CHAPTER I

BEGINNINGS

The aim of this introductory chapter is two-fold. First it serves to introduce some of the basic terminology and notation of context-free, EOL and ETOL grammars, which is a necessary prerequisite for the reader to have any understanding of the remainder of this book. Any terminology and notation not found in Section I.1 will either be defined at the time of usage or is assumed to be standard.

Second in Section I.2 the relationship between form theory and grammatical similarity is discussed briefly. Towards this end various approaches to grammatical similarity that have been taken previously are discussed. It culminates by proposing form theory as another approach to grammatical similarity and demonstrates how some of the problems that have been tackled fit into this framework.

Thus the study of grammatical similarity can be seen to be one of the aims of form theory. However it should be borne in mind that form theory has two further equally important aims. The first of these is simply an attempt to gain a better understanding of generative devices and the second is to obtain a deeper knowledge of the context-free, EOL and ETOL languages.

It is to be hoped that this book not only serves to whet the reader's appetite but that it also convinces the reader that form theory has already made contributions to each of these three areas.

I.1 Basic Terminology and Notation

Much of the terminology required for reading this book is standard and can be found in Aho and Ullman (1972), Berstel (1979), Ginsburg (1966), Harrison (1978), Herman and Rozenberg (1975), Hopcroft and Ullman (1979), Rozenberg and Salomaa (1980) and Salomaa (1973). However the notation differs in one important respect, namely sequential and parallel rewriting are distinguished by the use of $\Rightarrow$ and $\Rightarrow$ respectively, for the rewrite relations.

Before introducing the notation and terminology for context-free and L grammars in Sections I.1.1 and I.1.2, respectively, some of the basic notation and terminology for alphabets, words, language families and operations, and production schemes is reviewed.

Definition: Alphabets, Words and Length

An alphabet Σ is a finite non-empty set of symbols or letters. A word x over Σ is a finite, possibly empty, sequence of letters from Σ. The empty word, that is the empty sequence, is denoted by λ. By Σ^* we denote the set of words over Σ and by Σ^+ we denote $\Sigma^* - \{\lambda\}$. The length of a word x over Σ, denoted by $|x|$, is the number of symbols in x, hence $|\lambda| = 0$. For a in Σ and x a word over Σ the a-length of x, denoted by $|x|_a$ is the number of a's in x. Similarly for $\Delta \subseteq \Sigma$ the Δ-length of x, denoted by $|x|_\Delta$, is defined as $\sum_{a \text{ is in } \Delta} |x|_a$, hence in the case that $\Delta = \emptyset$, the empty set, $|x|_\emptyset = 0$.

For an alphabet Σ and an integer $i \geq 0$, Σ^i is the set of all words over Σ whose length is exactly i. Similarly by $\Sigma^{\leq i}$ we denote the set of all words over Σ whose length is at most i.

Notation:

Let A be an arbitrary set, then $\#A$ denotes the cardinality of A.

Definition: Operation on Words

Let Σ be an alphabet and x a word over Σ. Then by $mi(x)$ we denote the mirror image of x, also called the reversal of x.

Let Σ and Δ be alphabets. Then a map $h: \Sigma^* \to \Delta^*$ is a homomorphism if $h(\lambda) = \lambda$ and for all x,y in Σ^*, $h(xy) = h(x)h(y)$. It is a letter-to-letter homomorphism if $h(a)$ is in Δ, for all a in Σ and an isomorphism if it is letter-to-letter and one-to-one onto. A map $f: \Sigma^* \to 2^{\Delta^*}$ is a substitution if $f(\lambda) = \{\lambda\}$ and for all x,y in

Σ^*, $f(xy) = f(x)f(y)$. If $f(a)$ is finite for all a in Σ then f is a finite substitution. If $f(a) \subseteq \Delta$ for all a in Σ then f is a finite-letter substitution and if furthermore $f(a) \cap f(b) = \emptyset$ for all a,b in Σ, $a \neq b$, then f is a disjoint-finite-letter substitution (dfl-substitution). If $f(a)$ is a regular language (see Definition below) for all a in Σ then f is said to be a regular substitution.

Definition: Languages and Operations

A language is a subset of Σ^*, for some alphabet Σ. Let $L \subseteq \Sigma^*$ be a language and $h: \Sigma^* \to \Delta^*$ a homomorphism. Then $h(L) = \{h(x): x \text{ is in } L\}$. Similarly if $h: \Delta^* \to \Sigma^*$ is a homomorphism, then $h^{-1}(L) = \{y: y \text{ in } \Delta^* \text{ such that } h(y) \text{ is in } L\}$. Let L_1 and L_2 be two languages then by L_1L_2 we denote the catenation of L_1 and L_2, defined by: $L_1L_2 = \{x_1x_2: x_i \text{ is in } L_i, i = 1, 2\}$ and by $L_1 \cup L_2$ the union of L_1 and L_2, defined by: $L_1 \cup L_2 = \{x: x \text{ is in } L_1 \text{ or } x \text{ is in } L_2\}$. Let L be a language, then its (star) closure L^* is defined as: $L^* = \{x_1x_2\ldots x_m: m \geq 0, x_i \text{ is in } L, 1 \leq i \leq m\}$. For a language $L \subseteq \Sigma^*$ and a regular language $R \subseteq \Sigma^*$ (see Definition below), we denote by $L \cap R$ the language $\{x: x \text{ is in } L \text{ and } x \text{ is in } R\}$.

Definition: Finite State Acceptors and Regular Sets

A finite state acceptor (fsa) is a quintuple $M = (Q,\Sigma,\delta,q_0,F)$ where Q is a finite set of states, Σ is the input alphabet, $\delta: Q \times \Sigma \to 2^Q$ is the transition function, q_0 in Q is the start state and $F \subseteq Q$ is the set of accepting states.

A configuration of M is an ordered pair (q,x) where q is the current state of M and x is the input remaining to be read, that is q is in Q and x is in Σ^*.

Let (p,x) and (q,y) be two configurations of M, we say there is a move from (p,x) to (q,y) in M, denoted $(p,x) \vdash (q,y)$, if $x = ay$ for some a in Σ and q is in $\delta(p,a)$.

This is extended to move sequences in the usual way to give $\vdash^i$, $\vdash^+$ and $\vdash^*$. A word x in Σ^* is accepted if:

$(q_0, x) \vdash^* (q, \lambda)$ for some q in F.

The language of M, denoted $L(M)$ is defined as:

$L(M) = \{x: (q_0, x) \vdash^* (q,\lambda) \text{ for some } q \text{ in } F\}$.

The collection of all languages, which can be generated by fsa is denoted by $\mathcal{L}$(REG) and is known as the family of regular sets.

Definition: A-transducers and Gsms

It is straightforward to generalize finite state acceptors in such a way that words can be read at each move rather than input symbols. Moreover each generalised fsa can always be replaced by an equivalent fsa (one accepting the same language) satisfying the original definition. When output is included however it is the generalised fsa which is considered.

An a-transducer is a sextuple $M = (Q,\Sigma,\Delta,H,q_0,F)$ where Q is a finite set of states, Σ is the input alphabet, Δ is the output alphabet, $H \subseteq Q \times \Sigma^* \times \Delta^* \times Q$ is a finite set of transitions, q_0 in Q is the start state and $F \subseteq Q$ is the set of accepting states.

A configuration of M is a triple (q,x,z) in $Q \times \Sigma^* \times \Delta^*$, where q is the current state, x is the remaining input and z is the present output.

Let (p,x,w) and (q,y,z) be two configurations. Then we say there is a move from (p,x,w) to (q,y,z) in M, denoted $(p,x,w) \vdash (q,y,z)$ if $x = uy$ for some u in Σ^*, $z = wv$ for some v in Δ^* and (p,u,v,q) is in H.

As before this is extended to $\vdash^i$, $\vdash^+$, and $\vdash^*$.

