continuous  $\Sigma$ -homomorphism from  $\mathcal{P}(A)$  to  $B$ .
3.  $\mathcal{P}(T_\Sigma)$  is free in the  $\sqcup$ -continuous  $\Sigma$ -algebras with  $\sqcup$ -continuous  $\Sigma$ -homomorphisms.
4. For  $\omega \in S^+$ ,  $\mathcal{P}(T_\Sigma(X_\omega))$  is the free  $\sqcup$ -continuous  $\Sigma$ -algebra with generators  $X_\omega$ .

For a  $\Sigma$ -algebra  $A$  with derived alphabet  $D(\Sigma)$ , we can define the  $D(\Sigma)$ -algebra of functions over  $A$ . This algebra, denoted by  $\mathcal{F}(A)$ , has as carriers of sort  $\langle \omega, s \rangle$  the set of all functions  $A_{\omega_1} \times \dots \times A_{\omega_{|\omega|}} \rightarrow A_s$  with operations for each  $f \in \Sigma_{\langle \omega, s \rangle}$ ,  $f'$  is  $f_A$ ;  $\pi_i^\omega$  is the  $i$ th projection function; and  $c_{\omega, \nu, s}(f, f_{\omega_1}, \dots, f_{\omega_{|\omega|}}) = f \circ (f_{\omega_1}, \dots, f_{\omega_{|\omega|}})$  for  $f \in \langle \omega, s \rangle$ , and  $f_{\omega_i} \in \langle \nu, \omega_i \rangle$ , the usual composition of functions. The unique  $D(\Sigma)$ -homomorphism  $DT_\Sigma(X) \rightarrow \mathcal{F}(A)$  is called *derop<sub>A</sub>*, and it associates with each

$t \in T_\Sigma(X_\omega)_s$ , the *derived operation*  $t_A$ .  $t_A: A_{\omega_1} \times \cdots \times A_{\omega_{|\omega|}} \rightarrow A_s$  by  $t_A(a_{\omega_1}, \dots, a_{\omega_{|\omega|}}) = \bar{a}_\omega(t)$ , where  $\bar{a}_\omega: T_\Sigma(X_\omega)_s \rightarrow A_s$  is the unique  $\Sigma$ -homomorphism such that  $\bar{a}_\omega(x_{i,\omega_i}) = a_{\omega_i}$ .  $\mathcal{F}$  can be iterated to form the  $D^n(\Sigma)$ -algebra  $\mathcal{F}^n(A)$ . If  $A$  is a  $\Delta$ -continuous  $\Sigma$ -algebra with  $\sqcup$ -complete carriers, then  $\mathcal{F}_\Delta(A)$  denotes  $\mathcal{F}(A)$ . Each carrier is ordered in the usual way:  $f \subseteq g$  if and only if  $f(a_{\omega_1}, \dots, a_{\omega_{|\omega|}}) \sqsubseteq g(a_{\omega_1}, \dots, a_{\omega_{|\omega|}})$  for all  $a_{\omega_i} \in A_{\omega_i}$ .  $\mathcal{F}_\Delta^n(A)$  is also a  $\Delta$ -continuous  $D^n(\Sigma)$ -algebra with  $\sqcup$ -complete carriers. Given a  $\sqcup$ -continuous  $\Sigma$ -algebra  $A$ , any tree language  $L \subseteq T_\Sigma(X_\omega)_s$  can be interpreted as the derived operation,  $A_{\omega_1} \times \cdots \times A_{\omega_{|\omega|}} \rightarrow A_s$ , denoted by  $L_A$  with  $L_A(a_1, \dots, a_{|\omega|}) = \sqcup_{t \in L} t_A(a_1, \dots, a_{|\omega|})$  where  $t_A$  is the derived operation given above.

### 1.2.3 Context-free tree grammars

An *S-sorted context-free tree grammar* is a quadruple  $G = (\Sigma, \mathcal{F}, P, F_1)$ , where  $\Sigma$  and  $\mathcal{F}$  are disjoint finite  $S$ -sorted alphabets,  $F_1 \in \mathcal{F}_{(\lambda,s)}$ , and  $P$  is a finite set of productions of the form  $F(x_{1,\omega_1} \cdots x_{|\omega|,\omega_{|\omega|}}) \rightarrow \tau$ , where  $F \in \mathcal{F}_{(\omega,s)}$ ,  $\tau \in T_{\Sigma \cup \mathcal{F}}(X_\omega)_s$ ,  $\omega \in S^*$ , and  $s \in S$ . For each  $F \in \mathcal{F}_{(\omega,s)}$ , we define  $rhs(F) = \{\tau \mid F(x_{1,\omega_1}, \dots, x_{|\omega|,\omega_{|\omega|}}) \rightarrow \tau \in P\}$ . For  $\mathcal{F} = \{F_1, \dots, F_n\}$  we define  $i\nu$  and  $is$  to be  $\nu$  and  $s$  such that  $F_i \in \mathcal{F}_{(\nu,s)}$ .

The fixed point characterization of the tree language defined by the context-free tree grammars lifts the right hand sides of the productions into the set of derived  $\Sigma$ -algebras. Engelfriet and Schmidt [24,25] show how to obtain the tree languages associated with each nonterminal of the grammar by solving for the fixed point of a mapping from the powerset of the subset algebra associated with each nonterminal to that same algebra.

The function *COMB* is used to lift the right hand sides of the productions from  $T_{\Sigma \cup \mathcal{F}}(X_{i\nu})_{is}$  to  $T_{D(\Sigma \cup \mathcal{F})}$ . *COMB* is defined recursively as follows:

**Definition 5**  $COMB_{\nu}^{\Sigma}: T_{\Sigma}(X_{\nu}) \rightarrow T_{D(\Sigma)}$  by:

1.  $COMB_{\nu}^{\Sigma}(x_{i,\nu_i}) = \pi_i^{\nu};$
2.  $COMB_{\nu}^{\Sigma}(f) = c_{\lambda,\nu,s}(f),$  for  $f \in \Sigma_{\langle\lambda,s\rangle};$
3.  $COMB_{\nu}^{\Sigma}(f(t_1, \dots, t_{|w|})) = c_{\omega,\nu,s}(f COMB_{\nu}^{\Sigma}(t_1) \cdots COMB_{\nu}^{\Sigma}(t_{|w|})),$  for  $f \in \Sigma_{\langle\omega,s\rangle}.$

□

The subset algebra  $\mathcal{P}(T_{\Sigma \cup \mathcal{F}}(X_{i\nu})_{is})$  is associated with each nonterminal,  $F_i$ , of the grammar. Each of these algebras is a  $\Delta$ -continuous  $\Sigma$ -algebra with  $\sqcup$ -complete carriers so the cartesian product  $\prod_{i=1}^n \mathcal{P}(T_{\Sigma \cup \mathcal{F}}(X_{i\nu})_{is})$  with component-wise  $\subseteq$  as the ordering and  $(\emptyset, \dots, \emptyset) = \perp$  as the minimal element is a  $\sqcup$ -complete poset.

With a grammar  $G = (\Sigma, \mathcal{F}, P, F_1)$ , we associate two mappings  $M_{G,IO}$  and  $M_{G,OI}$  such that  $M_{G,IO(OI)}: \prod_{i=1}^n \mathcal{P}(T_{\Sigma}(X_{i\nu})_{is}) \rightarrow \prod_{i=1}^n \mathcal{P}(T_{\Sigma}(X_{i\nu})_{is})$ . These mappings will correspond to the *Inside-out* and *Outside-in* derivations in the grammars.  $M_{G,IO(OI)}$  is defined as follows:

**Definition 6** First define  $M_{OI}(\sigma): \prod_{i=1}^n \mathcal{P}(T_{\Sigma}(X_{i\nu})_{is}) \rightarrow \mathcal{P}(T_{\Sigma}(X_{i\nu})_{is})$  and similarly  $M_{IO}$  such that for all  $(b_1, \dots, b_n) = \mathbf{b} \in \prod_{i=1}^n \mathcal{P}(T_{\Sigma}(X_{i\nu})_{is})$ , and all  $\sigma \in T_{D(\Sigma)}(\mathcal{F}')_{\langle\nu,s\rangle}$ ,

1. If  $\sigma = a \in D(\Sigma)_{\langle\lambda,\langle\nu,s\rangle\rangle}$ , then  $M_{OI}(\sigma)(\mathbf{b}) = a(x_{i,\nu_i} \cdots x_{|\nu|,\nu_{|\nu|}});$
2. If  $\sigma = F_i \in \mathcal{F}_{\langle i\nu, is \rangle}$ ,  $\nu = i\nu$ ,  $s = is$ , then  $M_{OI}(\sigma)(\mathbf{b}) = \mathbf{b}_i;$
3. If  $\sigma = \pi_i^{\nu} \in D(\Sigma)_{\langle\lambda,\langle\nu,\nu_i\rangle\rangle}$ , then  $M_{OI}(\sigma)(\mathbf{b}) = x_{i,\nu_i};$
4. If  $\sigma = c_{\omega,\nu,s}(\sigma'_1 \cdots \sigma'_{|w|})$ ,  $c_{\omega,\nu,s} \in D(\Sigma)_{\langle\langle\omega,s\rangle\langle\nu,\omega_1\rangle \cdots \langle\nu,\omega_{|w|}\rangle, \langle\nu,s\rangle\rangle}$ ,  $\sigma' \in T_{D(\Sigma)}(\mathcal{F}')_{\langle\omega,s\rangle}$ ,  $\sigma_i \in T_{D(\Sigma)}(\mathcal{F}')_{\langle\nu,\omega_i\rangle}$ , then

$$M_{OI}(\sigma)(\mathbf{b}) = M_{OI}(\sigma')(\mathbf{b}) \stackrel{\Leftarrow}{OI} (M_{OI}(\sigma_1)(\mathbf{b}), \dots, M_{OI}(\sigma_{|w|})(\mathbf{b})).$$

$M_{OI}(\sigma)$  is extended to sets,  $R$ , by  $\hat{M}_{OI}(R)(\mathbf{b}) = \bigcup_{\sigma \in R} (M_{OI}(\sigma)(\mathbf{b}))$ . Finally  $M_{G,OI}(\mathbf{b}) = (\hat{M}_{OI}(R_1)(\mathbf{b}), \dots, \hat{M}_{OI}(R_n)(\mathbf{b}))$  where  $R_i = \text{COMB}_{i\nu}^{\Sigma \cup \mathcal{F}}(\text{rhs}(F_i))$ .

□

Engelfriet and Schmidt show that the power set is  $\Delta$ -continuous with  $\sqcup$ -complete carriers and apply the fixed point theorem to get the minimal fixed point of  $M_{G,IO(OI)}$  equal to  $\bigcup_{i=0}^{\infty} M_{G,IO(OI)}^i(\perp)$ , denoted by  $|G_{IO(OI)}|$ .  $M_{IO(OI)}(F_i)(|G_{IO(OI)}|)$  is then the subset of  $T_{\Sigma}(X)$  associated with the nonterminal  $F_i$  in the context-free tree grammar  $G$ . Given an alphabet  $\Sigma$ , the Engelfriet-Schmidt hierarchy of languages on  $\Sigma$  is those languages derived from grammars of the form  $G = (D^n(\Sigma), \mathcal{F}, P, F_1)$ . The language  $L$  is  $IO(n)$  if  $L = M_{IO}(F_1)(|G_{IO}|)$  for some context free tree grammar  $G = (D^n(\Sigma), \mathcal{F}, P, F_1)$ . The language  $L$  is  $OI(n)$  if  $L = M_{OI}(F_1)(|G_{OI}|)$  for some context free tree grammar  $G = (D^n(\Sigma), \mathcal{F}, P, F_1)$ .

Engelfriet and Schmidt established the following relationships between their tree hierarchy and the already known hierarchy.

1.  $OI(0) = IO(0)$  = the recognizable tree languages;
2.  $OI(1) = OI$  tree languages;
3.  $IO(1) = IO$  tree languages;
4. For all  $n \geq 0$ ,  $L \in IO(n)$  implies  $L \in IO(n+1)$ , the same is true for  $OI$  as well.

They prove the following theorem which relates the tree languages of Engelfriet-Schmidt hierarchy to the string languages defined by their yields.

**Theorem 1** (Engelfriet and Schmidt) *Let  $\Sigma$  be an  $S$ -sorted alphabet,  $A$  a  $\Sigma$ -algebra,  $h_A$  the  $\Sigma$ -homomorphism  $T_{\Sigma} \rightarrow A$ . For any  $s \in S$  and  $n \geq 0$ , a subset  $A_s$  is  $IO(n)$*

if and only if it is the image under  $h_A \circ YIELD^n$  of a recognizable tree language in  $T_{D^n(\Sigma), t_n(s)}$ . In particular, a tree language over  $\Sigma$  is  $IO(n)$  if and only if it is  $YIELD^n(L)$  for some recognizable tree language  $L$  over  $D^n(\Sigma)$  (of appropriate sort).

□

**Example 6**  $G = (\Sigma, \mathcal{F}, P, F_1)$  where  $\Sigma_{\langle \lambda, s \rangle} = \{a, b\}$ ,  $\Sigma_{\langle ss, s \rangle} = \{f\}$ ,  $\mathcal{F}_{\langle \lambda, s \rangle} = \{F_1, F_3\}$ ,  $\mathcal{F}_{\langle s, s \rangle} = \{F_2\}$  and  $P$  is the set of productions  $\{F_1 \rightarrow F_2(F_3), F_2(x_{1,s}) \rightarrow f(x_{1,s}, x_{1,s}), F_3 \rightarrow a, F_3 \rightarrow b\}$ .

Then the  $D(\Sigma)$  productions are:

$$COMB_{\lambda}^{\Sigma}(rhs(F_1)) = \{c_{s, \lambda, s}(F_2 \ c_{\lambda, \lambda, s}(F_3))\}$$

$$COMB_s^{\Sigma}(rhs(F_2)) = \{c_{ss, s, s}(f \ \pi_1^s \ \pi_1^s)\}$$

$$COMB_{\lambda}^{\Sigma}(rhs(F_3)) = \{c_{\lambda, \lambda, s}(a), c_{\lambda, \lambda, s}(b)\}$$

The inside-out and outside-in languages defined by the grammar are:

$$\begin{aligned} M_{G, IO}(\perp) &= (\hat{M}_{IO}(\{c_{s, \lambda, s}(F_2 \ c_{\lambda, \lambda, s}(F_3))\})(\perp), \\ &\hat{M}_{IO}(\{c_{ss, s, s}(f \ \pi_1^s \ \pi_1^s)\})(\perp), \hat{M}_{IO}(\{c_{\lambda, \lambda, s}(a), c_{\lambda, \lambda, s}(b)\})(\perp)) \end{aligned}$$

so

$$\begin{aligned} M_{G, IO}^0(\perp) &= \perp \\ M_{G, IO}^1(\perp) &= (\emptyset, \{f(x_{1,s} \ x_{1,s})\}, \{a, b\}) \\ M_{G, IO}^2(\perp) &= (\{f(aa), f(bb)\}, \{f(x_{1,s}x_{1,s})\}, \{a, b\}) \end{aligned}$$

and

$$\begin{aligned} M_{G, OI}(\perp) &= (\hat{M}_{OI}(\{c_{s, \lambda, s}(F_2 \ c_{\lambda, \lambda, s}(F_3))\})(\perp), \\ &\hat{M}_{OI}(\{c_{ss, s, s}(f \ \pi_1^s \ \pi_1^s)\})(\perp), \hat{M}_{OI}(\{c_{\lambda, \lambda, s}(a), c_{\lambda, \lambda, s}(b)\})(\perp)) \end{aligned}$$

so

$$M_{G,OI}^0(\perp) = \perp$$

$$M_{G,OI}^1(\perp) = (\emptyset, \{f(x_{1,s}, x_{1,s})\}, \{a, b\})$$

$$M_{G,OI}^2(\perp) = (\{f(ab), f(ba), f(aa), f(bb)\}, \{f(x_{1,s}, x_{1,s})\}, \{a, b\})$$

□

### 1.3 $n$ -Dimensional Forests, Grammars, and Derivations

Baldwin [12] generalized the concepts of tree grammars and the frontier function ( $\mathcal{Y}$ ) to define an entire hierarchy of tree and forest languages that he called the Hypertree Hierarchy. We call this hierarchy the  $n$ -dimensional forest hierarchy. He used the grammars to generate regular  $n$ -dimensional forests. Applying the frontier function on the forest language generated by a specific  $n$ -dimensional grammar produces an  $(n - 1)$ -dimensional language. Each successive application of the frontier function produces another language of dimension 1 less than before. Halting this process at dimension 1 results in a string language, since dimension 1 of the hierarchy corresponds to strings. Of particular interest are the results, from [12], that the 1-dimensional frontier of a 2-dimensional regular language is a context-free string language, and the 1-dimensional frontier of a 3-dimensional regular language is an *IO* Macro language[27].

These results together with Theorem 1 imply that the string languages of Engelfriet and Schmidt,  $IO(0)$ , correspond to the string languages defined by the 2-dimensional forest languages and the string languages defined by the *IO* tree languages of Engelfriet and Schmidt  $IO(1)$  correspond to the string languages defined by

the 3-dimensional forest languages . We will show that these results generalize to containment for the languages of higher level and give a definition of *OI*  $n$ -dimensional languages for which similar results are true.

### 1.3.1 $n$ -Dimensional trees and forests

As mentioned earlier, Baldwin [12] generalized the concept of tree grammars and the frontier function ( $\mathcal{Y}$ ) and defined hierarchy of tree and forest languages. The trees form a hierarchy of structures,  $H_1(\Sigma), H_2(\Sigma), \dots$ , along with  $H_0(\Sigma) = \Sigma$ , called the  *$n$ -dimensional trees over  $\Sigma$* .

**Definition 7** *The  $n$ -dimensional trees over  $\Sigma$ ,  $H_n(\Sigma)$ , for  $n \geq 0$  are the smallest sets that satisfy*

$$H_n(\Sigma) = \Sigma \quad \text{for } n = 0$$

$$H_n(\Sigma) = \Sigma \cup \Sigma[_n H_{n-1}(H_n(\Sigma)) ] \quad \text{for } n > 0$$

□

The bracketed expression  $H_{n-1}(H_n(\Sigma))$  can be expanded to get:

$$H_{n-1}(H_n(\Sigma)) = H_n(\Sigma) \cup H_n(\Sigma)[_n H_{n-2}(H_{n-1}(H_n(\Sigma))) ] .$$

Continued expansion of  $H_{n-2}(H_{n-1}(H_n(\Sigma)))$  yields an expression containing  $H_1(H_2(\dots H_n(\Sigma) \dots))$ . The expression,  $H_k(\dots H_n(\Sigma) \dots)$  for  $1 \leq k \leq n$ , denotes the  *$n$ -dimensional forests of degree  $k$*  and will be represented by  $H_n^k(\Sigma)$ . In the sequel, when  $\Sigma$  is fixed by context we will write  $H_n^k$  instead of  $H_n^k(\Sigma)$ . We also consider the bracketed expressions to be optional so the expression for  $n$ -dimensional trees over  $\Sigma$  for  $n > 0$  becomes  $H_n = \Sigma[_n H_n^{n-1} ]$ . Now extend the definition of  $n$ -dimensional

forests of degree  $k$  to include  $k = 0$  and  $k = n + 1$  as follows: for  $k = 0$ , note that  $H_n^0 = H_0(H_n^1) = H_n^1$  and, for  $k = n + 1$ , define  $H_n^{n+1} = \Sigma$ . Then for  $1 \leq k \leq n$  expand  $H_n^k$  by the definition to get

$$H_n^k = H_k(H_n^{k+1}) = H_n^{k+1}[_k H_n^{k-1}]$$

which expands to

$$H_n^k = \Sigma[_n H_n^{n-1}] \cdots [_k H_n^{k-1}].$$

Since the bracketed expressions are optional, given  $t = a[_n t_n] \cdots [_k t_k] \in H_n^k$ ,  $a \in \Sigma$ ,  $t_i \in H_n^{i-1}$ , any one of the  $t_i$  may not be present. If this occurs, we use  $t_i = \emptyset$  to express this fact.

For  $T = \langle T_k \rangle_{1 \leq k \leq n}$ , a family of disjoint sets of *variables*, we define the set of  $n$ -dimensional trees with variables from  $T$  as follows:

**Definition 8** *The  $n$ -dimensional trees  $H_n(\Sigma, T)$  over  $\Sigma$  with variables from  $T$  for  $n \geq 0$  are the smallest sets that satisfy*

$$H_n(\Sigma, T) = \Sigma \quad \text{for } n = 0$$

$$H_n(\Sigma, T) = \Sigma \cup T_n \cup \Sigma[_n H_{n-1}(H_n(\Sigma, T), T)] \quad \text{for } n > 0.$$

□

We generalize this definition to  $n$ -dimensional forests of degree  $k$  with variables from  $T$ .

Considering the bracketed expressions to be optional, first expand  $H_{n-1}(H_n(\Sigma, T), T)$  as we did above to get:

$$H_{n-1}(H_n(\Sigma, T), T) = T_{n-1} \cup H_n(\Sigma, T)[_{n-1} H_{n-2}(H_{n-1}(H_n(\Sigma, T), T), T)].$$

This can be expanded until it contains the expression  $H_1(H_2(\cdots H_n(\Sigma, T) \cdots, T), T)$ . As before, we use  $H_n^k(\Sigma, T)$  to denote  $H_k(H_{k+1}(\cdots H_n(\Sigma, T) \cdots, T), T)$  and when  $\Sigma$  and  $T$  are understood we use  $H_n^k$ . So we have  $H_n = T_n \cup \Sigma[_n H_n^{n-1}]$ , for  $n > 0$ ,  $H_n^0(\Sigma, T) = H_n(H_n^1(\Sigma, T), T) = H_n^1(\Sigma, T)$ , and in addition we define  $H_n^{n+1}(\Sigma, T) = \Sigma$ .

Now expand  $H_n^k(\Sigma, T)$  to get

$$\begin{aligned} H_n^k(\Sigma, T) &= H_k(H_n^{k+1}(\Sigma, T), T) \\ &= T_k \cup H_n^{k+1}(\Sigma, T)[_k H_{k-1}(H_k(H_n^{k+1}(\Sigma, T), T), T)]. \end{aligned}$$

Using  $H_n^k$  for  $H_n^k(\Sigma, T)$ , we get the expression below for the  $n$ -dimensional forests of degree  $k$  with variables from  $T$ .

$$\begin{aligned} H_n^k &= \Sigma[_n H_n^{n-1}] \cdots [_k H_n^{k-1}] \\ &\cup T_n[_{n-1} H_n^{n-2}] \cdots [_k H_n^{k-1}] \\ &\cup T_{n-1}[_{n-2} H_n^{n-3}] \cdots [_k H_n^{k-1}] \\ &\vdots \\ &\cup T_{k+1}[_k H_n^{k-1}] \\ &\cup T_k \end{aligned}$$

From this, it can be shown that  $H_n^k \subseteq H_n^l$  for  $l \leq k$ , and  $H_l^k \subseteq H_n^k$  for  $l \leq n$ .

If we agree that  $T_{n+1} = \Sigma$ ,  $H_n^k$  can be further simplified to:

$$H_n^k = \bigcup_{i=k}^{n+1} T_i[_{i-1} H_n^{i-2}] \cdots [_k H_n^{k-1}].$$

Notice that a variable  $x \in T_k$  can only precede a left bracket  $([_k)$  whose subscript is less than  $k$  or precede a right bracket  $]_k)$ . In this case, we say that  $x$  is on the  $k^{th}$

frontier.

### 1.3.2 The $n$ -ary tree representation for $n$ -dimensional forests of degree $k$ and the depth function

For the examples and illustrations in this text, we will use a representation of the  $n$ -dimensional forests that derives from their representation by  $n$ -ary trees. This is a generalization of the well known representation of the usual ordered trees,  $T_\Sigma$ , in Section 1.1 by binary trees. Given a forest  $t_1 \cdots t_n$  of trees from  $T_\Sigma$ , the binary tree representation,  $\text{FB}(t_1 \cdots t_n)$ , of this forest is given by:

$$\text{FB}(t_1 \cdots t_n) = \begin{cases} \lambda & \text{if } n = 0; \\ a(\text{FB}(t_{11} \cdots t_{1m}) \text{FB}(t_2 \cdots t_n)) & \text{if } t_1 = a(t_{11} \cdots t_{1m}). \end{cases}$$

This generalizes to  $t \in H_n^k$  by  $\text{nFB}$  which takes  $H_n^k$  to the set of  $n$ -ary trees by:

$$\text{nFB}(t) = \begin{cases} \lambda & \text{if } t = \emptyset; \\ a(\text{nFB}(t_n) \text{nFB}(t_{n-1}) \cdots \text{nFB}(t_k) \lambda^{k-1}) & \text{if } t = a[_n t_n ] \cdots [_k t_k ]. \end{cases}$$

Using this representation, the forest

$$g[_3 + [_2 x_2^\lambda ]_1 * [_2 x_2^1 ]_1 x_2^\lambda [_1 x_1^\lambda ] ] ] ] ] ]_2 g[_2 a[_1 + [_2 b[_1 c[_1 x_1^\lambda ] ] ] ]_1 x_1^\lambda ] ] ] ]$$

in  $H_3^2(\Sigma, X)$  with  $X_k = \{x_k^\rho \mid \rho \in \{1, \dots, k-1\}^*\}$ , and  $X = \langle X_k \rangle_{1 \leq k \leq 2}$ , has 3-ary representation:

$$g(+(\lambda x_2^\lambda (\lambda \lambda * (\lambda x_2^1 (\lambda \lambda x_2^\lambda (\lambda \lambda x_1^\lambda)) \lambda)) \lambda) \lambda) a(\lambda \lambda + (\lambda b(\lambda \lambda c(\lambda \lambda x_1^\lambda)) x_1^\lambda)) \lambda)$$

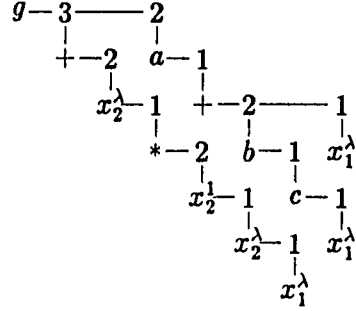


Figure 1.1: Pictorial Representation of a Forest

From all this, we get a pictorial representation. The rule is: for the forest  $t = a[_n t_n] \cdots [_k t_k] \in H_n^k$ ,  $\text{nFB}(t) = a(\text{nFB}(t_n) \cdots \text{nFB}(t_k) \lambda^{k-1})$  is an  $n$ -ary tree whose pictorial representation is:  $\frac{a-n-\cdots-k}{\text{nFB}(t_n) \text{nFB}(t_k)}$ . We leave out the numbers for the missing subtrees. The pictorial representation for the above forest is in Figure 1.1.

There is also a depth function associated with these forests.

**Definition 9** For  $t \in H_n^k(\Sigma, T)$ :

$$\text{depth}(t) = \begin{cases} 0 & t = \emptyset \\ 1 + \max \{ \text{depth}(t_n), \dots, \text{depth}(t_k) \} & t = a[_n t_n] \cdots [_k t_k] \wedge a \in \Sigma \cup T \end{cases}$$

□

The depth of the tree in Figure 1.1 is 7.

### 1.3.3 Paths in $n$ -dimensional forests

The set  $X = \langle X_k \rangle_{1 \leq k \leq n}$  where  $X_k = \{x_k^\rho \mid \rho \in \{1, \dots, k-1\}\}$  will be used later by the substitution operation on  $n$ -dimensional forests. The  $\rho$  associated with each

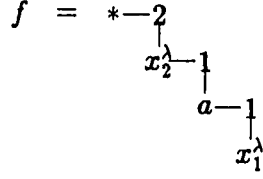


Figure 1.2: Substitution of a Set into a Forest

variable,  $x_k^\rho \in X_k$ , is called a *path*. For each forest,  $t$ , there is a set of legal paths denoted by  $shape(t)$ . For any  $x_k^\rho$  and  $t \in H_k^l$  we say that  $\rho \in treeshape(t)$ , if the tree whose root is at the “end of the path”  $\rho$  is in  $H_k^k$ . We use  $(t)_\rho$  to denote this tree in  $t$ . More formally, we define  $(t)_\rho$  as follows:

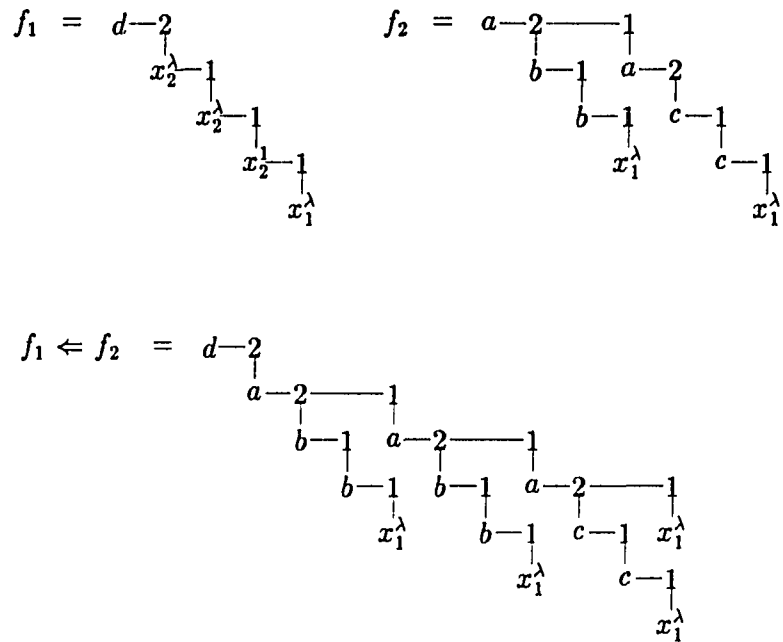
**Definition 10** If  $t = a[_n t_n] \cdots [_k t_k] \in H_n^k$ , and  $\rho \in \{1, \dots, n-1\}^*$  then

$$(a[_n t_n] \cdots [_k t_k])_\rho = \begin{cases} (t_j)_{\rho'}, & t_j \neq \emptyset \wedge \rho = j\rho' \\ a[_n t_n] & \rho = \lambda \wedge a \in \Sigma \cup X_n \cup X_{n+1} \\ error & \rho \notin treeshape(t) \end{cases}$$

□

**Example 7** Let  $f = *[_2 x_2^\lambda[_1 a[_1 x_1^\lambda]]] \in H_3^2$ , then  $(f)_{21} = a$ , and both  $(f)_1$  and  $(f)_{211}$  are errors.  $(f)_1$  is an error since  $*$  does not have a 1 subtree, and  $(f)_{211}$  is an error since  $x_1^\lambda$  is not in  $H_2$ . See Figure 1.2 for the pictorial representation of the forest in this example.

□



**Figure 1.3: Substitution of a Forest into a Forest**

#### 1.3.4 $n$ -Dimensional forest substitution

The substitution of one  $n$ -dimensional forest into another is a generalization of the definition given in Section 1.1.

**Definition 11** For  $n, k, k' \in \mathcal{N}$ ,  $f_1 \in H_n^k$  and  $f_2 \in H_n^{k'}$ ,  $f_1 \Leftarrow f_2$  denotes the result of substituting  $(f_2)_\rho$  for each  $x_n^\rho$  in  $f_1$ .



Figure 1.3 is an illustration of this. Note that  $(f_2)_\rho$  may not be defined. If so, the substitution is not defined.

For  $f_1 \in H_n^k$ , and  $L_2 \subseteq H_n^{k'}$ , we define the substitution of the set  $L_2$  into the forest  $f_1$  by:

**Definition 12** Let  $f_1 \in H_n^k$ ,  $L_2 \subseteq H_n^{k'}$ .

The IO substitution of  $L_2$  into  $f_1$ , denoted by  $f_1 \stackrel{\text{IO}}{\Leftarrow} L_2$ , is defined to be the forest language:  $\{f_1 \Leftarrow f_2 \mid f_2 \in L_2\}$ .

The OI substitution of  $L_2$  into  $f_1$ , denoted by  $f_1 \stackrel{\text{OI}}{\Leftarrow} L_2$ , is defined recursively by:

1. If  $f_1 = x_n^\rho [_{n-1} t_{n-1}] \cdots [_{k} t_k]$  then,  

$$f_1 \stackrel{\text{OI}}{\Leftarrow} L_2 = \{(f_2)_\rho [_{n-1} s_{n-1}] \cdots [_{k} s_k] \mid f_2 \in L_2, s_i \in t_i \stackrel{\text{OI}}{\Leftarrow} L_2\}$$
2. If  $f_1 = a [_{n} t_n] \cdots [_{k} t_k]$  then,  

$$f_1 \stackrel{\text{OI}}{\Leftarrow} L_2 = \{a [_{n} s_n] \cdots [_{k} s_k] \mid s_i \in t_i \stackrel{\text{OI}}{\Leftarrow} L_2\}$$

□

Figure 1.4 illustrates that these two concepts are in fact different.

Finally, for two forest languages we define the substitution of one into the other.

**Definition 13** Let  $L_1 \subseteq H_n^k$  and  $L_2 \subseteq H_n^{k'}$ .

The IO substitution of  $L_2$  into  $L_1$ , denoted by  $L_1 \stackrel{\text{IO}}{\Leftarrow} L_2$ , is defined to be the set  $\cup_{f_1 \in L_1} (f_1 \stackrel{\text{IO}}{\Leftarrow} L_2)$ .

The OI substitution of  $L_2$  into  $L_1$ , denoted by  $L_1 \stackrel{\text{OI}}{\Leftarrow} L_2$ , is defined to be the set  $\cup_{f_1 \in L_1} (f_1 \stackrel{\text{OI}}{\Leftarrow} L_2)$ .

□

Figure 1.5 shows the difference between IO and OI substitution for sets.

The correspondence between forest substitution and ordinary tree substitution as defined in Section 1.1 can be seen by considering the forest  $L_2$  as a structure which holds the individual objects from the sets of trees  $L_1, \dots, L_{|\omega|}$  of tree substitution and replacing  $x_{i, \omega_i}$  by  $x_n^{1^{(i-1)}}$ . Under this correspondence,  $t[t_1, \dots, t_{|\omega|}]$  becomes  $t \Leftarrow$

$$f_1 = \begin{array}{c} a-2 \\ | \\ x_2^\lambda-1 \\ | \\ x_2^\lambda-1 \\ | \\ x_1^\lambda \end{array} \quad L_2 = \{ \begin{array}{c} b-1 \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} c-1 \\ | \\ x_1^\lambda \end{array} \}$$

$$f_1 \stackrel{\Leftarrow}{\underset{IO}{\rightleftharpoons}} L_2 = \{ \begin{array}{c} a-2 \\ | \\ b-1 \\ | \\ b-1 \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} a-2 \\ | \\ c-1 \\ | \\ c-1 \\ | \\ x_1^\lambda \end{array} \}$$

$$f_1 \stackrel{\Leftarrow}{\underset{OI}{\rightleftharpoons}} L_2 = \{ \begin{array}{c} a-2 \\ | \\ b-1 \\ | \\ b-1 \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} a-2 \\ | \\ b-1 \\ | \\ c-1 \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} a-2 \\ | \\ c-1 \\ | \\ c-1 \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} a-2 \\ | \\ c-1 \\ | \\ b-1 \\ | \\ x_1^\lambda \end{array} \}$$

Figure 1.4: Substitution of a Set into a Forest

$$L_1 = \left\{ \begin{array}{c} d-2 \\ | \\ a-2 \text{ --- } 1 \\ | \quad | \\ x_2^\lambda \text{ --- } 1 \quad x_2^\lambda \text{ --- } 1 \\ | \quad | \quad | \\ x_2^\lambda \text{ --- } 1 \quad x_1^\lambda \\ | \\ x_1^\lambda \end{array} \right\}, \left\{ \begin{array}{c} a-2 \\ | \\ x_2^1 \text{ --- } 1 \\ | \\ x_2^1 \text{ --- } 1 \\ | \\ x_1^\lambda \end{array} \right\}$$

$$L_1 \stackrel{\Leftarrow}{\underset{IO}{L_2}} = \left\{ \begin{array}{c} d-2 \\ | \\ a-2 \text{ --- } 1 \\ | \quad | \\ b-1 \quad b-1 \\ | \quad | \\ c-1 \quad x_1^\lambda \\ | \\ x_1^\lambda \end{array} , \begin{array}{c} d-2 \\ | \\ a-2 \text{ --- } 1 \\ | \quad | \\ c-1 \quad c-1 \\ | \quad | \\ c-1 \quad x_1^\lambda \\ | \\ x_1^\lambda \end{array} , \begin{array}{c} a-2 \\ | \\ c-1 \\ | \\ c-1 \\ | \\ x_1^\lambda \end{array} \right\}$$

$$L_1 \stackrel{\Leftarrow}{O_I} L_2 = \{ \begin{array}{c} d-2 \\ | \\ a-2 \text{ --- } 1 \\ | \quad | \\ b-1 \quad b-1 \\ | \quad | \\ c-1 \quad x_1^\lambda \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} d-2 \\ | \\ a-2 \text{ --- } 1 \\ | \quad | \\ b-1 \quad c-1 \\ | \quad | \\ c-1 \quad x_1^\lambda \\ | \\ x_1^\lambda \end{array}, \dots, \begin{array}{c} d-2 \\ | \\ a-2 \text{ --- } 1 \\ | \quad | \\ c-1 \quad b-1 \\ | \quad | \\ c-1 \quad x_1^\lambda \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} a-2 \\ | \\ c-1 \\ | \\ c-1 \\ | \\ x_1^\lambda \end{array} \}$$

Figure 1.5: Substitution of a Set into a Set

$$\left( \left( a \begin{array}{c} \downarrow \\ x_2^\lambda \end{array} \leftarrow a \begin{array}{c} \downarrow \\ x_2^\lambda \end{array} \right) \leftarrow b \right) = \left( a \begin{array}{c} \downarrow \\ x_1^\lambda \end{array} \leftarrow b \right) = a \begin{array}{c} \downarrow \\ x_1^\lambda \end{array}$$

while

$$\left( a \begin{array}{c} \downarrow \\ x_2^\lambda \end{array} \leftarrow \left( a \begin{array}{c} \downarrow \\ x_2^\lambda \end{array} \leftarrow b \right) \right) \text{ is not defined}$$

Figure 1.6: Nonassociativity of Substitution in General

$t_1[_1 t_2[_1 \dots [_1 t_{|\omega|}] \dots ]]$ ,  $t_{IO(OI)}(L_1, \dots, L_{|\omega|})$  becomes  $t_{IO(OI)}\{t_1[_1 t_2[_1 \dots [_1 t_{|\omega|}] \dots ]]\mid t_i \in L_i\}$  and,  $L_{IO(OI)}(L_1, \dots, L_{|\omega|})$  becomes  $L_{IO(OI)}\{t_1[_1 t_2[_1 \dots [_1 t_{|\omega|}] \dots ]]\mid t_i \in L_i\}$ .

If no errors are encountered in the search operation, both types of substitution are associative. Figure 1.6 illustrates the fact that substitution is not associative when an error is encountered in a search operation. If no error is encountered,  $OI$  substitution is associative as with the traditional trees. In the  $IO$  case associativity comes from the fact that the trees used in the substitutions are held together in forests.

**Lemma 1** For  $f_1 \in H_n^k$ ,  $L_2 \subseteq H_n^{k'}$ , and  $L_3 \subseteq H_n^{k''}$ , if both  $f_1 \xleftrightarrow{IO} L_2$  and  $L_2 \xleftrightarrow{IO} L_3$  have no errors, then  $f_1 \xleftrightarrow{IO} (L_2 \xleftrightarrow{IO} L_3) = (f_1 \xleftrightarrow{IO} L_2) \xleftrightarrow{IO} L_3$ .

*Proof :*

$$\begin{aligned} f_1 \xleftrightarrow{IO} (L_2 \xleftrightarrow{IO} L_3) &= \{f_1 \leftarrow f \mid f \in (L_2 \xleftrightarrow{IO} L_3)\} \\ &= \{f_1 \leftarrow f \mid f \in \cup_{f_2 \in L_2} (f_2 \xleftrightarrow{IO} L_3)\} \\ &= \{f_1 \leftarrow f \mid f \in \cup_{f_2 \in L_2} \{f_2 \leftarrow f_3 \mid f_3 \in L_3\}\} \end{aligned}$$

$$\begin{aligned}
&= \{f_1 \Leftarrow (f_2 \Leftarrow f_3) \mid f_2 \in L_2 \wedge f_3 \in L_3\} \\
&= \{(f_1 \Leftarrow f_2) \Leftarrow f_3 \mid f_2 \in L_2 \wedge f_3 \in L_3\} \\
&= (f_1 \Leftarrow_{IO} L_2) \Leftarrow_{IO} L_3
\end{aligned}$$

□

We use this lemma in the following theorem to prove that *IO* forest substitution is associative when no errors are encountered.

**Theorem 2** For  $L_1 \subseteq H_n^k$ ,  $L_2 \subseteq H_n^{k'}$ , and  $L_3 \subseteq H_n^{k''}$ , if  $L_1 \Leftarrow_{IO} L_2$  and  $L_2 \Leftarrow_{IO} L_3$  have no errors, then

$$L_1 \Leftarrow_{IO} (L_2 \Leftarrow_{IO} L_3) = (L_1 \Leftarrow_{IO} L_2) \Leftarrow_{IO} L_3.$$

*Proof :*

$$\begin{aligned}
L_1 \Leftarrow_{IO} (L_2 \Leftarrow_{IO} L_3) &= \cup_{f_1 \in L_1} (f_1 \Leftarrow_{IO} (L_2 \Leftarrow_{IO} L_3)) \\
&= \cup_{f_1 \in L_1} ((f_1 \Leftarrow_{IO} L_2) \Leftarrow_{IO} L_3) \\
&= \cup_{f_1 \in L_1} (\{f_1 \Leftarrow_{IO} f_2 \mid f_2 \in L_2\} \Leftarrow_{IO} L_3) \\
&= (\{f_1 \Leftarrow_{IO} f_2 \mid f_1 \in L_1 \wedge f_2 \in L_2\}) \Leftarrow_{IO} L_3 \\
&= (L_1 \Leftarrow_{IO} L_2) \Leftarrow_{IO} L_3
\end{aligned}$$

□

The proof of associativity for the *OI* case requires two lemmas. The first lemma states that if no errors are encountered, then we may substitute into a tree before or after a search operation and still get the same result. This lemma is then used to prove a lemma corresponding to the *IO* lemma above.

**Lemma 2** For  $\rho \in \{1, \dots, n-1\}^*$ ,  $f \in H_n^k$  and  $L \in H_n^{k'}$ , if  $(f)_\rho$  is not an error and  $f \Leftarrow_{OI} L$  contains no errors, then  $(f \Leftarrow_{OI} L)_\rho = (f)_\rho \Leftarrow_{OI} L$ .

*Proof* : The proof is by induction on the length of the path  $\rho$ .

For  $\rho = \lambda$  the length is 0.

1. If  $f = a[_n t_n] \cdots [_k t_k]$ , then

$$\begin{aligned}
 & \left( a[_n t_n] \cdots [_k t_k] \overleftarrow{\overline{OI}} L \right)_\lambda \\
 &= \left( \left\{ a[_n s_n] \cdots [_k s_k] \mid s_i \in t_i \overleftarrow{\overline{OI}} L \right\} \right)_\lambda \\
 &= \left\{ a[_n s_n] \mid s_n \in t_n \overleftarrow{\overline{OI}} L \right\} \\
 &= (f)_\rho \overleftarrow{\overline{OI}} L
 \end{aligned}$$

2. If  $f = x_n^{\rho'}[_{n-1} t_{n-1}] \cdots [_k t_k]$ , then

$$\begin{aligned}
 & \left( x_n^{\rho'}[_{n-1} t_{n-1}] \cdots [_k t_k] \overleftarrow{\overline{OI}} L \right)_\lambda \\
 &= \left( \left\{ (f_2)_{\rho'}[_{n-1} s_{n-1}] \cdots [_k s_k] \mid s_j \in t_j \overleftarrow{\overline{OI}} L \wedge f_2 \in L \right\} \right)_\lambda \\
 &= \left\{ (f_2)_{\rho'} \mid f_2 \in L \right\} \\
 &= (f)_\lambda \overleftarrow{\overline{OI}} L
 \end{aligned}$$

Now assume that the lemma is true for all paths of length less than the length of  $\rho$ ,

and let  $\rho = j\rho'$ , then  $f = t[_j t_j] \cdots [_k t_k]$  so

$$\begin{aligned}
 & \left( t[_j t_j] \cdots [_k t_k] \overleftarrow{\overline{OI}} L \right)_{j\rho'} \\
 &= \left( \left\{ t[_j s_j] \cdots [_k s_k] \mid s_i \in t_i \overleftarrow{\overline{OI}} L \right\} \right)_{j\rho'} \\
 &= \left( \left\{ s_j \mid s_j \in t_j \overleftarrow{\overline{OI}} L \right\} \right)_{\rho'} \\
 &= \left( t_j \overleftarrow{\overline{OI}} L \right)_{\rho'} \\
 &= (t_j)_{\rho'} \overleftarrow{\overline{OI}} L \\
 &= (f)_\rho \overleftarrow{\overline{OI}} L.
 \end{aligned}$$

□

**Lemma 3** For  $f_1 \in H_n^k$ ,  $L_2 \subseteq H_n^{k'}$ , and  $L_3 \subseteq H_n^{k''}$ , if both  $f_1 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_2$  and  $L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3$  have no errors, then  $f_1 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} (L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3) = (f_1 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_2) \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3$ .

*Proof* : The proof is by induction on the depth of the forest.

If  $\text{depth}(f) = 1$ , then

1. if  $f = x_n^\rho$ , then

$$\begin{aligned}
 x_n^\rho \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} (L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3) &= \left\{ (f_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3)_\rho \mid f_2 \in L_2 \right\} \\
 &= \left\{ (f_2)_\rho \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3 \mid f_2 \in L_2 \right\} \\
 &= (f \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_2) \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3
 \end{aligned}$$

2. if  $f = a$ , then  $a \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} (L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3) = (a \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_2) \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3 = a$ .

If  $\text{depth}(f) = m$  and we assume that it is true for all  $m' < m$ , then

1. if  $f = a[n \ t_n] \cdots [k \ t_k]$ , then

$$\begin{aligned}
 f \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} (L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3) &= \left\{ a[n \ s_n] \cdots [k \ s_k] \mid s_i \in t_i \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} (L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3) \right\} \\
 &= \left\{ a[n \ s_n] \cdots [k \ s_k] \mid s_i \in (t_i \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_2) \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3 \right\} \\
 &= (f \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_2) \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3
 \end{aligned}$$

2. if  $f = x_n^\rho[n_{-1} \ t_{n-1}] \cdots [k \ t_k]$ , then

$$\begin{aligned}
 f \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} (L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3) &= \left\{ t[n_{-1} \ s_{n-1}] \cdots [k \ s_k] \mid s_i \in t_i \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} (L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3) \wedge t \in (L_2 \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3)_\rho \right\} \\
 &= \left\{ t[n_{-1} \ s_{n-1}] \cdots [k \ s_k] \mid s_i \in (t_i \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_2) \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3 \wedge t \in (L_2)_\rho \stackrel{\Leftarrow}{\underset{OI}{\Leftarrow}} L_3 \right\}
 \end{aligned}$$

and

$$\begin{aligned}
& (f \stackrel{\Leftarrow}{O_I} L_2) \stackrel{\Leftarrow}{O_I} L_3 \\
&= \left\{ t' [_{n-1} s'_{n-1} ] \cdots [ _k s'_k ] \mid s'_i \in t_i \stackrel{\Leftarrow}{O_I} (L_2 \stackrel{\Leftarrow}{O_I} L_3) \wedge t' \in (L_2)_\rho \right\} \stackrel{\Leftarrow}{O_I} L_3 \\
&= \left\{ t [_{n-1} s_{n-1} ] \cdots [ _k s_k ] \mid s_i \in (t_i \stackrel{\Leftarrow}{O_I} L_2) \stackrel{\Leftarrow}{O_I} L_3 \wedge t \in (L_2)_\rho \stackrel{\Leftarrow}{O_I} L_3 \right\}.
\end{aligned}$$

So the two are equal.

□

**Theorem 3** For  $L_1 \subseteq H_n^k$ ,  $L_2 \subseteq H_n^{k'}$ , and  $L_3 \subseteq H_n^{k''}$ , if  $L_1 \stackrel{\Leftarrow}{O_I} L_2$  and  $L_2 \stackrel{\Leftarrow}{O_I} L_3$  have no errors, then

$$L_1 \stackrel{\Leftarrow}{O_I} (L_2 \stackrel{\Leftarrow}{O_I} L_3) = (L_1 \stackrel{\Leftarrow}{O_I} L_2) \stackrel{\Leftarrow}{O_I} L_3.$$

*Proof :*

$$\begin{aligned}
& L_1 \stackrel{\Leftarrow}{O_I} (L_2 \stackrel{\Leftarrow}{O_I} L_3) \\
&= \cup_{f_1 \in L_1} \left\{ f_1 \stackrel{\Leftarrow}{O_I} \left\{ L_2 \stackrel{\Leftarrow}{O_I} L_3 \right\} \right\} \\
&= \cup_{f_1 \in L_1} \left\{ \left\{ f_1 \stackrel{\Leftarrow}{O_I} L_2 \right\} \stackrel{\Leftarrow}{O_I} L_3 \right\} \\
&= \cup_{f \in (f_1 \stackrel{\Leftarrow}{O_I} L_2) \wedge f_1 \in L_1} \left\{ f \stackrel{\Leftarrow}{O_I} L_3 \right\} \\
&= (L_1 \stackrel{\Leftarrow}{O_I} L_2) \stackrel{\Leftarrow}{O_I} L_3.
\end{aligned}$$

□

### 1.3.5 The frontier of an $n$ -dimensional forest

The frontier of an  $n$ -dimensional forest was defined by Baldwin [12]. Taking the frontier of an  $n$ -dimensional forest yields an  $(n - 1)$ -dimensional forest. This process

may be repeated  $n - 1$  times to yield a 1-dimensional object corresponding to the usual string.

**Definition 14** *The  $n$ -dimensional frontier function  $f: \bigcup_{k=1}^n H_n^k(\Sigma) \rightarrow \bigcup_{l=1}^{n-1} H_{n-1}^l(\Sigma)$  is defined as:*

$$f(a[_n t_n ] \cdots [_k t_k ]) = \begin{cases} (f(t_n) \Leftarrow f(t_{n-1})) [_{n-2} f(t_{n-2}) ] \cdots [_k f(t_k) ] & t_n \neq \emptyset \\ a[_{n-1} f(t_{n-1}) ] \cdots [_k f(t_k) ] & t_n = \emptyset \end{cases}$$

□

In fact, if  $t \in H_n^k$ ,  $k < n$ , then  $f(t) \in H_{n-1}^k$  and if  $t \in H_n^n$ , then  $f(t) \in H_{n-1}^{n-1}$ .

If  $f(t_{n-1}) = \emptyset$ , then  $f(t_n) \Leftarrow f(t_{n-1}) = f(t_n)$ . If not, then the  $f(t_{n-1})$  will be in  $H_{n-1}^{n-2} = H_{n-2}(H_{n-1}^{n-1})$ , that is, an  $(n-2)$ -dimensional tree of  $(n-1)$ -dimensional trees.  $\Leftarrow$  is the substitution defined in the previous section. Each  $x_{n-1}^\rho \in X_{n-1}$  appearing in  $f(t_n)$  is replaced by the tree  $(f(t_{n-1}))_\rho$ . We say that a tree is *frontierable* if no errors are encountered during the fronttering operation.

Figures 1.7 and 1.8 show a 3-dimensional tree along with its 1 and 2-dimensional frontiers. The computation of the 2-dimensional frontier of the tree is in Figure 1.7. The actual frontier is the first tree in Figure 1.8 and the remainder of this figure shows the computation of the 1-dimensional frontier from the 2-dimensional frontier. The 1-dimensional frontier may also be viewed as the string oaoaoao.

$$\begin{aligned}
& f( \begin{array}{c} * - 3 \\ | \\ * - 3 \text{ --- } 2 \\ | \quad | \\ * - 3 \text{ --- } 2 \quad * - 2 \text{ --- } 1 \\ | \quad | \quad | \quad | \\ x_2^\lambda \quad * - 2 \text{ --- } 1 \quad o - 1 \quad x_1^\lambda \\ | \quad | \quad | \quad | \\ x_2^\lambda \text{ --- } 1 \quad x_1^\lambda \quad * - 2 \text{ --- } 1 \\ | \quad | \quad | \quad | \\ x_2^\lambda \text{ --- } 1 \quad x_1^\lambda \quad * - 2 \text{ --- } 1 \quad x_1^\lambda \\ | \quad | \quad | \quad | \\ x_1^\lambda \quad * - 2 \text{ --- } 1 \quad x_1^\lambda \\ | \quad | \\ a - 1 \quad o - 1 \\ | \quad | \\ x_1^\lambda \quad x_1^\lambda \end{array} ) \\
&= f( \begin{array}{c} * - 3 \text{ --- } 2 \\ | \quad | \\ * - 2 \text{ --- } 1 \\ | \quad | \\ x_2^\lambda \quad * - 2 \text{ --- } 1 \\ | \quad | \\ x_2^\lambda \text{ --- } 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda \text{ --- } 1 \\ | \\ x_1^\lambda \end{array} ) \Leftarrow f( \begin{array}{c} * - 2 \text{ --- } 1 \\ | \quad | \\ o - 1 \quad x_1^\lambda \\ | \quad | \\ * - 2 \text{ --- } 1 \\ | \quad | \\ * - 2 \text{ --- } 1 \quad x_1^\lambda \\ | \quad | \\ * - 2 \text{ --- } 1 \quad o - 1 \\ | \quad | \\ a - 1 \quad o - 1 \\ | \quad | \\ x_1^\lambda \quad x_1^\lambda \end{array} ) \\
&= ( x_2^\lambda \Leftarrow f( \begin{array}{c} * - 2 \text{ --- } 1 \\ | \quad | \\ x_2^\lambda \text{ --- } 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda \text{ --- } 1 \\ | \\ x_1^\lambda \end{array} ) ) \Leftarrow f( \begin{array}{c} * - 2 \text{ --- } 1 \\ | \quad | \\ o - 1 \quad x_1^\lambda \\ | \quad | \\ * - 2 \text{ --- } 1 \\ | \quad | \\ * - 2 \text{ --- } 1 \quad x_1^\lambda \\ | \quad | \\ * - 2 \text{ --- } 1 \quad o - 1 \\ | \quad | \\ a - 1 \quad o - 1 \\ | \quad | \\ x_1^\lambda \quad x_1^\lambda \end{array} ) \\
&= \begin{array}{c} * - 2 \\ | \\ x_2^\lambda \text{ --- } 1 \\ | \\ x_2^\lambda \text{ --- } 1 \\ | \\ x_1^\lambda \end{array} \Leftarrow \begin{array}{c} * - 2 \text{ --- } 1 \\ | \quad | \\ o - 1 \quad x_1^\lambda \\ | \quad | \\ * - 2 \text{ --- } 1 \\ | \quad | \\ * - 2 \text{ --- } 1 \quad x_1^\lambda \\ | \quad | \\ * - 2 \text{ --- } 1 \quad o - 1 \\ | \quad | \\ a - 1 \quad o - 1 \\ | \quad | \\ x_1^\lambda \quad x_1^\lambda \end{array}
\end{aligned}$$

Figure 1.7: Computation of a 2-dimensional frontier

### 1.3.6 $n$ -Dimensional grammars

Baldwin also defined regular  $n$ -dimensional forest languages and  $IO$  forest languages corresponding to the  $IO$  macro languages of Fischer [27] and Engelfriet-Schmidt [24,25]. He extended the concept of tree languages to include dimensions greater than two.

**Definition 15** *An  $n$ -dimensional tree grammar of degree  $k$ , denoted  $G_n^k$ , is a quadruple  $(\Sigma, \mathcal{F}, P, S)_n^k$  where*

1.  $\Sigma$  is the terminal alphabet;
2.  $\mathcal{F} = \langle \mathcal{F}_l \rangle_{1 \leq l \leq n}$  is a ranked set called the nonterminal alphabet;
3.  $S \in \mathcal{F}_k$  is the start symbol;
4.  $P = \langle P_l \rangle_{1 \leq l \leq n}$  is a ranked set of productions of the form  $F \rightarrow \alpha \in P_l$  for  $F \in \mathcal{F}_l$  and  $\alpha \in H_n^l(\Sigma, \mathcal{F} \cup X)$ .

□

The derivation relation for  $n$ -dimensional forest languages is much the same as that of Regular and Context-free string languages. The nonterminals of degree  $n$  are replaced by trees of the same degree if the replacement is allowed by the productions of the grammar.

**Definition 16** *The  $IO$  derivation relation,  $H_n^k(\Sigma, \mathcal{F} \cup X) \xRightarrow{G_n^k} H_n^k(\Sigma, \mathcal{F} \cup X)$  for  $n \geq k > 0$ , over the grammar  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , is defined by: If  $\alpha F \beta \in H_n^k(\Sigma, \mathcal{F} \cup X)$  with  $F \in \mathcal{F}_l$  and  $F \rightarrow \gamma \in P_l$ , then  $\alpha F \beta \xRightarrow{G_n^k} \alpha \gamma \beta$ .*

□

$$\begin{aligned}
& f( \begin{array}{c} * - 2 \\ | \\ * - 2 \text{---} 1 \\ | \quad | \\ o - 1 \quad * - 2 \text{---} 1 \\ | \quad | \quad | \\ * - 2 \text{---} 1 \quad o - 1 \quad x_1^\lambda \\ | \quad | \quad | \\ * - 2 \text{---} 1 \quad a - 1 \quad o - 1 \quad x_1^\lambda \\ | \quad | \quad | \\ x_1^\lambda \quad x_1^\lambda \quad x_1^\lambda \end{array} ) \\
& f( \begin{array}{c} * - 2 \\ | \\ * - 2 \text{---} 1 \\ | \quad | \\ o - 1 \quad * - 2 \text{---} 1 \\ | \quad | \quad | \\ * - 2 \text{---} 1 \quad o - 1 \quad x_1^\lambda \\ | \quad | \quad | \\ * - 2 \text{---} 1 \quad a - 1 \quad o - 1 \quad x_1^\lambda \\ | \quad | \quad | \\ x_1^\lambda \quad x_1^\lambda \quad x_1^\lambda \end{array} ) \Leftarrow f( \begin{array}{c} * - 2 \text{---} 1 \\ | \\ o - 1 \quad x_1^\lambda \\ | \\ * - 2 \text{---} 1 \\ | \quad | \\ * - 2 \text{---} 1 \quad o - 1 \\ | \quad | \\ a - 1 \quad o - 1 \quad x_1^\lambda \\ | \quad | \\ x_1^\lambda \quad x_1^\lambda \end{array} ) \\
& = \begin{array}{c} o - 1 \\ | \\ f( \begin{array}{c} * - 2 \text{---} 1 \\ | \\ * - 2 \text{---} 1 \quad o - 1 \\ | \quad | \\ a - 1 \quad o - 1 \quad x_1^\lambda \\ | \quad | \\ x_1^\lambda \quad x_1^\lambda \end{array} ) \Leftarrow x_1^\lambda \Leftarrow ( \begin{array}{c} o - 1 \\ | \\ f( \begin{array}{c} * - 2 \text{---} 1 \\ | \\ * - 2 \text{---} 1 \quad o - 1 \\ | \quad | \\ a - 1 \quad o - 1 \quad x_1^\lambda \\ | \quad | \\ x_1^\lambda \quad x_1^\lambda \end{array} ) \Leftarrow x_1^\lambda \Leftarrow x_1^\lambda ) \\
\end{array} \\
& = \begin{array}{c} o - 1 \\ | \\ (((a - 1 \Leftarrow o - 1) \Leftarrow o - 1) \Leftarrow x_1^\lambda) \Leftarrow ( \begin{array}{c} o - 1 \\ | \\ (((a - 1 \Leftarrow o - 1) \Leftarrow o - 1) \Leftarrow x_1^\lambda) \end{array} ) \\
\end{array} \\
& = \begin{array}{c} o - 1 \\ | \\ a - 1 \\ | \\ o - 1 \\ | \\ o - 1 \\ | \\ a - 1 \\ | \\ o - 1 \\ | \\ o - 1 \\ | \\ x_1^\lambda \end{array}
\end{aligned}$$

Figure 1.8: Computation of the 1-dimensional frontier

This definition is extended to the reflexive transitive closure  $\xRightarrow{*}$  in the usual way. The language generated by  $G_n^k$  is  $\{\beta \in H_n^k(\Sigma) \mid S \xRightarrow{*}_{G_n^k} \beta\}$ . Any such  $\beta$  is called a *sentence* or *word* in the language. If  $S \xRightarrow{*} \omega$  and  $\omega \in H_n^k(\Sigma, X \cup \mathcal{F})$ , then  $\omega$  is called a *sentential form*.

As an example of an  $n$ -dimensional grammar, we give the 3-dimensional grammar of degree 3 for the language whose 1-dimensional frontier is the *IO* string language  $\{\omega \in (o, a)^* \mid \text{the number of } a\text{'s in } \omega \text{ is a power of 2}\}$  and the *IO* string language  $\{(o^{l_1} a o^{l_2})^{2^n} \mid n \geq 0, l_1, l_2 \geq 0\}$ . Using Baldwin's derivation technique we will get the *IO* string language and not the *OI* string language. Later we will give the definition of *OI* derivations and show how to get the language above from the grammar.

**Grammar 1** *An example of a 3-dimensional grammar of degree 3 is*

$$G_3^3 = (\{*, o, a\}, \{\langle S, F \rangle_3, \langle A \rangle_2\}, P, S)$$

*with  $P$  containing the productions listed below:*

1.  $S \rightarrow *[_3 F[_2 A[_1 x_1^\lambda ] ] ]$ ,
2.  $F \rightarrow *[_3 F[_2 *[_2 x_2^\lambda[_1 x_2^\lambda[_1 x_1^\lambda ] ] ] ][_1 x_1^\lambda ] ] ]$ ,
3.  $F \rightarrow *[_3 x_2^\lambda ]$ ,
4.  $A \rightarrow *[_2 o[_1 A[_1 x_1^\lambda ] ] ]$ ,
5.  $A \rightarrow *[_2 A[_1 o[_1 x_1^\lambda ] ] ]$ ,
6.  $A \rightarrow *[_2 a[_1 x_1^\lambda ] ]$

□

Associated with each of these grammars is a hierarchy of tree languages obtained by taking the successive frontiers of the trees generated by the productions. The 3-dimensional tree of Figure 1.6 can be derived from this grammar. The 2-dimensional tree in the figure is then in the 2-dimensional language associated with the grammar, and the 1-dimensional tree is in the 1-dimensional language.

These 1-dimensional languages form a hierarchy called the *Algebraic Language Hierarchy*. The level 1 languages are those whose grammars are  $G_1^1$ . And the level  $n$  languages are those whose grammars are  $G_n^1$ . Baldwin showed that ALH level 1 corresponds to Regular Languages, ALH level 2 corresponds to Context Free Languages, and ALH level 3 corresponds to the *IO* macro languages.

### 1.3.7 $n$ -Dimensional *OI* and *IO* derivations

As stated earlier, Baldwin's derivations proceed by completing all the substitutions before taking any frontiers. This is the *IO* mode of derivation. We have generalized the work of Baldwin to include both *OI* and *IO* derivations. We will use the frontier operation to drive the derivation; this means that a grammar,  $G_n^k$ , will produce a forest language in  $H_{n-1}^k$ , and a grammar,  $G_n^n$ , will produce a forest language in  $H_{n-1}^{n-1}$ . In the sequel, unless otherwise specified,  $H_n^k$  means  $H_n^k(\Sigma, X \cup \mathcal{F})$ .

The frontier operation which drives the derivation must deal with the nonterminals which appear in the forest being derived. Since nonterminals may have more than one right hand side, the frontier operation is now a relation. Its definition follows:

**Definition 17** *The frontier relation for  $n$ -dimensional grammars*

$f: \bigcup_{k=1}^n H_n^k \rightarrow \bigcup_{l=1}^{n-1} H_{n-1}^l$  is given as follows:

$$f(F[_j t_j] \dots [_k t_k]) = \bigcup_{\alpha \in \{\alpha | F \rightarrow \alpha \in P\}} \{f(\alpha[_j t_j] \dots [_k t_k])\}$$

$$f(a[_n t_n] \dots [_k t_k]) = \begin{cases} \left( f(t_n)_{IO(OI)} f(t_{n-1}) \right) [_{n-2} f(t_{n-2})] \dots [_k f(t_k)] & t_n \neq \emptyset \\ a[_{n-1} f(t_{n-1})] \dots [_k f(t_k)] & t_n = \emptyset \end{cases}$$

□

Another, perhaps more subtle, difference is in the substitution used by the frontier relation. Consider the expression  $f(t_n)_{IO(OI)} f(t_{n-1})$  in the definition above. If the substitutions are performed using *IO* substitution, then we will have the hierarchy of languages as defined by Baldwin. On the other hand, if we use *OI* substitution we have a new hierarchy called the *OI n-dimensional language hierarchy*. To distinguish which type of substitution is to be used we will use  $\xleftarrow{IO}$  or  $\xleftarrow{OI}$  in the substitution operation. It will be clear from the text which is meant if no substitution operation is present.

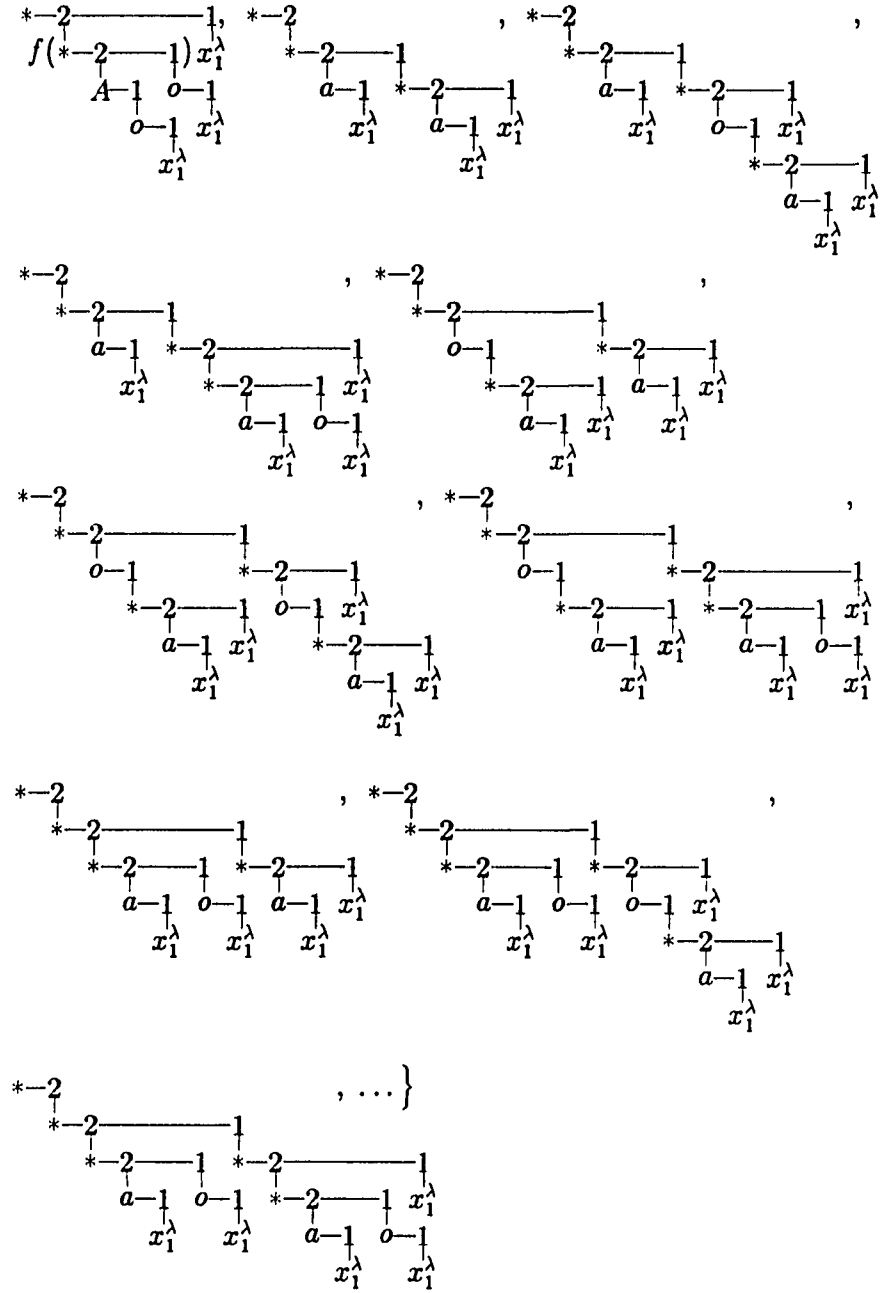
**Definition 18** Given an *n*-dimensional tree grammar  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , the context free *n*-dimensional forest language generated by  $G_n^k$  is given by  $\{\omega \in H_{n-1}^k(\Sigma) \mid f(S) = \omega\}$ . If the substitution used is  $\xleftarrow{IO}$ , then the language is Inside-out. If the substitution is  $\xleftarrow{OI}$ , then the language is Outside-in.

A language,  $L$ , is *n*-dimensional *OI* if there is a grammar  $G_{n'}^k = (\Sigma, \mathcal{F}, P, S)$  for some  $n' \leq n$  such that the  $n^{\text{th}}$  frontier of  $S$  is  $L$ . And similarly for *IO*.