In this case however we are not so much interested in the word pairs accepted by M as the transformation of input words to output words. For each x in Σ^*, let $M(x) = \{z: (q_0,x,\lambda) \vdash^* (q,\lambda,z)$, for some q in F and z in $\Delta^*\}$ and for each language $L \subseteq \Sigma^*$ let $M(L) = \bigcup_{x \text{ in } L} M(x)$. The mapping M from 2^{Σ^*} into 2^{Δ^*} so defined is called an a-transducer mapping.

A gsm (generalized sequential machine) is an a-transducer in which (i) all states are accepting states (hence M is given as a quintuple) and (ii) $H \subseteq Q \times \Sigma \times \Delta^* \times Q$, that is it is based directly on the finite state acceptor. Otherwise its definition is analogous to that of the a-transducer. Note that there is no accepted standard definition of a gsm. As will be seen we will use the most convenient definition in our proofs, however it is straightforward though laborious to convert these into gsms according to the present definition.

Definition: Pushdown Acceptors

A pushdown acceptor (pda) is a sextuple $M = (Q,\Sigma,\Gamma,H,Z_0,q_0)$ where Q is a finite set of states, Σ is the input alphabet, Γ is the pushdown alphabet, $H \subseteq Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \times \Gamma^* \times Q$ is a finite set of moves or transitions, Z_0 in Γ is the initial pushdown symbol and q_0 in Q is the start state.

A configuration of M is a triple (q,x,γ) in $Q \times \Sigma^* \times \Gamma^*$, where q is the current state, x is the remaining input and γ is the current pushdown. Note that the left end of γ corresponds to the tope of the pushdown.

Let (p,x,γ) and (q,y,γ') be two configurations. We write $(p,x,\gamma) \vdash (q,y,\gamma')$ if $x = zy, \gamma = Z\delta$ and $\gamma' = \delta'\delta$ where (p,z,Z,δ',q) is in H. Note that z is in $\Sigma \cup \{\lambda\}$. We can define $\vdash^i$, $\vdash^+$ and $\vdash^*$ in the usual way. The language accepted by M with empty pushdown, Null(M), is defined by:

$$\text{Null}(M) = \{x: (q_0,x,Z_0) \vdash^* (q,\lambda,\lambda), \text{ for some } q \text{ in } Q\}.$$

It is well known that the collection of all Null(M) for all pda M is the family of context-free languages, $\mathcal{L}$(CF).

A pda M is a deterministic pda (dpda) if

(i) for all (p,z,Z) in $Q \times (\Sigma \cup \{\lambda\}) \times \Gamma$ there is at most one move (p,z,Z,γ,q) in H for some γ and q, and

(ii) for all p in Q and Z in Γ if there is a move (p,λ,Z,γ,q) in H for some γ and q then for all a in Σ, there is no move (p,a,Z,γ',q') in H for any γ' and q'.

Definition: Language Families and Operations

A language family $\mathcal{L}$ is a collection of languages which satisfies the following weak condition:

If $L \subseteq \Sigma^*$ is a language in $\mathcal{L}$, Δ is an arbitrary alphabet with $\#\Sigma = \#\Delta$ and $h: \Sigma^* \to \Delta^*$ is an isomorphism, then h(L) is in $\mathcal{L}$.

Thus $\mathcal{L}$ is closed under renaming.

We say $\mathcal{L}$ is closed under:

(i) union if for all L_1, L_2 in $\mathcal{L}$, $L_1 \cup L_2$ is in $\mathcal{L}$,

(ii) intersection with regular sets, if for all L in $\mathcal{L}$ and for all regular languages R, $L \cap R$ is in $\mathcal{L}$,

(iii) catenation if for all L_1, L_2 in $\mathcal{L}$, L_1L_2 is in $\mathcal{L}$,

(iv) star closure is for all L in $\mathcal{L}$, L^* is in $\mathcal{L}$,

(v) homomorphism if for all Σ, for all $L \subseteq \Sigma^*$ in $\mathcal{L}$ and for all homomorphisms $h: \Sigma^* \to \Delta^*$ for some Δ, h(L) is in $\mathcal{L}$,

(vi) inverse homomorphism if for all Σ, for all $L \subseteq \Sigma^*$ in $\mathcal{L}$ and for all homomorphisms $h: \Delta^* \to \Sigma^*$ for some Δ, $h^{-1}(L)$ is in $\mathcal{L}$,

(vii) a-transducer mappings if for all Σ, for all $L \subseteq \Sigma^*$ in $\mathcal{L}$ and for all a-transducers $M: \Sigma^* \to 2^{\Delta^*}$ for some alphabet Δ, M(L) is in $\mathcal{L}$.

If $\mathcal{L}$ is closed under (i)-(vi) then it is said to be a full AFL (Abstract Family of Languages).

If $\mathcal{L}$ is closed under (i), (ii), and (vi) then is said to be a full semi-AFL; $\mathcal{L}$ is a full semi-AFL iff it is closed under a-transducer mappings.

A final operation is the wedge operation of two language families. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two language families, then $\mathcal{L}_1 \vee \mathcal{L}_2$ (the wedge of $\mathcal{L}_1$ and $\mathcal{L}_2$) is defined as:

$$\mathcal{L}_1 \vee \mathcal{L}_2 = \{L_1 \cup L_2 : L_i \text{ is in } \mathcal{L}_i,\ i = 1, 2\}.$$

Definition: Closure and Language Families

Let $\mathcal{L}$ be a family of languages and X be a subset of the operations (i)-(vii) of the previous definition. Then the X-closure of $\mathcal{L}$ is the smallest family of languages containing $\mathcal{L}$ and closed under each of the operations in X. In particular we speak of the homomorphic closure of $\mathcal{L}$, denoted $\mathcal{H}(\mathcal{L}) = \{h(L): L \text{ is in } \mathcal{L},\ L \subseteq \Sigma^*,\ h: \Sigma^* \to \Delta^* \text{ is a homomorphism for some alphabet } \Delta\}$. This notion is often well defined for a collection of languages $\mathcal{L}$, which is not a family. For example given a single language $L \subseteq \Sigma^*$ say, then the full semi-AFL closure of L, denoted by $\mathcal{S}(L)$, is the smallest full semi-AFL containing L. In this case we say that $\mathcal{L} = \mathcal{S}(L)$ is a full principal semi-AFL, with full generator L. Let $\mathcal{L}$ be the family of all alphabets and X = {∪,•,*} where • denotes catenation, then the X-closure of $\mathcal{L}$ is well known to be the family of regular languages, denoted $\mathcal{L}$(REG). Moreover $\mathcal{L}$(REG) is a full AFL.

Definition: Production Schemes

The notion underlying context-free, EOL and ETOL grammars is that of a production scheme.

A production scheme is a (n+3)-tuple $G = (V,\Sigma,P_1,\ldots P_n,S)$, for some $n \geq 1$, where V is an alphabet, $\Sigma \subseteq V$ is the terminal alphabet, $V - \Sigma$ is the nonterminal alphabet, P_i is a finite subset of $V \times V^*$, for all i, $1 \leq i \leq n$ and S in $V - \Sigma$ is the start or sentence symbol.

Each member of each P_i is called a production and each (X,α) in P_i is usually written as $X \to \alpha$.

For context-free grammars we have $n = 1$ and $P = P_1$ is further restricted so that $P \subseteq (V-\Sigma) \times V^*$. For EOL grammars we also have $n = 1$ and $P = P_1$ satisfies a "completeness" condition, namely for all X in V there is some production $X \to \alpha$ in P for some α in V^*. Apart

from this basic distinguishing feature context-free and EOL grammars are only distinguished by their rewrite relations.

Let $G = (V,\Sigma,P,S)$ be a production scheme, then a production $X \to \alpha$ in P is called an X-production. The X-productions of G are all X-productions in P. This notion is easily extended to production schemes with $n > 1$. When specifying the X-productions of a scheme G we often write them as: $X \to \alpha_1|\alpha_2|\ldots|\alpha_r$, that is in BNF notation.