□

$$\begin{aligned}
f(S) &= \left\{ f\left( \begin{array}{c} * - 3 \\ | \\ F - 2 \\ | \\ A - 1 \\ | \\ x_1^\lambda \end{array} \right) \right\} = \left\{ f\left( \begin{array}{c} F - 2 \\ | \\ A - 1 \\ | \\ x_1^\lambda \end{array} \right) \right\} \\
&= \left\{ f\left( \begin{array}{c} * - 3 - 2 \\ | \quad | \\ x_2^\lambda \quad A - 1 \\ | \quad | \\ x_1^\lambda \end{array} \right), f\left( \begin{array}{c} * - 3 \quad \quad 2 \\ | \quad \quad | \\ F - 2 \quad \quad A - 1 \\ | \quad \quad | \\ * - 2 \quad \quad 1 \\ | \quad \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \end{array} \right) \right\} \\
&= \left\{ x_2^\lambda \stackrel{OI}{\Leftarrow} f\left( \begin{array}{c} A - 1 \\ | \\ x_1^\lambda \end{array} \right), f\left( \begin{array}{c} F - 2 \\ | \\ * - 2 \quad \quad 1 \\ | \quad \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \end{array} \right) \stackrel{OI}{\Leftarrow} f\left( \begin{array}{c} A - 1 \\ | \\ x_1^\lambda \end{array} \right) \right\} \\
&= \left\{ \begin{array}{c} * - 2 \quad \quad 1 \\ | \quad \quad | \\ a - 1 \quad x_1^\lambda \\ | \quad \quad | \\ x_1^\lambda \end{array}, \begin{array}{c} * - 2 \quad \quad 1 \\ | \quad \quad | \\ o - 1 \quad x_1^\lambda \\ | \quad \quad | \\ * - 2 \quad \quad 1 \\ | \quad \quad | \\ a - 1 \quad x_1^\lambda \\ | \quad \quad | \\ x_1^\lambda \end{array}, \begin{array}{c} * - 2 \quad \quad 1 \\ | \quad \quad | \\ * - 2 \quad \quad 1 \\ | \quad \quad | \\ a - 1 \quad o - 1 \\ | \quad \quad | \\ x_1^\lambda \quad x_1^\lambda \end{array} \right\} \\
x_2^\lambda \stackrel{OI}{\Leftarrow} &\left\{ \begin{array}{c} * - 2 \quad \quad 1 \\ | \quad \quad | \\ o - 1 \quad x_1^\lambda \\ | \quad \quad | \\ f(* - 2 \quad \quad 1) \\ | \quad \quad | \\ o - 1 \quad x_1^\lambda \\ | \quad \quad | \\ A - 1 \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} * - 2 \quad \quad 1 \\ | \quad \quad | \\ o - 1 \quad x_1^\lambda \\ | \quad \quad | \\ f(* - 2 \quad \quad 1) \\ | \quad \quad | \\ A - 1 \quad x_1^\lambda \\ | \quad \quad | \\ o - 1 \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} * - 2 \quad \quad 1 \\ | \quad \quad | \\ f(* - 2 \quad \quad 1) \quad x_1^\lambda \\ | \quad \quad | \\ o - 1 \quad o - 1 \\ | \quad \quad | \\ A - 1 \quad x_1^\lambda \\ | \quad \quad | \\ x_1^\lambda \end{array} \right\}
\end{aligned}$$

Figure 1.9: Computation of  $f(S)$  using  $OI$  substitution

Figure 1.10: Completion of  $f(S)$  using  $OI$  substitution

$$\begin{aligned}
f(A \downarrow x_1^\lambda) &= \left\{ f\left( \begin{array}{c} * - 2 \text{---} 1 \\ | \\ a \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array} \right), f\left( \begin{array}{c} * - 2 \text{---} 1 \\ | \\ o \text{---} 1 \text{---} x_1^\lambda \\ | \\ A \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array} \right), f\left( \begin{array}{c} * - 2 \text{---} 1 \\ | \\ A \text{---} 1 \text{---} x_1^\lambda \\ | \\ o \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array} \right) \right\} \\
&= \left\{ \begin{array}{c} * - 2 \text{---} 1 \\ | \\ a \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} * - 2 \text{---} 1 \\ | \\ o \text{---} 1 \text{---} x_1^\lambda \\ | \\ * - 2 \text{---} 1 \text{---} x_1^\lambda \\ | \\ a \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} * - 2 \text{---} 1 \\ | \\ o \text{---} 1 \text{---} x_1^\lambda \\ | \\ f(* - 2 \text{---} 1) \\ | \\ o \text{---} 1 \text{---} x_1^\lambda \\ | \\ A \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} * - 2 \text{---} 1 \\ | \\ o \text{---} 1 \text{---} x_1^\lambda \\ | \\ f(* - 2 \text{---} 1) \\ | \\ A \text{---} 1 \text{---} x_1^\lambda \\ | \\ o \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array}, \right. \\
&\quad \left. \begin{array}{c} * - 2 \text{---} 1 \\ | \\ * - 2 \text{---} 1 \text{---} x_1^\lambda \\ | \\ a \text{---} 1 \text{---} o \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \quad x_1^\lambda \end{array}, \begin{array}{c} * - 2 \text{---} 1 \\ | \\ f(* - 2 \text{---} 1) \text{---} x_1^\lambda \\ | \\ o \text{---} 1 \text{---} o \text{---} 1 \text{---} x_1^\lambda \\ | \\ A \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array}, \begin{array}{c} * - 2 \text{---} 1 \\ | \\ f(* - 2 \text{---} 1) \text{---} x_1^\lambda \\ | \\ A \text{---} 1 \text{---} o \text{---} 1 \text{---} x_1^\lambda \\ | \\ o \text{---} 1 \text{---} x_1^\lambda \\ | \\ x_1^\lambda \end{array} \right\} \\
&= \dots
\end{aligned}$$

Figure 1.11: Computation of  $f(A \downarrow x_1^\lambda)$  using  $OI$  substitution

As an example of a language in this new hierarchy, consider the grammar of Section 1 on page 38. Figures 1.9, 1.10, 1.11, and 1.12 combined are an example derivation from this grammar. Those objects which can be derived from the subforest  $A[{}_1 x_1^\lambda]$  are shown in Figure 1.11, while those which can be derived from  $F[{}_2 * [{}_2 x_2^\lambda [{}_1 x_2^\lambda [{}_1 x_1^\lambda]]] [{}_1 x_1^\lambda]]$  are in Figure 1.11.

#### 1.4 Containment of the Engelfriet-Schmidt Hierarchy in the Baldwin-Schoenberger Hierarchy

In this section we show that the tree language hierarchy of Engelfriet and Schmidt as described in the earlier section is completely contained in the forest language hierarchy of Baldwin and Schoenberger. First we show how to convert the projections to the path variables in the Baldwin-Schoenberger system so that the paths will be correct. Second we show how to convert trees in the Engelfriet-Schmidt system to forests in the Baldwin-Schoenberger system. Then we show that any tree derived from the start symbol of an Engelfriet-Schmidt grammar has an equivalent tree in the Baldwin-Schoenberger system that can be derived from the corresponding Baldwin-Schoenberger grammar. Finally, we show that the string languages defined by the Engelfriet-Schmidt grammars are the same as the string languages defined by their equivalent Baldwin-Schoenberger grammars.

##### 1.4.1 Conversion of the projections, $\pi_i^\omega$ , into path variables, $x_k^\rho$

In the conversion from Engelfriet-Schmidt, see Section 1.2.3, to Baldwin-Schoenberger, each projection,  $\pi_i^\omega$ , in an Engelfriet-Schmidt tree is converted to a variable,  $x_k^\rho$ , in the equivalent Baldwin-Schoenberger forest. The value of  $k$  is determined by

$$\begin{aligned}
& f(F-2 \begin{array}{c} \text{---} 1 \\ | \\ * - 2 \text{---} 1 \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \end{array} ) \\
&= \left\{ f( \begin{array}{c} * - 3 - 2 \text{---} 1 \\ | \quad | \quad | \\ x_2^\lambda \quad * - 2 \text{---} 1 \quad x_1^\lambda \\ | \quad | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \end{array} ), f( \begin{array}{c} * - 3 \text{---} 2 \\ | \quad | \quad | \\ F-2 \quad * - 2 \text{---} 1 \\ | \quad | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \quad x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \quad | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \quad x_2^\lambda - 1 \quad x_1^\lambda \end{array} ) \right\} \\
&= \left\{ x_2^\lambda, f(F-2 \begin{array}{c} \text{---} 1 \\ | \\ * - 2 \text{---} 1 \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \end{array} ) \right\} \xrightarrow{OI} \left\{ \begin{array}{c} * - 2 \text{---} 1 \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \end{array} \right\} \\
&= \left\{ \begin{array}{c} * - 2 \text{---} 1 \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \end{array}, \begin{array}{c} * - 2 \text{---} 1 \\ | \quad | \quad | \\ x_2^\lambda - 1 \quad * - 2 \text{---} 1 \quad x_1^\lambda \\ | \quad | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \quad x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \end{array}, \begin{array}{c} * - 2 \text{---} 1 \\ | \quad | \quad | \quad | \\ x_2^\lambda - 1 \quad * - 2 \text{---} 1 \quad * - 2 \text{---} 1 \quad x_1^\lambda \\ | \quad | \quad | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \quad x_2^\lambda - 1 \quad x_1^\lambda \\ | \quad | \quad | \quad | \\ x_2^\lambda - 1 \quad x_1^\lambda \quad x_2^\lambda - 1 \quad x_1^\lambda \end{array}, \dots \right\} \\
&= \dots
\end{aligned}$$

Figure 1.12:  $f(F[{}_2 * [{}_2 x_2^\lambda [{}_1 x_2^\lambda [{}_1 x_1^\lambda ] ] ] [{}_1 x_1^\lambda ] ] )$  using  $OI$  substitution

the level of the derived alphabet at which  $\pi_i^\omega$  was introduced. In other words,  $k$  is such that  $\pi_i^\omega \in D^k(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}$  such that  $\omega \neq \lambda$ . The path,  $\rho$ , is then determined by both  $k$  and  $i$ . For a projection,  $\pi_i^\omega \in D^k(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}$  such that  $\omega \neq \lambda$ ,  $\rho = \text{path}(i, k)$  as calculated using the following definition.

**Definition 19** *For all  $i \in \mathcal{N}$  and  $k = 0, 1, \dots$ , the value of  $\text{path}(i, k) \in \mathcal{N}^*$  is defined recursively by:*

1.  $\text{path}(1, k) = \lambda$ ;
2.  $\text{path}(i, 1) = 1^{i-1}$ ;
3.  $\text{path}(i, k) = k \text{path}(i - 1, k - 1)$ .

□

**Example 8**

$$\pi_2^{sss} \in D^1(\Sigma)_{\langle \lambda, \langle sss, s \rangle \rangle} \text{ translates to } x_2^1,$$

$$\pi_1^{\langle s, s \rangle \langle s, s \rangle} \in D^2(\Sigma)_{\langle \lambda, \langle \langle s, s \rangle \langle s, s \rangle, \langle s, s \rangle \rangle \rangle} \text{ translates to } x_3^\lambda$$

and,

$$\pi_3^{\langle s, s \rangle \langle s, s \rangle \langle s, s \rangle} \in D^2(\Sigma)_{\langle \lambda, \langle \langle \langle s, s \rangle \langle s, s \rangle \langle s, s \rangle, \langle s, s \rangle \rangle \rangle \rangle} \text{ translates to } x_3^{21}.$$

□

#### 1.4.2 Conversion of Engelfriet-Schmidt trees to Baldwin-Schoenberger forests

The function  $T_n$  converts trees in the Engelfriet-Schmidt hierarchy to forests in the Baldwin-Schoenberger hierarchy. Languages at level  $n - 2$  in the Engelfriet-Schmidt hierarchy will be of dimension  $n$  in the Baldwin-Schoenberger hierarchy. We

will need to convert a tree from  $T_{D^{n-2}(\Sigma) \cup \mathcal{F}}(X_S)$  to a forest in  $H_n^n(D^{n-2}(\Sigma) \cup \{*\}, \mathcal{F} \cup X)$ . We do this by surrounding the variables with  $n$  brackets, converting the projections to path variables, and leaving the constants unchanged.

Before this conversion can take place, we first insert the symbol  $\pi_1^\lambda \in \Sigma_{(\lambda, \lambda)}$  at the right end of every “string” in the Engelfriet-Schmidt trees. This symbol is used to generate the 1-dimensional variable  $x_1^\lambda$ . Remember, this variable represents the end of a string and indicates that new strings may only be appended at the end of a string.

**Example 9** *The Engelfriet-Schmidt trees*

$$g(g(a \ a) \ g(a \ a)),$$

$$c_{s, \lambda, s}(c_{s, s, s}(c_{s, s, s}(g \ \pi_1^{ss} \ \pi_1^{ss}) \ c_{ss, s, s}(g \ \pi_1^{ss} \ \pi_1^{ss})) \ c_{\lambda, \lambda, s}(a))$$

and

$$c_{\langle s, s \rangle, \langle s, s \rangle, \langle s, s \rangle}(F \ c_{\langle s, s \rangle, \langle s, s \rangle, \langle s, s \rangle}(c_{s, s, s} \ \pi_1^{\langle s, s \rangle} \ \pi_1^{\langle s, s \rangle}))$$

become

$$g(g(a \ a \ \pi_1^\lambda) \ g(a \ a \ \pi_1^\lambda) \ \pi_1^\lambda)$$

$$c_{s, \lambda, s}(c_{s, s, s}(c_{ss, s, s}(g \ \pi_1^{ss} \ \pi_1^{ss} \ \pi_1^\lambda) \ c_{ss, s, s}(g \ \pi_1^{ss} \ \pi_1^{ss} \ \pi_1^\lambda) \ \pi_1^\lambda) \ c_{\lambda, \lambda, s}(a \ \pi_1^\lambda) \ \pi_1^\lambda)$$

and

$$c_{\langle s, s \rangle, \langle s, s \rangle, \langle s, s \rangle}(F \ c_{\langle s, s \rangle, \langle s, s \rangle, \langle s, s \rangle}(c_{s, s, s} \ \pi_1^{\langle s, s \rangle} \ \pi_1^{\langle s, s \rangle} \ \pi_1^\lambda) \ \pi_1^\lambda).$$

□

We also need a variant of subtraction. The reason for this will become apparent as we proceed.

**Definition 20** For all  $j, n \in \mathcal{N}$ ,  $n \dot{-} j$  is defined as:

$$n \dot{-} j = \begin{cases} n - j & \text{if } j < n, \\ 1 & \text{if } j \geq n. \end{cases}$$

□

**Definition 21**  $T_n: T_{D(D^{n-3}(\Sigma) \cup \mathcal{F}) \cup \{\pi_1^\lambda\}}(X_s) \rightarrow H_n^n(D(D^{n-3}(\Sigma) \cup \{*\}), \mathcal{F} \cup X)$  for  $n \geq 3$ , is defined recursively by:

1.  $T_n(F_i) = *|_n F_i|$ , for  $F_i \in \mathcal{F}_{\langle \omega, s \rangle}$ ;
2.  $T_n(\pi_i^\omega) = \begin{cases} *|_{n-1} \cdots |_{k+2} x_{k+1}^\lambda |_{k+1} x_{k+1}^\rho | \cdots | & \text{for } \pi_i^\omega \in D^k(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}, \omega \neq \lambda \\ & \wedge \rho = \text{path}(i, k), \text{ or } \pi_i^\omega = \pi_1^\lambda, \\ x_{n-1}^{\text{path}(i, n-2)} & \text{for } \pi_i^\omega \in D^{n-2}(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle} \wedge \omega \neq \lambda, \end{cases}$
3.  $T_n(x_{i, \omega_i}) = x_n^{\text{path}(i, n-1)}$ ,
4.  $T_n(f) = f$ , for  $f \in D^k(\Sigma)_{\langle \omega, s \rangle}$ ,
5.  $T_n(c_{\omega, \nu, s}(\sigma \sigma_1 \cdots \sigma_{|\omega|})) = c_{\omega, \nu, s}[|_n T_n(\sigma) |_{n-1} T_n(\sigma_1) |_{n-2} \cdots |_{n-|\omega|} T_n(\sigma_{|\omega|}) | \cdots | ]$ .

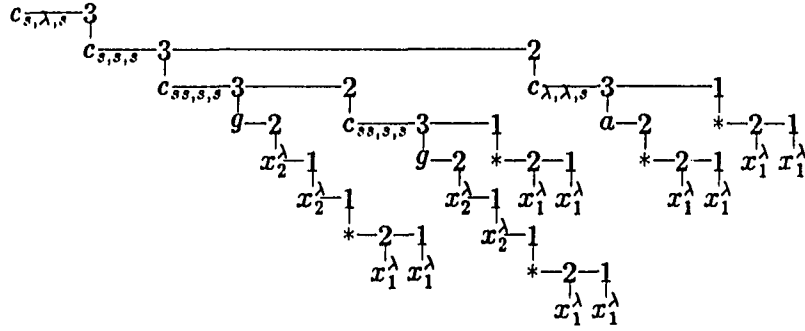
□

In the examples below, an augmented tree from  $T_{D(\Sigma \cup \mathcal{F})}$  is converted to a forest in  $H_2^2$  and an augmented tree from  $T_{D(D(\Sigma) \cup \mathcal{F})}$  is converted to a forest in  $H_3^3$ .

**Example 10** We convert the Engelfriet-Schmidt tree from example 5 on page 12 to its Baldwin-Schoenberger equivalent.

$$T_3 \left( c_{s, \lambda, s}(c_{s, s, s}(c_{ss, s, s}(g \pi_1^{ss} \pi_1^{ss} \pi_1^\lambda) c_{ss, s, s}(g \pi_1^{ss} \pi_1^{ss} \pi_1^\lambda) \pi_1^\lambda) c_{\lambda, \lambda, s}(a \pi_1^\lambda) \pi_1^\lambda) \right)$$

converts to:

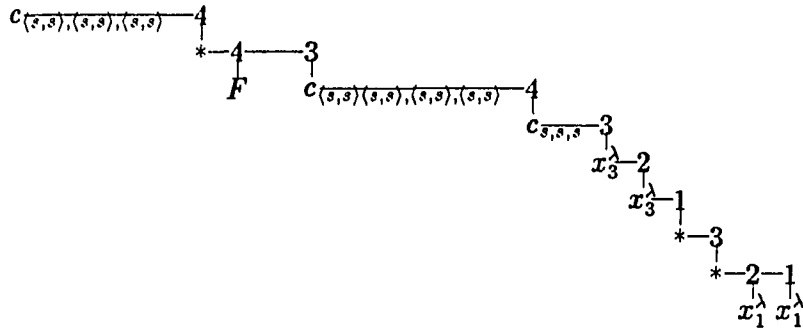


□

**Example 11**

$$T_4(c_{\langle s, s \rangle, \langle s, s \rangle, \langle s, s \rangle}(F c_{\langle s, s \rangle, \langle s, s \rangle, \langle s, s \rangle}(c_{s, s, s} \pi_1^{\langle s, s \rangle} \pi_1^{\langle s, s \rangle} \pi_1^\lambda \pi_1^\lambda)))$$

converts to:



□

In what follows the symbols  $\xleftarrow{IO}$  and  $\xleftarrow{OI}$  will be used for both Baldwin-Schoenberger forest substitution and Engelfriet-Schmidt tree substitution. From the context, it will be clear which is meant.

During an Engelfriet-Schmidt derivation, the  $\pi_i^\omega \in D^{n-2}(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}$  are converted to  $x_{i, \omega_i}$  which are in turn replaced during the substitution operation.  $T_n$  converts these  $\pi_i^\omega$  into  $x_{n-1}^{path(i, n-2)}$  which are then used in the substitution operation of the Baldwin-Schoenberger derivations. The following lemma establishes the fact that the paths created by  $T_n$  and  $path$  are indeed the correct paths.

**Lemma 4** Let  $t = t_1[ \underset{n-2}{\cdot} t_2[ \underset{n-3}{\cdot} t_3[ \underset{n-4}{\cdot} \cdots [ \underset{n-l}{\cdot} t_l ] \cdots ] ] ] \in H_{n-2}^{n-2}$ . Then for all  $n \in \mathcal{N}$ , and  $i = 1, \dots, l$ ,  $(t)_{path(i, n-2)} = t_i$ .

*Proof:* The proof is by induction on  $n$ .

*Basis:* If  $n = 1$ , then  $path(i, n-2) = 1^{i-1}$  and  $(t)_{1^{i-1}} = t_i$ , giving the desired result.

*Induction Step:* If  $n > 1$ , then  $path(i, n-2) = \lambda$  for  $i = 1$ , and  $(n-2) path(i-1, n-3)$  for  $i > 1$ . If  $path(i, n-2) = \lambda$ , then  $(t)_\lambda = t_i$  and we are done. If  $path(i, n-2) = (n-2) path(i-1, n-3)$ , then

$$(t)_{(n-2) path(i-1, n-3)} = \left( t_2[ \underset{n-3}{\cdot} t_3[ \underset{n-4}{\cdot} \cdots [ \underset{n-l}{\cdot} t_l ] \cdots ] ] \right)_{path(i-1, n-3)}$$

which is  $t_i$  as desired.

□

An Engelfriet-Schmidt derivation involves the computation of  $L_{IO(OI)}^{\Leftarrow}(L_1, \dots, L_{|\omega|})$  where  $L, L_1, \dots, L_{|\omega|} \in T_{D^{n-2}(\Sigma)}(X_S)$ .  $L$  is the set of trees that can be derived from  $\sigma$  and each  $L_i$  is the set of trees that can be derived from  $\sigma_i$  in an expression of the form  $c_{\omega, \nu, s}(\sigma \sigma_1 \cdots \sigma_{|\omega|})$ .  $T_n \left( L_{IO(OI)}^{\Leftarrow}(L_1, \dots, L_{|\omega|}) \right)$  is then the set of corresponding Baldwin-Schoenberger forests. The substitution should correspond to  $T_n(L)_{IO(OI)}^{\Leftarrow} T_n(L_1)[ \underset{n-2}{\cdot} T_n(L_2)[ \underset{n-3}{\cdot} \cdots [ \underset{n-|\omega|}{\cdot} T_n(L_{|\omega|}) ] \cdots ] ]$ . In fact, these two sets are equal since any path in  $T_n(L)$  which must be evaluated in the right hand side of the expression will be  $path(i, n-2)$  which, according to the just proved lemma, will find an object in  $T_n(L_i)$  while  $x_{i, \omega_i}$  in  $L$  will be replaced by an object in  $L_i$ . This proves the following corollary to the lemma.

**Corollary 1** Let  $L, L_1, \dots, L_l \subseteq T_{D^{n-2}(\Sigma)}(X_S)$ , then

$$\begin{aligned} & T_n(L_{IO(OI)}^{\Leftarrow}(L_1, \dots, L_l)) \\ &= T_n(L)_{IO(OI)}^{\Leftarrow} T_n(L_1)[ \underset{n-2}{\cdot} T_n(L_2)[ \underset{n-3}{\cdot} \cdots [ \underset{n-l}{\cdot} T_n(L_l) ] \cdots ] ]. \end{aligned}$$

□

### 1.4.3 Construction of a Baldwin-Schoenberger forest grammar from an Engelfriet-Schmidt tree grammar

We are now ready to give the construction which transforms a grammar for an Engelfriet-Schmidt tree language into a grammar for a Baldwin-Schoenberger forest language. Given an Engelfriet-Schmidt tree grammar,  $G = (D^{n-3}(\Sigma), \mathcal{F}, P, F_1)$ , we form the Baldwin-Schoenberger forest grammar,  $G_n^n = (\Sigma', \mathcal{F}', P', F'_1)$ , as follows:

$\Sigma' = D^{n-2}(\Sigma) \cup \{*\}$  where  $*$  is a symbol not appearing elsewhere in the grammar;

$\mathcal{F}' = \mathcal{F}'_n = \mathcal{F}$ ;

$P' = \{F'_i \rightarrow T_n(t) \mid t \text{ is the augmented form of } t' \text{ where}$   
 $t' \in COMB_{\nu}^{D^{n-1}(\Sigma) \cup \mathcal{F}}(rhs(F_i)), F_i \in \mathcal{F}_{(\nu,s)}\}$ ;

$F'_1 = F_1$ .

Notice that all nonterminals or function symbols are in  $\mathcal{F}'_n$ .

**Example 12** Consider example 6 on page 17.  $G = (\Sigma, \mathcal{F}, P, F_1)$  where  $\Sigma_{\langle \omega, s \rangle} = \{a, b\}$ ,  $\Sigma_{\langle s, s, s \rangle} = \{f\}$ ,  $\mathcal{F}_{\langle \lambda, s \rangle} = \{F_1, F_3\}$ ,  $\mathcal{F}_{\langle s, s \rangle} = \{F_2\}$  and  $P$  is the set of productions  $\{F_1 \rightarrow F_2(F_3), F_2(x_{1,s}) \rightarrow f(x_{1,s}, x_{1,s}), F_3 \rightarrow a, F_3 \rightarrow b\}$ .

The  $D(\Sigma)$  productions are:

$$COMB_{\lambda}^{\Sigma}(rhs(F_1)) = \{c_{s,\lambda,s}(F_2 c_{\lambda,\lambda,s}(F_3))\}$$

$$COMB_s^{\Sigma}(rhs(F_2)) = \{c_{ss,s,s}(f \pi_1^s \pi_1^s)\}$$

$$COMB_{\lambda}^{\Sigma}(rhs(F_3)) = \{c_{\lambda,\lambda,s}(a), c_{\lambda,\lambda,s}(b)\}$$

The Baldwin-Schoenberger grammar for this language has

$$\Sigma = \{c_{s,\lambda,s}, c_{ss,s,s}, c_{\lambda,\lambda,s}, f, a, b, *\};$$

$$\mathcal{F} = \{F_1, F_2, F_3\};$$

$P$  contains the following productions:

$$\begin{array}{ll}
 F_1 \rightarrow c_{\overline{s},\lambda,\overline{s}} \begin{array}{c} 3 \\ * \begin{array}{c} 3-2 \\ F_2 \end{array} \end{array} & F_2 \rightarrow c_{\overline{ss},s,\overline{s}} \begin{array}{c} 3 \\ f \begin{array}{c} 2-1 \\ x_2^\lambda \begin{array}{c} 1 \\ x_2^\lambda \begin{array}{c} 1 \\ * \begin{array}{c} 2-1 \\ x_1^\lambda \end{array} \end{array} \end{array} \end{array} \\
 F_2 \rightarrow c_{\lambda,\lambda,\overline{s}} \begin{array}{c} 3 \\ * \begin{array}{c} 3-2 \\ F_3 \end{array} \end{array} & F_3 \rightarrow c_{\lambda,\lambda,\overline{s}} \begin{array}{c} 3 \\ a \begin{array}{c} 2-1 \\ * \begin{array}{c} 2-1 \\ x_1^\lambda \end{array} \end{array} \\
 F_3 \rightarrow c_{\lambda,\lambda,\overline{s}} \begin{array}{c} 3 \\ a \begin{array}{c} 2-1 \\ * \begin{array}{c} 2-1 \\ x_1^\lambda \end{array} \end{array} & F_3 \rightarrow c_{\lambda,\lambda,\overline{s}} \begin{array}{c} 3 \\ b \begin{array}{c} 2-1 \\ * \begin{array}{c} 2-1 \\ x_1^\lambda \end{array} \end{array}
 \end{array}$$

□

We want to show that if a tree  $t$  is in the Engelfriet-Schmidt level  $(n-2)$   $IO$  or  $OI$  language of grammar  $G$ , then  $T_n(t)$  is in the Baldwin-Schoenberger  $n$ -dimensional  $IO(OI)$  forest language of  $G_n^n$ . We first show that if  $t$  can be derived from an Engelfriet-Schmidt tree  $\sigma$ , then  $T_n(t)$  can be derived from a Baldwin-Schoenberger forest  $T_n(\sigma)$ . We then use this result to prove the above statement.

For an Engelfriet-Schmidt tree grammar  $G = (\Sigma, \mathcal{F}, P, F_1)$ , the  $IO$  language,  $L$ , defined by  $G$  is  $M_{IO}(F_1)(|G_{IO}|)$  where  $|G_{IO}| = \cup_{i=0}^{\infty} M_{G,IO}^i(\perp)$ , the minimal fixed point of  $M_{G,IO}$ .  $M_{g,IO}(\perp) = (\hat{M}_{IO}(R_1)(\perp), \dots, \hat{M}_{IO}(R_{|\mathcal{F}|})(\perp))$  where  $R_i = COMB_{iv}^{\Sigma \cup \mathcal{F}}(rhs(F_i))$ , for all  $i = 1, \dots, |\mathcal{F}|$ .  $\hat{M}_{IO}$  is  $M_{IO}$  extended to sets. Both  $COMB$  and  $M_{IO}$  are defined on page 15 in Section 1.2.3. In what follows, we will use  $M$  for  $M_{G,IO(OI)}$ ,  $\hat{M}$  for  $\hat{M}_{IO(OI)}$ , and allow the label on  $\overleftarrow{IO}$ ,  $\overleftarrow{OI}$  to distinguish which form of substitution is being used.

**Theorem 4** Given an Engelfriet-Schmidt tree grammar  $G = (D^{n-3}(\Sigma), \mathcal{F}, P, F_1)$ , let  $G_n^n = (\Sigma', \mathcal{F}', P', F_1')_n^n$  be the  $n$ -dimensional Baldwin-Schoenberger grammar constructed as above, then for  $\sigma \in T_{D(D^{n-3}(\Sigma))}(\mathcal{F}')$  and  $t \in T_{D^{n-3}(\Sigma)}$  and some  $q$ ,

$$t \in (M(\sigma)(M^q(\perp))) \Rightarrow T_{n-1}(t) \in f(T_n(\sigma)).$$

*Proof:* First we show that

$$t \in (M(\sigma)(M^q(\perp))) \rightarrow T_{n-1}(t) \in f(T_n(\sigma))$$

The proof is by induction on  $q$ .

*Basis:* If  $q = 0$ , then  $M(\sigma)(M^0(\perp)) = M(\sigma)(\perp)$ . So  $\sigma = a \in D^{n-1}(\Sigma)_{(\lambda, s)}$  and  $T_n(a) = T_{n-1}(a) = a$ , and since  $f(a) = a$ , we have the desired result.

*Induction Step:* Assume  $t \in M(\sigma)(M^q(\perp))$  implies  $T_{n-1}(t) \in f(T_n(\sigma))$ . We must show that  $t \in M(\sigma)(M^{q+1}(\perp))$  implies that  $T_{n-1}(t) \in f(T_n(\sigma))$ . We do this by induction on the depth of  $\sigma$ .

*Basis:* If  $d(\sigma) = 1$ , then  $\sigma = a \in D^{n-3}(\Sigma)_{(\lambda, s)}$  or  $\sigma = F_i \in \mathcal{F}_{(\omega, s)}$ . If  $\sigma = a$ , then there is nothing to prove since this is the same as the basis step. If  $\sigma = F_i$ , then  $M(\sigma)(M^{q+1}(\perp)) = (M^{q+1}(\perp))_i$ . So letting  $k$  be the number of nonterminals in  $\mathcal{F}$  we have:

$$\begin{aligned} t \in (M^{q+1}(\perp))_i &= M(M^q(\perp))_i \\ &= (\hat{M}(R_1)(M^q(\perp)), \dots, \hat{M}(R_k)(M^q(\perp)))_i \\ &= \hat{M}(R_i)(M^q(\perp)) \end{aligned}$$

where  $R_i = \text{COMB}_v^{D^{n-3}(\Sigma) \cup \mathcal{F}}(F_i)$ . So for some  $r \in R_i$ ,  $t \in M(r)(M^q(\perp))_i$  and by the induction hypothesis:

$$T_{n-1}(t) \in f(T_n(r))$$

$$\begin{aligned}
&\in f(*[{}_n F_i]) \\
&= f(T_n(\sigma)).
\end{aligned}$$

*Induction Step:* Assume that  $t \in M(\sigma)(M^{q+1}(\perp))$  implies that  $T_{n-1}(t) \in f(T_n(\sigma))$  for  $\sigma$  such that  $d(\sigma) \leq p$  and consider  $\sigma$  such that  $d(\sigma) = p + 1$ . Then  $\sigma = c_{\omega, \nu, s}(\sigma' \sigma_1 \cdots \sigma_{|\omega|})$  and

$$\begin{aligned}
M(\sigma)(M^{q+1}(\perp)) & \\
= M(\sigma')(M^{q+1}(\perp))_{IO(\overline{OI})} \left( M(\sigma_1)(M^{q+1}(\perp)) \cdots M(\sigma_{|\omega|})(M^{q+1}(\perp)) \right). & \quad (1.1)
\end{aligned}$$

Since each  $\sigma_i$  and  $\sigma'$  has depth  $\leq p$  we know that  $T_{n-1}(\tau_i) \in f(T_n(\sigma_i))$  for  $\tau_i \in M(\sigma_i)(M^{q+1}(\perp))$  and  $T_{n-1}(\tau') \in f(T_n(\sigma'))$  for  $\tau' \in M(\sigma')(M^{q+1}(\perp))$ . Now,  $\sigma'$  can be  $a \in D^n(\Sigma)_{\langle \lambda, \langle \omega, s \rangle \rangle}$ ,  $F_i \in \mathcal{F}_{\langle \omega, s \rangle}$  or neither of these.

If  $\sigma' = a \in D^n(\Sigma)_{\langle \lambda, \langle \omega, s \rangle \rangle}$ , equation 1.1 is

$$M(\sigma)(M^{q+1}(\perp)) = \{a(\tau_1 \cdots \tau_{|\omega|}) \mid \tau_i \in M(\sigma_i)(M^{q+1}(\perp))\}$$

and for  $t \in M(\sigma)(M^{q+1}(\perp))$

$$\begin{aligned}
&T_{n-1}(t) \\
&\in \left\{ a[{}_{n-1} T_{n-1}(\tau_1)[{}_{n-2} T_{n-1}(\tau_2)[{}_{n-3} \cdots [{}_{n-|\omega|} T_{n-1}(\tau_{|\omega|})] \cdots ] ] \right. \\
&\quad \left. \mid \tau_i \in M(\sigma_i)(M^{q+1}(\perp)) \right\} \\
&\subseteq a[{}_{n-1} T_{n-1}(\sigma_1)[{}_{n-2} T_{n-1}(\sigma_2)[{}_{n-3} \cdots [{}_{n-|\omega|} T_{n-1}(\sigma_{|\omega|})] \cdots ] ]
\end{aligned}$$

since by the induction hypothesis, each  $T_{n-1}(\tau_i) \in f(T_n(\sigma_i))$ .

Now

$$f(T_n(\sigma)) = f(T_n(c_{\omega, \nu, s}(\sigma' \sigma_1 \cdots \sigma_{|\omega|})))$$

$$\begin{aligned}
&= f(c_{\omega, \nu, s} [{}_n a [{}_{n-1} T_n(\sigma_1) [{}_{n-2} T_n(\sigma_2) [{}_{n-3} \cdots [{}_{n-|\omega|} T_n(\sigma_{|\omega|}) ] \cdots ] ] ] ) \\
&= c_{\omega, \nu, s} [{}_n a [{}_{n-1} f(T_n(\sigma_1)) [{}_{n-2} f(T_n(\sigma_2)) [{}_{n-3} \cdots [{}_{n-|\omega|} f(T_n(\sigma_{|\omega|})) ] \cdots ] ] ]
\end{aligned}$$

So  $T_{n-1}(t) \in f(T_n(\sigma))$  as desired.

If  $\sigma' = F_i$ , then

$$\begin{aligned}
&M(\sigma)(M^{q+1}(\perp)) \\
&= M(F_i)(M^{q+1}(\perp))_{IO(OI)} \left( M(\sigma_1)(M^{q+1}(\perp)), \dots, M(\sigma_{|\omega|})(M^{q+1}(\perp)) \right)
\end{aligned} \tag{1.2}$$

By the corollary,  $t \in M(\sigma)(M^{q+1}(\perp))$  implies that

$$\begin{aligned}
&T_{n-1}(t) \\
&\in \left\{ T_{n-1}(\tau'_{IO(OI)}(\tau_1, \dots, \tau_2)) \right. \\
&\quad \left. | \tau \in M(F_i)(M^{q+1}(\perp)), \tau_i \in M(\sigma_i)(M^{q+1}(\perp)), i = 1, \dots, |\omega| \right\} \\
&= \left\{ T_{n-1}(\tau)_{IO(OI)} T_{n-1}(\tau_1) [{}_{n-2} T_{n-1}(\tau_2) [{}_{n-3} \cdots [{}_{n-|\omega|} T_{n-1}(\tau_{|\omega|}) ] \cdots ] \right. \\
&\quad \left. | \tau \in M(F_i)(M^{q+1}(\perp)), \tau_i \in M(\sigma_i)(M^{q+1}(\perp)), i = 1, \dots, |\omega| \right\} \\
&\in \left\{ f(T_{n-1}(\tau))_{IO(OI)} f(T_{n-1}(\tau_1)) [{}_{n-2} f(T_{n-1}(\tau_2)) [{}_{n-3} \cdots [{}_{n-|\omega|} f(T_{n-1}(\tau_{|\omega|})) ] \cdots ] \right. \\
&\quad \left. | \tau \in M(F_i)(M^{q+1}(\perp)), \tau_i \in M(\sigma_i)(M^{q+1}(\perp)), i = 1, \dots, |\omega| \right\} \\
&\in \left\{ f(T_n(c_{\omega, \nu, \sigma}(\tau \tau_1 \cdots \tau_{|\omega|}))) \right. \\
&\quad \left. | \tau \in M(F_i)(M^{q+1}(\perp)), \tau_i \in M(\sigma_i)(M^{q+1}(\perp)), i = 1, \dots, |\omega| \right\}
\end{aligned}$$

as desired.