Finally we introduce two conventions, which are used throughout these notes.

Convention: λ-convention

Given two languages L_1 and L_2 we say that they are equal (modulo λ) if $L_1 - \{\lambda\} = L_2 - \{\lambda\}$. Similarly we say two language families $\mathcal{L}_1$ and $\mathcal{L}_2$ are equal (modulo λ and ∅) if for every $L_1 - \{\lambda\} \neq \emptyset$ in $\mathcal{L}_1$ there is an L_2 in $\mathcal{L}_2$ such that $L_1 - \{\lambda\} = L_2 - \{\lambda\}$ and vice versa.

Notational Convention:

In the following unless specified otherwise we have assumed the following notational conventions:

Terminal symbols are represented by early lower case Roman letters and

Nonterminal symbols by early upper case Roman letters.

Symbols which may be either terminal or nonterminal are represented by late upper case Roman letters.

Terminal words are represented by late lower case Roman letters and words which may or may not be terminal by lower case Greek letters.

II.1.1 Context-Free Grammars and Languages

Definition: Context-free Grammars

A context-free grammar is an ordered pair $(G,\Rightarrow)$ where G is a production scheme (V,Σ,P,S) with $P \subseteq (V-\Sigma) \times V^*$ and $\Rightarrow$ is the sequential rewrite relation defined as follows:

For all α,β in V^* we write $\alpha \Rightarrow_G \beta$ (or simply $\alpha \Rightarrow \beta$ if G is understood) if:

$\alpha = \alpha_1 C \alpha_2$, $\beta = \alpha_1 \gamma \alpha_2$ for some α_1,α_2,γ in V^*, C in $V - \Sigma$ and $C \to \gamma$ in P.

If α_1 is in Σ^* then we may write $\alpha \overset{L}{\Rightarrow} \beta$, that is β is obtained by a leftmost rewrite of α and if α_2 is in Σ^* then we may write $\alpha \overset{R}{\Rightarrow} \beta$, that is β is obtained by a rightmost rewrite of α. We will write G rather than $(G,\Rightarrow)$ when it is clear that a context-free grammar is meant.

Definition: Derivations

Given a context-free grammar $(G,\Rightarrow)$, where $G = (V,\Sigma,P,S)$ we extend $\Rightarrow$, $\overset{L}{\Rightarrow}$ and $\overset{R}{\Rightarrow}$ to sequences of rewrite steps.

For all $i \geq 1$ and for all α,β in V^* we write $\alpha \overset{i}{\Rightarrow} \beta$ if:

either $i = 1$ and $\alpha \Rightarrow \beta$

or $i > 1$ and there exists γ in V^* such that $\alpha \Rightarrow \gamma$ and $\gamma \overset{i-1}{\Rightarrow} \beta$.

For all α,β in V^*, we write $\alpha \overset{+}{\Rightarrow} \beta$ if there exists an $i \geq 1$ such that $\alpha \overset{i}{\Rightarrow} \beta$. By convention we write $\alpha \overset{0}{\Rightarrow} \alpha$ for all α in V^* and we write $\alpha \overset{*}{\Rightarrow} \beta$ if either $\alpha \overset{+}{\Rightarrow} \beta$ or $\alpha = \beta$.

In a similar manner we can define $\overset{L}{\Rightarrow}i$, $\overset{L}{\Rightarrow}+$, $\overset{L}{\Rightarrow}*$, $\overset{R}{\Rightarrow}i$, $\overset{R}{\Rightarrow}+$, and $\overset{R}{\Rightarrow}*$.

Whenever $\alpha \overset{*}{\Rightarrow} \beta$ for some α,β in V^* we say that β is derived from α and when $\alpha \overset{+}{\Rightarrow} \beta$ we say that β is properly derived from α. In both cases we say that $\alpha \overset{*}{\Rightarrow} \beta$ or $\alpha \overset{+}{\Rightarrow} \beta$ is a derivation in G. If $\alpha = S$ we say that it is a sentential derivation and that β is a sentential form.

Occasionally we need to specify the sentential forms in derivations more precisely in which case we write a derivation as a derivation sequence:

$$\alpha = \alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_r = \beta$$

where $\alpha \overset{r}{\Rightarrow} \beta$ in G, for some $r \geq 0$.

Definition: Sub-grammars

Let $G = (V,\Sigma,P,S)$ be a grammar. We say $G' = (V',\Sigma',P',S)$ is a sub-grammar of G if $V' \subseteq V$, $\Sigma' \subseteq \Sigma$ and $P' \subseteq P$. For a nonterminal A in $V-\Sigma$ we say that the sub-grammar of G induced by A, denoted by G_A, is the grammar $G_A = (V,\Sigma,P,A)$.

Definition: Derivation Trees and Distinct Derivations

Let $G = (V,\Sigma,P,S)$ be a context-free grammar and τ be a tree with oriented and directed edges and with node labels taken from $V \cup \{\lambda\}$. Then τ is said to be a G-derivation tree or a derivation tree in G if the following conditions hold:

(i) the root is labelled with S,

(ii) the leaves are labelled from $V \cup \{\lambda\}$,

(iii) the non-leaf or internal nodes are labelled from $V - \Sigma$, and

(iv) for all non-leaf nodes u: if u is labelled with some A from $V - \Sigma$ and the sons of u in left to right order, $u_1,\ldots,u_r$, $r \geq 1$ are labelled with $X_1,\ldots,X_r$ respectively, then $A \to X_1 \ldots X_r$ is in P.

The leaf nodes of τ when read in left to right order yield a sentential form of G if τ is a derivation tree for G. We call this the frontier of τ and refer to the specific frontier by $fr(\tau)$.

Since each A in $V - \Sigma$ defines a sub-grammar G_A of G we also allow any nonterminal to be the label of a root node, in this case we write: τ is a G_A-derivation tree.

We say two derivation trees τ_1 and τ_2 for two grammars G_1 and G_2 (not necessarily distinct) are equally shaped if the non-leaf nodes of τ_1 can be relabelled to give τ_2 and vice versa.

Let $d_1: A \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_{r-1} \Rightarrow \alpha$ and $d_2: A \Rightarrow \beta_1 \Rightarrow \ldots \Rightarrow \beta_{s-1} \Rightarrow \beta$ be two derivations in G. Then d_1 and d_2 are distinct if their corresponding G_A-derivation trees τ_α and τ_β satisfy the following condition:

τ_α is not a tree-prefix of τ_β and τ_β is not a tree-prefix of τ_α.

Definition: Ambiguity

Let $G = (V,\Sigma,P,S)$ be a context-free grammar. A word x in Σ^* is said to be ambiguous with respect to G if there are two different G-derivation trees τ_1 and τ_2 with $fr(\tau_1) = fr(\tau_2) = x$.

G is said to be ambiguous if there is a word in Σ^*, which is ambiguous with respect to G, otherwise G is said to be unambiguous.

Definition: Context-free Languages and Length Sets

Given a context-free grammar $G = (V,\Sigma,P,S)$ the language generated by G, denoted $L(G,\Rightarrow)$, is defined by:

$$L(G,\Rightarrow) = \{x: x \text{ is in } \Sigma^* \text{ and } S \Rightarrow^* x\}.$$

Similarly the length set generated by G, denoted $LS(G,\Rightarrow)$, is defined by:

$$LS(G,\Rightarrow) = \{|x|: x \text{ is in } \Sigma^* \text{ and } S \Rightarrow^* x\}.$$

Let Σ be an alphabet and L be an arbitrary subset of Σ^*, then we say that L is a context-free language if there exists a context-free grammar G such that $L(G,\Rightarrow) = L$.

Definition: Families of Context-free Grammars

Let $G = (V,\Sigma,P,S)$ be a context-free grammar. G is a linear grammar if $P \subseteq (V-\Sigma) \times (\Sigma^* \cup \Sigma^*(V-\Sigma)\Sigma^*)$, a right linear grammar if $P \subseteq (V-\Sigma) \times (\Sigma^* \cup \Sigma^*(V-\Sigma))$ and a left linear grammar if $P \subseteq (V-\Sigma) \times (\Sigma^* \cup (V-\Sigma)\Sigma^*)$.

We say G is an empty or trivial grammar if $L(G,\Rightarrow) = \emptyset$ or $\{\lambda\}$ and otherwise G is nonempty or nontrivial. G is said to be finite if $L(G,\Rightarrow)$ is finite and infinite otherwise. We say G is reduced if the following conditions hold:

(i) For all X in V there is a derivation $S \Rightarrow^* \alpha X \beta$ for some α and β in V^*; X is reachable.

(ii) For all A in $V - \Sigma$ there is a derivation $A \Rightarrow^* x$ where x is in Σ^*; A is useful.

G is self-embedding if G is reduced and there exists A in $V - \Sigma$ together with a derivation $A \Rightarrow^+ uAv$, for some u and v in Σ^+. We say a reduced grammar G is non-self-embedding if it is not self-embedding.

G is expansive if it is reduced and there exists a nonterminal A in $V - \Sigma$ together with a derivation $A \Rightarrow^+ uAvAw$ in G, for some u,v,w in Σ^*, where $uvw \neq \lambda$.

We say G is non-expansive if it is not expansive.

Definition: Families of Context-free Languages

We denote by $\mathcal{L}(CF)$ the family of all context-free languages, that is:

$$\mathcal{L}(CF) = \{L(G,\Rightarrow): G \text{ is a context-free grammar}\}.$$

A number of the families of context-free grammars define proper sub-families of $\mathcal{L}(CF)$. Thus we obtain:

(i) $\mathcal{L}(FIN)$, the family of finite languages, is defined by:
$\mathcal{L}(FIN) = \{L(G,\Rightarrow): G \text{ is finite}\}$.

(ii) $\mathcal{L}(REG)$, the family of regular languages is defined by:
$\mathcal{L}(REG) = \{L(G,\Rightarrow): G \text{ is right linear}\}$
$= \{L(G,\Rightarrow): G \text{ is left linear}\}$
$= \{L(G,\Rightarrow): G \text{ is non-self-embedding}\}$

(iii) $\mathcal{L}(LIN)$, the family of linear languages, is defined by:
$\mathcal{L}(LIN) = \{L(G,\Rightarrow): G \text{ is linear}\}$.

(iv) $\mathcal{L}(DB)$, the family of derivation bounded languages, is defined by:
$\mathcal{L}(DB) = \{L(G,\Rightarrow): G \text{ is non-expansive}\}$.

II.1.2 EOL and ETOL Grammars and Languages

Definition: EOL Grammars

An EOL grammar (Extended Zero-sided Lindenmayer grammar) is an ordered pair $(G,\Rightarrow)$, where G is a production scheme (V,Σ,P,S) with $P \subseteq V \times V^*$ also satisfying the condition that for all X in V there is an α in V^* such that $X \rightarrow \alpha$ is in P, and $\Rightarrow$ is the parallel rewrite relation, defined as follows:

For all α,β in V^* we write $\alpha \Rightarrow_G \beta$ (or simply $\alpha \Rightarrow \beta$ if G is understood) if:

$\alpha = X_1 \ldots X_m$, where X_i is in V, $1 \leq i \leq m$,
$\beta = \beta_1 \ldots \beta_m$, for some β_i in V^*, $1 \leq i \leq m$, and
$X_i \rightarrow \beta_i$ is in P, $1 \leq i \leq m$.

As for context-free grammars we will often write G rather than $(G,\Rightarrow)$ when it is clear from the context that an EOL grammar is intended.

Definition: ETOL Grammars

An ETOL grammar (Extended Tabled Zero-sided Lindenmayer grammar) is an ordered pair $(G,\Rightarrow)$, where

(1) for some $n \geq 1$, $G = (V,\Sigma,P_1,\ldots,P_n,S)$ is a production scheme, where the P_i are called tables,
(2) for all i, $1 \leq i \leq n$, $G_i = (V,\Sigma,P_i,S)$ is an EOL grammar, and
(3) $\Rightarrow$ the parallel rewrite relation is defined by: for all α,β in V^*, $\alpha \Rightarrow_G \beta$ if there is some i, $1 \leq i \leq n$ such that $\alpha \Rightarrow_{G_i} \beta$. Usually we will write $\alpha \Rightarrow_{P_i} \beta$ or simply $\alpha \Rightarrow \beta$ when the table used is not important.

We say that G is a n-tabled ETOL grammar. Note that an EOL grammar is a one-tabled ETOL grammar.

A sub-grammar of an ETOL grammar is defined analogously to the notion of a sub-grammar of a context-free grammar.

Definition: Derivations

Given an n-tabled ETOL grammar $(G,\Rightarrow)$ where $G = (V,\Sigma,P_1,\ldots,P_n,S)$ we extend $\Rightarrow$ to sequences of rewrite steps as follows.

For all $i \geq 1$ and for all α,β in V^* we write $\alpha \Rightarrow^i \beta$ if:

either $i = 1$ and $\alpha \Rightarrow \beta$
or $i > 1$ and there exists γ in V^* such that $\alpha \Rightarrow \gamma$ and $\gamma \Rightarrow^{i-1} \beta$.

For any α,β in V^* we write $\alpha \Rightarrow^+ \beta$ if there exists an $i \geq 1$ such that

$\alpha \Rightarrow^{i} \beta$. By convention we write $\alpha \Rightarrow^{0} \alpha$ for all α in V* and we write $\alpha \Rightarrow^{*} \beta$ if either $\alpha \Rightarrow^{+} \beta$ or $\alpha = \beta$.

Whenever $\alpha \Rightarrow^{*} \beta$ for some α, β in V* we say that β is <u>derived from</u> α and when $\alpha \Rightarrow^{+} \beta$ we say β is <u>properly derived</u> from α. In both cases we say that $\alpha \Rightarrow^{*} \beta$ or $\alpha \Rightarrow^{+} \beta$ is a <u>derivation in G</u>. In a similar manner we can extend $\Rightarrow_{P_i}$ to $\Rightarrow^{+}_{P_i}$ and $\Rightarrow^{*}_{P_i}$. In this case we say that $\alpha \Rightarrow^{*}_{P_i} \beta$ or $\alpha \Rightarrow^{+}_{P_i} \beta$ is a <u>derivation in P_i</u>.

A sentential derivation is a derivation $S \Rightarrow^{*} \beta$ (or $S \Rightarrow^{+} \beta$), in which case β is said to be <u>sentential form</u>.

Whenever we need to specify a particular derivation we write $\alpha \Rightarrow^{*} \beta$ as a derivation sequence.

$$\alpha = \alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \alpha_r = \beta$$

where $\alpha \Rightarrow^{r} \beta$ in G, for some $r \geq 0$.

<u>Definition: Nonterminal Derivations</u>

Let $G = (V, \Sigma, P_1, \ldots, P_n, S)$ be an n-tabled ETOL grammar and $\alpha \Rightarrow^{+} \beta$ be a derivation in G for some α and β in V*.

We write $\alpha \Rightarrow^{+}_{nt} \beta$, that is a <u>nonterminal derivation (nt-derivation)</u> in G, if there exists a derivation sequence:

$$\alpha = \alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_r = \beta$$

for some $r > 0$ such that each α_i, $1 \leq i \leq r$ contains a nonterminal.

We write $\alpha \Rightarrow^{+}_{tnt} \beta$, that is a <u>totally nonterminal derivation (tnt-derivation)</u> in G, if for every derivation sequence

$$\alpha = \alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_r = \beta$$

for some $r > 0$, each α_i, $1 \leq i < r$ contains a nonterminal.

This notion can be extended to an <u>nt-</u> and a <u>tnt-derivation in P_i</u> in the obvious way.