If  $\sigma'$  is neither  $a$  nor  $F_i$ , then

$$\begin{aligned}
&M(c_{\omega, \nu, s}(\sigma' \sigma_1 \cdots \sigma_{|\omega|}))(M^{q+1}(\perp)) = \\
&M(\sigma')(M^{q+1}(\perp))_{IO(OI)} \left( M(\sigma_1)(M^{q+1}(\perp)), \dots, M(\sigma_{|\omega|})(M^{q+1}(\perp)) \right)
\end{aligned} \tag{1.3}$$

and by an argument similar to that used for  $F_i$ , we get that

$$T_{n-1}(\sigma) \subseteq f(T_n(\sigma)).$$

This proves the induction step for the depth induction which finishes the induction step for  $q$ .

Now we show that for some  $q$

$$T_{n-1}(t_n) \in f(T_n(\sigma)) \rightarrow t \in M(\sigma)(M^q(\perp))$$

We do this by showing that  $T_n(t) \in f(T_n(\sigma))$  with  $q$  applications of the derivation rule:

$$f(F[_n t_n ] \cdots [_k t_k ]) = \bigcup_{\alpha \in \{\alpha | F \rightarrow \alpha \in P\}} \{ \alpha [_n t_n ] \cdots [_k t_k ] \},$$

then  $t \in M(\sigma)(M^q(\perp))$ . The proof is by induction on  $q$ .

*Basis:* If  $T_{n-1}(t) \in f(T_n(\sigma))$  with no applications of the production rule, then  $\sigma$  contains no nonterminals so  $M(\sigma)(M^q(\perp))$  will never access any of the objects in  $M^q(\perp)$  so certainly  $t \in M(\sigma)(M^0(\perp))$ .

*Induction Step:* Assume that if  $T_{n-1}(t) \in f(T_n(\sigma))$  with  $q$  applications of the production rule, then  $t \in M(\sigma)(M^q(\perp))$ , and let  $T_{n-1}(t) \in f(T_n(\sigma))$  with  $q + 1$  applications of the rule. The proof is by induction on the depth of  $\sigma$ .

*Basis:* Suppose  $d(\sigma) = 1$ , then for any substitution to take place, it must be that  $\sigma = F_i$  and  $T_n(\sigma) = *[_n F_i ]$ . So

$$T_{n-1}(t) \in f(*[_n F_i ]) = \{ f(T_n(\alpha)) \mid F_i \rightarrow T_n(\alpha) \in P \}$$

and  $T_{n-1}(t) \in f(T_n(\alpha))$  for some  $\alpha$  such that  $F_i \rightarrow \alpha \in P$  by fewer than  $q + 1$  applications of the production rule. So by induction,  $t \in M(\alpha)(M^{q+1}(\perp)) = \hat{M}(R_i)(M^q(\perp))$  which contains  $M(\alpha)M^q(\perp)$  so  $t \in M(F_i)(M^{q+1}(\perp))$  as desired.

*Induction Step:* Assume that for  $\sigma$  of depth  $p$ ,  $T_{n-1}(t) \in f(T_n(\sigma))$  by  $q+1$  applications of the rule implies that  $t \in M(\sigma)(M^{q+1}(\perp))$ . Let  $\sigma$  of depth  $p+1$  be such that  $T_{n-1}(t) \in f(T_n(\sigma))$  by  $q+1$  applications of the rule. Now  $\sigma = c_{\omega, \nu, s}(\sigma' \sigma_1 \cdots \sigma_{|\omega|})$  means  $\sigma' = a$  or  $\sigma' = F_i$ , or neither of these. If  $\sigma' = F_i$ , then

$$T_n(\sigma) = c_{\omega, \nu, s} \left[ \begin{array}{c} * \\ \left[ \begin{array}{c} F_i \\ \left[ \begin{array}{c} T_n(\sigma_1) \\ \vdots \\ T_n(\sigma_{|\omega|}) \end{array} \right] \end{array} \right] \end{array} \right] \left[ \begin{array}{c} T_n(\sigma_1) \\ \vdots \\ T_n(\sigma_{|\omega|}) \end{array} \right] \cdots \left[ \begin{array}{c} T_n(\sigma_{|\omega|}) \\ \vdots \\ T_n(\sigma_{|\omega|}) \end{array} \right] \left[ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right] \left[ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right] \end{array} \right]$$

and

$$\begin{aligned} T_{n-1}(t) &= f(T_n(\sigma)) \\ &= f(T_n(F_i))_{IO(OI)} \left[ \begin{array}{c} f(T_n(\sigma_1)) \\ \vdots \\ f(T_n(\sigma_{|\omega|})) \end{array} \right] \left[ \begin{array}{c} f(T_n(\sigma_1)) \\ \vdots \\ f(T_n(\sigma_{|\omega|})) \end{array} \right] \cdots \left[ \begin{array}{c} f(T_n(\sigma_{|\omega|})) \\ \vdots \\ f(T_n(\sigma_{|\omega|})) \end{array} \right] \left[ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right] \left[ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right] \\ &= f(T_n(\alpha))_{IO(OI)} \left[ \begin{array}{c} f(T_n(\sigma_1)) \\ \vdots \\ f(T_n(\sigma_{|\omega|})) \end{array} \right] \left[ \begin{array}{c} f(T_n(\sigma_1)) \\ \vdots \\ f(T_n(\sigma_{|\omega|})) \end{array} \right] \cdots \left[ \begin{array}{c} f(T_n(\sigma_{|\omega|})) \\ \vdots \\ f(T_n(\sigma_{|\omega|})) \end{array} \right] \left[ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right] \left[ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right] \end{aligned}$$

for  $F_i \rightarrow \alpha \in P$  and by the induction hypothesis

$$\begin{aligned} T_{n-1}(t) &\in T_{n-1}(t')_{IO(OI)} \left[ \begin{array}{c} T_{n-1}(t_1) \\ \vdots \\ T_{n-1}(t_{|\omega|}) \end{array} \right] \left[ \begin{array}{c} T_{n-1}(t_1) \\ \vdots \\ T_{n-1}(t_{|\omega|}) \end{array} \right] \cdots \left[ \begin{array}{c} T_{n-1}(t_{|\omega|}) \\ \vdots \\ T_{n-1}(t_{|\omega|}) \end{array} \right] \left[ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right] \left[ \begin{array}{c} \vdots \\ \vdots \\ \vdots \end{array} \right] \\ &= T_{n-1}(t')_{IO(OI)}(t_1, \dots, t_{|\omega|}) \end{aligned}$$

for  $t_i \in M(\sigma_i)(M^{q+1}(\perp))$ ,  $t' \in M(\alpha)(M^{q+1}(\perp))$ . By the induction hypothesis we then conclude that

$$t_{IO(OI)}(t_1, \dots, t_{|\omega|}) \in M(\sigma)(M^{q+1}(\perp))$$

as desired.

If  $\sigma'$  is  $a$  or neither  $a$  nor  $F_i$ , then the first application of the rule occurs in a context identical to that of the case where  $\sigma' = F_i$ . This is a consequence of the

definition of the function,  $T_n$ , which translates from Engelfriet-Schmidt trees to Baldwin-Schoenberger forests. This completes the proof of the theorem.

□

**Theorem 5** *Given a grammar  $G = (D^{n-3}(\Sigma), \mathcal{F}, P, F_1)$ , let  $G_n^n = (\Sigma', \mathcal{F}', P', F_1')_3^1$  be the  $(n + 2)$ -dimensional grammar constructed as indicated in the construction, then*

$$T_{n-1}(M(F_1)(|G_{IO(OI)}|)) = f(T_n(F_1'))$$

*Proof:* Since  $|G_{IO(OI)}| = \bigcup_{i=0}^{\infty} M^i(\perp)$ , we need only apply the previous theorem.

□

#### 1.4.4 The string languages are equivalent

In this section we show that the string languages defined by the *YIELDS* of the Engelfriet-Schmidt languages are the same as the string languages defined by the equivalent Baldwin-Schoenberger language. A string  $\omega \in \Sigma^*$  has Baldwin-Schoenberger equivalent  $\omega_1[_1 \omega_2[_1 \omega_3[_1 \dots [_1 \omega_{|\omega|}[_1 x_1^\lambda ] ] \dots ] ] ]$ . First, we will show that the Baldwin-Schoenberger equivalent of a string,  $\omega$ , which is the yield,  $\mathcal{Y}$ , of a tree,  $t$ , in  $T_\Sigma$  is the same as the frontier,  $f$ , of the Baldwin-Schoenberger equivalent of  $t$ . Next, we show that the frontier of the Baldwin-Schoenberger equivalent of any tree  $t$  in the Engelfriet-Schmidt hierarchy is the same as the Baldwin-Schoenberger equivalent of the *YIELD* of the tree. Finally, we show that the strings formed, from Engelfriet-Schmidt trees, by taking successive yields until a string is the result, are equivalent to the 1-dimensional objects obtained by taking successive frontiers of the equivalent Baldwin-Schoenberger forests.

**Lemma 5** *Let  $\sigma \in T_\Sigma$ , then the 1-dimensional Baldwin-Schoenberger forest equivalent of  $\mathcal{Y}(\sigma)$  is  $T_2(\sigma)$ .*

*Proof:* The proof is by induction on the depth of  $\sigma$ .

*Basis:* If  $d(\sigma) = 1$ , then  $\sigma \in \Sigma_{\langle \lambda, s \rangle}$ , and  $\mathcal{Y}(\sigma) = \sigma$ . The equivalent Baldwin-Schoenberger forest is  $\sigma[{}_1 x_1^\lambda]$ , and  $f(T_2(\sigma\pi_1^\lambda)) = f(\sigma[{}_1 * [{}_2 x_1^\lambda] [{}_1 x_1^\lambda] ] ) = \sigma[{}_1 x_1^\lambda]$ . Thus the lemma is true for  $\sigma$  of depth 1.

*Induction Step:* Assume the theorem true for  $\sigma$  such that  $d(\sigma) \leq p$  and consider  $\sigma$  such that  $d(\sigma) = p + 1$ . Then  $\sigma = a(t_1 \cdots t_{|\omega|})$  where  $a \in \Sigma_{\langle \omega, s \rangle}$ , and for  $i = 1, \dots, |\omega|$ ,  $t_i \in T_\Sigma$  and has depth less than  $p + 1$ .

$\mathcal{Y}(\sigma) = \sigma_1 \cdots \sigma_{|\omega|}$ , where for  $i = 1, \dots, |\omega|$ ,  $\sigma_i = \mathcal{Y}(t_i)$ . For those  $t_i$  of depth one,  $f(T_2(\sigma_i))$  will be  $\sigma_i$ , and for those of depth greater than one, we will get the string equivalent in the Baldwin-Schoenberger hierarchy, that is,

$$\sigma_{i,1}[{}_1 \sigma_{i,2}[{}_1 \cdots [{}_1 \sigma_{i,|\sigma_i|}[{}_1 x_1^\lambda] ] \cdots ] ] .$$

This gives

$$\begin{aligned} f(T_2(\sigma)) &= f(a[{}_2 T_2(t_1)[{}_1 T_2(t_2)[{}_1 \cdots [{}_1 T_2(t_{|\omega|})[{}_1 * [{}_2 x_1^\lambda] [{}_1 x_1^\lambda] ] \cdots ] ] ) \\ &= f(T_2(t_1)[{}_1 T_2(t_2)[{}_1 \cdots [{}_1 T_2(t_{|\omega|})[{}_1 * [{}_2 x_1^\lambda] [{}_1 x_1^\lambda] ] \cdots ] ] ) . \end{aligned}$$

For those  $\sigma \in \Sigma_{\langle \lambda, s \rangle}$  the frontier appends  $\sigma_i$  to the string formed by

$$f(T_2(t_{i+1})[{}_1 \cdots [{}_1 T_2(t_{|\omega|})[{}_1 * [{}_2 x_1^\lambda] [{}_1 x_1^\lambda] ] \cdots ] ) .$$

While for those whose depth is greater than one, the frontier replaces the  $x_1^\lambda$  at the frontier with

$$f(T_2(t_{i+1})[{}_1 \cdots [{}_1 T_2(t_{|\omega|})[{}_1 * [{}_2 x_1^\lambda] [{}_1 x_1^\lambda] ] \cdots ] ) .$$

The result is

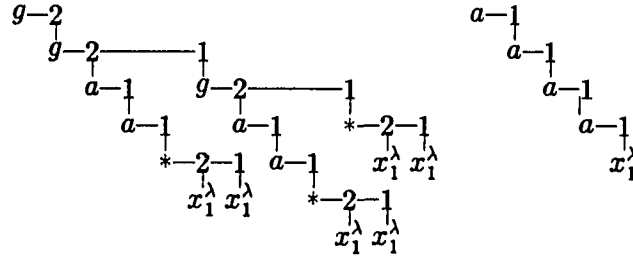
$$f(T_2(\sigma)) = \sigma_{1,1} [ \cdots \sigma_{1,|\sigma_1|} [ \sigma_{2,i} \cdots \sigma_{2,|\sigma_2|} [ \cdots \sigma_{|\omega|,1} [ \cdots \sigma_{|\omega|,|\sigma_{|\omega|}|} [ x_1^\lambda ] \cdots ] \cdots ] \cdots ] \cdots ]$$

which is the string equivalent of  $\mathcal{Y}(\sigma)$  as desired.

□

The next example is of a 2-dimensional Baldwin-Schoenberger forest, its frontier, and its Engelfriet-Schmidt equivalent along with it's yield.

**Example 13** The tree  $g(g(a a) g(a a)) \in T_\Sigma$  has Baldwin-Schoenberger equivalent and frontier:



while  $\mathcal{Y}(g(g(a a) g(a a))) = aaaa$  which has the same Baldwin-Schoenberger string equivalent.

□

In the following lemma we use the expression, *identical for the purpose of taking frontiers*. By this we mean that the forests are identical except that one is surrounded by  $n$  brackets, making the  $n - 1$  frontiers the same. For example, consider the trees  $*[{}_n *[_{n-1} *[_{n-2} x_{n-3}^\lambda ][_{n-3} x_{n-3}^\rho ] ]$ , and  $*[_{n-1} *[_{n-2} x_{n-3}^\lambda ][_{n-3} x_{n-3}^\rho ] ]$ . The  $(n - 1)$  frontiers of these forests are the same. An even more important example is that of the trees  $*[_n x_{n-1}^\lambda ][_{n-1} x_{n-2}^{(n-2)(n-3)} ]$  and  $x_{n-2}^{(n-2)(n-3)}$ . The  $(n - 1)$  frontiers of these are

also the same. In the following lemma, we show that two forests are the same for the purposes of taking frontiers because we are ultimately interested in the frontiers of these objects.

**Lemma 6** *Let  $\sigma \in T_{D^{n-2}(\Sigma)}$ ,  $n > 2$ , then  $T_{n-1}(YIELD(\sigma))$  is the same as  $f(T_n(\sigma))$  for the purposes of taking frontiers.*

*Proof:* The proof is by induction on the depth of  $\sigma$ .

*Basis:* If  $d(\sigma) = 1$ , then  $\sigma \in D^{n-2}(\Sigma)_{\langle \lambda, s \rangle}$ , so  $\sigma \in \Sigma_{\langle \omega, s \rangle}$ , or  $\sigma = \pi_i^\omega \in D^k(\Sigma)_{\langle \lambda, s \rangle}$  for  $k \leq n - 2$ .

If  $\sigma = \pi_i^\omega \in D^{n-2}(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}$  and  $\omega \neq \lambda$ , then

$$\begin{aligned} T_{n-1}(YIELD(\sigma)) &= T_{n-1}(x_{i, \omega_i}) \\ &= x_{n-1}^{path(i, n-2)}. \end{aligned}$$

And

$$\begin{aligned} f(T_n(\sigma)) &= f(x_{n-1}^{path(i, n-2)}) \\ &= x_{n-1}^{path(i, n-2)}. \end{aligned}$$

If  $\sigma = \pi_i^\omega \in D^{n-3}(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}$ , then

$$\begin{aligned} T_{n-1}(YIELD(\sigma)) &= T_{n-1}(x_{i, \omega_i}) \\ &= x_{n-2}^{path(i, n-3)}. \end{aligned}$$

And

$$\begin{aligned} f(T_n(\sigma)) &= f(x_{n-2}^{path(i, n-3)}) \\ &= * \left[ {}_{n-1} x_{n-2}^\lambda \right] {}_{n-2} x_{n-2}^{path(i, n-3)} \end{aligned}$$

which is the same for the purpose of taking frontiers.

If  $\sigma = \pi_i^\omega \in D^k(\Sigma)_{\langle \lambda, \langle \omega, \omega_i \rangle \rangle}$  and  $k < n - 3$ , or  $\sigma = \pi_1^\lambda$ , then

$$\begin{aligned} T_{n-1}(YIELD(\sigma)) &= T_{n-1}(\pi_i^\omega) \\ &= * [_{n-2} \cdots [_{k+2} x_{k+1}^\lambda ]_{k+1} x_{k+1}^{path(i,k)} \cdots ] ], \end{aligned}$$

while

$$\begin{aligned} f(T_n(\sigma)) &= f(* [_{n-1} * [_{n-2} \cdots [_{k+2} x_{k+1}^\lambda ]_{k+1} x_{k+1}^{path(i,k)} ] \cdots ] ]) \\ &= * [_{n-1} * [_{n-2} \cdots [_{k+2} x_{k+1}^\lambda ]_{k+1} x_{k+1}^{path(i,k)} ] \cdots ] ] \end{aligned}$$

which again is the same for purposes of taking frontiers.

Finally, if  $\sigma \in D^{n-2}(\Sigma)_{\langle \lambda, \langle \lambda, s \rangle \rangle}$ , then  $T_{n-1}(YIELD(\sigma)) = \sigma$ , and  $f(T_n(\sigma)) = f(\sigma) = \sigma$ .

*Induction Step:* If  $d(\sigma) = p + 1$ , then  $\sigma = c_{\omega, \nu, s}(t \ t_1 \cdots t_{|\omega|})$  and if  $t$  has depth 1, then

$$\begin{aligned} T_{n-1}(YIELD(\sigma)) &= T_{n-1}(t(YIELD(t_1) \cdots YIELD(t_{|\omega|}))) \\ &= t [_{n-1} T_{n-1}(YIELD(t_1)) [_{n-2} \cdots [_{n-|\omega|} T_{n-1}(YIELD(t_{|\omega|})) [_{n-(|\omega|-1)} T_{n-1}(\pi_1^\lambda) ] ] \cdots ] ] \\ &= t [_{n-1} f(T_n(t_1)) [_{n-2} f(T_n(t_2)) [_{n-3} \cdots [_{n-|\omega|} f(T_n(t_{|\omega|})) [_{n-(|\omega|-1)} f(T_n(\pi_1^\lambda)) ] ] \cdots ] ] ] \end{aligned}$$

and

$$\begin{aligned} f(T_n(\sigma)) &= f(T_n(t(t_1 \cdots t_{|\omega|} \pi_1^\lambda))) \\ &= f(c_{\omega, \nu, s} [_{n-1} t [_{n-1} T_n(t_1) [_{n-2} \cdots [_{n-|\omega|} T_n(t_{|\omega|}) [_{n-(|\omega|-1)} T_n(\pi_1^\lambda) ] ] \cdots ] ] ] \\ &= t [_{n-1} f(T_n(t_1)) [_{n-2} f(T_n(t_2)) [_{n-3} \cdots [_{n-|\omega|} f(T_n(t_{|\omega|})) [_{n-(|\omega|-1)} f(T_n(\pi_1^\lambda)) ] ] \cdots ] ] ] \end{aligned}$$

which is what we want.

If  $d(t) > 1$ , then

$$T_{n-1}(YIELD(\sigma)) = T_{n-1}(YIELD(t) \Leftarrow (YIELD(t_1), \dots, YIELD(t_{|\omega|}))).$$

Since the substitutions are for  $x_i, \omega_i$ , this has Baldwin-Schoenberger equivalent

$$\begin{aligned} & T_{n-1}(YIELD(t)) \\ \Leftarrow & T_{n-1}(YIELD(t_1))|_{n-2} \cdots |_{n-|\omega|} T_{n-1}(YIELD(t_{|\omega|}))|_{n-(|\omega|-1)} T_{n-1}(\pi_1^\lambda) \cdots \end{aligned}$$

which is

$$\begin{aligned} & f(T_n(\sigma)) \\ \Leftarrow & f(T_n(t_1))|_{n-2} f(T_n(t_2))|_{n-3} \cdots |_{n-|\omega|} f(T_n(t_{|\omega|}))|_{n-(|\omega|-1)} f(T_n(\pi_1^\lambda)) \cdots \end{aligned}$$

which in turn is  $f(T_n(\sigma))$  as in the argument for  $d(t) = 1$ .

□

Finally, we show that the string obtained from a tree in the Engelfriet-Schmidt hierarchy by taking successive *YIELD*s, is equivalent to the string in the Baldwin-Schoenberger hierarchy obtained by taking successive frontiers of the equivalent Baldwin-Schoenberger forest.

**Theorem 6** *Let  $\sigma \in T_{D^{n-2}(\Sigma)}$ , then  $YIELD^{n-1}(\sigma) = f^{n-1}(T_n(\sigma))$ .*

*Proof:* The proof is by induction on  $n$ .

*Basis:* If  $n=2$ , then by the first lemma  $YIELD(\sigma) = f(T_2(\sigma))$ .

*Induction Step:* Assume that  $f^{n-1}(T_n(\sigma)) = YIELD^{n-1}(\sigma)$  for all  $n \geq 2$  and consider

$$f^n(T_{n+1}(\sigma)).$$

$$\begin{aligned} f^n(T_{n+1}(\sigma)) &= f^{n-1}(f(T_{n+1}(\sigma))) \\ &= f^{n-1}(T_{n-1}(YIELD(\sigma))) \\ &= YIELD^{n-1}(YIELD(\sigma)) \\ &= YIELD^n(\sigma) \end{aligned}$$

as desired.

□

This last theorem is important in that it establishes that the Engelfriet-Schmidt tree languages are strictly contained in the Baldwin-Schoenberger forest languages.

**Theorem 7** *The Engelfriet-Schmidt tree languages are strictly contained in the Baldwin-Schoenberger forest languages.*

*Proof:* Consider the language defined by the Baldwin-Schoenberger grammar:  $G_3^1 = (\{*, a\}, \mathcal{F}, P, S)_3^1$ , where  $\mathcal{F}_1 = \{G\}$ ,  $\mathcal{F}_2 = \{F\}$ ,  $\mathcal{F}_3 = \{S\}$ , and

$$P = \{S \rightarrow *[_3 F ][_1 F ], F \rightarrow *[_2 G ], G \rightarrow a[_1 G ], G \rightarrow x_1^\lambda\}.$$

The language defined by  $G_3^1$  is a set of forests, there is no Engelfriet-Schmidt equivalent grammar since the Engelfriet-Schmidt grammars do not define forests.

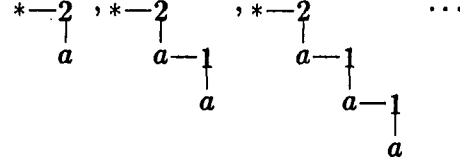
□

The Baldwin-Schoenberger grammars are even more powerful than this example indicates. The language  $G_3^3 = (\{*, a\}, \mathcal{F}, P, S)_3^3$  where  $\mathcal{F}_1 = \{G\}$ ,  $\mathcal{F}_2 = \{F\}$ ,  $\mathcal{F}_3 = \{S\}$  and

$$P = \{S \rightarrow *[_3 F ], F \rightarrow *[_2 G ], G \rightarrow a[_1 G ], G \rightarrow a\}$$

has a special property as indicated by the next theorem.

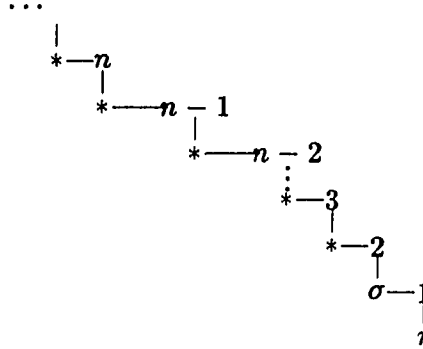
**Theorem 8** *The language*



*cannot be produced by a grammar which does not have forest nonterminals.*

*Proof:* Suppose that there is a grammar  $G_n^n = (\Sigma, \mathcal{F}, P, S)_n^n$ , such that  $f^{n-2}(S) = \{ *[_2 a ], *[_2 a[_1 a ] ], *[_2 a[_1 a[_1 a ] ] ], \dots \}$  such that  $\mathcal{F} = \mathcal{F}_n$ .

To produce trees of the form  $*[_2 a[_1 a[_1 a[_1 \dots [_1 a ] ] ] ] ]$  the grammar must produce a tree of the form:



This is because until the  $(n-1)^{st}$  frontier is taken, no subtree of the form  $*[_1 \dots ]$  can be copied. So this tree must be present in the correct position at the time the original tree is generated in order to produce the trees  $*[_2 a[_1 \dots ] ]$ .

So let us look at the  $*[_2 \sigma[_1 \tau ] ]$ . If  $\sigma$  is a nonterminal then  $f^{n-2}(\sigma)$  must be a single nonterminal ( $a$ ), otherwise the resulting tree would not be in our language. This means that  $f^{n-2}(\tau)$  must be  $a[_1 a[_1 a[_1 \dots [_1 a ] ] ] ]$ . From this we see that  $\tau$

cannot be a nonterminal since forests in  $H_n^n$  cannot produce objects in  $H_1^1$  until the  $(n-1)^{\text{st}}$  frontier.  $\tau$  must be  $\alpha[ \tau_1 ]$  where  $f^{n-2}(\alpha) = a$ . We can apply this argument to  $\tau_1$  and find that the grammar must have a production of infinite length which is not possible.

□

## 2 THE RELATIONSHIP BETWEEN THE *IO* AND *OI* N-DIMENSIONAL LANGUAGE HIERARCHIES

In this chapter we discuss the relationship between the *OI*  $n$ -dimensional language hierarchy and the *IO*  $n$ -dimensional language hierarchy. Fischer [27] introduced the *IO* and *OI* macro languages. He showed the existence of an *OI* macro grammar for a language which has no *IO* macro grammar, and an *IO* macro grammar for a language which has no *OI* macro grammar. Rounds [51,52,53] showed how these macro grammars can be viewed as tree grammars for tree languages whose yields are the string languages of Fischer. Engelfriet-Schmidt [24,25] extended this to the hierarchy discussed in the previous chapter while Baldwin [12] extended it to the *IO*  $n$ -dimensional hierarchy also discussed in the previous chapter. None of these authors was able to throw any more light on the relationship between *IO* and *OI* languages than Fischer.

It was necessary to develop the Baldwin-Schoenberger hierarchy in the *OI* direction before this situation could be changed. Section 1 contains an example of an *IO* language at dimension 3 which is not *OI* at dimension 3, but is *OI* at dimension 4. In section 2 we show how the *IO*  $n$ -dimensional hierarchy may be viewed as a subset of the *OI*  $n$ -dimensional hierarchy. Finally, in section 3 we show that the string languages defined by the *OI* 3-dimensional languages are the same as the *OI* macro languages.

### 2.1 An *IO* Macro Language Which is Not *OI*

In this section we give an example, from Fischer [27], of an *OI* macro language which is not *IO*. We then give the 3-dimensional *IO* language whose 1 dimensional frontier is the same language. Next we show how to convert this 3-dimensional grammar into a 4-dimensional *OI* grammar whose 1-dimensional frontier is the language.

Our example language is  $L = \{b^m(a b^m)^{2^m-1} \mid m \geq 1\}$ . The *IO* macro grammar for this language is  $\tilde{G} = (\Sigma, \mathcal{F}, \rho, S, P)$  where:

$$\Sigma = \{a, b\},$$

$$\mathcal{F} = \{S, F, G\},$$

$$\mathcal{V} = \{x\},$$

$$\rho = \{(S, 0), (F, 1), (G, 1)\},$$

$$P = \{S \rightarrow F(b), F(x) \rightarrow G(F(x b)) \mid G(x), G(x) \rightarrow x a x\}.$$

An *IO* derivation from this grammar first produces  $G(G(\cdots G(b^m) \cdots))$ . This expression contains exactly  $m$   $G$ s. Each  $G$  then doubles  $b^m a b^m$  with the final result being  $b^m(ab^m)^{2^m-1}$ .

The Baldwin-Schoenberger *IO* 3-dimensional grammar for this language is  $G_3^3 = (\Sigma, \mathcal{F}, P, S)_3^3$  where

$$\Sigma = \{a, b, *\},$$

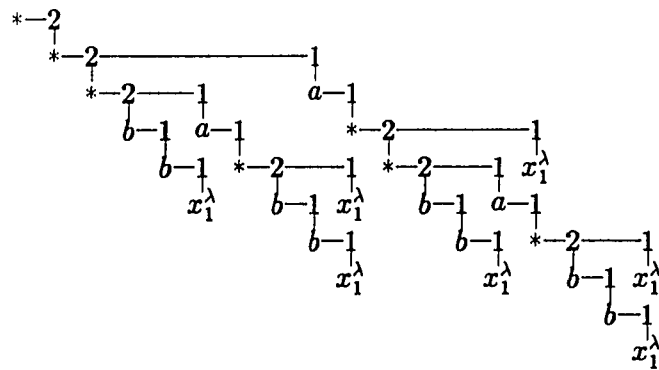
$$\mathcal{F} = \mathcal{F}_2 \cup \mathcal{F}_3, \text{ where } \mathcal{F}_2 = \{F, G\}, \text{ and } \mathcal{F}_3 = \{S\},$$

$P$  contains the following productions:

$$F \rightarrow * - \underset{\substack{| \\ G}}{3} - \underset{\substack{| \\ x_2^\lambda}}{2}$$

$$G \rightarrow \begin{array}{c} * - 2 \\ \quad \downarrow \lambda \\ \quad x_2 - 1 \\ \quad \quad \downarrow \\ \quad \quad a - 1 \\ \quad \quad \quad \downarrow \lambda \\ \quad \quad \quad x_2 - 1 \\ \quad \quad \quad \quad \downarrow \lambda \\ \quad \quad \quad \quad x_1 \end{array}$$

## The forest



is in this language. It has a 1-dimensional frontier of

$$b[ \mid b[ \mid a[ \mid b[ \mid b[ \mid a[ \mid b[ \mid b[ \mid a[ \mid b[ \mid b[ \mid x_1^\lambda ] ] ] ] ] ] ] ] ] ] ].$$

The grammar,  $\tilde{G}_4^4 = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_4^3$ , where

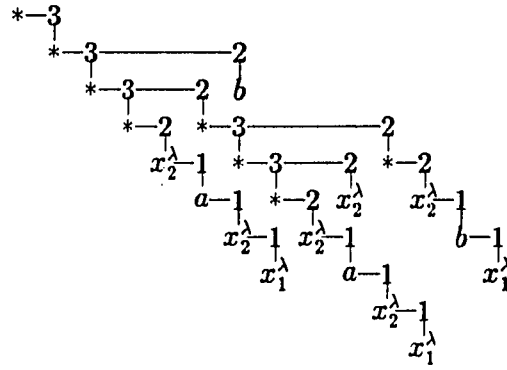
$$\tilde{\Sigma} = \{a, b, *\}$$

$$\tilde{\mathcal{F}} = \tilde{\mathcal{F}}_2 \cup \tilde{\mathcal{F}}_3, \text{ where } \tilde{\mathcal{F}}_2 = \{F, G\}, \text{ and } \tilde{\mathcal{F}}_3 = \{S, \tilde{S}\},$$

$\tilde{P} = \{\tilde{S} \rightarrow S\}$  along with the productions from the  $G_3^3$ .

This grammar is a 4-dimensional grammar whose 3-dimensional frontier is the language generated by  $G_3^3$ .

The tree:



has the tree given above as its 2-dimensional frontier.

This inclusion is possible because *IO* substitution requires that all substitutions be made from the same forest in the substitution set. The conversion of the *IO* grammars of dimension  $n$  to *OI* grammars of dimension  $(n + 1)$  is as follows:

Let  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  be an *IO* grammar, then  $\tilde{G}_{n+1}^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})$  where

$$\tilde{\Sigma} = \Sigma \cup \{*\},$$

$$\tilde{\mathcal{F}} = \mathcal{F} \cup \{\tilde{S}\}, \tilde{S} \in \tilde{\mathcal{F}}_k,$$

$$\tilde{P} = \{\tilde{S} \rightarrow S\} \cup P.$$

This grammar is of level  $k$ , dimension  $n + 1$  since  $H_n^k \subset H_{n+1}^k$  (see Section 1.3.1.) In the next section we will prove that  $(n - 1)$ -dimensional frontier of the *OI* language derived from  $\tilde{G}_{n+1}^k$  is the same as the *IO* language derived from  $G_n^k$ .

## 2.2 $IO$ Dimension $n$ Is a Subset of $OI$ Dimension $(n + 1)$

In this section we prove that for each  $n$ -dimensional  $IO$  language,  $L_1$ , there is an  $(n + 1)$ -dimensional language,  $L_2$ , such that the  $n$ -dimensional frontier of  $L_2$  is  $L_1$ . This is a direct result of the fact that the trees obtained from an  $IO$  derivation are the same as those obtained from first substituting for all of the nonterminals in a tree and then taking the frontier of the tree. This is the way that Baldwin [12] did all of his derivations.

### 2.2.1 The Baldwin derivation relation and its relation to the Baldwin-Schoenberger derivation relation

**Definition 22** The  $IO$  derivation relation of Baldwin: *The Baldwin  $IO$  derivation relation over a grammar  $G_n^k$ , denoted  $B \Rightarrow$  is defined as:  $H_n^k B \Rightarrow H_n^k$  if  $n \geq k > 0$  such that, if  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  and  $\alpha F \beta \in H_n^k$  where  $F \in \mathcal{F}_j$  and  $F \rightarrow \gamma \in P$ , then  $\alpha F \beta B \Rightarrow \alpha \gamma \beta$ .*

*The  $n$ -dimensional forest languages defined in this manner are the Regular  $n$ -dimensional languages.*

*The  $(n - 1)$ -dimensional forest language defined by  $G_n^k$  is  $\{f_{n-1}(\sigma) \mid S B \Rightarrow \sigma\}$ .*

□

The languages defined in this manner are the same as the  $IO$   $(n - 1)$ -dimensional languages of chapter 1. In other words, if we generate the trees completely before taking the frontier we get the same language as that from the  $IO$  derivation. The following theorem affirms this fact.

**Theorem 9** Let  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  be an  $n$ -dimensional forest grammar, then

$$\left\{ f_{n-1}(t) \mid S \xrightarrow{\bullet} t \right\} = f_{n-1,IO}(S).$$

□

Before proving this theorem we need the following lemma.

**Lemma 7** Let  $t \in H_n^k(\Sigma, \mathcal{F})$ , then

$$\left\{ f_{n-1}(t) \mid \sigma \xrightarrow{\bullet} t \right\} = f_{n-1,IO}(\sigma).$$

*Proof:* The proof is by induction on the length of the derivation. First we show that

$$\left\{ f_{n-1}(t) \mid \sigma \xrightarrow{\bullet} t \right\} \subseteq f_{n-1,IO}(\sigma).$$

*Basis:* If  $\sigma \xrightarrow{\bullet} t$  by 0 Baldwin derivation steps, then they certainly are the same.

*Induction Step:* Assume that if  $\sigma \xrightarrow{\bullet} t$  by  $p$  derivation steps, then  $f_{n-1}(t) \in f_{n-1,IO}(\sigma)$ .

And assume that  $\sigma \xrightarrow{\bullet} t$  by  $p+1$  derivation steps.

If  $\sigma = F[_j \sigma_j] \cdots [_k \sigma_k]$  and  $F \rightarrow \alpha = a[_n \sigma_n] \cdots [_{j+1} \sigma_{j+1}]$  is the production used, then

$$\begin{aligned} \sigma & \xrightarrow{\bullet} \alpha[_j \sigma_j] \cdots [_k \sigma_k] \\ & \xrightarrow{p} a[_n t_n] \cdots [_k t_k] \end{aligned}$$

where  $\alpha \xrightarrow{\bullet} a[_n t_n] \cdots [_{j+1} t_{j+1}]$  and  $\sigma_i \xrightarrow{\bullet} t_i, i = k, \dots, j$  by less than  $p+1$  steps.

So

$$f_{n-1}(t) = \begin{cases} f_{n-1}(t_n) \Leftarrow f_{n-1}(t_{n-1})[_{n-2} f_{n-1}(t_{n-2})] \cdots [_k f_{n-1}(t_k)] & t_n \neq \emptyset \\ a[_{n-1} f_{n-1}(t_{n-1})] \cdots [_k f_{n-1}(t_k)] & t_n = \emptyset \end{cases}$$

with  $f_{n-1}(t_i) \in f_{n-1,IO}(\sigma_n)$ . So  $f_{n-1}(t_n) \Leftarrow f_{n-1}(t_{n-1}) \in f_{n-1}(\sigma'_n) \Leftarrow_{IO} f_{n-1}(\sigma'_{n-1})$  by the definition of  $IO$  substitution. Therefore,  $f_{n-1}(t) \in f_{n-1,IO}(\sigma)$ .

If  $\sigma = a[_n \sigma_n] \cdots [_k \sigma_k]$ , then we can apply the above argument to each of the  $\sigma_i$  to get the desired result.

Next we show that

$$f_{n-1,IO}(\sigma) \subseteq \left\{ f_{n-1}(t) \mid \sigma \xrightarrow{\bullet} t \right\}.$$

*Basis:* Suppose  $t \in f_{n-1,IO}(\sigma)$  after 0 applications of the production rule

$$f_{n-1,IO}(F[_j \sigma_j] \cdots [_k \sigma_k]) = \bigcup_{\alpha \in \{ \alpha \mid F \rightarrow \alpha \in P \}} \left\{ f_{n-1,IO}(\alpha[_j \sigma_j] \cdots [_k \sigma_k]) \right\}.$$

then clearly  $t \in \left\{ f_{n-1}(t) \mid \sigma \xrightarrow{\bullet} t \right\}$  as desired.

*Induction Step:* Suppose  $t \in f_{n-1,IO}(\sigma)$  after  $p$  applications of the production rule implies that  $t \in \left\{ f_{n-1}(t) \mid \sigma \xrightarrow{\bullet} t \right\}$ . Let  $t \in f_{n-1,IO}(\sigma)$  after  $p+1$  applications of the production rule.

If  $\sigma = F[_j \sigma_j] \cdots [_k \sigma_k]$ , then

$$f_{n-1,IO}(\sigma) = \left\{ f_{n-1,IO}(a[_n \sigma_n] \cdots [_{j+1} \sigma_{j+1}] [_j \sigma_j] \cdots [_k \sigma_k]) \mid F \rightarrow a[_n \sigma_n] \cdots [_{j+1} \sigma_{j+1}] \in P \right\},$$

and

$$t \in f_{n-1,IO}(a[_n \sigma_n] \cdots [_{j+1} \sigma_{j+1}] [_j \sigma_j] \cdots [_k \sigma_k])$$

for some specific  $a[_n \sigma_n] \cdots [_{j+1} \sigma_{j+1}]$  such that  $F \rightarrow a[_n \sigma_n] \cdots [_{j+1} \sigma_{j+1}] \in P$ . So

$$t \in \begin{cases} f_{n-1}(\sigma_n) \Leftarrow_{IO} f_{n-1}(\sigma_{n-1})[_{n-2} f_{n-1}(\sigma_{n-2})] \cdots [_k f_{n-1}(\sigma_k)] & \sigma_n \neq \emptyset \\ a[_{n-1} f_{n-1}(\sigma_{n-1})] \cdots [_k f_{n-1}(\sigma_k)] & \sigma_n = \emptyset \end{cases}$$

where each  $f_{n-1}(\sigma_i) \subseteq \left\{ f_{n-1}(t_i) \mid \sigma_i \xrightarrow{\cdot} t_i \right\}$ ,  $i = k, \dots, n$ .

This means that

$$t = a[_{n-1} \ t_{n-1} ] \cdots [ _k \ t_k ]$$

where, in the case of line 2, each

$$t_i \in f_{n-1}(\sigma_i) \subseteq \left\{ f_{n-1}(t_i) \mid \sigma_i \xrightarrow{\cdot} t_i \right\},$$

for  $i = k, \dots, n-1$  so

$$t \in \left\{ f_{n-1}(a[_{n-1} \ t_{n-1} ] \cdots [ _k \ t_k ]) \mid \sigma_i \xrightarrow{\cdot} t_i \right\}.$$

For line 1, each  $t_i \in f_{n-1}(\sigma_i) \subseteq \left\{ f_{n-1}(t_i) \mid \sigma_i \xrightarrow{\cdot} t_i \right\}$ ,  $i = k, \dots, n-2$  and

$$\begin{aligned} a[_{n-1} \ t_{n-1} ] &\in f_{n-1,IO}(\sigma_n) \stackrel{\text{def}}{=} f_{n-1,IO}(\sigma_{n-1}) \\ &= \{ t_n \Leftarrow t_{n-1} \mid t_n \in f_{n-1,IO}(\sigma_n) \wedge t_{n-1} \in f_{n-1,IO}(\sigma_{n-1}) \} \\ &\subseteq \left\{ t_n \Leftarrow t_{n-1} \mid t_n \in \left\{ f_{n-1,IO}(t_n) \mid \sigma_n \xrightarrow{\cdot} t_n \right\} \right. \\ &\quad \left. \wedge t_{n-1} \in \left\{ f_{n-1,IO}(t_{n-1}) \mid \sigma_{n-1} \xrightarrow{\cdot} t_{n-1} \right\} \right\} \\ &= \left\{ f_{n-1}(t_n) \Leftarrow f_{n-1}(t_{n-1}) \mid \sigma_n \xrightarrow{\cdot} t_n \wedge \sigma_{n-1} \xrightarrow{\cdot} t_{n-1} \right\}. \end{aligned}$$

So  $t \in \left\{ f_{n-1}(a[ _n \ t_n ] \cdots [ _k \ t_k ]) \mid \sigma \xrightarrow{\cdot} a[ _n \ t_n ] \cdots [ _k \ t_k ] \right\}$  as desired.

Now if  $\sigma = a[ _n \ \sigma_n ] \cdots [ _k \ \sigma_k ]$ , then we can apply the above argument to each of the  $\sigma_i$  to get the desired result.

□

This theorem shows that the Baldwin-Schoenberger hierarchy does indeed contain the languages described by Baldwin. The next theorem states that Baldwin's hierarchy is completely contained in the  $OI$  hierarchy.

**Theorem 10** *Let  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , there is a grammar,  $\tilde{G}_{n+1}^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_{n+1}^k$ , such that the  $IO$  language,  $f_{IO}(S)$ , is equal to the frontier of the  $OI$  language,  $f_{OI}(\tilde{S})$ .*

*Proof:* First, construct  $G_{n+1}^k$  as indicated in the previous section. Next, note that  $f_{OI}(S)$  produces the same set as the Baldwin derivation, since no substitutions take place during the derivation. Now apply the previous theorem.

□.

### 2.2.2 An $OI$ language which has no $IO$ grammar at any dimension

In this section we demonstrate the existence of an  $OI$  language which has no  $IO$  grammar. This language is the language of Section 1.3.6,  $L = \{(b^m a b^m)^{2^m} \mid m \geq 1\}$ . This language is not the  $(n - 1)$ st frontier of any  $IO$   $n$ -dimensional grammar of the Baldwin type or Baldwin-Schoenberger type. We say of neither type because in the last section we showed that the  $IO$  Baldwin-Schoenberger languages and the Baldwin languages are the same. We use the Baldwin definition in the rest of this section.

Baldwin [12] showed the following facts:

1.  $L$  is the Baldwin language defined by the grammar  $G_1^1$  if and only if there is a regular grammar  $G$  such that if  $\omega$  is in the language of  $G_1^1$  then  $\omega$  with any brackets and  $x_1^1$ s removed is in the language derived from  $G$ .
2.  $L$  is the 1-dimensional frontier of the Baldwin language derived from the grammar  $G_2^k$  if and only if (subject to the constraints of 1 above) there is a context-free grammar for  $L$ .
3.  $L$  is the 1-dimensional frontier of the Baldwin language derived from the grammar  $G_3^k$  if and only if (subject to the constraints of 1 above) there is an  $IO$

macro grammar for  $L$ .

Fischer [27] proved that the language  $L$  does not have an  $IO$  macro grammar. Therefore,  $L$  does not have an  $IO$   $n$ -dimensional grammar for  $n \leq 3$ . In Section 1.3.6 we gave the  $OI$  3-dimensional grammar for the language whose 1-dimensional frontier is  $L$ . Therefore,  $L$  does have an  $OI$   $n$ -dimensional grammar. We will then have the fact that the  $IO$  hierarchy is a proper subset of the  $OI$  hierarchy.

The proof uses the concept of completed grammar. Before giving the proof, we first define completed grammars along with the concepts necessary for the understanding of their construction. The construction of a completed grammar for a Baldwin language requires the refinement of the definition of paths and the construction of a normal form grammar.

**Definition 23** A grammar,  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , is said to be in normal form if its productions are of the form  $F \rightarrow \sigma$  where  $\sigma = \Sigma$ , or  $\Sigma[_n G]$  for some  $G \in \mathcal{F}_{n-1}$  whenever  $F \in \mathcal{F}_n$  and  $\sigma = H$ , or  $H[_m G]$  or  $x_m^\rho$  for some  $H \in \mathcal{F}_{m+1}$ ,  $G \in \mathcal{F}_{m-1}$ , and  $x_m^\rho \in X_m$  whenever  $F \in \mathcal{F}_m$  for all  $n > m \geq 1$ .

□

**Definition 24** (Baldwin [12]) A grammar,  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , is said to be a completed grammar if, during any extract operation on any element of the Baldwin language defined by  $G_n^k$ , a piece of the structure derived from any nonterminal is copied, then all of the structure derived from that nonterminal is copied.

□

If a forest is in  $H_n^k$ , then it is of the form  $a[_n t_n] \cdots [_k t_k]$  and the legal paths into this forest would be of the form  $\lambda, (n-1)\rho_{n-1}, (n-2)\rho_{n-2}, \dots, (k+1)\rho_{k+1}$ , or  $k\rho_k$

where  $\lambda$  chooses the tree  $a[n \ t_n]$  and  $i\rho_i$  chooses the subtree at the end of the path  $\rho_i$  in the forest  $t_i$ . Then  $\rho_i$  must be of the form  $\lambda, (n-1)\rho_{i,n-1}, (n-2)\rho_{i,n-2}, \dots, (i)\rho_{i,i}$ , or  $(i-1)\rho_{i,i-1}$  where  $\lambda$  chooses the tree  $a[n \ t'_n]$  from  $t_i$  and  $j\rho_{i,j}$  chooses the subtree at the end of the path  $\rho_{i,j}$  in the  $j^{th}$  subforest in  $t_i$ . Combining this information we see that if a path,  $\alpha i j \beta$ , is legal, then  $j$  must be such that  $n-1 \geq j \geq i-1$ . Of course this does not guarantee that the particular subforest will be present, but it does guarantee that there will be no paths to subforests that cannot possibly be present. The next definition, from Baldwin [12] makes these notions more precise.

**Definition 25** *The set of legal  $n$ -paths of degree  $k$ , denoted,  $P_n^k$ , is the smallest set such that:*

1.  $P_n^n = \{\lambda\}$ , if  $n \geq 0$ ,
2.  $P_n^k = P_n^{k+1} \cup kP_n^{k-1}$ , if  $n > k > 0$ ,
3.  $P_n^0 = P_n^1$ , if  $n > 0$ .

*The set of path variables of degree  $k$ , denoted,  $X_k$ , is equal to the set  $\{x_k^\rho \mid \rho \in P_k^{k-1}\}$ .*

□

Baldwin [12] gives the algorithm for construction of a normal form grammar from a given grammar. We give a simplified version of the algorithm given in that document. The algorithm is a two step algorithm. The first step is to use well known techniques to remove productions of the form  $A \rightarrow B$  where both  $A$  and  $B$  are in  $\mathcal{F}_1$ . This step is not included here. The second step is the application of one of the transformations given below.

So, given a grammar,  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , the normal form grammar,  $\tilde{G}_n^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^k$  has

$$\tilde{\Sigma} = \Sigma,$$

$\tilde{\mathcal{F}} = \mathcal{F}$  plus the new nonterminals added by the transformations,

$\tilde{P} = P$  minus any production not in the correct form plus those added by the transformations,

$$\tilde{S} = S.$$

Let  $P''$  be the set of all productions from  $P$  that are not already in the correct form. Apply one of the transformations given below to the productions in  $P''$  until  $P''$  is empty. The new nonterminals added by the transformations should not appear elsewhere in the grammar.

**Transformation 1:** If  $A \rightarrow a[_n \sigma] \in P''$ , for some  $A \in \mathcal{F}_n$ , and  $\sigma \notin \mathcal{F}_{n-1}$ , then add  $A_2$  to  $\mathcal{F}_{n-1}$ , and  $A \rightarrow a[_n A_2]$  to  $\tilde{P}$ . If  $\sigma = H$ ,  $H[_{n-1} G]$ , or  $x_{n-1}^\rho$ , for some  $H \in \mathcal{F}_n$ ,  $G \in \mathcal{F}_{n-2}$ , and  $x_{n-1}^\rho \in X_{n-1}$ , add  $A_2 \rightarrow \sigma$  to  $\tilde{P}$  otherwise add  $A_2 \rightarrow \sigma$  to  $P''$ .

**Transformation 2:** If  $A \rightarrow \sigma \in P''$ , for some  $A \in \mathcal{F}_m$ , for  $n > m \geq 1$  and  $\sigma \notin \mathcal{F}_{m+1}$  or  $\sigma \notin X_m$ , add  $A_2$  to  $\mathcal{F}_{m+1}$ , add  $A \rightarrow A_2$  to  $\tilde{P}$  and if  $\sigma$  is  $H$ ,  $H[_{m+1} G]$  or  $x_{m+1}^\rho$  for some  $H \in \mathcal{F}_{m+2}$ ,  $G \in \mathcal{F}_m$  and  $x_{m+1}^\rho \in X_{m+1}$  add  $A_2 \rightarrow \sigma$  to  $\tilde{P}$  otherwise add  $A_2 \rightarrow \sigma$  to  $P''$ .

**Transformation 3:** If  $A \rightarrow \sigma[_m \tau] \in P''$ , for some  $A \in \mathcal{F}_m$ , for  $n > m \geq 1$ , then

- If  $\sigma \notin \mathcal{F}_{m+1}$  and  $\tau \notin \mathcal{F}_{m-1}$  then add  $A_2$  to  $\mathcal{F}_{m+1}$  and  $A_3 \rightarrow \mathcal{F}_{m-1}$  and add  $A \rightarrow A_2[_m A_3]$  to  $\tilde{P}$ . If  $\sigma$  is the correct form for  $A_2 \in \mathcal{F}_{m+1}$ , then add  $A_2 \rightarrow \sigma$  to  $\tilde{P}$  otherwise add  $A_2 \rightarrow \sigma$  to  $P''$ . If  $\tau$  is the correct form for  $A_3 \rightarrow \mathcal{F}_{m-1}$ , then add  $A_3 \rightarrow \tau$  to  $\tilde{P}$  otherwise add  $A_3 \rightarrow \tau$  to  $P''$ .
- If  $\sigma \in \mathcal{F}_{m+1}$  and  $\tau \notin \mathcal{F}_{m-1}$ , then add  $A_3$  to  $\mathcal{F}_{m-1}$  and add  $a \rightarrow \sigma[_m A_3]$  to  $\tilde{P}$ . If  $\tau$  is the correct form for  $A_3 \rightarrow \mathcal{F}_{m-1}$ , then add  $A_3 \rightarrow \tau$  to  $\tilde{P}$  otherwise add  $A_3 \rightarrow \tau$  to  $P''$ .
- If  $\sigma \notin \mathcal{F}_{m+1}$  and  $\tau \in \mathcal{F}_{m-1}$ , then add  $A_2$  to  $\mathcal{F}_{m+1}$ , and add  $A \rightarrow A_2[_m \tau]$  to  $\tilde{P}$ . If  $\sigma$  is the correct form for  $A_2 \in \mathcal{F}_{m+1}$ , then add  $A_2 \rightarrow \sigma$  to  $\tilde{P}$  otherwise add  $A_2 \rightarrow \sigma$  to  $P''$ .

Since each production is of finite length, we know that this process will eventually terminate with  $P''$  empty. In addition, we know that the new productions add no nondeterminism to the grammar so the language defined by  $\tilde{G}_n^k$  is the same as that defined by  $G_n^k$ .

We now show how to construct a completed grammar from a normal form grammar. The IO language defined by the new grammar is the same as the IO language defined by the old grammar. Given a grammar  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  in normal form, construct  $\tilde{G}_n^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^k$  as directed below.

Each nonterminal in  $\tilde{\mathcal{F}}$  is an ordered triple. The first item is a nonterminal from  $\tilde{\mathcal{F}}$ . The second item is the dimension of the frontier at which this nonterminal first produces a single item from  $\Sigma$  (a *singleton*). The third item is a set of pairs,  $(x_j^p, d)$ , the set of path variables that might be derived from the nonterminal and must be satisfied outside of the structure derived from the nonterminal. The object copied by

the path variable will be a singleton for the first time at the frontier indicated by the dimension in the pair.

Consider the forest,  $a[_n t_n ][_{n-1} t_{n-1} ] \cdots [_k t_k ]$ . When the  $(n - 1)$ -dimensional frontier is taken, the path variables from  $X_{n-1}$  appearing in the frontier of  $t_n$  must be evaluated in the frontier of  $t_{n-1}$ . In fact, they can only be evaluated in the frontier of  $t_{n-1}$  and any part of the frontier of  $t_{n-1}$  not copied will be discarded. In addition, any object copied from the frontier of  $t_{n-1}$  will be of the form  $a[_{n-1} \sigma_{n-1} ]$  where  $\sigma_{n-1}$  may or may not be present. If  $\sigma_{n-1}$  is not present then the object copied will not be discarded at the next frontier. The result of this frontier operation is a forest  $a[_{n-1} t'_{n-1} ][_{n-2} t'_{n-2} ] \cdots [_k t'_k ]$  with the same observations applying to the  $(n - 2)$ -dimensional frontier. This means that if we have a production  $A \rightarrow B[_j C ]$  with  $A \in \mathcal{F}_j$ ,  $B \in \mathcal{F}_{j+1}$ , and  $C \in \mathcal{F}_{j-1}$ , then any path variables from  $X_j$  derived from  $B$  and not satisfied within the structure derived from  $B$  must be satisfied in the structure derived from  $C$ . If  $C$  is expanded so that there is a nonterminal from  $\mathcal{F}_j$  at the end of each of these paths, then the grammar will be completed. It is only necessary to expand for the path variables from  $X_j$  since these are the only path variables which will be evaluated in the structure derived from  $C$ .

$$\tilde{\Sigma} = \Sigma,$$

$$\tilde{\mathcal{F}}_i = \{ \langle \langle A, l, \alpha \rangle \rangle \} \text{ for all } A \in \mathcal{F}_i, l \in N \text{ and } \alpha \subseteq \nabla \times N \text{ where } \nabla \text{ is the set of all } X\text{s} \\ \text{actually appearing in } G_n^k, \text{ and } N = \{1, \dots, n\},$$

$\tilde{P}$  is constructed as indicated below,

$\tilde{S} \in \tilde{\mathcal{F}}_k$  does not occur elsewhere.

Construct  $\tilde{P}$  as follows:

1. If  $A \rightarrow a \in P$  where  $a \in \Sigma$ , then for all  $\alpha \subseteq \nabla \times N$

$$\langle\langle A, n, \alpha \rangle\rangle \rightarrow a \in \tilde{P}$$

since  $a$  is a singleton and this nonterminal derives no unsatisfied path variables.

2. If  $A \rightarrow x_{k'}^\rho \in P$ , then for all  $\alpha \subseteq \nabla \times N$  such that  $(x_{k'}^\rho, l) \in \alpha$

$$\langle\langle A, l, \alpha \rangle\rangle \rightarrow x_{k'}^\rho \in \tilde{P}$$

since  $x_{k'}^\rho$  is always satisfied outside of  $x_{k'}^\rho$  and if the object which replaces  $x_{k'}^\rho$  is first a singleton at dimension  $l$ , then the object derived from this nonterminal will be a singleton at dimension  $l$ .

3. If  $A \rightarrow B \in P$ , then for all  $\alpha \subseteq \nabla \times N$

$$\langle\langle A, l, \alpha \rangle\rangle \rightarrow \langle\langle B, l, \alpha \rangle\rangle \in \tilde{P}$$

since the structure derived from  $A$  must behave as the structure derived from  $B$  behaves.

4. If  $A \rightarrow a[_n B] \in P$ , then for all  $\alpha \subseteq \nabla \times N$  and  $n > l > 0$

$$\langle\langle A, l, \alpha \rangle\rangle \rightarrow a[_n \langle\langle B, l, \alpha \rangle\rangle] \text{ and } \langle\langle A, n-1, \alpha \rangle\rangle \rightarrow a[_n \langle\langle B, n, \alpha \rangle\rangle] \in \tilde{P}$$

since the  $a$  will be discarded at the dimension  $n$  frontier.

5. If  $A \rightarrow B[_{k'} C] \in P$  for some  $n > k' > 0$ , then

(a) for all  $\alpha \subseteq \nabla \times N$ ,  $n \geq l > k'$  and  $n \geq j > k'$

$$\langle\langle A, k', \alpha \rangle\rangle \rightarrow \langle\langle B, l, \alpha \rangle\rangle[_{k'} \langle\langle C, j, \alpha \rangle\rangle] \in \tilde{P}$$

since if  $B$  first produces a singleton at dimension  $l$ ,  $n \geq l > k'$  and  $C$  first produces one at dimension  $j$ ,  $n \geq j > k'$ , then  $A$  will not produce one until dimension  $k'$ ; any path variables which must be evaluated outside of the structures derived from  $B$  and  $C$  must also be evaluated outside of the structure derived  $A$ .

(b) for all  $\alpha \subseteq \nabla \times N$ ,  $n \geq l > k'$  and  $k' \geq j > 0$

$$\langle\langle A, j, \alpha \rangle\rangle \rightarrow \langle\langle B, l, \alpha \rangle\rangle|_{k'}, \langle\langle C, j, \alpha \rangle\rangle \in \tilde{P}$$

since  $A$  will not produce a singleton until  $C$  does.

(c) for all  $k' \geq l > 0$ ,  $\alpha_A$  and  $\alpha_B, \subseteq \nabla \times N$  such that

$$\begin{aligned} \alpha_B &= (\alpha_A - X_{k'} \times N) \cup \{(x_{k'}^\rho, l) \mid (\rho, l) \in p_C\} \\ p_C &= \{\rho \mid x_{k'}^\rho \in \nabla\} \times N \end{aligned}$$

together with the  $\mathcal{E}$  function as defined below

$$\langle\langle A, l, \alpha_A \rangle\rangle \rightarrow \langle\langle B, l, \alpha_B \rangle\rangle|_{k'}, \mathcal{E}_{k'-1}(C, p_C, \alpha_A) \subseteq \tilde{P}.$$

Since  $B$  does not produce a singleton until dimension  $l$ ,  $k' \geq l > 0$ , it must derive subforests at dimension  $k' + 1$  or greater. Therefore, there may be path variables from  $X_{k'}$  that must be evaluated in the structure derived from  $C$ . In addition,  $A$  will first produce a singleton when  $B$  does. The set  $p_C$  contains pairs,  $(\rho, d)$ , where  $\rho$  is a path which may be evaluated in the structure derived from  $C$  and  $d$  is a dimension at which the structure copied by the path variable will first be a singleton. The set  $\alpha_B$  contains pairs,  $(x_{k'}^\rho, d)$ , where the only path variables from  $X_{k'}$  are those from  $p_C$  which first produce a singleton at dimension  $l$ .

6.  $\tilde{S} \rightarrow \langle\langle S, l, \emptyset \rangle\rangle \in \tilde{P}$  for all  $n \geq l \geq 1$ .

7. These are all the elements of  $\tilde{P}$ .

The purpose of the  $\mathcal{E}$  function is to evaluate the nonterminal for all the paths in the set  $p$ . The initial calls to  $\mathcal{E}$  are generated by step 5-(c) with  $p$  containing those paths which might be evaluated by some frontier in the structure derived from  $C$ . This means that if  $B$  in  $B[k, C]$  derives a path variable from  $X_k$  that must be evaluated in the structure derived from  $C$ , then  $\mathcal{E}(C, p_C, \alpha_A)$  will have nonterminals from  $\mathcal{F}'_k$  in those positions. Thus, no subpart of a structure derived from a nonterminal will not be copied.  $\mathcal{E}$  is a set of finite functions  $\{\mathcal{E}_{k''}\}_{n \geq k'' > 0}$  where for all  $A \in \mathcal{F}_{k''}$ ,  $p \subseteq \{\rho \mid x_k^\rho \in \nabla\} \times N$ , and  $\alpha \in \nabla \times N$ ,  $\mathcal{E}_{k''}(A, p, \alpha)$  is the smallest set such that

1. If  $A \rightarrow B[k'', C] \in P$ , then for all  $\alpha$ ,  $p_B$  and  $p_C$  such that

$$\begin{aligned} p &= p_B \cup \{(k''\rho, l) \mid (\rho, l) \in p_C\} \\ p_B &\subseteq P_k^{k''+1} \times N \\ p_C &\subseteq P_k^{k''-1} \times N \end{aligned}$$

such that if  $(\rho, l) \in p_B \cup p_C$ , then  $\rho$  is the suffix of a path in  $\nabla \cap X_k$

$$\mathcal{E}_{k''+1}(B, p_B, \alpha)[_{k''} \mathcal{E}_{k''-1}(C, p_C, \alpha)] \subseteq \mathcal{E}_{k''}(A, p, \alpha).$$

The sets  $p_B$  and  $p_C$  contains those pairs,  $(\rho, d)$ , whose paths can be evaluated in the structure derived from  $B$  and  $C$  respectively, and  $p$  is the set of paths that can be evaluated in the structure derived from  $A$ . The last line restricts the paths under consideration to suffixes of paths which can actually occur in the grammar.

2. If  $A \rightarrow B \in P$  where  $A \in \mathcal{F}_{k''}$ ,  $B \in \mathcal{F}_{k''+1}$ , then for all  $p \subseteq P_k^{k''+1} \times N$  such that if  $(\rho, l) \in p$ , then  $\rho$  is the suffix of a path in  $\nabla \cap X_k$  and  $\alpha \subseteq \nabla \times N$

$$\mathcal{E}_{k''+1}(B, p, \alpha) \subseteq \mathcal{E}_{k''}(A, p, \alpha).$$

In this situation we restrict  $p$  to those paths in the grammar which can be evaluated in the structure derived from  $B$ .

3. If  $C \in \mathcal{F}_{k''}$ , then for all  $\alpha \subseteq \nabla \times N$ ,  $j \in N$ , and  $\beta \subseteq (\{\lambda\} \cup \{\rho \mid \rho \in P_k^{k''}, \rho \notin P_k^{k''+1} \text{ and } \rho \text{ is the suffix of a path in } \nabla \cap X_k\}) \times \{j\}$ ,

$$\{\langle C, j, \alpha \rangle\} \subseteq \mathcal{E}_{k''}(C, \beta, \alpha).$$

The structure derived from  $C$  will be a singleton at the dimension  $j$  frontier and it will be copied at dimension  $k$ , so the copied object will be a singleton at dimension  $j$ . The set  $\beta$  contains those paths which could be satisfied by the structure derived from  $C$ . If  $\beta = \{\lambda\}$ , then the structure will be copied at dimension  $k$  frontier. If  $\beta = \emptyset$ , the structure will be dropped at the dimension  $k$  frontier. If  $\beta \neq \{\lambda\}$  and  $\beta \neq \emptyset$ , then the dimension  $k$  frontier might not be defined since  $C$  might not derive a structure in which it is possible to evaluate the path.

□

Baldwin gives the following replacement for step 3 in the  $\mathcal{E}$  function. Using these steps guarantees the existence of  $f^{n-k}(\alpha)$  for all  $\alpha$  such that  $S \xrightarrow{B} \alpha$ . This revision of the  $\mathcal{E}$  functions only produces something when there is an object at the end of the paths being evaluated. The replacement steps follow:

3. If  $C \in \mathcal{F}_k$ , then  $k > k'' > 0$  and for all  $\alpha \subseteq \nabla \times N$  and  $\beta \subseteq (\{\lambda\} \cup \{\rho \mid \rho \in P_i^{k''}, \rho \notin P_i^{k''+1} \text{ and } \rho \text{ in the suffix of a path in } \nabla \cap X_i\}) \times \{j\}$  where  $\beta \neq \emptyset$  and  $\beta \notin \{\lambda\} \times N$

$$\mathcal{E}_{k''}(C, \beta, \alpha),$$

4. If  $C \in \mathcal{F}_{k''}$  and  $n \geq k'' > k$ , then

$$\mathcal{E}_{k''}(C, \emptyset, \alpha) = \{\langle\langle C, j, \alpha \rangle\rangle \mid j \in N\}$$

and

$$\mathcal{E}_{k''}(C, (\{\lambda\}, j), \alpha) = \{\langle\langle C, j, \alpha \rangle\rangle\}.$$

□

**Theorem 11** *There is no IO grammar,  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  such that the language  $L = \{(b^+ a b^+)^{2^m} \mid m \geq 1\}$  is the  $(n-1)^{st}$  frontier of the IO language derived from  $G_n^k$ .*

*Proof:* If we assume that there is a grammar,  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  such that  $L = f^{n-1}(S)$ . We may assume that  $G_n^k$  is a completed grammar and that  $f^{n-2}(S)$  exists. A completed grammar has the property that during any copy operation, if any piece of a structure derived from a nonterminal is copied, then the entire structure is copied. We also know that to produce  $2^m$ ,  $m \geq 1$ , copies of any string, copying must be used. This means that each string from  $b^+ a b^+$  occurring in a string in the language must be generated by a different nonterminal, or the  $bs$  must be added or dropped by subsequent frontier operations after the initial string is copied. If each of these strings is generated by a different nonterminal, then we must have an indeterminate number of nonterminals in the grammar. This cannot be. If  $bs$  are added or dropped

by subsequent frontier operations, then to produce a string having each sequence of  $bs$  different from every other sequence of  $bs$ , they must be copied into  $2^m$  different places. But again the grammar must produce  $2^m$  different objects,  $m \geq 1$ , none of which was produced by copying. Again the grammar must have an indeterminate number of nonterminals.

□

### 2.3 $OI$ 3-Dimensional = $OI$ Macro

In this section we show that the 1-dimensional languages defined by the 2-dimensional  $OI$  grammars correspond to the context free languages, and the 1-dimensional languages defined by the 1-dimensional frontiers of the 3-dimensional  $OI$  grammars correspond to the  $OI$  macro languages as defined by Fischer. Since a 2-dimensional  $OI$  grammar contains only the  $x_1^\lambda$  path variable any  $OI$  derivation will be the same as the  $IO$  derivation and *vice versa*. This means that  $OI$  dimension 2 is equal to  $IO$  dimension 2 and since the  $IO$  dimension 2 languages correspond to the context free languages [12], so do the  $OI$  dimension 2 languages.

For the 3-dimensional languages we will show how to construct an  $OI$  macro grammar from an  $OI$  3-dimensional grammar. This is a three step process. The first step is the construction of a normal form grammar for the language; the second step is the construction of a grammar, similar to a completed grammar, which has the property that if a production contains the "sentential form"  $B[\sigma]$  and  $B$  generates any path variables which are not satisfied within the structure generated by  $B$ , then  $\sigma$  will be expanded no further than the longest of these paths. In fact, each such path will end in a nonterminal. Finally, we show how to convert this grammar to an  $OI$

macro grammar.

The construction of a normal form grammar is given in the previous section, so we will assume that  $G_3^k$  is in normal form and show how to construct the modified completed grammar. Let  $G_3^k = (\Sigma, \mathcal{F}, P, S)_3^k$ , construct the new grammar  $\tilde{G}_3^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_3^k$  as follows:

$$\tilde{\Sigma} = \Sigma;$$

$\tilde{\mathcal{F}}$  = those nonterminals added during the construction of  $\tilde{P}$ ;

$\tilde{S} \in \tilde{\mathcal{F}}_k$  does not appear elsewhere in the grammar;

$\tilde{P}$  contains the productions indicated below.

1. If  $A \rightarrow a \in P$ , then add  $A$  to  $\tilde{\mathcal{F}}_3$  and  $A \rightarrow a$  to  $\tilde{P}$ .
2. If  $A \rightarrow x_1^\lambda \in P$ , then add  $A$  to  $\tilde{\mathcal{F}}_1$  and  $A \rightarrow x_1^\lambda$  to  $\tilde{P}$ .
3. If  $A \rightarrow x_2^\rho \in P$ , then add  $A$  to  $\tilde{\mathcal{F}}_2$  and  $A \rightarrow x_2^\rho$  to  $\tilde{P}$ .
4. If  $A \rightarrow B \in P$  for some  $A \in \mathcal{F}_j$ ,  $B \in \mathcal{F}_{j+1}$ , then add  $A$  to  $\tilde{\mathcal{F}}_j$ ,  $B$  to  $\tilde{\mathcal{F}}_{j+1}$  and  $A \rightarrow B$  to  $\tilde{P}$ .
5. If  $A \rightarrow a[_3 B] \in P$ , then add  $A$  to  $\tilde{\mathcal{F}}_3$  and  $A \rightarrow a[_3 B]$  to  $\tilde{P}$ .
6. If  $A \rightarrow B[_1 C] \in P$ , then add  $A$  to  $\tilde{\mathcal{F}}_1$  and  $A \rightarrow B[_1 C]$  to  $\tilde{P}$ .
7. If  $A \rightarrow B[_2 C] \in P$ , then add  $A$  to  $\tilde{\mathcal{F}}_2$  and
  - (a) for each  $a$  such that  $B \rightarrow a \in P$ , add  $A \rightarrow a[_2 C]$  to  $\tilde{P}$ ,

- (b) for each  $B \rightarrow a[_3 G] \in P$ , add  $\tilde{B}$  to  $\tilde{\mathcal{F}}_3$ , and  $\tilde{B} \rightarrow a[_3 G]$  to  $\tilde{P}$ . Finally add  $A \rightarrow \tilde{B}[_2 \tau]$  to  $\tilde{P}$  where  $\tau \in H_3^1$  is built as follows.