<u>Definition: Derivation Trees and Distinct Derivations</u>

We only deal with derivation trees for EOL grammars, since the extension to the case of ETOL grammars is straighforward.

Let $G = (V, \Sigma, P, S)$ be an EOL grammar and τ be a tree with oriented and directed edges and with node labels taken from $V \cup \{\lambda\}$. Then τ is said to be a <u>G-derivation</u> tree or a <u>derivation tree in G</u> if the following conditions hold:

(i) the root is labelled with S,

(ii) the nodes are labelled from $V \cup \{\lambda\}$,

(iii) all the leaves are at the same distance from the root,

(iv) for all non-leaf nodes u; if u is labelled with some X from V and the sons of u in left to right order $u_1,\ldots,u_r$, $r \geq 1$ are labelled with $X_1,\ldots,X_r$, respectively, then $X \rightarrow X_1\ldots X_r$ is in P; if u is labelled with λ then it has one son which is also labelled with λ.

The leaves of τ, when read in left to right order yield a sentential form of G if τ is a G-derivation tree. We call this the frontier of τ, written fr(τ). The notion of equally shaped derivation trees is defined as for context-free derivation trees, see Section 1.1. However note that internal nodes may be labelled with terminals in the EOL case.

Let d_1: $S \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_{r-1} \Rightarrow \alpha$ and

d_2: $S \Rightarrow \beta_1 \Rightarrow \ldots \Rightarrow \beta_{s-1} \Rightarrow \beta$

be two derivations in G. We say d_1 and d_2 are distinct if their corresponding trees τ_α and τ_β satisfy:

τ_α is not a tree-prefix of τ_β and

τ_β is not a tree-prefix of τ_α.

Because we are dealing with parallel derivations we can simply say:

d_1 and d_2 are distinct if d_1 is not a prefix of

d_2 and d_2 is not a prefix of d_1.

This of course extends naturally to any two derivations, not necessarily sentential, in G.

Definition: Ambiguity

Let G = (V,Σ,P,S) be an EOL grammar. A word x in Σ* is said to be ambiguous with respect to G if there are two different G-derivation trees τ_1 and τ_2 such that $fr(\tau_1) = fr(\tau_2) = x$. G is said to be ambiguous if there is a word in Σ* which is ambiguous with respect to G, otherwise G is said to be unambiguous.

Definition: EOL and ETOL Languages and Length Sets

Given an EOL grammar G = (V,Σ,P,S) the language generated by G, denoted L(G,⇒), is defined by:

$L(G,\Rightarrow) = \{x: x \text{ is in } \Sigma^* \text{ and } S \Rightarrow^* x\}$.

Similarly the length set generated by G, denoted LS(G,⇒), is defined by:

$LS(G,\Rightarrow) = \{|x|: x \text{ in } \Sigma^* \text{ and } S \Rightarrow^* x\}$.

Let Σ be an alphabet and L be an arbitrary subset of Σ^*, then we say that L is an EOL language if there exists an EOL grammar G such that $L = L(G,\Rightarrow)$.

Each of these notions can be extended to the ETOL case in the obvious way.

Definition: Families of EOL and ETOL Grammars

Let $G = (V,\Sigma,P_1,\ldots,P_n,S)$ be an n-tabled ETOL grammar.

We say G is propagating if for all i, $1 \leq i \leq n$, and for all $X \rightarrow \alpha$ in P_i, $\alpha \neq \lambda$.

We say G is deterministic if for all i, $1 \leq i \leq n$, and for all X in V there is exactly one production $X \rightarrow \alpha$ in P_i, for some α.

We refer to EPTOL, EDTOL, EPDTOL, EPOL, EDOL and EPDOL grammars, where P and D indicate propagating and deterministic, respectively.

A n-tabled TOL grammar $G = (\Sigma,P_1,\ldots,P_n,\sigma)$ is an n-tabled ETOL grammar in which $V = \Sigma$ and S has been replaced by a starting word σ. An OL grammar $G = (\Sigma,P,\sigma)$ is a one-tabled TOL grammar. In both cases $L(G,\Rightarrow)$ consists of all words in all G-derivations.

We say G, an ETOL grammar, is empty if $L(G,\Rightarrow) = \emptyset$ or $\{\lambda\}$, otherwise G is nonempty. G is finite if $L(G,\Rightarrow)$ is finite and infinite otherwise.

We say $G = (V,\Sigma,P_1,\ldots,P_n,S)$ is reduced if for all X in V there is a derivation $S \overset{*}{\Rightarrow} \alpha X \beta$, for some α and β in V^*; we say X is reachable. This notion of a reduced grammar should be compared with the corresponding one for context-free grammars in Section 1.1.

We say $G = (V,\Sigma,P_1,\ldots,P_n,S)$ is looping if there is a reachable symbol X in V with a derivation $X \overset{+}{\Rightarrow} X$ in G, and G is expansive if there is some reachable symbol X in V with a derivation $X \overset{+}{\Rightarrow} \alpha X \beta X \gamma$, for some α,β and γ in V^*.

We say $G = (V,\Sigma,P_1,\ldots,P_n,S)$ is separated if for all i, $1 \leq i \leq n$, $X \rightarrow \alpha$ in P_i implies (i) α is in $\Sigma \cup (V-\Sigma)^*$ and (ii) X in Σ implies α is not in Σ. G is synchronized if for all a in Σ, $a \overset{+}{\Rightarrow} \alpha$ implies α is not in Σ^*. G is short if for all i, $1 \leq i \leq n$, $X \rightarrow \alpha$ in P_i implies $|\alpha| \leq 2$. G is said to be binary if each production is of one of the types $A \rightarrow \lambda$, $A \rightarrow a$, $A \rightarrow B$, $A \rightarrow BC$ or $a \rightarrow A$ where a is in Σ and A,B and C are in $V - \Sigma$.

Definition: Families of EOL and ETOL Languages

We denote by $\mathcal{L}$(EOL) and $\mathcal{L}$(ETOL) the families of all EOL and ETOL languages, respectively that is:

$\mathcal{L}$(EOL) = {L(G,⇛): G is an EOL grammar}

and

$\mathcal{L}$(ETOL) = {L(G,⇛): G is an ETOL grammar}.

Similarly we obtain $\mathcal{L}$(OL) and $\mathcal{L}$(TOL) and with the propagating and deterministic restrictions we obtain $\mathcal{L}$(EPOL), $\mathcal{L}$(EDOL), etc.

I.2 Notions of Grammatical Similarity

In this section a rapid survey of the different notions of grammatical similarity for context-free grammars is given. This culminates in Section 2.6 with the introduction of the topic of these lecture notes, namely a definition of similarity based on collections of grammars rather than on the grammars alone. With the exception of the similarity notions discussed in Sections 2.1, 2.4 and 2.6, these notions have not been applied to EOL or ETOL grammars.

I.2.1 Weak and Structural Equivalence

Given two grammars G_1 and G_2, the simplest notion of grammatical similarity is that $L(G_1,\Rightarrow) = L(G_2,\Rightarrow)$; we say that G_1 and G_2 are (weak) equivalent in this case. However this notion is too primitive, since grammars which are very different in structure can be related in this way. For example, consider G_1 and G_2 given as follows:

G_1: $S \to \lambda$; $S \to aS$;

and

G_2: $S \to \lambda$; $S \to aSaSa$; $S \to a$; $S \to aa$.

Now $L(G_1,\Rightarrow) = L(G_2,\Rightarrow) = a^*$, however the derivation trees generated by each grammar are clearly very different.

Because of this, a second notion of grammatical similarity was suggested by McNaughton [1967]. This was further investigated by Knuth [1967] and Paull and Unger [1968].

For an arbitrary context-free grammar $G = (V,\Sigma,P,S)$ the parenthesized version of G, denoted by $G_{()}$, is defined as $G_{()} = ((V \cup \{(,)\},\Sigma \cup \{(,)\},P_{()},S)$ where

$P_{()} = \{A \to (\alpha) : A \to \alpha \text{ is in } P\}$.