Define  $Eval: \mathcal{F}_1 \times 1^* \rightarrow \text{powerset}(\mathcal{F}_2)$  recursively by:

$$Eval(C, \rho) = \begin{cases} \{H \mid C \rightarrow H \in P \vee C \rightarrow H[_1 G] \in P\} & \text{if } \rho = \lambda \\ Eval(G, \rho') \quad \forall G \ni C \rightarrow H[_1 G] \in P & \text{if } \rho \neq \lambda \end{cases}$$

Then  $\tau = C_0[_1 C_1[_1 \dots [_1 C_m] \dots ]]$ , where

$m = \max \{j \mid Eval(C, 1^j) = \emptyset \wedge x_2^{1^j} \in \nabla\}$ ,  $C_j \in \mathcal{F}_2'$ , and  $C_j \rightarrow H \in P'$  for every  $H \in Eval(C, 1^j)$ .

**Theorem 12** *The OI 3-dimensional language derived from  $\tilde{G}_n^k$  constructed as indicated above is the same as the OI 3-dimensional language derived from  $G_n^k$ .*

*Proof:* Using the lemma below  $f(S) = f(\tilde{S})$ .

□

**Lemma 8** *Let  $G_3^k = (\Sigma, \mathcal{F}, P, S)_3^k$ , then if  $\tilde{G}_3^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_3^k$  is the language constructed from  $G_3^k$  using the construction given above, and  $A \in \mathcal{F}$ , then  $f_{OI, G_3^k}(A) = f_{OI, \tilde{G}_3^k}(A)$ .*

*Proof:* The proof is by induction on the number of derivation steps.

*Basis:* If  $\sigma \in f_{OI, G_3^k}(A)$  by one derivation step, then  $\sigma = a$ ,  $x_1^\lambda$ , or  $x_2^\rho$  so  $A \rightarrow a$ ,  $A \rightarrow x_1^\lambda$ , or  $A \rightarrow x_2^\rho$  are productions in  $G_3^k$  and by the construction are also in  $\tilde{G}_3^k$ . Clearly  $f_{OI, G_3^k}(A) = f_{OI, \tilde{G}_3^k}(A)$ .

*Induction Step:* Assume that if  $\sigma \in f_{OI, G_3^k}(A)$  after  $m$  derivation steps, then  $\sigma \in f_{OI, \tilde{G}_3^k}(A)$ . Let  $\sigma \in f_{OI, G_3^k}(A)$  after  $m+1$  steps.

1. If  $A \rightarrow B \in P$  is used, then  $\sigma \in f_{OI, G_3^k}(B)$  after  $m$  steps, so  $\sigma \in f_{OI, \hat{G}_3^k}(B) \subseteq f_{OI, \hat{G}_3^k}(A)$ .
2. If  $A \rightarrow a[_3 B] \in P$  is used, then  $\sigma \in f_{OI, G_3^k}(B)$  after  $m$  steps, so  $\sigma \in f_{OI, \hat{G}_3^k}(B) \subseteq f_{OI, \hat{G}_3^k}(A)$ .
3. If  $A \rightarrow B[_1 C]$  is used then  $\sigma \in f_{OI, G_3^k}(B)[_1 f_{OI, G_3^k}(C)]$  by less than  $m+1$  steps, so  $\sigma \in f_{OI, \hat{G}_3^k}(B)[_1 f_{OI, \hat{G}_3^k}(C)] \subseteq f_{OI, \hat{G}_3^k}(A)$ .
4. If  $A \rightarrow B[_2 C]$  and  $B \rightarrow a$  is used, then  $\sigma \in a[_2 f_{OI, G_3^k}(C)]$  by less than  $m$  steps and so  $\sigma \in a[_2 f_{OI, \hat{G}_3^k}(C)] \subseteq f_{OI, \hat{G}_3^k}(A)$ .
5. If  $A \in B[_2 C]$  and  $B \rightarrow a[_3 G]$  is used, then  $\sigma \in f_{OI, G_3^k}(G) \xrightarrow{OI} f_{OI, G_2^k}(C)$ . From the induction hypothesis, we know that  $f_{OI, G_3^k}(G) \subseteq f_{OI, \hat{G}_3^k}(G)$ . In addition, if  $x_2^\rho$  appears in  $f_{OI, G_3^k}(G)$ , then it will be replaced by an object located at  $\rho$  in a tree from  $f_{OI, G_3^k}(C)$ . The replacement tree will be one derived from a nonterminal in  $Eval(C, \rho)$ . But  $(\tau)_\rho$  is  $C|_\rho$  where  $C|_\rho \rightarrow H$  for all  $H \in Eval(C, \rho)$  is a production in  $P'$ . Combining this information, we see that  $f_{OI, G_3^k}(G) \xrightarrow{OI} f_{OI, G_3^k}(C) \subseteq (f_{OI, \hat{G}_3^k}(G) \xrightarrow{OI} f_{OI, \hat{G}_3^k}(\tau)) \subseteq f_{OI, \hat{G}_3^k}(A)$ .

Therefore,  $f_{OI, G_3^k}(A) \subseteq f_{OI, \hat{G}_3^k}(A)$ .

We next show that  $f_{OI, \hat{G}_3^k}(A) \subseteq f_{OI, G_3^k}(A)$ .

*Basis:* If  $\sigma \in f_{OI, \hat{G}_3^k}(A)$  by one derivation step, then  $\sigma = a$ ,  $x_1^\lambda$ , or  $x_2^\rho$ . So  $A \rightarrow a$ ,  $A \rightarrow x_1^\lambda$ , or  $A \rightarrow x_2^\rho$  are productions in  $P'$  and are therefore in  $P$ . Therefore,  $\sigma \in f_{OI, G_3^k}(A)$ .

*Induction Step:* Assume that  $\sigma \in f_{OI, \hat{G}_3^k}(A)$  after less than  $m$  derivation steps, then  $\sigma \in f_{OI, G_3^k}(A)$ . Let  $\sigma \in f_{OI, G_3^k}(A)$  after  $m+1$  steps.

1. If  $A \rightarrow B$  is used, then  $\sigma \in f_{OI, \tilde{G}_3^k}(B)$  after  $m$  steps, so  $\sigma \in f_{OI, G_3^k}(B) \subseteq f_{OI, G_3^k}(A)$ .
2. If  $A \rightarrow a[_3 B ]$  is used, then  $\sigma \in f_{OI, \tilde{G}_3^k}(B)$  after  $m$  steps, so  $\sigma \in f_{OI, G_3^k}(B) \subseteq f_{OI, G_3^k}(A)$ .
3. If  $A \rightarrow B[_1 C ]$  is used, the  $\sigma \in f_{OI, \tilde{G}_3^k}(B)[_1 f_{OI, \tilde{G}_3^k}(C) ]$  by less than  $m+1$  steps, so  $\sigma \in f_{OI, G_3^k}(B)[_1 f_{OI, G_3^k}(C) ] \subseteq f_{OI, G_3^k}(A)$ .
4. If  $A \rightarrow a[_2 C ]$  is used, then there exist  $B \in \mathcal{F}_3$  such that  $B \rightarrow a \in P$  and  $A \rightarrow B[_2 C ] \in P$ . From this we know

$$\begin{aligned}
 \sigma \in f_{OI, \tilde{G}_3^k}(a[_2 C ]) &\subseteq a[_2 f_{OI, \tilde{G}_3^k}(C) ] \\
 &\subseteq a[_2 f_{OI, G_3^k}(C) ] \\
 &\subseteq f_{OI, G_3^k}(B[_2 C ]) \\
 &\subseteq f_{OI, G_3^k}(A).
 \end{aligned}$$

5. If  $A \rightarrow \tilde{B}[_2 \tau ]$  is used, then there exist  $B \in \mathcal{F}_3$  such that  $B \rightarrow a[_3 G ] \in P$ ,  $A \rightarrow B[_2 C ] \in P$  and  $\tilde{B} \rightarrow a[_3 G ] \in P'$ . In addition,  $\tau$  is such that  $(\tau)_{1^j} = C_j$  and  $C_j \rightarrow H \in P$  for all  $H \in Eval(C, 1^j)$ . Therefore,  $\sigma \in f_{OI, \tilde{G}_3^k}(B[_2 \tau ]) = f_{OI, \tilde{G}_3^k}(\tilde{B}) \stackrel{\Leftarrow}{OI} f_{OI, G_3^k}(\tau)$  where  $f_{OI, \tilde{G}_3^k}(\tilde{B}) \subseteq f_{OI, G_3^k}(B)$ , and if  $x_2^{1^j}$  appears in  $f_{OI, \tilde{G}_3^k}(\tilde{B})$ , then if  $(\tau)_{1^j}$  is defined, some tree in  $f_{OI, G_3^k}(C)$  will have an object at  $1^j$ . Therefore,  $\sigma \in f_{OI, G_3^k}(B) \stackrel{\Leftarrow}{OI} f_{OI, G_3^k}(C) \subseteq f_{OI, G_3^k}(A)$ .

□

**Theorem 13**  $L$  is the 1-dimensional frontier of the OI language defined by  $G_3^k = (\Sigma, \mathcal{F}, P, S)_3^k$  if and only if there is an OI macro grammar for  $L$ .

*Proof:* If  $L$  is an  $OI$  macro grammar, then the transformation described in Baldwin [12] will produce an  $OI$  3-dimensional grammar for the language. We give a slightly different construction here.

If  $L$  has  $OI$  macro grammar  $G = (\Sigma, \mathcal{F}, V, \rho, S, P)$  where  $\Sigma$  is the set of terminals,  $\mathcal{F}$  is the set of function symbols,  $V$  is the set of arguments,  $\rho$  is the ranking function for  $\mathcal{F}$ ,  $S$  is the start symbol and  $P$  is the set of productions in  $OI$  standard form, then hypertree grammar  $G_3^3 = (\Sigma', \mathcal{F}', P', S')_3^3$  as indicated below is the desired grammar.

$$\Sigma' = \Sigma \cup \{*\},$$

$$\mathcal{F}'_1 = \emptyset, \mathcal{F}'_2 = \emptyset, \text{ and } \mathcal{F}'_3 = \mathcal{F},$$

$P'$  is the set of productions constructed below,

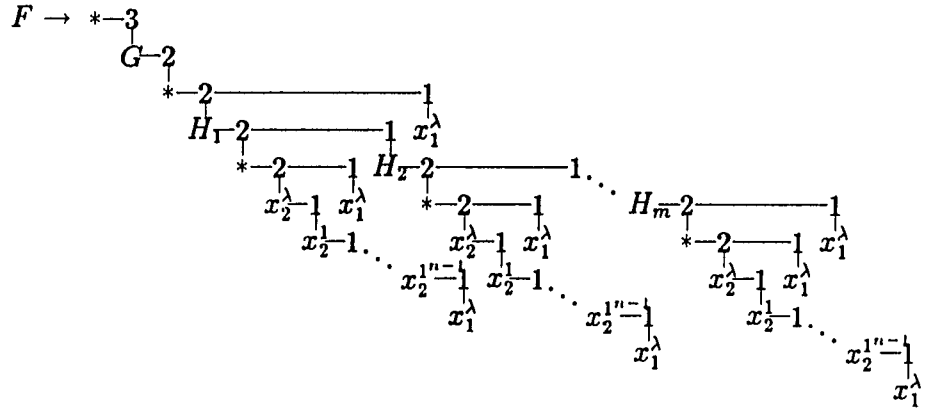
$$S' = S.$$

The productions in  $P$  are in  $OI$  standard form so they are in one of the following forms:

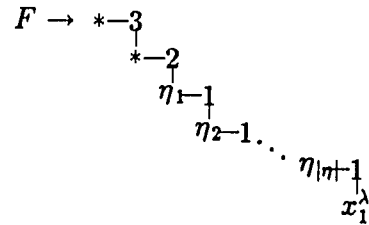
1.  $F(x_1, \dots, x_n) \rightarrow G(H_1(x_1, \dots, x_n), \dots, H_m(x_1, \dots, x_m))$ , for  $m, n \geq 0$ ,
2.  $F(x_1, \dots, x_n) \rightarrow \eta$ ,  $\eta \in (\Sigma \cup V)^*$ , for  $n \geq 0$ .

So the productions in  $P'$  are constructed as follows:

1. If  $F(x_1, \dots, x_n) \rightarrow G(H_1(x_1, \dots, x_n), \dots, H_m(x_1, \dots, x_m))$ , for  $m, n \geq 0$  is a production in  $P$ , then  $P'$  has the production:



2. If  $F(x_1, \dots, x_n) \rightarrow \eta$ ,  $\eta \in (\Sigma \cup V)^*$ , for  $n \geq 0$  is a production in  $P$ , then  $P'$  has the production:



where  $\eta_i = \eta_i$  if  $\eta_i \in \Sigma$ , and  $\eta_i = x_2^{1^{j-1}}$  if  $\eta_i = x_j$ .

If  $L$  is the 1-dimensional frontier of an  $OI$  3-dimensional grammar, then  $L$  has a modified completed grammar  $G_3^k = (\Sigma, \mathcal{F}, P, S)_3^k$ . We construct an  $OI$  macro grammar,  $G' = (\Sigma', \mathcal{F}', V', \rho', S', P')$  as follows:

$$\Sigma' = \Sigma,$$

$$\mathcal{F}' = \{S', D\} \cup \mathcal{F} \times \{1, 2\} \times \{1, 2, 3\} \times \{0, 1, 2, 3\}^{m+1},$$

$$V' = \{y_0, \dots, y_m\} \text{ where } y_i \text{ does not appear elsewhere in the grammar, and } m = \max\{j \mid x_2^{1^j} \text{ is in the grammar}\},$$

$\rho'$  maps all  $\mathcal{F}'$  to  $m$ ,

$S'$  does not appear elsewhere in the grammar,

$P'$  contains the productions described below.

Each nonterminal in the grammar is an  $m + 3$ -tuple  $\langle\langle A, i, j, a_0, \dots, a_m \rangle\rangle$ . The first object is the nonterminal with which the production is associated. The object in the second position indicates whether or not this nonterminal produces a singleton. The object in the third position represent whether the frontier produces a singleton, a string ending in  $x_1^\lambda$ , or a string not ending in  $x_1^\lambda$ . And the objects in the last  $m + 1$  positions represent whether the the object copied by this path variable produces a singleton, a string ending in  $x_1^\lambda$ , or a string not ending in  $x_1^\lambda$ . A zero in one of the last  $m + 1$  positions means that the argument associated with that path variable is not used. In the nonterminal  $\langle\langle A, i, j, a_0, \dots, a_m \rangle\rangle$ , the pair  $(i, j)$  may be interpreted using the table below. The  $\times$ s indicate those combinations which make sense and are used in the grammar.

	1: $f(A)$ is a singleton	2: $f(f(A))$ is a string ending in $x_1^\lambda$	3: $f(f(A))$ is a string not ending in $x_1^\lambda$
1: $A$ produces a singleton	$\times$		
2: $A$ does not produce a singleton	$\times$	$\times$	$\times$

The productions in  $P'$  are given by the rules which follow. We use the notation  $\vec{a}$  for  $a_0, \dots, a_m$ .

1. If  $A \rightarrow a \in P$ , for  $A \in \mathcal{F}_3$ , and  $a \in \Sigma$ , then for all  $\vec{a}$

$$\langle\langle A, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \rightarrow a \in P',$$

2. If  $A \rightarrow x_1^\lambda \in P$ , for  $A \in \mathcal{F}_1$ , then for all  $\vec{a}$

$$\langle\langle A, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \lambda \in P',$$

3. If  $A \rightarrow x_2^\rho \in P$ , for  $A \in \mathcal{F}_2$ , and  $\rho = 1^j$  then for all  $\vec{a}$  such that  $a_j \neq 0$

$$\langle\langle A, 2, a_j, \vec{a} \rangle\rangle(\vec{y}) \rightarrow y_j \in P',$$

4. If  $A \rightarrow B \in P$ , then for all  $\vec{a}$  the following are in  $P'$

$$\langle\langle A, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \langle\langle B, 1, 1, \vec{a} \rangle\rangle(\vec{y})$$

$$\langle\langle A, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \langle\langle B, 2, 1, \vec{a} \rangle\rangle(\vec{y})$$

$$\langle\langle A, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \langle\langle B, 2, 2, \vec{a} \rangle\rangle(\vec{y})$$

$$\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \langle\langle B, 2, 3, \vec{a} \rangle\rangle(\vec{y}),$$

5. If  $A \rightarrow a[{}_3 B] \in P$ , then for all  $\vec{a}$  the following are in  $P'$

$$\langle\langle A, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \langle\langle B, 1, 1, \vec{a} \rangle\rangle(\vec{y})$$

$$\langle\langle A, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \langle\langle B, 2, 1, \vec{a} \rangle\rangle(\vec{y})$$

$$\langle\langle A, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \langle\langle B, 2, 2, \vec{a} \rangle\rangle(\vec{y})$$

$$\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) \rightarrow \langle\langle B, 2, 3, \vec{a} \rangle\rangle(\vec{y}),$$

6. If  $A \rightarrow B[_1 C] \in P$ , then for all  $\vec{a}$  the following are in  $P'$

$$\begin{aligned}
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 2, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 3, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 2, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 3, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 2, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \langle\langle C, 2, 3, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle B, 2, 3, \vec{a} \rangle\rangle(\vec{y}),
\end{aligned}$$

7. If  $A \rightarrow a[_2 C] \in P$ , then for all  $\vec{a}$  the following are in  $P'$

$$\begin{aligned}
\langle\langle A, 2, 1, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle C, 1, 1, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 1, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle C, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 2, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle C, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \\
\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) &\rightarrow \langle\langle C, 2, 3, \vec{a} \rangle\rangle(\vec{y}),
\end{aligned}$$

8. If  $A \rightarrow B[_2 \tau] \in P$ , such that  $\tau = C_0[_1 \cdots [_1 C_j] \cdots ]$  where  $C_i \in \mathcal{F}_2$  for  $m \geq j \geq 0$ , then for all  $\vec{a}, \vec{b} = b_0, \dots, b_j, 0, \dots, 0$ , and  $\vec{c} = c_0, \dots, c_j, 0, \dots, 0$  the

following are in  $P'$

$$\langle\langle A, 2, 1, \vec{a} \rangle\rangle(\vec{y}) \rightarrow$$

$$\langle\langle B, 2, 1, \vec{b} \rangle\rangle(\langle\langle C_0, c_0, b_0, \vec{a} \rangle\rangle(\vec{y}), \dots, \langle\langle C_j, c_j, b_j, \vec{a} \rangle\rangle(\vec{y}), D, \dots, D, )$$

$$\langle\langle A, 2, 2, \vec{a} \rangle\rangle(\vec{y}) \rightarrow$$

$$\langle\langle B, 2, 2, \vec{b} \rangle\rangle(\langle\langle C_0, c_0, b_0, \vec{a} \rangle\rangle(\vec{y}), \dots, \langle\langle C_j, c_j, b_j, \vec{a} \rangle\rangle(\vec{y}), D, \dots, D, )$$

$$\langle\langle A, 2, 3, \vec{a} \rangle\rangle(\vec{y}) \rightarrow$$

$$\langle\langle B, 2, 3, \vec{b} \rangle\rangle(\langle\langle C_0, c_0, b_0, \vec{a} \rangle\rangle(\vec{y}), \dots, \langle\langle C_j, c_j, b_j, \vec{a} \rangle\rangle(\vec{y}), D, \dots, D, ),$$

$$9. S' \rightarrow \langle\langle S, i, j, \vec{0} \rangle\rangle \text{ for all } i, j,$$

10. These are all of the productions.

□

### 3 MORE ABOUT THE NATURE OF THE *OI* LANGUAGES

In this chapter we investigate further the *OI* grammars. The first section shows that there is a completed grammar form for the *OI* languages. This grammar form is very close to that for the *IO* languages. The next section concerns the closure properties of these languages. The most important result of this section is that they are closed under intersection with the regular languages as defined by Baldwin. In the third section we show how to remove dead symbols from the grammars. Finally, we describe an extension of the  $n$ -dimensional grammars that allows the definition of grammars for a larger class of languages.

#### 3.1 *OI* Completed Grammars and Related Theorems

There is also a completed grammar for the *OI* languages. The major difference between the *OI* completed grammar and the *IO* completed grammar is the expansion of  $C$  in productions of the form  $A \rightarrow B[_{n-1} C ]$ . If the nonterminal  $B$  has productions of the form  $B \rightarrow a[_n G ]$ , then any derivation from  $B[_{n-1} C ]$  which uses this production will use  $f(G) \stackrel{\text{OI}}{\Leftarrow} f(C)$ . The *IO* completed grammar replaces the nonterminal  $C$  by trees, each of which is an expansion of  $C$  along all paths appearing in the structure derived from  $G$  and not satisfied in that structure. If we do this for the *OI* completed grammar, we do not get *OI* substitution. So, we replace  $C$  by a tree which has

nonterminals at the end of each such path. These nonterminals will have productions of the form  $C_\rho \rightarrow H$  where  $H$  is any nonterminal located at  $(C)_\rho$ . This preserves the  $OI$  substitution. Productions of the form  $B[k \ C]$  where  $k < n - 1$  still use the same evaluation strategy as the  $IO$  completed grammar, since no substitution will be made from these structures until after the first  $OI$  frontier is taken. All subsequent frontiers are necessarily  $IO$  since the substitution is of a single tree into a single tree.

**Definition 26** *A grammar,  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , is said to be an  $OI$  completed grammar if, during any extract operation on any element of the  $OI$  language defined by  $G_n^k$ , a piece of the structure derived from any nonterminal is copied, then all of the structure derived from that nonterminal is copied.*

**Theorem 14** *If  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  is an  $OI$   $n$ -dimensional grammar, then there is an  $OI$  completed grammar of dimension  $n$ ,  $\tilde{G}_n^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^k$ , defining the same language.*

*Proof:* Without loss of generality, assume that  $G_n^k$  is a normal form grammar. The grammar  $\tilde{G}_n^k$  has

$$\tilde{\Sigma} = \Sigma,$$

$$\tilde{\mathcal{F}} = \{ \langle \langle A, i, \alpha \rangle \rangle \mid A \in \mathcal{F}_{k'}, i \in N, \alpha \subseteq \nabla \times N \} \text{ where } N = \{1, 2, \dots, n\}, k' \in N \text{ and } \nabla$$

is the set of  $X$ s actually appearing in the grammar together with the nonterminals added at step 6,

$\tilde{P}$  is the set of productions described below,

$\tilde{S} \in \tilde{\mathcal{F}}_k$  does not appear elsewhere in the grammar.

$\tilde{P}$  contains the following productions:

1. If  $A \rightarrow a \in P$ , for  $A \in \mathcal{F}_n$ ,  $a \in \Sigma$ , then for all  $\alpha \subseteq \nabla \times N$ ,

$$\langle\langle A, n, \alpha \rangle\rangle \rightarrow a \in \tilde{P}.$$

2. If  $A \rightarrow x_{k'}^p \in P$ , for  $A \in \mathcal{F}_{k'}$ ,  $x_{k'}^p \in X_{k'}$ , then for all  $\alpha \subseteq \nabla \times N$  such that  $(x_{k'}^p, i) \in \alpha$ ,

$$\langle\langle A, i, \alpha \rangle\rangle \rightarrow x_{k'}^p \in \tilde{P}.$$

3. If  $A \rightarrow a[_n B] \in P$ , for  $A \in \mathcal{F}_n$ ,  $B \in \mathcal{F}_{n-1}$ ,  $a \in \Sigma$ , then for all  $\alpha \subseteq \nabla \times N$ ,

$$\langle\langle A, i, \alpha \rangle\rangle \rightarrow a[_n \langle\langle B, i, \alpha \rangle\rangle] \in \tilde{P}$$

and for all  $\alpha \subseteq \nabla \times N$ ,

$$\langle\langle A, n-1, \alpha \rangle\rangle \rightarrow a[_n \langle\langle B, n, \alpha \rangle\rangle] \in \tilde{P}.$$

4. If  $A \rightarrow B \in P$ , for  $A \in \mathcal{F}_{k'}$ ,  $B \in \mathcal{F}_{k'-1}$ ,  $a \in \Sigma$ , then for all  $\alpha \subseteq \nabla \times N$ ,  $i \in \{1, \dots, n\}$ ,

$$\langle\langle A, i, \alpha \rangle\rangle \rightarrow \langle\langle B, i, \alpha \rangle\rangle \in \tilde{P}.$$

5. If  $A \rightarrow B[_{k'} C] \in P$ , for  $A \in \mathcal{F}_{k'}$ ,  $C \in \mathcal{F}_{k'-1}$ ,  $B \in \mathcal{F}_{k'+1}$ ,  $1 \leq k' \leq n-2$ , then

- (a) for all  $\alpha \subseteq \nabla \times N$ ,  $k' < i \leq n$ , and  $k' + 1 \leq j \leq n$ ,

$$\langle\langle A, k', \alpha \rangle\rangle \rightarrow \langle\langle B, i, \alpha \rangle\rangle[_{k'} \langle\langle C, j, \alpha \rangle\rangle] \in \tilde{P},$$

- (b) for all  $\alpha \subseteq \nabla \times N$ ,  $k' < i \leq n$ , and  $1 \leq j \leq k'$ ,

$$\langle\langle A, j, \alpha \rangle\rangle \rightarrow \langle\langle B, i, \alpha \rangle\rangle[_{k'} \langle\langle C, j, \alpha \rangle\rangle] \in \tilde{P}.$$

(c) for all  $1 \leq i \leq k'$ , and  $\alpha_A$  and  $\alpha_B \subseteq \nabla \times N$ , such that

$$\begin{aligned}\alpha_B &= (\alpha_A - X_{k'} \times N) \cup \{(x_{k'}^\rho, i) \mid (\rho, i) \in p_C\} \\ p_C &= \{\rho \mid x_{k'}^\rho \in \nabla\} \times N,\end{aligned}$$

using the  $\mathcal{E}$  functions defined on page 83 for  $IO$  completed grammars

$$\langle\langle A, i, \alpha_A \rangle\rangle \rightarrow \langle\langle B, i, \alpha_B \rangle\rangle \big|_{n-1} \mathcal{E}_{k'-1}'(C, p_C, \alpha_A) \mid \subseteq \tilde{P},$$

6. If  $A \rightarrow B \big|_{n-1} C \mid$  then,

(a) for all  $\alpha \subseteq \nabla \times N$ ,

$$\langle\langle A, n-1, \alpha \rangle\rangle \rightarrow \langle\langle B, n, \alpha \rangle\rangle \big|_{n-1} \langle\langle C, n, \alpha \rangle\rangle \mid \in \tilde{P},$$

(b) for all  $\alpha \subseteq \nabla \times N$ ,  $1 \leq j \leq n-1$ ,

$$\langle\langle A, j, \alpha \rangle\rangle \rightarrow \langle\langle B, n, \alpha \rangle\rangle \big|_{n-1} \langle\langle C, j, \alpha \rangle\rangle \mid \in \tilde{P},$$

(c) for all  $\alpha_A$  and  $\alpha_B \subseteq \nabla \times N$ ,  $n-1 \geq i \geq 1$  such that

$$\begin{aligned}\alpha_B &= (\alpha_A - X_{n-1} \times N) \cup \{(x_{n-1}^\rho, i) \mid (\rho, i) \in p_C\} \\ p_C &= \{\rho \mid x_{n-1}^\rho \in \nabla\} \times N\end{aligned}$$

where  $\tau_{C, p_C, \alpha_A}$  is as described below,

$$\langle\langle A, i, \alpha_A \rangle\rangle \rightarrow \langle\langle B, i, \alpha_B \rangle\rangle \big|_{n-1} \tau_{C, p_C, \alpha_A} \mid \in \tilde{P},$$

7.  $\tilde{S} \rightarrow \langle\langle S, i, \emptyset \rangle\rangle \in \tilde{P}$  for all  $i \in N$

8. These are all of the elements of  $\tilde{P}$ .

The forest  $\tau_{C,p_i,\alpha_i}$  is constructed in a manner similar to that used to construct the forest  $\tau$  in the proof that the 3-dimensional *OI* languages correspond to the *OI* macro languages. This forest has at the end of each path from  $P_{n-1}^{n-2}$ , that might be derived from  $B$ , a new nonterminal having productions that generate the set of forests at  $(\tau_{C,p_i,\alpha_A})_\rho$ .  $\mathcal{E}val$  evaluates the nonterminal  $C$  for the path  $\rho$  and produces the set of nonterminals at the end of the path.

$\mathcal{E}val : \mathcal{F}_j \times (P_{n-1}^{n-2} \times N) \times \text{powerset}(\nabla \times N) \rightarrow \text{powerset}(\mathcal{F}_{n-1}')$  is defined recursively by:

$$\mathcal{E}val(C, (\lambda, i), \alpha) = \{ \langle H, i, \alpha \rangle \mid C \rightarrow H \in P, \text{ or } C \rightarrow H|_{n-2} G \mid \in P \} \text{ for } C \in \mathcal{F}_{n-2},$$

$$\mathcal{E}val(C, (\lambda, i), \alpha) \supseteq \mathcal{E}val(H, (\lambda, i), \alpha) \text{ for } C \in \mathcal{F}_k, k \leq n-2, \text{ for all } H \in \mathcal{F}_{k+1} \text{ such that } C \rightarrow H|_k G \mid \in P \text{ or } C \rightarrow H \in P,$$

$$\mathcal{E}val(C, (k\rho, i), \alpha) \supseteq \mathcal{E}val(G, (\rho, i), \alpha) \text{ for } C \in \mathcal{F}_k, k \leq n-2, \text{ for all } G \in \mathcal{F}_{k-1} \text{ such that } C \rightarrow H|_k G \mid \in P,$$

$$\mathcal{E}val(C, (j\rho, i), \alpha) \supseteq \mathcal{E}val(H, (j\rho, i), \alpha) \text{ for } C \in \mathcal{F}_k, k > j \geq n-2, \text{ for all } H \in \mathcal{F}_{k+1} \text{ such that } C \rightarrow H|_k G \mid \in P \text{ or } C \rightarrow H \in P,$$

The only *OI* substitutions are at the  $(n-1)$  frontier. For subsequent frontiers the tree is fully expanded so the substitutions are those of a single tree into a single tree. It is for this reason that the  $\mathcal{E}$  function is used to expand the subforests at dimension less than  $n-1$ , and the  $\mathcal{E}val$  function must be used for the subforests at dimension  $n-1$ .

As in the *IO* case, if a structure derived from a nonterminal is copied, then all of that structure is copied. If  $\sigma \in f(\langle A, i, \alpha \rangle)$ , then  $\sigma \in f(A)$  and the set of path

variables derived from  $A$  and not satisfied in the structure derived from  $A$  is a subset of  $\alpha$ . In addition, if  $\sigma \in f(A)$ , then there is a nonterminal such that  $\sigma \in f(\langle\langle A, i, \alpha \rangle\rangle)$ .

□

Baldwin uses the completed grammars to prove a series of theorems which show that the *IO* grammars can be simplified considerably. The simplification involves restricting the paths to a certain subset of  $\{1, \dots, (n-1)\}^*$  and restricting the degree to the terminals. If we are only interested in the  $(n-k)$ th frontier of the language, then all nonterminals of degree  $k$  or greater can be moved to degree  $n$  or  $n-1$  and all paths of dimension  $k+1$  or greater need only be from  $\{(n-1)\}^*$ . These paths are *linear paths* and the grammar is said to be *linear at dimension  $k+1$* . These results also hold for the *OI* languages.

**Theorem 15** *If  $L = f^{n-k}(S)$ , for some  $n$ -dimensional grammar,  $G_n^{k'} = (\Sigma, \mathcal{F}, P, S)_n^{k'}$ , then there is a grammar  $\tilde{G}_n^{k''} = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^{k''}$ , such that  $\tilde{\mathcal{F}}_i = \emptyset$  for all  $n-1 > i \geq k$  and  $L = f^{n-k}(\tilde{S})$ .*

*Proof:* Without loss of generality assume that  $G_n^{k'}$  is a completed grammar for  $L$  and let  $\tau_j$  be the tree which has  $(\tau)_\rho = x_j^\rho$  if  $x_j^\rho$  is in the grammar and  $*$  otherwise. Construct the grammar  $\tilde{G}_n^{k''} = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^{k''}$  as follows:

$$\tilde{\Sigma} = \Sigma \cup \{*\},$$

$$\tilde{\mathcal{F}}_i = \langle \mathcal{F}_i \rangle_{k \leq i \leq n-1} \text{ and } \tilde{\mathcal{F}}_i = \mathcal{F}_i \text{ if } k > i > 0, \text{ or } i = n,$$

$\tilde{P}$  is as indicated in the construction below,

$$\tilde{S} = S.$$

$\tilde{P}$  contains the following productions:

1. If  $A \rightarrow \alpha$ , for  $A \in \mathcal{F}_i$ ,  $k > i > 0$ , and  $i = n$ , then  $\tilde{P}$  contains  $A \rightarrow \alpha$ .

2. If  $A \rightarrow \alpha$ , for  $A \in \mathcal{F}_i$ ,  $n > i \geq k$ , then  $\tilde{P}$  contains

$$A \rightarrow *[_n *[_{n-1} *[_{n-2} \cdots *[_{i+1} \alpha ]_i \tau_i ] \cdots ]_{n-3} \tau_{n-3} ] ]_{n-2} \tau_{n-2} ] ]_{n-1} \tau_{n-1} ].$$

The productions of the completed grammar have the property that  $\alpha$  does not reference any object outside of itself until the  $i$ th frontier. At this point, the enclosing brackets will be gone. So we see that  $L = f^{n-k}(S')$  as desired.

□

**Corollary 2** If  $L = f^{n-1}(S)$ , for some  $n$ -dimensional grammar,  $G_n^{k'} = (\Sigma, \mathcal{F}, P, S)_n^{k'}$ , then there is a grammar,  $\tilde{G}_n^{k''} = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^{k''}$ , such that  $\tilde{\mathcal{F}}_i = \emptyset$  if  $i \neq n$  and  $i \neq n-1$  and  $L = f^{n-1}(\tilde{S})$ .

*Proof:* Replace  $n - k$  by  $n - 1$  in the previous theorem.

□

**Corollary 3** If  $L = f^{n-k}(S)$ , for some  $n$ -dimensional grammar,  $G_n^{k'} = (\Sigma, \mathcal{F}, P, S)_n^{k'}$ , with  $k' > k$ , then there is a grammar,  $\tilde{G}_n^n = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^n$ , such that  $L = f^{n-k}(\tilde{S})$ .

*Proof:* If  $k' > k$ , then the theorem places  $\tilde{S} \in \tilde{\mathcal{F}}_n$  or  $\tilde{\mathcal{F}}_{n-1}$  and  $k'' = n$  or  $n - 1$  in the previous theorem. If  $k'' = n$ , then the corollary is true. If  $k'' = n - 1$ , then  $\tilde{S} \in \tilde{\mathcal{F}}_n$  is new symbol with production  $\tilde{S} \rightarrow [_n S ]$ .

□

**Theorem 16** If  $L = f^{n-k}(S)$  for some  $n$ -dimensional grammar  $G_n^{k'} = (\Sigma, \mathcal{F}, P, S)_n^{k'}$ , then there is a grammar,  $\tilde{G}_n^n = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^n$ , such that

$$f^{n-k}(S) = *[_k *[_{k-1} \cdots *[_{k''} L ] \cdots ] ]$$

where  $k' + 1 \geq k'' \geq 1$ .

*Proof:* The grammar  $\tilde{G}_n^n = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^n$  is as follows:

$$\tilde{\Sigma} = \Sigma \cup \{*\},$$

$$\tilde{\mathcal{F}}_i = \mathcal{F}_i \text{ for } n > i > 0, \text{ and } \tilde{\mathcal{F}}_n = \mathcal{F}_n \cup \{\tilde{S}\},$$

$$\tilde{P} = P \cup \{\tilde{S} \rightarrow *[_n *[_{n-1} \cdots *[_{k''} S ] \cdots ]\},$$

$\tilde{S}$  does not occur elsewhere in the grammar.

Since  $k' + 1 \geq k'' \geq 1$ , the added production is well formed, and at the  $k'$ th frontier the language defined by  $\tilde{G}_n^n$  will be:  $*[_k *[_{k-1} \cdots *[_{k''} L ] \cdots ]$ .

□

As stated previously, if the languages of interest are the string languages only, then we do not need to consider all possible paths for the path variables. We need only consider those paths from  $\{(k-1)\}^*$  for each set of path variables  $X_k$  appearing in the grammar. More formally:

**Definition 27** *The set of linear paths of dimension  $n$  denoted,  $LP_n$ , is defined by*

$$LP_n = \{(n-1)\}^*.$$

*The set of linear path variables of dimension  $n$  denoted,  $LX_n$ , is defined by  $LX_n =$*

$$\{x_n^\rho \mid \rho \in LP_n\}.$$

*A grammar,  $G_n^k$ , is linear at dimension  $i$ , if  $x_j^\rho \in LX_j$  for all  $j > i$ .*

*A grammar,  $G_n^k$ , is a linear grammar, if it is linear at dimension 1.*

□

**Theorem 17** *If  $L = f^{n-k}(S)$ , for  $G_n^{k'} = (\Sigma, \mathcal{F}, P, S)_n^{k'}$ , then there is a linear grammar,  $\tilde{G}_n^{k''} = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^{k''}$ , at dimension  $k+1$  such that  $L = f^{n-k}(\tilde{S})$ .*

*Proof:* Assume that  $G_n^{k'} = (\Sigma, \mathcal{F}, P, S)_n^{k'}$  is a completed grammar for  $L$ . Then  $\tilde{G}_n^{k''} = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^{k''}$  as indicated below is a linear grammar at dimension  $k + 1$ .

$$\tilde{\Sigma} = \Sigma \cup \{*\},$$

$$\tilde{\mathcal{F}} = \mathcal{F},$$

$\tilde{P}$  is as indicated in the construction below,

$$\tilde{S} = S.$$

Order the  $X$ s at each dimension greater than  $k$ . Refer to the first  $x_{k''}^p$  by  $\pi_0^{k''}$ , the second by  $\pi_1^{k''}$ , ..., for  $n \geq k'' \geq k + 1$ . Then  $\tilde{P}$  has the following productions.

1. If  $A \rightarrow B \in P$ , for  $B \in \mathcal{F} \cup \Sigma$ , then  $A \rightarrow B \in \tilde{P}$ .
2. If  $A \rightarrow a[_n \alpha] \in P$ , for  $a \in \Sigma$ , then  $a \in a[_n \alpha] \in \tilde{P}$ .
3. In  $A \rightarrow B[_i \alpha] \in P$ , for  $B \in \mathcal{F}_{i+1}$ ,  $n > i > k$ , and  $B = \langle\langle F, j, \beta \rangle\rangle$  and  $i \geq j > 0$ , then  $A \rightarrow B[_i C_0[_{i-1} C_1[_{i-1} \cdots [_{i-1} C_m] \cdots ] ] ] \in \tilde{P}$  where  $C_l$  is  $*$  if  $(\pi_l^i, j)$  is not in  $\beta$  for any  $j$ , and  $C_l$  is the nonterminal  $(\alpha)_p$  where  $\pi_l^i = x_i^p$  if  $(\pi_l^i, j) \in \beta$ . If  $B = \langle\langle F, j, \beta \rangle\rangle$ , then the structure generated by  $B$  is a singleton after the dimension  $j$  frontier and any set of variables appearing in this structure and not satisfied within it is a subset of  $\beta$ . This means that these  $x_j^p$  will be satisfied in  $\alpha$ , so  $\alpha$  is replaced by the indicated structure which has the nonterminals correctly placed. If  $n \geq j \geq i$ , the structure generated by  $B$  will be a singleton before the  $i$ th frontier. This means that the structure generated by  $\alpha$  will be used, as is, in the frontier and not as a substitution set. Therefore, if  $n \geq j \geq 0$ , then  $A \rightarrow B[_i \alpha] \in \tilde{P}$ .

4. If  $A \rightarrow B[_i \alpha] \in P$ , for  $B \in \mathcal{F}_{i+1}$  and  $k \geq i > 0$ , then  $A \rightarrow B[_i \alpha] \in \tilde{P}$ .
5. If  $A \rightarrow x_i^\rho \in P$ , for  $n > i > k$ , then  $A \rightarrow x_i^{(i-1)^l} \in \tilde{P}$  where  $x_i^\rho = \pi_i^i$ .
6. If  $A \rightarrow x_i^\rho \in P$ , for  $k \geq i \geq 1$ , then  $A \rightarrow x_i^\rho \in \tilde{P}$ .

A simple induction proof shows that  $L = f^{n-k}(\tilde{S})$ .

□

**Corollary 4** *There is a linear grammar for all of the OI string languages.*

*Proof:* Replace  $k$  by 1 in the previous theorem and note that the paths on the variables in  $X_2$  and  $X_1$  are always linear.

□

### 3.2 Closure Properties for the OI Languages

This section concerns the closure properties of the OI  $n$ -dimensional languages. First we show that they are closed under unions, next we show that they are not closed under intersection with another OI-language, but they are closed under intersection with the regular  $n$ -dimensional languages.

**Theorem 18** *Let  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  and  $G_n^{k'} = (\Sigma', \mathcal{F}', P', S')_n^{k'}$  be OI grammars, then there exists an OI grammar  $\tilde{G}_n^{k''} = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^{k''}$  such that  $f(\tilde{S}) = f(S) \cup f(S')$ .*

*Proof:* Let  $\tilde{G}_n^{k''}$  be as follows:

$$\tilde{\Sigma} = \Sigma \cup \Sigma',$$

$$\tilde{\mathcal{F}}_i = \mathcal{F}_i \cup \mathcal{F}'_i, \tilde{\mathcal{F}}_{k''} = \mathcal{F}_{k''} \cup \mathcal{F}'_{k''} \cup \{\tilde{S}\} \text{ where } k'' = \min(k, k'),$$

$$\tilde{P}_i = P_i \cup P'_i \cup \{\tilde{S} \rightarrow S, \tilde{S} \rightarrow S'\}.$$

Clearly this is a grammar for  $f(\tilde{S})$ .

□

It is not possible to define an *OI* grammar for the intersection of two *OI*  $n$ -dimensional languages. Consider the well known context free languages:

$$\{a^n b^m c^m \mid n \geq 1, m \geq 1\}$$

and

$$\{a^m b^m c^n \mid n \geq 1, m \geq 1\}.$$

These languages have *OI* grammars of dimension 2 but their intersection

$$\{a^n b^n c^n \mid n \geq 1\}$$

does not wince it is not a context free language. The next theorem uses the techniques of the completed grammar to show that the *OI*  $n$ -dimensional languages are closed under intersection with the regular  $n$ -dimensional languages. Remember that the regular  $n$ -dimensional languages are those produced using the Baldwin derivation rule  $B \Rightarrow$  as defined in Section 22 on page 71.

**Theorem 19** *If  $L = f^{n-k}(S)$  by an *OI* derivation from the  $n$ -dimensional grammar  $G_n^{k'} = (\Sigma, \mathcal{F}, P, S)_n^{k'}$ , and  $M$  is the language defined using the Baldwin derivation technique from the grammar,  $G_k^k = (\Sigma', \mathcal{F}', P', S')_k^k$ , for  $k \leq n - 1$ , then there is an *OI*  $n$ -dimensional grammar  $\tilde{G}_n^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^k$  such that  $L \cap M = f^{n-k}(S)$  by an *OI* derivation.*

*Proof:* Assume that  $G_n^{k'}$  and  $G_k^k$  are normal form grammars. Then construct  $\tilde{G}_n^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^k$  as follows:

$$\tilde{\Sigma} = (\Sigma' \cap \Sigma) \cup \{a \in \Sigma \mid a|_n C \text{ is a rhs for some } C \in \mathcal{F}_{n-1}\},$$

$$\tilde{\mathcal{F}}_i = \{\langle\langle A, A', l, \alpha \rangle\rangle\} \text{ for all } k \geq i \geq 1, A \in \mathcal{F}_i, A' \in \mathcal{F}'_i, l \in N, \text{ and } \alpha \subseteq \nabla \times \mathcal{F}' \times N,$$

where  $\nabla$  is the set of all  $X$ s actually appearing in  $G_n^k$  and used in one of the frontier operations, and  $N = \{1, \dots, n\}$ ,

$$\tilde{\mathcal{F}}_i = \{\langle\langle A, A', l, \alpha \rangle\rangle\} \text{ for all } n \geq i > k, A \in \mathcal{F}, A' \in \mathcal{F}'_i, l \in N, \text{ and } \alpha \subseteq \nabla \times \mathcal{F}' \times N,$$

$\tilde{P}$  is defined in the construction below,

$\tilde{S}$  does not appear elsewhere in the grammar.

The new grammar has the following properties:

- If  $\langle\langle A, A', j, \alpha \rangle\rangle \in \tilde{\mathcal{F}}$  and  $\beta \in f(\langle\langle A, A', j, \alpha \rangle\rangle)$ , then  $\beta \in f(A)$  and  $A' \xrightarrow{B} f^{n-1-k}(\beta')$  where  $\beta'$  is  $\beta$  with all  $x_i^p$  that reference a subtree outside of  $\beta$  replaced by a structure derived from  $F'$  such that  $(x_i^p, F', j) \in \alpha$ .
- If  $(x_i^p, F', j) \in \alpha$  then if  $x_i^p$  is in the structure derived from  $\langle\langle A, A', j, \alpha \rangle\rangle$  and must be satisfied outside of that structure, it will be replaced by a subtree generated from a nonterminal  $\langle\langle C, F', j, \alpha' \rangle\rangle$  for some  $C$  and  $\alpha'$ .
- The  $j$  represents the level of the first frontier operation for  $\beta'$  that returns a singleton.

$\tilde{P}$  is as follows:

1. If  $A \rightarrow a \in P$  and  $A' \rightarrow a \in P'$  for  $a \in \Sigma \cup \Sigma'$ ,  $A \in \mathcal{F}_n$ , and  $A' \in \mathcal{F}'_k$ , then for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$

$$\langle\langle A, A', n, \alpha \rangle\rangle \rightarrow a \in \tilde{P}.$$

The nonterminal  $\langle\langle A, A', n, \alpha \rangle\rangle$  derives  $a$ ,  $a \in f(A)$  and  $A' \xrightarrow{B} f^{n-1-k}(a)$ . In addition, this nonterminal derives no unsatisfied path variables and  $a$  is a singleton.

2. If  $A \rightarrow x_j^\rho \in P$  and  $A' \rightarrow x_j^\rho \in P'$  for  $A \in \mathcal{F}_j$ , and  $A' \in \mathcal{F}'_j$ , then for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$

$$\langle\langle A, A', j, \alpha \rangle\rangle \rightarrow x_j^\rho \in \tilde{P}.$$

In this case  $j < k$  and the path variable is not involved in any of the frontier operations and is thus treated as a terminal by the frontier operation.

3. If  $A \rightarrow x_j^\rho \in P$  for  $A \in \mathcal{F}_j$ ,  $j \geq k$ , then for all  $F' \in \mathcal{F}'_k$ ,  $l \in N$ , and  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$  such that  $(x_j^\rho, F', l) \in \alpha$

$$\langle\langle A, F', l, \alpha \rangle\rangle \rightarrow x_j^\rho \in \tilde{P}.$$

These path variables will take part in the frontier operation and  $x_j^\rho$  is generated by this nonterminal so  $(x_j^\rho, F', l)$  must be in each  $\alpha$  for each  $F'$ , and  $l$ . If the object replacing  $x_j^\rho$  is first a singleton at dimension  $l$ , then the object derived from this nonterminal will be a singleton at dimension  $l$ .

4. If  $A \rightarrow B \in P$  and  $A' \rightarrow B' \in P'$  for  $A \in \mathcal{F}_i$ ,  $B \in \mathcal{F}_{i+1}$ ,  $A' \in \mathcal{F}'_i$ , and  $B' \in \mathcal{F}'_{i+1}$ , for  $k > i > 0$ , then for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$  and  $l \in N$

$$\langle\langle A, A', l, \alpha \rangle\rangle \rightarrow \langle\langle B, B', l, \alpha \rangle\rangle \in \tilde{P}.$$

The structure derived from  $A$  must behave as the structure derived from  $B$ .

5. If  $A \rightarrow B \in P$  for  $A \in \mathcal{F}_i$ ,  $B \in \mathcal{F}_{i+1}$ , and  $n > i \geq k$ , then for all  $l \in N$ ,  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ , and  $F' \in \mathcal{F}'_k$

$$\langle\langle A, F', l, \alpha \rangle\rangle \in \langle\langle B, F', l, \alpha \rangle\rangle \in \tilde{P}.$$

If a forest derived from  $A'[_{k-1} G_{k-1}][_{k-2} G_{k-2}] \cdots [_1 G_1]$  in the  $k$ -dimensional grammar is the  $(n-1-k)$ -dimensional frontier of a forest derived in the  $n$ -dimensional grammar, then the  $(n-1-k)$ -dimensional frontier of the forest derived from  $A$  must be the same forest as that directly derived from  $F'$ .

6. If  $A \rightarrow a[_n B] \in P$ , then for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ ,  $n > l \geq 1$ , and  $F' \in \mathcal{F}'_k$

$$\langle\langle A, F', l, \alpha \rangle\rangle \rightarrow a[_n \langle\langle B, F', l, \alpha \rangle\rangle] \in \tilde{P}$$

along with

$$\langle\langle A, F', n-1, \alpha \rangle\rangle \in a[_n \langle\langle B, F', n, \alpha \rangle\rangle].$$

If the structure derived from  $\langle\langle B, F', l, \alpha \rangle\rangle$  is not a singleton until after the  $n$ th frontier, then the structure derived from  $\langle\langle A, F', l, \alpha \rangle\rangle$  will be one at the same frontier. If  $\langle\langle B, F', n, \alpha \rangle\rangle$  is a singleton immediately, then  $a[_n \langle\langle B, F', n, \alpha \rangle\rangle]$  will be one at the  $(n-1)$ -frontier.

7. If  $A \rightarrow B[_j C]$  and  $A' \rightarrow B'[_j C'] \in P'$ , for  $k > j \geq 1$ , then for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ , and  $l, m \in N$  such that  $i = \min(j, l, m)$

$$\langle\langle A, A', i, \alpha \rangle\rangle \rightarrow \langle\langle B, B', l, \alpha \rangle\rangle[_j \langle\langle C, C', m, \alpha \rangle\rangle] \in \tilde{P}.$$

If  $j = \min(j, l, m)$ , then the structure derived from  $\langle\langle B, B', l, \alpha \rangle\rangle[_j \langle\langle C, C', m, \alpha \rangle\rangle]$  will be a singleton at the  $j$ th frontier and it will be the value of the structure derived from  $\langle\langle C, C', m, \alpha \rangle\rangle$ . If  $l$  or  $m$  is the minimum, then at the structure derived from  $\langle\langle C, C', m, \alpha \rangle\rangle$  will be used in the  $j$ th frontier so the resulting structure will be a singleton at the  $l$ th frontier. Also since  $k > j \geq l$ , this frontier is not taken during the calculation of  $f^{n-1-k}(\langle\langle A, A', i, \alpha \rangle\rangle)$ .

8. If  $A \rightarrow B[_{n-1} C] \in P$  and  $A' \rightarrow a[_{n-1} C'] \in P'$ , then  $k = n - 1$  and

(a) If  $B \rightarrow a \in P$  for  $a \in \Sigma' \cap \Sigma$ , then for  $n - 1 > l \geq 1$

$$\langle\langle A, A', l, \alpha \rangle\rangle \rightarrow a[_{n-1} \langle\langle C, C', l, \alpha \rangle\rangle] \in \tilde{P}$$

and

$$\langle\langle A, A', n - 1, \alpha \rangle\rangle \rightarrow a[_{n-1} \langle\langle C, C', n, \alpha \rangle\rangle] \in \tilde{P}.$$

If  $B \rightarrow a$  then the structure derived from  $A$  and  $A'$  will be in the both languages if the structures derived from  $a[_{n-1} C]$  and  $a[_{n-1} C']$  are.

(b) if  $B \rightarrow b[_n G]$ , then for all  $n - 1 \geq l \geq 1$ ,  $\alpha_A$  and  $\alpha_B \subseteq \nabla \times \mathcal{F}' \times N$ , and  $p_C \subseteq P_{n-1}^{n-2} \times \mathcal{F}' \times N$  such that

$$\begin{aligned} \alpha_B &= (\alpha_A - (X_{n-1} \times \mathcal{F}' \times N)) \cup \{(x_{n-1}^\rho, C', i) \mid (\rho, C, i) \in p_C\} \\ p_C &\subseteq \{\rho \mid x_{n-1}^\rho \in \nabla\} \times \mathcal{F}' \times N \end{aligned}$$

using the *Eval* function defined below to construct  $\tau_{C, p_C, \alpha_A}$

$$\langle\langle A, A', i, \alpha_A \rangle\rangle \rightarrow \langle\langle B, A', i, \alpha_B \rangle\rangle[_{n-1} \tau_{C, p_C, \alpha_A}] \subseteq \tilde{P}.$$

If  $B \rightarrow b[_n G]$  then the structure generated by  $B$  will use the structure generated by  $C$  for substitution. This structure must be the same as that derived from  $A'$ . The unresolved  $x_{n-1}^\rho$  in this structure will be replaced by structures derived from  $F' \in \mathcal{F}'$  such that  $(x_{n-1}^\rho, F', j) \in \alpha$  and  $\langle\langle C, F', j, \alpha \rangle\rangle$  is generated by the nonterminal in that position in  $\tau_{C, p_C, \alpha_A}$ .

9. If  $a \rightarrow B[_j C] \in P$ ,  $n - 1 > j > k$ , then

- (a) for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ ,  $F' \in \mathcal{F}'_k$ ,  $n > i > j$ , and  $n > l \geq j$

$$\langle\langle A, F', j, \alpha \rangle\rangle \rightarrow \langle\langle B, F', i, \alpha \rangle\rangle[_j \langle\langle C, F', l, \alpha \rangle\rangle] \in \tilde{P}.$$

If both  $\langle\langle B, F', i, \alpha \rangle\rangle$  and  $\langle\langle C, F', l, \alpha \rangle\rangle$  derive structures which are singletons before the  $j$ th frontier, then the entire structure will be a singleton at the  $j$ th frontier.

- (b) for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ ,  $F' \in \mathcal{F}'_k$ ,  $n > i > j$ , and  $j \geq l \geq 1$

$$\langle\langle A, F', l, \alpha \rangle\rangle \rightarrow \langle\langle B, F', i, \alpha \rangle\rangle[_j \langle\langle C, F', l, \alpha \rangle\rangle] \in \tilde{P}.$$

If  $\langle\langle C, F', l, \alpha \rangle\rangle$  derives a structure which is not a singletons until after the  $j$ th frontier and  $\langle\langle B, F', i, \alpha \rangle\rangle$  is one before the  $j$ th frontier, then the entire structure will be a singleton at the  $l$ th frontier.

- (c) for all  $\alpha_A$  and  $\alpha_B \subseteq \nabla \times \mathcal{F}' \times N$ ,  $F' \in \mathcal{F}'_k$ ,  $j \geq i \geq 1$ , and  $p_C \subseteq P_j^{j-1} \times \mathcal{F}' \times N$  such that

$$\begin{aligned} \alpha_B &= (\alpha_A - (X_j \times \mathcal{F}' \times N)) \cup \{(x_j^\rho, C', i) \mid (\rho, C, i) \in p_C\} \\ p_C &\subseteq \{\rho \mid x_j^\rho \in \nabla\} \times \mathcal{F}' \times N \end{aligned}$$

and using the  $\mathcal{E}$  function defined below.

$$\langle\langle A, F', i, \alpha_A \rangle\rangle \rightarrow \langle\langle B, F', i, \alpha_B \rangle\rangle[_j \mathcal{E}_{j-1}(C, p_C, \alpha_A)] \subseteq \tilde{P}.$$

These structures take part in the frontiering after the initial derivation frontier. So, any substitutions will be *IO* and thus the  $\mathcal{E}$  function produces a set of trees each of which is a tree with all paths from  $P_j^{j-1}$ , that might be evaluated in the structure derived from  $C$ , expanded.

10. If  $A \rightarrow B[n-1, C] \in P$ ,  $n-1 > k$ , then

(a) for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ ,  $F' \in \mathcal{F}'_k$

$$\langle\langle A, F', n-1, \alpha \rangle\rangle \rightarrow \langle\langle B, F', n, \alpha \rangle\rangle[n-1, \langle\langle C, F', n, \alpha \rangle\rangle] \in \tilde{P}.$$

(b) for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ ,  $F' \in \mathcal{F}'_k$  and  $1 \leq j < n-1$

$$\langle\langle A, F', j, \alpha \rangle\rangle \rightarrow \langle\langle B, F', n, \alpha \rangle\rangle[n-1, \langle\langle C, F', j, \alpha \rangle\rangle] \in \tilde{P}.$$

(c) for all  $\alpha_A$  and  $\alpha_B \subseteq \nabla \times \mathcal{F}' \times N$ ,  $F' \in \mathcal{F}'_k$ ,  $n-1 > j \geq 1$ , and  $p_C \subseteq P_{n-1}^{n-2} \times \mathcal{F}' \times N$  such that

$$\alpha_B = (\alpha_A - (X_{n-1} \times \mathcal{F}' \times N)) \cup \{(x_{n-1}^\rho, C', i) \mid (\rho, C, i) \in p_C\}$$

$$p_C \subseteq \{\rho \mid x_{n-1}^\rho \in \nabla\} \times \mathcal{F}' \times N$$

and using the  $\mathcal{Eval}$  function defined below to construct  $\tau_{C, p_C, \alpha_A}$ .

$$\langle\langle A, F', i, \alpha_A \rangle\rangle \rightarrow \langle\langle B, F', i, \alpha_B \rangle\rangle[j, \tau_{C, p_C, \alpha_A}] \in \tilde{P}.$$

These structures take part in the initial derivation frontier. So, any substitutions will be  $OI$  and thus the  $\mathcal{Eval}$  function produces a set of trees each of which is a tree with all paths from  $P_j^{j-1}$ , that might be evaluated in the structure derived from  $C$ , expanded.

11.  $\tilde{S} \rightarrow \langle\langle S, S', i, \emptyset \rangle\rangle \in \tilde{P}$  for all  $i \in N$ .

12. These are all of the productions in  $\tilde{P}$ .

The purpose of the  $\mathcal{E}$  function is to evaluate the nonterminal for all the paths in the set  $p$ . The initial calls to  $\mathcal{E}$  are generated by step 9-(c) with  $p$  containing

those paths which might be evaluated by some frontier in the structure derived from  $C$ . This means that if  $B$  in  $B[k, C]$  derives a path variable from  $X_k$  that must be evaluated in the structure derived from  $C$ , then  $\mathcal{E}(C, p_C, \alpha_A)$  will have nonterminals from  $\mathcal{F}'_k$  in those positions. Thus, no subpart of a structure derived from a nonterminal will not be copied.  $\mathcal{E}$  is a set of finite functions  $\{\mathcal{E}_{k''}\}_{n \geq k'' > 0}$  where for all  $A \in \mathcal{F}_{k''}$ ,  $p \subseteq \{\rho \mid x_j^\rho \in \nabla\} \times \mathcal{F}' \times N$ , and  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ ,  $\mathcal{E}_{k''}(A, p, \alpha)$  is the smallest set such that

1. If  $A \rightarrow B[k'', C] \in P$ , then for all  $\alpha, p_B$  and  $p_C$  such that

$$\begin{aligned} p &= p_B \cup \{(k''\rho, C, l) \mid (\rho, C, l) \in p_C\} \\ p_B &\subseteq P_k^{k''+1} \times \mathcal{F}' \times N \\ p_C &\subseteq P_k^{k''-1} \times \mathcal{F}' \times N \end{aligned}$$

such that if  $(\rho, C, l) \in p_B \cup p_C$ , then  $\rho$  is the suffix of a path in  $\nabla \cap X_k$

$$\mathcal{E}_{k''+1}(B, p_B, \alpha)[k'', \mathcal{E}_{k''-1}(C, p_C, \alpha)] \subseteq \mathcal{E}_{k''}(A, p, \alpha).$$

2. If  $A \rightarrow B \in P$  where  $A \in \mathcal{F}_{k''}$ ,  $B \in \mathcal{F}_{k''+1}$ , then for all  $p \subseteq P_k^{k''+1} \times \mathcal{F}' \times N$  such that if  $(\rho, C, l) \in p$ , then  $\rho$  is the suffix of a path in  $\nabla \cap X_k$  and  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$

$$\mathcal{E}_{k''+1}(B, p, \alpha) \subseteq \mathcal{E}_{k''}(A, p, \alpha).$$

3. If  $C \in \mathcal{F}_{k''}$ , then for all  $\alpha \subseteq \nabla \times \mathcal{F}' \times N$ ,  $F' \mathcal{F}'_l$  when  $l \leq k$ , or  $F' \in \mathcal{F}'_k$  when  $l > k$ , and  $i \in N$ ,

$$\mathcal{E}_{k''}(C, \emptyset, \alpha) = \{\langle\langle C, F', i, \alpha \rangle\rangle\}$$

and

$$\mathcal{E}_{k''}(C, \{(\lambda, F', i)\}, \alpha) = \{\langle\langle C, F', i, \alpha \rangle\rangle\}$$

The forest,  $\tau_{C,p_i,\alpha_A}$ , is constructed in a manner similar to that used in the completed grammar. This forest has, at the end of each path from  $P_{n-1}^{n-2}$  that might be unresolved in the structure derived from  $B$ , a new nonterminal having productions that generate the set of forests at  $(\tau_{C,p_i,\alpha_A})_\rho$ .  $\mathcal{Eval}$  evaluates the nonterminal  $C$  for the path  $\rho$  and produces the set of nonterminals at the end of the path.

$\mathcal{Eval} : \mathcal{F}_j \times (P_{n-1}^{n-2} \times \mathcal{F}' \times N) \times \text{powerset}(\nabla \times \mathcal{F}' \times N) \rightarrow \text{powerset}(\mathcal{F}'_{n-1})$  is defined recursively by:

$$\mathcal{Eval}(C, (\lambda, F', i), \alpha) = \{ \langle \langle H, F', i, \alpha \rangle \rangle \mid C \rightarrow H \in P, \text{ or } C \rightarrow H[_{n-2} G] \in P \} \text{ for } C \in \mathcal{F}_{n-2},$$

$$\mathcal{Eval}(C, (\lambda, F', i), \alpha) \supseteq \mathcal{Eval}(H, (\lambda, F', i), \alpha) \text{ for } C \in \mathcal{F}_j, j \leq n-2, \text{ for all } H \in \mathcal{F}_{j+1} \text{ such that } C \rightarrow H[_j G] \in P \text{ or } C \rightarrow H \in P,$$

$$\mathcal{Eval}(C, (k\rho, F', i), \alpha) \supseteq \mathcal{Eval}(G, (\rho, F', i), \alpha) \text{ for } C \in \mathcal{F}_j, j \leq n-2, \text{ for all } G \in \mathcal{F}_{j-1} \text{ such that } C \rightarrow H[_j G] \in P,$$

$$\mathcal{Eval}(C, (k\rho, i), \alpha) \supseteq \mathcal{Eval}(H, (k\rho, i), \alpha) \text{ for } C \in \mathcal{F}_j, j > k \geq n-2, \text{ for all } H \in \mathcal{F}_{j+1} \text{ such that } C \rightarrow H[_j G] \in P \text{ or } C \rightarrow H \in P,$$

□

### 3.3 Removal of Dead Symbols from the *OI* Grammars

Another topic of interest is the removal of dead symbols. Baldwin has already done this for the *IO* languages. In fact, the completed *IO* grammars which use the modified  $\mathcal{E}$  function have the additional property that  $f^{n-k}(\alpha)$  is defined for all  $\alpha$  in the language.

A symbol  $f \in \mathcal{F}_k$  is dead if it cannot generate a tree in  $H_{n-1}^k$ . Consider the grammar with productions  $S \rightarrow *[_3 F[_2 D ] ]$ , and  $F \rightarrow *[_3 a ]$ . This grammar defines the *OI* language  $\{a\}$ .  $D$  is a dead symbol but  $F$  is not. In an *OI* grammar, a dead symbol is only important if it must be used in a derivation. The definition of a dead symbol in an *OI* grammar is an extension of that of Fischer [27].

**Definition 28** Let  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  be an *OI* grammar. Let  $\nabla_{n-1}$  be the set of path variables from  $X_{n-1}$  that actually appear in  $G_{n-1}^k$ . For each  $\eta \in H_{n-1}^j$ , let  $\nabla_\eta = \{x_{n-1}^\rho \mid x_{n-1}^\rho \text{ appears in } \eta\}$ . A symbol  $F_j \in \mathcal{F}_j$  is dead relative to a set  $\nabla \in \nabla_{n-1}$ , if for no  $\eta \in H_{n-1}^j$  with  $\nabla_\eta \subseteq \nabla$  is  $\eta \in f(F)$ .

$F$  is dead if it is dead relative to  $\nabla_{n-1}$ .

□

**Lemma 9** Let  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  be an *OI*  $n$ -dimensional grammar. It is decidable for each  $F \in \mathcal{F}_j$  and each  $\nabla \in \nabla_{n-1}$  whether or not  $F$  is dead relative to  $\nabla$ .

*Proof:* Given  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , construct the grammar  $\tilde{G}_{n-1}^{n-1} = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_{n-1}^{n-1}$  where

$\tilde{\Sigma} = \Sigma \cup \{*, \delta\}$  where  $*$  and  $\delta$  are symbols not appearing elsewhere in the grammar,

$\tilde{\mathcal{F}}_k = \mathcal{F}_k$ ,  $\tilde{\mathcal{F}}_{n-1} = \mathcal{F}_{n-1} \cup \{\tilde{S}, \tilde{D}\}$ , where  $\tilde{S}$  and  $\tilde{D}$  are symbols not appearing elsewhere in the grammar,

$\tilde{P}_k = P_k$ ,  $\tilde{P}_{n-1} = P_{n-1} \cup \{\tilde{S} \rightarrow *[_n *[_{n-1} \cdots *[_{j+1} F ] \cdots ] ][_{n-1} \tau ]\}$  where  $\tau$  is a tree

in  $H_{n-1}^{n-2}$  such that for all  $\rho \in \nabla$

$$(\tau)_\rho = \begin{cases} \delta & \text{if } x_{n-1}^\rho \in \nabla \\ \tilde{D} & \text{if } x_{n-1}^\rho \notin \nabla. \end{cases}$$

$F$  is dead relative to  $\nabla$  if and only if  $f(\tilde{S}) = \emptyset$ . This is a decidable property since  $\emptyset$  is a regular set. Suppose  $f(\tilde{S}) \neq \emptyset$ , then there exists  $t \in H_{n-1}^{n-1}$  such that

$$\begin{aligned} t &\in f(\tilde{S}) \\ &= f(*[{}_n * [{}_{n-1} \cdots * [{}_{j+1} F] \cdots] \mid {}_n \tau]) \\ &= * [{}_{n-1} \cdots * [{}_{j+1} f(F)] \cdots] \xrightarrow{OI} f(\tau). \end{aligned}$$

So there must be  $\eta \in H_{n-1}^j$  such that  $\eta \in f(F)$  and  $t = * [{}_{n-1} \cdots * [{}_{j+1} \eta] \cdots] \xrightarrow{OI} \tau$ . For those  $x_{n-1}^\rho$  appearing in  $\eta$ , it must be that  $(\tau)_\rho = \delta$  since no derivations are possible from  $\tilde{D}$ . This means that  $\nabla_\eta \subseteq \nabla$ , so  $F$  is not dead relative to  $\nabla$ .

Conversely, if  $F$  is not dead relative to  $\nabla$ , then  $\eta \in f(F)$  for some  $\eta \in H_{n-1}^j$ ,  $\nabla_\eta \subseteq \nabla$ . Clearly,

$$* [{}_{n-1} \cdots * [{}_{j+1} \eta] \cdots] \in f(* [{}_n * [{}_{n-1} \cdots * [{}_{j+1} F] \cdots] \mid {}_n \tau])$$

so

$$* [{}_{n-1} \cdots * [{}_{j+1} \eta] \cdots] \xrightarrow{IO} \tau \in f(\tilde{S})$$

since  $(\tau)_\rho = \delta$  for each  $x_{n-1}^\rho \in \nabla$  and  $f(\tilde{S}) \neq \emptyset$ .

□

The next lemma uses this lemma to make  $OI$   $n$ -dimensional grammars with no dead symbols.

**Lemma 10** *If  $L = f(S)$  for the OI  $n$ -dimensional grammar  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$ , then there exists an OI  $n$ -dimensional grammar  $\tilde{G}_n^k = (\tilde{\Sigma}, \tilde{\mathcal{F}}, \tilde{P}, \tilde{S})_n^k$  with no dead symbols such that  $f(\tilde{S}) = f(S)$ .*

*Proof:* Assume that  $G_n^k$  is a completed grammar. Construct  $\tilde{G}_n^k$  as follows:

$\tilde{\Sigma} = \Sigma \cup \{\delta\}$  where  $\delta$  is a symbol not appearing elsewhere in the grammar,

$\tilde{\mathcal{F}}_i = \{\langle\langle A, \alpha \rangle\rangle \mid A \in \mathcal{F}_i \text{ and } \alpha \subseteq \nabla_{n-1}\}$  that are added during the construction of  $\tilde{P}$  and  $\nabla_{n-1}$  is the set of path variables from  $X_{n-1}$  that actually appear in  $G_n^k$ ,

$\tilde{P}$  is as described below

$\tilde{S} = \langle\langle S, \emptyset \rangle\rangle$ .