Thus $L(G_{()},\Rightarrow)$ is a linear parenthetical coding of the derivation trees of $L(G,\Rightarrow)$.

Given two context-free grammars G_1 and G_2 we say that they are structurally equivalent if $L(G_{1,()},\Rightarrow) = L(G_{2,()},\Rightarrow)$. In other words, for every derivation tree in G_1 with a terminal frontier there is an equally shaped derivation tree in G_2, and vice versa.

It is decidable whether or not two context-free grammars are structurally equivalent as proved in the three papers mentioned above.

An even more restrictive notion of structural equivalence was introduced in Ginsburg and Harrison [1967]. This we now define.

Let $G = (V,\Sigma,P,S)$ be a context-free grammar in which the productions in P are numbered in some arbitrary but unique way. Define the bracketed version of G, denoted by $G_{[]}$, as follows:

Let $\Delta = \{[_i,]_i : i \leq i \leq \#P\}$ and $G_{[]} = (V \cup \Delta, \Sigma \cup \Delta, P_{[]}, S)$, where $P_{[]} = \{A \to [_i \alpha]_i : A \to \alpha$ is the ith production in $P\}$.

In this case each word in $L(G_{[]},\Rightarrow)$ not only contains enough information to reconstruct the shape of its derivation tree (as is the case for $L(G_{(\ )},\Rightarrow)$) but also enough information to label the internal nodes of the tree correctly. The brackets $[_i$ and $]_i$ are known as phrase markers in linguistics.

We say two context-free grammars $G_i = (V_i,\Sigma,P_i,S_i)$, $i = 1, 2$ are strongly structurally equivalent if there is a numbering of the productions of P_1 and P_2 such that under this numbering $L(G_{1,[]},\Rightarrow) = L(G_{2,[]},\Rightarrow)$. Strong structural equivalence is decidable since there are only finitely many numberings and for each such numbering the bracketed version of a grammar defines a simple deterministic language. The transformation of $G_{[]}$ into an s-grammar (simple deterministic grammar) is straightforward. Korenjak and Hopcroft [1966] have proved that it is decidable whether or not two s-grammars generate the same language. More recently Olshansky and Pnueli [1977] and Harrison, Havel and Yehudai []979] have also provided proofs of this result.

While the parenthetical and bracketed versions of grammars provide a stricter notion of grammatical similarity, they are both too restrictive. This comes about in two different ways. First their languages must be identical and second, the derivation trees must be identical up to an isomorphic renaming of their nonterminals.

In the next three sections we consider the affect of relaxing these restrictions somewhat.

However before doing this we mention in passing a notion due to Blattner [1976], which strictly speaking is based on languages rather than grammars. Let L and L' be context-free languages. They are said to be structurally similar if there are a-transducers M and M' such that $M(L) = L'$ and $M'(L') = L$. One of her results is: if L has a structurally similar set of sentential forms and L and L' are structurally similar then L' has a structurally similar set of sentential forms. These results are of some interest for grammar form theory because of the results in Sections II.4.2 and II.4.3 on principality and semi-AFLs.

II.2.2 Covers

In the area of compiler theory and practice the notions of cover and syntax-directed translation have considerable importance. In both cases this involves a concept of grammatical similarity. Let us first examine the notion of cover.

Let $G = (V,\Sigma,P,S)$ be a context-free grammar, Δ be a set of labels with $\#\Delta = \#P$, and label each production in P arbitrarily but uniquely with some label from Δ. Let (G,Δ) denote this labelled version of G. Let $D:S = \alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_r = x$ in Σ^* be a derivation in (G,Δ), where $d_1,\ldots,d_r$ are the labels corresponding to the productions used in D in that order. We say $d = d_1 \ldots d_r$ is a parse, derivation word or Szilard word of (G,Δ) and write $S \Rightarrow^{d} x$. If D is a left derivation then d is a left parse and if D is a right derivation then d is a right parse. $Sz(G,\Delta) = \{d : d \text{ in } \Delta^* \text{ is a parse in } (G,\Delta)\}$ is called the Szilard language of (G,Δ), $LSz(G,\Delta) = \{d : d \text{ in } \Delta^* \text{ is a left parse in } (G,\Delta)\}$ is the left Szilard language, and similarly we obtain the right Szilard language. See Salomaa [1973] for further information on Szilard languages.

We can now define the notion of left cover following Aho and Ullman [1972]. Let $G_i = (V_i,\Sigma,P_i,S_i)$, $i = 1, 2$ be two context-free grammars, Δ_i, $i = 1, 2$ be their label sets and (G_i,Δ_i) be their labelled versions, $i = 1, 2$. We say G_2 left covers G_1 if there is a homomorphism h from Δ_2^* to Δ_1^* such that:

(a) if $S_2 \overset{L}{\Rightarrow}^{d_2} x$ in G_2, then $S_1 \overset{L}{\Rightarrow}^{h(d_2)} x$ in G_1, and

(b) for all d_1 in Δ_1^* such that $S_1 \overset{L}{\Rightarrow}^{d_1} x$ in G_1, there exists d_2 in Δ_2^* such that $h(d_2) = d_1$ and $S_2 \overset{L}{\Rightarrow}^{d_2} x$ in G_2.

Condition (b) is a surjectivity condition. Note that $L(G_1,\Rightarrow) = L(G_2,\Rightarrow)$ is implied by this definition.

Typical questions in this area are:

(i) For each LR(k) grammar G does there exist an LR(1) grammar G' such that G' right covers G? See Mickunas, Lancaster and Schneider [1976].

(ii) Can every λ-free grammar be right covered with a grammar in Greibach normal form? See Nijholt [1979f].

(iii) Given two grammars G and G' is it decidable whether or not G' left or right covers G? See Hunt III, Rosenkrantz and Szymanski [1976]. In the case or arbitrary grammars it is undecidable, but for sub-linear grammars it is in fact decidable.

The emphasis of the work of Hunt III, Rosenkrantz and Szymanski places the study of covers within the area of grammatical similarity, since the basic question (iii) of when two grammars are similar is tackled for the first time in their papers.

Since covering implies weak equivalence and question (iii) has been resolved negatively the usefulness of this notion of similarity is limited as a basis for a theory of grammatical similarity. This, we hasten to add, does not imply that it is an area to be abandoned, since questions of types (i) and (ii) remain meaningful in their original framework.

Other papers in this area are those of Hunt III, Rosenkrantz et al, those of Mickunas et al, those of Nijholt, and the following, Benson [1977], Gray and Harrison [1972], Haskell [1970], Reynolds [1968], Reynolds and Haskell [1970], Soisalon-Soininen [1979] and Ukkonen [1978, 1979].

I.2.3 Translations and Szilard Languages

Just as studies of the parsing process led to the notion of covering, studies of compiling led to the notion of syntax-directed translations.

The reader is referred to Aho and Ullman [1972], which deals with this topic in some detail and includes many of their own contributions.

Let $G = (V,\Sigma,P,S)$ and $G' = (V',\Sigma',P',S')$ be two grammars. We say (G,G',f) is a translation grammar if $f: P \to P'$ is a bijection. Since a translation is a set of pairs (x,x') of words x and x' over some alphabets Σ and Σ', respectively, we denote by $T(G,G',f)$ the translation defined by the translation grammar (G,G',f) and we define it as follows:

$$T(G,G',f) = \{(x,x'): S \Rightarrow^{d} x \text{ in } \Sigma^* \text{ in } G \text{ and } S' \Rightarrow^{f(d)} x' \text{ in } \Sigma'^* \text{ in } G'\}.$$

Often, for example see Aho and Ullman [1972], the relationship of the derivations is even more tightly controlled.