$\tilde{P}$  is constructed as follows:

1. If  $A \rightarrow a \in P$  for  $A \in \mathcal{F}_n$ ,  $a \in \Sigma$ , then for all  $\alpha \subseteq \nabla_{n-1}$

$$\langle\langle A, \alpha \rangle\rangle \rightarrow a \in \tilde{P}.$$

2. If  $A \rightarrow x_i^\rho \in P$  for  $A \in \mathcal{F}_i$ ,  $i < n - 1$ , then for all  $\alpha \subseteq \nabla_{n-1}$

$$\langle\langle A, \alpha \rangle\rangle \rightarrow x_i^\rho \in \tilde{P}.$$

3. If  $A \rightarrow x_{n-1}^\rho \in P$  for  $A \in \mathcal{F}_{n-1}$ , then for all  $\alpha \subseteq \nabla_{n-1}$  such that  $x_{n-1}^\rho \in \alpha$

$$\langle\langle A, \alpha \rangle\rangle \rightarrow x_{n-1}^\rho \in \tilde{P}.$$

4. If  $A \rightarrow B \in P$  for  $A \in \mathcal{F}_i$ ,  $B \in \mathcal{F}_{i+1}$ , then for all  $\alpha \subseteq \nabla_{n-1}$  such that  $B$  is not dead relative to  $\alpha$

$$\langle\langle A, \alpha \rangle\rangle \rightarrow \langle\langle B, \alpha \rangle\rangle \in \tilde{P}.$$

5. If  $A \rightarrow a[_n C] \in P$  for  $A \in \mathcal{F}_n$ ,  $C \in \mathcal{F}_{n-1}$ ,  $a \in \Sigma$ , then for all  $\alpha \subseteq \nabla_{n-1}$  such that  $C$  is not dead relative to  $\alpha$

$$\langle\langle A, \alpha \rangle\rangle \rightarrow a[_n \langle\langle C, \alpha \rangle\rangle] \in \tilde{P}.$$

6. If  $A \rightarrow a[_{n-1} C] \in P$  for  $A \in \mathcal{F}_{n-1}$ ,  $C \in \mathcal{F}_{n-2}$ ,  $a \in \Sigma$ , then for all  $\alpha \subseteq \nabla_{n-1}$  such that  $C$  is not dead relative to  $\alpha$

$$\langle\langle A, \alpha \rangle\rangle \rightarrow a[_{n-1} \langle\langle C, \alpha \rangle\rangle] \in \tilde{P}.$$

7. If  $A \rightarrow B[_{n-1} \tau] \in P$  for  $A \in \mathcal{F}_{n-1}$ ,  $B \in \mathcal{F}_n$ , then for all  $\alpha_A$  and  $\alpha_B \subseteq \nabla_{n-1}$  such that  $B$  is not dead relative to  $\alpha_B$ ,  $(\tau)_\rho$  is not dead relative to  $\alpha_A$  for all  $\rho \in \nabla_{n-1}$  such that  $(\tau)_\rho$  is defined and

$$(\tau')_\rho = \begin{cases} \delta & \text{if } \rho \in \alpha_B \\ \langle\langle (\tau)_\rho, \alpha_A \rangle\rangle & \text{if } \rho \in \alpha_B \end{cases}$$

$$\langle\langle A, \alpha \rangle\rangle \rightarrow \langle\langle B, \alpha_B \rangle\rangle[_{n-1} \tau'] \in \tilde{P}.$$

8. If  $A \rightarrow B[_j \tau] \in P$  for  $A \in \mathcal{F}_j$ ,  $B \in \mathcal{F}_{j+1}$ ,  $1 \leq j \leq n-2$ , then for all  $\alpha_A$ ,  $\alpha_B$  and  $\alpha_\rho \subseteq \nabla_{n-1}$  such that  $(\tau)_\rho$  is not dead relative to  $\alpha_\rho$ ,  $B$  is not dead relative to  $\alpha_B$ , and  $\alpha_A = \alpha_B \cup \{\alpha_\rho \mid (\tau)_\rho \in \mathcal{F}\}$  and  $\tau'$  such that  $(\tau')_\rho = \langle\langle (\tau)_\rho, \alpha_\rho \rangle\rangle$

$$\langle\langle A, \alpha_A \rangle\rangle \rightarrow \langle\langle B, \alpha_B \rangle\rangle[_{j, \tau'}] \in \tilde{P}.$$

We now show by induction on the number of derivation steps that  $f(S) = f(\tilde{S})$ . We first show that if  $\sigma \in f(A)$ , then  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$  for  $\alpha \supseteq \nabla_\sigma$ , then we show the converse. From this we can conclude that  $\sigma \in f(S)$  if and only if  $\sigma \in f(\langle\langle S, \emptyset \rangle\rangle)$ .

*Basis:* If  $\sigma \in f(A)$  after 1 derivation step, then  $A \rightarrow \sigma \in P$  and  $\langle\langle A, \alpha \rangle\rangle \rightarrow \sigma \in \tilde{P}$ . If  $\sigma \in \Sigma$ , or  $\sigma = x_i^\rho$ ,  $1 \leq i \leq n-2$ , then  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$  for all  $\alpha \subseteq \nabla_{n-1}$  since  $\nabla_\sigma = \emptyset$ . If  $\sigma = x_{n-1}^\rho$ , then  $\alpha$  is such that  $x_{n-1}^\rho \in \alpha$  so  $\alpha \supseteq \nabla_\sigma$  and again  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$ .

*Induction Step:* assume that  $\sigma \in f(A)$  after  $m$  or less derivation steps implies that  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$  for some  $\alpha \subseteq \nabla_{n-1}$  such that  $\alpha \supseteq \nabla_\sigma$ . Let  $\sigma \in f(A)$  after  $m+1$  derivation steps.

1. If  $A \rightarrow B$  is used, then  $\sigma \in f(B)$  after  $m$  derivation steps so  $\sigma \in f(\langle\langle B, \alpha \rangle\rangle)$  for  $\alpha \supseteq \nabla_\sigma$  and  $B$  is not dead relative to  $\alpha$  so  $\langle\langle A, \alpha \rangle\rangle \rightarrow \langle\langle B, \alpha \rangle\rangle \in \tilde{P}$  and  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$ .
2. If  $A \rightarrow a[_n C]$  is used, then  $\sigma \in f(C)$  after  $m$  derivation steps. So  $\sigma \in f(\langle\langle C, \alpha \rangle\rangle)$ , for  $\alpha \supseteq \nabla_\sigma$  and  $C$  is not dead relative to  $\alpha$ , so  $\langle\langle A, \alpha \rangle\rangle \rightarrow a[_n \langle\langle C, \alpha \rangle\rangle] \in \tilde{P}$  and  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$ .
3. If  $A \rightarrow a[_{n-1} C]$  is used, then  $\sigma \in a[_{n-1} f(C)]$  after  $m$  derivation steps. So  $\sigma \in a[_{n-1} f(\langle\langle C, \alpha \rangle\rangle)]$ , for  $\alpha \supseteq \nabla_\sigma$  and  $C$  is not dead relative to  $\alpha$ , so  $\langle\langle A, \alpha \rangle\rangle \rightarrow a[_{n-1} \langle\langle C, \alpha \rangle\rangle] \in \tilde{P}$  and  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$ .
4. If  $A \rightarrow B[_{n-1} \tau]$  is used, then  $\sigma \in f(B) \xleftrightarrow{IO} f(\tau)$  so  $\sigma = \sigma_1 \xleftrightarrow{IO} f(\tau)$  where  $\sigma_1 \in f(B)$  after less than  $m$  derivation steps and  $\sigma \in f(\langle\langle B, \alpha_B \rangle\rangle)$  for  $\alpha_B = \nabla_{\sigma_1}$ . In addition  $B$  is not dead relative to  $\alpha_B$ . For all  $\rho \in \alpha_B$ ,  $(\tau)_\rho \in \mathcal{F}$  and  $x_{n-1}^\rho$  in  $\sigma_1$  is replaced by  $\sigma_\rho \in f((\tau)_\rho)$  so  $\sigma_\rho \in f(\langle\langle (\tau)_\rho, \alpha_A \rangle\rangle)$  for  $\alpha_A = \bigcup_{\{\rho | x_{n-1}^\rho \in \alpha_B\}} \nabla_{\sigma_\rho}$  and  $(\tau)_\rho$  is not dead relative to  $\alpha_A$ . So  $\sigma \in f(\langle\langle B, \alpha_B \rangle\rangle) \xleftrightarrow{OI} f(\tau')$  where  $\tau'$  is as the in the construction and  $\langle\langle A, \alpha_A \rangle\rangle \rightarrow \langle\langle A, \alpha_B \rangle\rangle[_{n-1} \tau'] \in \tilde{P}$  so  $\sigma \in f(\langle\langle A, \alpha_A \rangle\rangle)$ .
5. If  $A \rightarrow B[_j \tau]$  is used, then  $\sigma \in f(B)[_j f(\tau)]$  so  $\sigma = \sigma_1[_j \sigma_2]$  where  $\sigma_1 \in f(B)$

after less than  $m$  derivation steps and  $\sigma_2 \in f(\tau)$  after less than  $m$  derivation steps. Thus  $\sigma_1 \in f(\langle\langle B, \alpha_B \rangle\rangle)$ ,  $\alpha_B = \nabla_{\sigma_1}$  and  $B$  is not dead relative to  $\alpha_B$ . By the construction of the completed grammar,  $\tau$  contains no path variables. Let  $\sigma_\rho \in (f(\tau))_\rho = (\sigma_2)_\rho$ . Then  $\sigma_\rho \in f(\langle\langle (\tau)_\rho, \nabla_{\sigma_\rho} \rangle\rangle)$ , and  $(\tau)_\rho$  is not dead relative to  $\nabla_{\sigma_\rho}$ , so  $\langle\langle A, \alpha_A \rangle\rangle \rightarrow \langle\langle B, \alpha_B \rangle\rangle_{[, \in ]\tilde{P}}$  where  $\alpha_A = \alpha_B \cup \{ \nabla_{\sigma_\rho} \mid (\tau)_\rho \in \mathcal{F} \}$  means that  $\sigma \in f(\langle\langle A, \alpha_A \rangle\rangle)$ .

We now show by induction on the number of derivation steps that  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$  implies  $\sigma \in f(A)$ .

*Basis:* Suppose  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$  after 1 derivation step. Then  $\langle\langle A, \alpha \rangle\rangle \rightarrow \sigma \in \tilde{P}$  so  $A \rightarrow \sigma \in P$  and  $\sigma \in f(A)$ .

*Induction Step:* Assume  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$  after  $m$  or less derivation steps implies that  $\sigma \in f(A)$ . Let  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$  after  $m + 1$  steps.

1. If  $\langle\langle A, \alpha \rangle\rangle \rightarrow \langle\langle B, \alpha \rangle\rangle$  is used, then  $\sigma \in f(\langle\langle B, \alpha \rangle\rangle)$  after  $m$  derivation steps so  $\sigma \in f(B)$  and  $A \rightarrow B \in P$  so  $\sigma \in f(A)$ .
2. If  $\langle\langle A, \alpha \rangle\rangle \rightarrow a_{[n} \langle\langle C, \alpha \rangle\rangle ]$  is used, then  $\sigma \in f(\langle\langle C, \alpha \rangle\rangle)$  after  $m$  derivation steps so  $\sigma \in f(C)$  and  $A \rightarrow a_{[n} C ] \in P$  so  $\sigma \in f(A)$ .
3. If  $\langle\langle A, \alpha \rangle\rangle \rightarrow a_{[n-1} \langle\langle C, \alpha \rangle\rangle ]$  is used, then  $\sigma \in a_{[n-1} f(\langle\langle C, \alpha \rangle\rangle) ]$  after  $m$  derivation steps so  $\sigma \in a_{[n-1} f(C) ]$  and  $A \rightarrow a_{[n-1} C ] \in P$  so  $\sigma \in f(A)$ .
4. If  $\langle\langle A, \alpha_A \rangle\rangle \rightarrow \langle\langle B, \alpha_B \rangle\rangle_{[n-1} \tau' ]$  is used, then  $\sigma \in f(\langle\langle B, \alpha_B \rangle\rangle) \stackrel{\text{OI}}{\Leftarrow} f(\tau')$  so  $\sigma = \sigma_1 \stackrel{\text{OI}}{\Leftarrow} f(\tau')$  and  $\sigma_1 \in f(B)$ . In addition  $(\tau')_{f\rho} = \langle\langle (\tau)_\rho, \alpha_A \rangle\rangle$  for some  $\tau$  such that  $A \rightarrow B_{[n-1} \tau ] \in P$  and  $\sigma_\rho \in f(\langle\langle (\tau)_\rho, \alpha_A \rangle\rangle)$  so  $\sigma_\rho \in f((\tau)_\rho)$  and  $\sigma \in f(B_{[n-1} \tau ]) \subseteq f(A)$  as desired.

This proves that  $\sigma \in f(\langle\langle A, \alpha \rangle\rangle)$  implies that  $\sigma \in f(A)$ .

Now we must show that  $\tilde{G}_n^k$  contains no dead symbols. Suppose  $\tilde{G}_n^k$  contains dead symbols, then there must be a productions which uses a dead symbol.

1. It cannot be any of the productions of the form  $\langle\langle A, \alpha \rangle\rangle \rightarrow \sigma$ , for  $\sigma \in \Sigma$  or  $X_i$ ,  $1 \leq i \leq n-1$ .
2. Suppose it is one of  $\langle\langle A, \alpha \rangle\rangle \rightarrow \langle\langle B, \alpha \rangle\rangle$ ,  $\langle\langle A, \alpha \rangle\rangle \rightarrow a[_n \langle\langle C, \alpha \rangle\rangle ]$ ,  $\langle\langle A, \alpha \rangle\rangle \rightarrow a[_{n-1} \langle\langle C, \alpha \rangle\rangle ]$ . Let us consider the rule  $\langle\langle A, \alpha \rangle\rangle \rightarrow \langle\langle B, \alpha \rangle\rangle$ . This rule comes from  $A \rightarrow B$  and  $B$  is not dead relative to  $\alpha$ , so there is some  $\sigma$  such that  $\nabla_\sigma \subseteq \alpha$  and  $\sigma \in f(B)$  so  $\sigma \in f(\langle\langle B, \alpha \rangle\rangle)$  which contradicts  $\langle\langle B, \alpha \rangle\rangle$  a dead symbol. A similar argument applies to the other productions in this list.
3. Suppose it is  $\langle\langle A, \alpha_A \rangle\rangle \rightarrow \langle\langle B, \alpha_B \rangle\rangle[_j \tau']$  where  $(\tau')_\rho = \langle\langle \tau \rangle_\rho, \alpha_\rho \rangle\rangle$  for all  $\sigma_\rho \subseteq \nabla_{n-1}$ ,  $\alpha_B \subseteq \nabla_{n-1}$ ,  $1 \leq j \leq n-2$ ,  $(\tau)_\rho$  is not dead relative to  $\alpha_\rho$ ,  $B$  is not dead relative to  $\alpha_B$  and  $\alpha_A = \alpha_B \cup \{\alpha_\rho \mid (\tau)_\rho \in \mathcal{F}\}$ . Then as above each of the nonterminals derives something so this production cannot contain the dead symbol.
4. Suppose it is  $\langle\langle A, \alpha_A \rangle\rangle \rightarrow \langle\langle B, \alpha_B \rangle\rangle[_{n-1} \tau']$  where  $(\tau')_\rho = \langle\langle \tau \rangle_\rho, \alpha_A \rangle\rangle$  and  $\alpha_A$  and  $\alpha_B \subseteq \nabla_{n-1}$  such that  $(\tau)_\rho$  is not dead relative to  $\alpha_A$ , and  $B$  is not dead relative to  $\alpha_B$  for all  $\rho \in \alpha_B$ . Then as above each of the nonterminals derives something so this production cannot contain the dead symbol.

□

We would like to know if the string languages of Baldwin-Schoenberger are contained in the string languages of Engelfriet-Schmidt. A simple answer to this question

is no, since the empty string, represented by  $x_1^\lambda$ , can be generated by the Baldwin-Schoenberger grammars but not by the Engelfriet-Schmidt grammars. This is why we have shown that the 3-dimensional grammars have string languages which are the same as the macro languages.

An even more elusive problem is that of typing or arity. In a production of the type  $A \rightarrow a$ , it is not clear what type the terminal  $a$  should have since the Baldwin-Schoenberger languages do not restrict the frontier on which the terminal may appear. The symbol  $a$  may appear as an internal node or as a leaf. In addition any nonterminal  $a$  appearing as an internal node may have a nondeterminate number of children. These two difficulties make it unclear at present how one might reconcile these two definitions completely.

### 3.4 A Further Extension of the $n$ -Dimensional Grammars

The language  $L = \{(a^- b a^-)^{2^{2^m}} \mid m \geq 1\}$  is thought to not be the string language of any  $OI$   $n$ -dimensional language. We already know from Hayashi [38] that it is not  $IO$  and that it is not  $OI$  for  $n = 1, 2, 3$ . Any grammar for  $L$  must use the full copying power of at least the 4-dimensional grammars. The set of strings in  $L$  which have the property that each substring  $(a'' b a'')$  has  $a''$ ,  $a''$  different from that in any other such substring are worth considering. First note that after the initial derivation frontier all subsequent frontiers are essentially  $IO$ . For this reason all of these different substrings must be produced at this first frontier or they are copied in such a way that each place into which they are copied is different which again necessitates the generation of exactly  $2^{2^m}$  distinct objects. It is for this reason that we believe  $L$  cannot be defined by any  $OI$  Baldwin-Schoenberger grammar.

If we augment the definitions of the  $n$ -dimensional grammars, frontiers, and  $(\tau)_\rho$  to include nonterminals which are not expanded until subsequent frontiers, we may easily define this language. The extended language definition is:

**Definition 29** An extended  $n$ -dimensional tree grammar of degree  $k$ , denoted  $G_n^k$ , is a quadruple  $(\Sigma, \mathcal{F}, P, S)_n^k$  where

1.  $\Sigma$  is the terminal alphabet;
2.  $\mathcal{F} = \langle \mathcal{F}_l^j \rangle_{1 \leq l \leq n}^{1 \leq j \leq l}$  is a ranked set called the nonterminal alphabet;
3.  $S \in \mathcal{F}_n^k$  is the start symbol;
4.  $P = \langle P_l^j \rangle_{1 \leq l \leq n}^{1 \leq j \leq l}$  is a ranked set of productions of the form  $F \rightarrow \alpha \in P_l^j$  for  $F \in \mathcal{F}_l^j$  and  $\alpha \in H_n^l(\Sigma, \langle \mathcal{F}_{l'}^{j'} \rangle \cup X^{l''})$ ,  $1 \leq l' \leq l$ , and  $1 \leq l'' \leq l - 1$ .

□

The nonterminals are now of dimension  $l$  degree  $j$  and are not expanded until the  $l$ th frontier. In other words,  $F \in \mathcal{F}_l^j$  is not expanded until the  $l$ -dimensional frontier obtaining true nondeterministic copying for the languages. This delay is incorporated into the frontier relation, giving the following definition.

**Definition 30** The extended frontier relation for  $n$ -dimensional grammars

$f: \bigcup_{k=1}^n H_n^k \rightarrow \bigcup_{l=1}^{n-1} H_{n-1}^l$  is given as follows:

$$f(F[_j t_j] \dots [_k t_k]) = \begin{cases} \bigcup_{\alpha \in \{\alpha | F \rightarrow \alpha \in P_n^{j'}\}} \{f(\alpha[_j t_j] \dots [_k t_k])\} & F \in \mathcal{F}_n^{j'} \\ F[_j f(t_j)] \dots [_k f(t_k)] & F \in \mathcal{F}_l^{j'} \wedge l < n \end{cases}$$

$$f(a[_n t_n] \cdots [_k t_k]) = \begin{cases} \left( f(t_n)_{IO(OI)} f(t_{n-1}) \right) [_{n-2} f(t_{n-2})] \cdots [_k f(t_k)] & t_n \neq \emptyset \\ a[_{n-1} f(t_{n-1})] \cdots [_k f(t_k)] & t_n = \emptyset \end{cases}$$

□

The evaluation of the substitution now requires that the paths be evaluated in structures containing nonterminals. Recall that the search function returns the tree in  $H_n^n$  that is at the end of the path on the tree being searched. Now it is possible to copy a nonterminal from  $\mathcal{F}_n^n$  as well as a forest from  $H_n^n$ . Note, the only nonterminals which may be copied are those producing trees and not forests. The new search function is:

**Definition 31** If  $t = a[_n t_n] \cdots [_k t_k] \in H_n^k$ , and  $\rho \in \{1, \dots, n-1\}^*$  then

$$(a[_n t_n] \cdots [_k t_k])_\rho = \begin{cases} (t_j)_{\rho'} & t_j \neq \emptyset \wedge \rho = j\rho' \\ a[_n t_n] & \rho = \lambda \wedge a \in \Sigma \cup X_n \cup X_{n+1} \cup \mathcal{F}_n^n \\ \text{error} & \rho \notin \text{treeshape}(t) \end{cases}$$

□

The original *OI* and *IO* languages are extended  $n$ -dimensional languages with  $\mathcal{F}_l^j = \emptyset$  for  $1 \leq l < n$ . These extended grammars are of interest mainly for the string languages that may be derived from them, since the first frontier,  $f_{IO(OI)}^{\Leftarrow}(S)$ , produces forests in  $H_{n-1}^k(\Sigma, X \cup \mathcal{F})$  as well as forests in  $H_{n-1}^k(\Sigma, X)$ . The *OI* string language for the grammar  $G_n^k = (\Sigma, \mathcal{F}, P, S)_n^k$  is then  $f_{IO}^{n-1}(S)$  and the *OI* string language is  $f_{OI}^{n-1}(S)$ . Under this new definition the  $n$ -dimensional *IO* languages are still the frontiers of  $(n+1)$ -dimensional *OI* languages. The result follows if we move the

nonterminals of dimension  $k$  to dimension  $k + 1$ , then the  $k + 1$ -dimensional frontier forces evaluation of the nonterminals of dimension  $k$  from the old  $n$ -dimensional grammars, but does no substitution until the next frontier.

We now give the extended *OI* 4-dimensional grammar for the language  $L = \{(a^* b a^*)^{2^{2^m}} \mid m \geq 1\}$ .

**Grammar 2**  $G_4^4 = (\Sigma, \mathcal{F}, P, U)_4^4$  where

$$\Sigma = \{*, a, b\};$$

$$\mathcal{F}_4^4 = \{U, T, S, A\}, \mathcal{F}_2^2 = \{B\};$$

$P$  contains the following productions:

1.  $U \rightarrow *[_4 *[_3 T[_2 B \mid \mid ]],$
2.  $T \rightarrow *[_4 S[_3 *[_4 *[_3 *[_2 x_2^\lambda[_1 x_2^\lambda[_1 x_1^\lambda \mid \mid \mid \mid \mid \mid ]],$
3.  $S \rightarrow A,$
4.  $A \rightarrow *[_4 *[_3 x_3^\lambda[_2 x_3^\lambda[_2 x_2^\lambda \mid \mid \mid ]],$
5.  $A \rightarrow *[_4 A[_3 *[_4 *[_3 x_3^\lambda[_2 x_3^\lambda[_2 x_2^\lambda \mid \mid \mid ][_3 x_3^\lambda \mid \mid ]],$
6.  $B \rightarrow *[_2 b[_1 x_1^\lambda \mid ]]$
7.  $B \rightarrow *[_2 B[_1 a[_1 x_1^\lambda \mid \mid ]]$
8.  $B \rightarrow *[_2 a[_1 B[_1 x_1^\lambda \mid \mid ]]$

□

The nonterminal  $B$  is not evaluated until the 2-dimensional frontier, so that each  $B$  will produce a different sequence of  $as$  and  $bs$ .

In fact, we can modify this grammar to produce the languages  $(a^* b a^*)^{2^{m-2}}$  where  $m \geq 1$ ,  $n \geq 2$ , and the tower of twos is of depth  $n - 2$ . For  $n = 4$  use the grammar as given. For  $n = 5$  add 1 to all numbers in the productions for the nonterminals  $U$ ,  $T$ ,  $S$ , and  $A$ ; replace  $B$  in the  $U$  production with

$$*[_5 *[_4 *[_3 *[_2 x_2^\lambda[_1 x_2^\lambda[_1 x_1^\lambda ] ] ] ] ] ] ]$$

and add the new start symbol  $U_5 \in \mathcal{F}_5^5$  along with the production

$$U_5 \rightarrow *[_5 *[_4 *[_3 U[_2 B ] ] ] ] ] .$$

The nonterminals  $U$ ,  $T$ ,  $S$ , and  $A$  are now in  $\mathcal{F}_5^5$  but  $B$  and its productions remain as before. Given a grammar for  $n$  we get the grammar for  $n + 1$  by adding 1 to all numbers in the production from  $P_n^n$ , replacing  $B$  in these productions with

$$*[_{n+1} *[_n \cdots *[_2 x_2^\lambda[_1 x_2^\lambda[_1 x_1^\lambda ] ] ] ] \cdots ] ]$$

and adding the new start symbol  $U_{n+1}$  with the production

$$U_{n+1} \rightarrow *[_{n+1} *[_n \cdots *[_3 U_n[_2 B ] ] ] \cdots ] ] .$$

## 4 CONCLUSIONS AND FUTURE WORK

We now know that there is an *OI*  $n$ -dimensional forest language hierarchy whose development parallels that of the *IO*  $n$ -dimensional forest hierarchy. In fact the *IO* languages of dimension  $n$  are contained in the *OI* languages of dimension  $(n + 1)$ . This containment is strict for both the string languages and the forest languages.

This *OI* hierarchy can be compared to the hierarchy described by Engelfriet and Schmidt. The Engelfriet-Schmidt tree languages are completely contained in the Baldwin-Schoenberger forest languages. The Engelfriet-Schmidt string languages are contained in the Baldwin-Schoenberger string languages, but we do not know if containment is strict. Further research is required to determine the answer to this question. Perhaps we need a machine oriented definition for both types of languages or an algebraic description for the Baldwin-Schoenberger languages. The algebraic definition would help resolve the type conflicts found in the attempts to reconcile the two hierarchies of string languages.

We do know that the *OI* 3-dimensional string languages are equivalent to the *OI* macro languages as described by Fischer. This gives hope that eventually we will be able to determine whether or not the containment of the Engelfriet-Schmidt hierarchy in the Baldwin-Schoenberger hierarchy is strict.

The generalization of the *OI* hierarchy required the extension of the frontier function to a frontier relation which expands the nonterminals as they are encountered in

the forests. This extension shows how close the taking of frontiers is to the derivation of trees or forests. The generalization of substitution of trees into trees has the property that both *OI* and *IO*  $n$ -dimensional substitution are associative. This result is different from the usual substitution which is associative in the *OI* case only.

The *OI* hierarchy has limited nondeterministic copying power since the frontiers taken after the derivation frontier are essentially *IO*. For this reason we extended the grammar definition to include nonterminals of dimension as well as degree. A nonterminal of dimension  $k$  is not expanded until the  $k$ -dimensional frontier. This allows a much greater degree of nondeterminism and these new languages contain the old languages. Further research is needed to determine their relationship to the regular languages and what normal form would prove useful.

## 5 BIBLIOGRAPHY

- [1] Aho, A. V. "Indexed grammars – an extension of context-free grammars". *Journal of the Association for Computing Machinery* 15(1968):647–671.
- [2] Aho, A. V. "Nested stack automata". *Journal of the Association for Computing Machinery* 16(1969):383–406.
- [3] Aho, A. V., and J. D. Ullman. "Automaton analogs of syntax directed translation schemata". *IEEE 9<sup>th</sup> Annual Symposium on Switching and Automata Theory* 9(1968):143–159.
- [4] Aho, A. V., and J. D. Ullman. *The Theory of Parsing, Translation, and Compiling*. Volume I: Parsing. Englewood Cliffs: Prentice-Hall, 1972.
- [5] Aho, A. V., and J. D. Ullman. *The Theory of Parsing, Translation, and Compiling*. Volume II. Compiling. Englewood Cliffs: Prentice-Hall, 1973.
- [6] Aho, A. V., and J. D. Ullman. *Principles of Compiler Design*. Reading: Addison-Wesley, 1979.
- [7] Arbib, M. A., and Y. Givon. "Algebra automata I: parallel programming as a prolegomena to the categorical approach". *Information and Control* 12(1968):331–345.
- [8] Bader, C., and A. Moura. "A generalization of Ogden's lemma". *Journal of the Association for Computing Machinery* 29(1982):404–407.
- [9] Baker, B. S. "Tree transductions and families of tree languages". *Proceedings of the Fifth Annual ACM Symposium on the Theory of Computing* Austin, Texas 5(1973):200–206.
- [10] Baker, B. S. "Tree transducers and tree languages". *Information and Control* 37(1978):241–266.
- [11] Baker, B. S. "Composition of top-down and bottom-up tree transductions". *Information and Control* 41(1979):186–213.

- [12] Baldwin, W. A. *Hypertrees - A Study in Language Specification*. Ph. D. thesis. Iowa State University, 1983.
- [13] Bar-Hillel, Y. "On formal properties of simple phrase structure grammars". *On Formal Properties of Simple Phrase Structure Grammars*. Reading: Addison-Wesley, 1964.
- [14] Berstel, J. *Transductions and Context-Free Languages*. Stuttgart: B. G. Teubner, 1979.
- [15] Birkhoff, G., and J. D. Lipson. "Heterogeneous algebras". *Journal of Combinatorial Theory* 8(1970):115-133.
- [16] Brainerd, B. "An analog of a theorem about context-free languages". *Information and Control* 11(1968):561-567.
- [17] Brainerd, W. S. "The minimalization of tree automata". *Information and Control* 13(1968):484-491.
- [18] Brainerd, W. S. "Tree generating regular systems". *Information and Control* 14(1969):217-231.
- [19] Burstall, R. M., and J. A. Goguen. *Algebras, Theories, and Freeness: an Introduction for Computer Scientists*. Technical Report University of Edinburgh Edinburgh, England February 1982.
- [20] Eilenberg, S., and J. B. Wright. "Automata in general algebras". *Information and Control* 11(1967):452-470.
- [21] Engelfriet, J. "Bottom-up and top-down tree transformations - a comparison". *Mathematical Systems Theory* 9(1974):198-231.
- [22] Engelfriet, J. "Some open questions and recent results on tree transducers and tree languages". *Formal Language Theory: Perspectives and Open Problems*. Ed. R. V. Book. New York: Academic Press, 1980. 241-286.
- [23] Engelfriet, J. "Hierarchies of hyper-AFL". *Journal of Computer and System Sciences* 30(1985):86-115.
- [24] Engelfriet, J., and E. M. Schmidt. "IO and OI. I". *Journal of Computer and System Sciences* 15(1977):328-353.
- [25] Engelfriet, J., and E. M. Schmidt. "IO and OI. II". *Journal of Computer and System Sciences* 16(1978):67-99.

- [26] Engelfriet, J., and G. Slutzki. "Extended macro grammars and stack controlled machines". *Journal of Computer and System Sciences* 29(1984):366-408.
- [27] Fischer, M. J. *Grammars with Macro-Like Productions*. Ph. D. thesis. Harvard University, 1968.
- [28] Ginsburg, S., and S. A. Greibach. "Abstract families of languages". *Memoirs of the American Mathematical Society (Studies in Abstract Families of Languages)* 87(1969):1-32.
- [29] Ginsburg, S., and S. A. Greibach. "Pre-AFL". *Memoirs of the American Mathematical Society (Studies in Abstract Families of Languages)* 87(1969):40-51.
- [30] Ginsburg, S., and S. A. Greibach. "Principal AFL". *Journal of Computer and System Sciences* 4(1970):308-338.
- [31] Ginsburg, S., S. A. Greibach, and M. A. Harrison. "One-way stack automata". *Journal of the Association for Computing Machinery* 14(1967):389-418.
- [32] Ginsburg, S., S. A. Greibach, and M. A. Harrison. "Stack automata and compiling". *Journal of the Association for Computing Machinery* 14(1967):172-201.
- [33] Goguen, J. A., and J. W. Thatcher. "Initial algebra semantics". *Proceedings of the First Annual ACM Symposium on the Theory of Computing* 1(1969):63-77.
- [34] Goguen, J. A., J. W. Thatcher, E. G. Wagner, and J. B. Wright. "Initial algebra semantics and continuous algebras". *Journal of the Association for Computing Machinery* 24(1977):68-95.
- [35] Greibach, S. A., and J. Hopcroft. "Independence of AFL operations". *Memoirs of the American Mathematical Society (Studies in Abstract Families of Languages)* 87(1969):233-247.
- [36] Guessarian, I. "Pushdown tree automata". *Mathematical Systems Theory* 16(1983):237-263.
- [37] Harrison, M. A. *Introduction to Formal Language Theory*. Reading: Addison-Wesley, 1978.
- [38] Hayashi, T. "On derivation trees of indexed grammars - an extension of the uvwxy-theorem". *Publ. RIMS Kyoto Univ.* 9(1973):61-92.
- [39] Hopcroft, J. E., and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Reading: Addison-Wesley, 1979.

- [40] Hu, S. *Elements of Modern Algebra*. San Francisco: Holden-Day, 1965.
- [41] Knuth, D. E., and P. B. Bendix. "Simple word problems in universal algebras". *Computational Problems in Abstract Algebra*. Ed. J. Leech. Oxford: Pergamon Press, 1970. 263-297.
- [42] Kosaraju, S. R. "Context-free preserving functions". *Mathematical Systems Theory* 9(1975):193-197.
- [43] Maibaum, T. S. E. "A generalized approach to formal languages". *Journal of Computer and System Sciences* 8(1974):409-439.
- [44] Maslov, A. N. "The hierarchy of indexed languages of an arbitrary level". *Soviet Math. Dokl. (in translation)* 15(1974):1170-1174.
- [45] Maslov, A. N. "Indexed grammars and Wijngaarden grammars". *Problems of Information Transmission (in translation)* 11(1975):81-89.
- [46] Maslov, A. N. "Multilevel stack automata". *Problems of Information Transmission (in translation)* 12(1976):55-62.
- [47] Mezei, J., and J. B. Wright. "Algebraic automata and context-free sets". *Information and Control* 11(1967):3-29.
- [48] Ogden, W. F. "Intercalation theorems for stack languages". *Proceedings of the First Annual ACM Symposium on the Theory of Computing* 1(1969):31-42.
- [49] O'Neil, T. E. *The Multidimensional Forest Languages*. Ph. D. thesis. Iowa State University, 1985.
- [50] Raynard-Smith, V. J. "Hypergrammars: an extension of macrogrammars". *Journal of Computer and System Sciences* 14(1977):130-149.
- [51] Rounds, W. C. "Context-free grammars on trees". *Proceedings of the First Annual ACM Symposium on the Theory of Computing* Marina del Ray, Ca. 1(1969):143-148.
- [52] Rounds, W. C. "Mappings and grammars on trees". *Mathematical Systems Theory* 4(1970):257-287.
- [53] Rounds, W. C. "Tree-oriented proofs of some theorems on context-free and indexed languages". *Proceedings of the Second Annual ACM Symposium on the Theory of Computing* Northhampton, Mass. 2(1970):109-116.

- [54] Schimpf, K. M. *A Parsing Method for Context-Free Tree Languages*. Ph. D. thesis. University of Pennsylvania, 1982.
- [55] Schimpf, K. M. "Tree pushdown automata". *Journal of Computer and System Sciences* 30(1985):25-40.
- [56] Strawn, G. " $n$ -dimensional trees and the frontier function". Manuscript. Department of Computer Science, Iowa State University, 1985.
- [57] Thatcher, J. W., and J. B. Wright. "Generalized finite automata theory with an application to a decision problem of second-order logic". *Mathematical Systems Theory* 2(1968):57-81.
- [58] Thatcher, J. W. "Generalized sequential machine maps". *Journal of Computer and System Sciences* 4(1970):339-367.
- [59] Thatcher, J. W. "Tree automata: an informal survey". *Currents in the Theory of Computing*. Ed. A. V. Aho. Englewood Cliffs: Prentice-Hall, 1973. 143-172.
- [60] Upton, R. A. "An extension of tree adjunct grammars". *Information and Control* 51(1981):248-274.
- [61] Wagner, E. G. "An algebraic theory of recursive definitions and recursive languages". *Proceedings of the Third Annual ACM Symposium on the Theory of Computing* Shaker Heights 3(1971):12-23.
- [62] Wand, M. "An algebraic formulation of the chomsky hierarchy". *Category Theory Applied to Computation and Control; Proceedings of the First International Symposium*, San Francisco. Ed. E. G. Manes. Vol. 25 of Lecture Notes in Computer Science. Ed. G. Goos and J. Hartmanis. New York: Springer-Verlag, 1975. 209-213.