A translation grammar (G,G',f) is said to be agreeable if:

$$S \Rightarrow^{d} x \text{ in } \Sigma^* \text{ in } G \text{ iff } S' \Rightarrow^{f(d)} x' \text{ in } \Sigma'^* \text{ in } G'.$$

Penttonen [1974] had proved that (G,G',f) is agreeable iff:

(i) there is a bijection $g: V - \Sigma \to V' - \Sigma'$ and

(ii) for all $p: A \to x_0 A_1 \ldots A_m x_m$ in P, where $m \geq 0$, the x_i are in Σ^* and the A_i are in $V - \Sigma$, and

for p': $B \to y_0 B_1 \ldots B_n y_n$ in P' such that $p' = f(p)$, then $m = n$, $g(A) = B$, and $(g(A_1),\ldots,g(A_m))$ is a permutation of $(B_1,\ldots,B_m)$.

These are exactly the conditions needed in the definition of a syntax-directed translation schema (see Aho and Ullman [1972, p. 218]), hence the notion of agreeableness is equivalent to requiring that (G,G',f) forms a syntax-directed translation schema.

Moreover Penttonen's [1974] characterization theorem implies that (G,G',f) is agreeable iff $Sz(G,\Delta)$ is <u>isomorphic</u> to $Sz(G',\Delta')$. This leads to a tie-in with Szilard languages.

Kriegel and Maurer [1976] proposed a one-sided version of agreeableness, since in a translation grammar (G,G',f) it is only necessary for each terminating derivation in G, the source language grammar, to have a corresponding terminating derivation in G', the object language grammar, but not necessarily vice versa.

We say (G,G',f) is a <u>fitting</u> translation grammar if whenever $S \Rightarrow {}^{d}x$ in Σ^* in G there is a derivation $S' \Rightarrow {}^{f(d)}x'$ in Σ'^* in G'.

It now follows that (G,G',f) is fitting iff $f(Sz(G,\Delta)) \subseteq Sz(G',\Delta')$, where f is extended to label sets and words over these label sets in a natural way.

Assuming $\Delta = \Delta'$, this reduces to the containment problem for Szilard languages of context-free grammars. Kriegel and Maurer [1976] show that this is decidable.

A refinement of this notion comes from observing that usually only left or right derivations are used in the source language grammar. Thus (G,G',f) is said to be <u>left (right) fitting</u> if whenever $S \overset{L}{\Rightarrow} {}^{d}x$ ($S \overset{R}{\Rightarrow} {}^{d}x$) in Σ^* in G then $S' \Rightarrow {}^{f(d)}x'$ in G', for some x' in Σ'^*.

Again whether a translation grammar (G,G',f) is left fitting reduces to a question concerning their Szilard languages, namely is $f(LSz(G,\Delta)) \subseteq Sz(G',\Delta')$. In Kriegel and Ottmann [1977] this also was shown to be decidable.

We say (G,G',f) is <u>(X,Y) fitting</u>, where X and Y are chosen from $\{L,R,\lambda\}$, if whenever $S \overset{X}{\Rightarrow} {}^{d}z$ in Σ^* in G, then $S' \overset{Y}{\Rightarrow} {}^{f(d)}z'$ in G' for some z' in Σ'^*, where $\overset{\lambda}{\Rightarrow}$ is equivalent $\Rightarrow$.

Linna [1977] has shown that (L,L) fitting is also decidable.

The area of syntax-directed translations has led to another notion of grammatical similarity based on Szilard languages of grammars, which can be called Szilard similarity. A number of cases have been shown to be decidable, therefore this notion is in this respect better than covering. However given a grammar G, all possible candidate

similar grammars must have the sane number of productions. It is this fact which causes us to look elsewhere.

I.2.4 Grammar Morphisms

Let $G = (V,\Sigma,P,S)$ and $G' = (V',\Sigma',P',S')$ be two grammars and $h: V^* \to V'^*$ be a homomorphism such that:

(i) $h(V - \Sigma) \subseteq V' - \Sigma'$,

(ii) $h(\Sigma) \subseteq \Sigma'^*$,

(iii) for all $A \to \alpha$ in P, $h(A \to \alpha) = A' \Rightarrow^+ \alpha'$ in G' where $h(A) = A'$ and $h(\alpha) = \alpha'$, and

(iv) $h(S) = S'$.

In this case, we say that h is a grammar morphism $h: G \to G'$.

This notion was first studied in Hotz [1966] and in the subsequent papers of Benson, Bertsch, Claus, Hotz, Nelson, Nivat, Ross, Schnorr and Walter. Recently Lange [1978] has considered L morphisms, while Kobuchi and his co-workers have studied a special case of L morphisms, called simulations, see Culik II and Kobuchi [1980], Kobuchi [1977], Kobuchi and Seki [1980], Kobuchi and Wood [1980] and Seki and Kobuchi [1980].

Its importance stems from the observation that a grammar $G = (V,\Sigma,P,S)$ defines in a natural way a category (called the free X-category or syntax category) in which the objects are words over V and the morphisms are essentially derivations in G. Moreover this category is a free strict monoidal category. From this viewpoint a grammar morphism is a functor between two syntax categories.

Of all the approaches to grammatical similarity in the previous sections, this one seems to be the most useful and, at the same time, the most natural. The fact that a grammar morphism is a functor for the corresponding syntax categories tends to confirm this.

In Section II.1 the two basic notions of interpretation for context-free grammars will be characterized in terms of particular grammar morphisms. To this end some special grammar morphisms need to be identified.

We say that a grammar morphism $h: G \to G'$, where $G = (V,\Sigma,P,S)$ and $G' = (V',\Sigma',P',S')$ is:

(i) fine if $h(A \to \alpha)$ is in P' for all $A \to \alpha$ in P,

(ii) length preserving if $h(V) \subseteq V'$

(iii) very fine if h is fine and length preserving,

(iv) closed if for every derivation $S' \Rightarrow^+ x'$ in Σ'^* in G' there is a derivation $S \Rightarrow^+ x$ in Σ^* in G such that $h(S \Rightarrow^+ x) = S' \Rightarrow^+ x'$ and

(v) terminal if $h(A) = A$ for all A in $V - \Sigma$.

Given two grammars G and G' it is clearly decidable whether or not there exists a length preserving grammar morphism $h: G \to G'$. Moreover a grammar morphism preserves the structural properties of the source grammar G. Recently Walter [1979] and his co-workers have begun a detailed investigation of this notion of similarity. As we shall see in Section II.1, whether one grammar is an s-interpretation of another is equivalent to whether there exists a very fine grammar morphism between them. Thus an in-depth study of grammar morphisms is long overdue.

An untapped research area is the consideration of collections of grammars based on grammar morphisms rather than grammar forms.

II.2.5 Topological Similarity

Kuroda [1973 a,b and, in particular, 1976] introduced a new notion of grammatical similarity, which he called topological similarity. A related notion was placed in the framework of syntax categories by Walter [1975] and this was followed up by Nelson [1980]. At this time it is unclear how useful these complex similarity measures will be.

We will briefly explain the basis of the similarity measures due to both Kuroda [1976] and Walter [1975].

Let $G = (V,\Sigma,P,S)$ be a context free grammar. For $Q \subseteq P$ and α,β in V^*, we write $\alpha \underset{Q}{\Rightarrow} {}^{d}\beta$ if $\alpha \Rightarrow {}^{d}\beta$ in G, where d is in Q^*. Similarly we write $\alpha \underset{Q}{\Rightarrow}^{*} \beta$ is such a d in Q^* exists. Let $\gamma: \alpha \Rightarrow^{*} \beta$ be a derivation in G, then recall that $\tau(\gamma)$ denotes its corresponding tree. In the following we will discuss topological similarity by way of derivation trees. However any proofs are better dealt with in terms of derivations, for example compare Kuroda [1976] with Nelson [1980] and Walter [1975].

Let $Tree(G) = \{\tau(\gamma): \gamma: S \Rightarrow^{*} x \text{ in } G \text{ with } x \text{ in } \Sigma^*\}$, that is all derivation trees of G with terminal frontiers. Let $Q \subseteq P$ and τ be in Tree(G), then τ_Q is the maximal tree-prefix of τ which only consists of productions in Q. For all $Q \subseteq P$ and all τ in Tree(G) τ_Q clearly exists. Of course it may be degenerate, that is consist of a single node labelled S.

For all $Q \subseteq P$ and for all τ and τ' in Tree(G) we write $\tau \leq_Q \tau'$ iff τ_Q is a tree-prefix of τ'_Q.

Clearly $\leq_Q$ is reflexive and transitive hence it is a pre-order (or quasi-order). Moreover $(Tree(G), \leq_Q)$ is a lattice, since

given τ and τ' their gcd and lcm under $\leq_Q$ exists and is unique.

Using the natural topology which is defined by a quasi-ordered set we have:

For all $Q \subseteq P$, for all $T \subseteq \text{Tree}(G)$:

$\text{Open}(T,Q) = \{\tau' : \tau' \text{ is in Tree}(G), \tau \text{ is in } T \text{ and } \tau \leq_Q \tau'\}$.

Tree(G) is the set of points of the toplogical space and the open sets are all Open(T,Q), for all $T \subseteq$ Tree(G). Each point τ in Tree(G) has a smallest open neighborhood, namely Open $(\{\tau\},Q)$. We denote this topology by $\mathfrak{J}(G,Q)$.

Since $(\text{Tree}(G), \leq_Q)$ is a lattice for every G and Q, a bijection f from Tree(G) to Tree(G') is said to be <u>structurally continuous</u> if for all $Q \subseteq P$ there exists $Q' \subseteq P'$ such that for all τ,τ' in Tree(G), $\tau \leq_Q \tau'$ implies $f(\tau) \leq_{Q'} f(\tau')$, that is it is order preserving. In this case f is continuous in the toplogical sense with respect to $\mathfrak{J}(G,Q)$ and $\mathfrak{J}(G',Q')$. Numerous definitions of the similarity of two grammars can be obtained on the basis of these definitions. Let us consider one such definition.

We say $G = (V,\Sigma,P,S)$ is <u>structurally similar</u> to $G' = (V',\Sigma',P',S')$, written $G \leq_t G'$, there exists a bijection f: Tree(G) $\to$ Tree(G') which is structurally continuous. We can define two grammars G and G' to be <u>structurally equivalent</u> if $G \leq_t G'$ and $G' \leq_t G$.

However at the time of writing it is not even known whether $\leq_t$ is decidable. In all the studies so far, with the exception of Nelson and Wood [1980], there has been no detailed investigation of any of the possible measures. In particular two basic questions have remained unanswered for most cases. First, is $\leq_t$ decidable and more generally are any of the measures proposed in Kuroda [1976] and Walter [1975] decidable? Second, are any of the measures meaningful in the sense that (i) they provide a nontrivial classification of the context-free grammars and (ii) they provide a consistent classification, for example an expansive grammar is never similar to a right linear grammar. In Nelson and Wood [1980] there is a partial answer to the first concern and a complete answer to the second, in both cases for a specific topological similarity measure.

Before closing this section we also glance briefly at Kuroda's notions.

Let $\text{Alltree}(G) = \{\tau(\gamma) : \gamma : A \Rightarrow^* x \text{ in } G \text{ for some } A \text{ in } V - \Sigma$ and $x \text{ in } \Sigma^*\}$,

that is all derivation trees with terminal frontiers having any non-terminal as their root symbol. Let $\pi \subseteq$ Alltree(G) and π be finite, then π is a pruning set and each member of π is a prune.

Let τ be in Tree(G) and $\pi \subseteq$ Alltree(G) be a pruning set. Notice that each prune p in π can only appear in τ as a tree-suffix since p has a terminal frontier. We prune τ with π by removing tree-suffixes of τ which are members of π. Moreover we do this in such a way that τ is pruned by π to the maximum extent. It can be proved that this leaves a unique tree prefix of τ which we denote by τ_π. For all pruning sets $\pi \subseteq$ Alltree(G) and for all τ and τ' in Tree(G) we write $\tau \leq_\pi \tau'$ iff τ_π is a tree prefix of τ'_π.

As with Walter's approach we can now define Open(T,π), for all $T \subseteq$ Tree(G) and all pruning sets $\pi \subseteq$ Alltree(G), by:

Open(T,π) = $\{\tau'$: τ' is in Tree(G), τ is in T and $\tau \leq_\pi \tau'\}$.

Again Open($\{\tau\}$,π) is the smallest open neighbourhood of the point τ and as before we can define a notion of similarity based on these ideas. We denote the pruning set topology with respect to π by $\mathfrak{J}(G,\pi)$.

Let G and G' be two grammars and f be a bijection f: Tree(G) $\rightarrow$ Tree(G'). We again say f is structurally continuous if:

> For all $\pi \subseteq$ Alltree(G), there exists $\pi' \subseteq$ Alltree(G') such that f is order-preserving with respect to π and π', that is if $\sigma \leq_\pi \tau$ in G then $f(\sigma) \leq_{\pi'} f(\tau)$ in G'. In other words f: $\mathfrak{J}(G,\pi) \rightarrow \mathfrak{J}(G',\pi')$ is continuous.

We say f is a structural homeomorphism iff f and f^{-1} are structurally continuous.

Again it is not known whether or not structural homeomorphism is decidable.

However Kuroda [1976] does not base his main notion of topological similarity solely on $\mathfrak{J}(G,\pi)$ but rather on the join of $\mathfrak{J}(G,\pi)$ with another topology $\mathfrak{J}^*$ generated by the "partial" trees of G. This new toplogy is denoted by $\mathfrak{J}^*(G,\pi)$. He demonstrates the remarkable result that if f: Tree(G) $\rightarrow$ Tree(G') is a structural homeomorphism then f is a structural *-homeomorphism, where structural *-homeomorphism is defined in the same way as structural homeomorphism, replacing $\mathfrak{J}(G,\pi)$ by $\mathfrak{J}^*(G,\pi)$ in the definition.

I.2.6 Grammar Collections

Cremers and Ginsburg [1975] introduced the notion of a grammar form as, among other aims, a new approach to the study of grammatical similarity. Each grammar G gives rise to a collection of grammars $\mathcal{G}(G)$, all of which are similar in a strong sense. For each G' in $\mathcal{G}(G)$ each production in G' is an "image" of some production in G, in fact if terminals are ignored there is essentially a very fine grammar morphism from G' to G. A grammar used to define a collection in this way is termed a grammar form.

It enables some basic similarity questions to be posed and solved, that is:

(i) is $\mathcal{G}(G_1) = \mathcal{G}(G_2)$?

(ii) is $\mathcal{L}(G_1) = \mathcal{L}(G_2)$,

where $\mathcal{L}(G_i) = \{L(G_i', \Rightarrow): G_i' \text{ is in } \mathcal{G}(G_i)\}$, i = 1, 2.

If the answer to question (i) is affirmative then G_1 and G_2 are very similar and we say they are strong form equivalent, whereas if the answer to question (ii) is affirmative, then G_1 and G_2 are similar but not very similar, hence we say they are form equivalent. These similarity definitions should be compared with those of structural and weak equivalence for grammars.

In the future perhaps the major impact of grammar forms will be seen to be the idea of grammar collections defined by a grammar. It is the purpose of these notes to demonstrate that this idea gives rise to many interesting and fundamental questions in grammar and language theory as well as contributing to the study of grammatical similarity. For example, two grammars G and G' are said to be $\mathcal{L}$-similar if $\mathcal{L}(G) = \mathcal{L}(G') = \mathcal{L}$ for some given language family $\mathcal{L}$. What does a grammar which is $\mathcal{L}(CF)$-similar, look like? Is it decidable whether or not G is $\mathcal{L}$-similar for a given $\mathcal{L}$? And so on. The questions explode.