7.6   Tree Domains and Gorn Trees

Derivation trees play a very important role in parsing theory and in the proof of a strong version of the pumping lemma for the context-free languages known as Ogden's lemma. Thus, it is important to define derivation trees rigorously. We do so using *Gorn trees*.
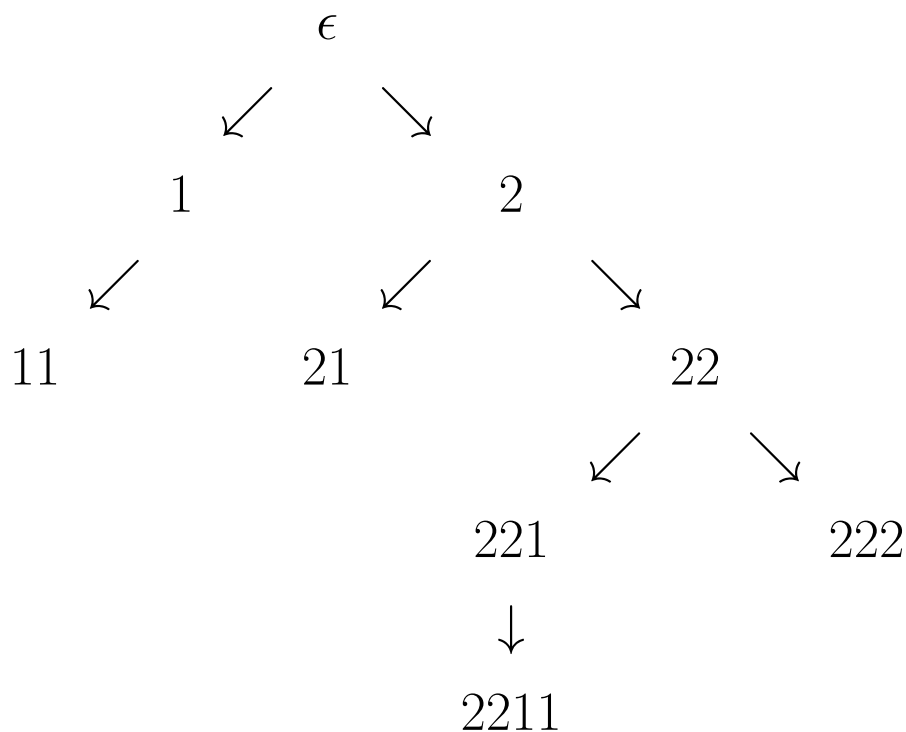
Let $\mathbf{N}_+ = \{1, 2, 3, \ldots\}$.

Definition 7.7. A *tree domain* D is a nonempty subset of strings in $\mathbf{N}^*_+$ satisfying the conditions:

(1) For all $u, v \in \mathbf{N}^*_+$, if $uv \in D$, then $u \in D$.

(2) For all $u \in \mathbf{N}^*_+$, for every $i \in \mathbf{N}_+$, if $ui \in D$ then $uj \in D$ for every j, $1 \leq j \leq i$.

The tree domain

$$D = \{\epsilon, 1, 2, 11, 21, 22, 221, 222, 2211\}$$

is represented as follows:

$$
\begin{array}{ccccccc}
 & & & \epsilon & & & \\
 & & \swarrow & & \searrow & & \\
 & 1 & & & & 2 & \\
\swarrow & & \swarrow & & & & \searrow \\
11 & & 21 & & & & 22 \\
 & & & & \swarrow & & \searrow \\
 & & & 221 & & & 222 \\
 & & & \downarrow & & & \\
 & & & 2211 & & & \\
\end{array}
$$

A tree labeled with symbols from a set Δ is defined as follows.

Definition 7.8. Given a set Δ of labels, a Δ-*tree* (for short, a *tree*) is a total function $t : D \to \Delta$, where D is a tree domain.

The domain of a tree t is denoted as $dom(t)$. Every string $u \in dom(t)$ is called a *tree address* or a *node*.

Let $\Delta = \{f, g, h, a, b\}$. The tree $t : D \to \Delta$, where D is the tree domain of the previous example and t is the function whose graph is

$$\{(\epsilon, f), (1, h), (2, g), (11, a), (21, a),$$
$$(22, f), (221, h), (222, b), (2211, a)\}$$

is represented as follows:

$$f$$

$$h \qquad g$$

$$a \qquad a \qquad f$$

$$h \qquad b$$

$$a$$
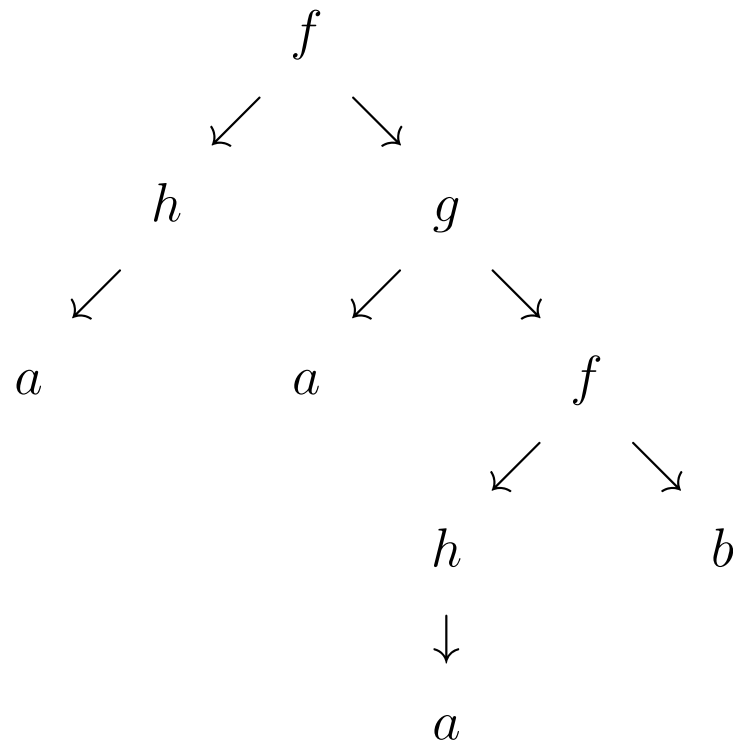
The *outdegree* (sometimes called *ramification*) $r(u)$ of a node u is the cardinality of the set

$$\{i \mid ui \in dom(t)\}.$$

Note that the outdegree of a node can be infinite. Most of the trees that we shall consider will be *finite-branching*, that is, for every node u, $r(u)$ will be an integer, and hence finite. If the outdegree of all nodes in a tree is bounded by n, then we can view the domain of the tree as being defined over $\{1, 2, \ldots, n\}^*$.

A node of outdegree 0 is called a *leaf*. The node whose address is ϵ is called the *root* of the tree. A tree is *finite* if its domain $dom(t)$ is finite. Given a node u in $dom(t)$, every node of the form ui in $dom(t)$ with $i \in \mathbf{N}_+$ is called a *son* (or *immediate successor*) of u.

Tree addresses are totally ordered *lexicographically*:
$u \leq v$ if either u is a prefix of v or, there exist strings
$x, y, z \in \mathbf{N}_+^*$ and $i, j \in \mathbf{N}_+$, with $i < j$, such that
$u = xiy$ and $v = xjz$.

In the first case, we say that u is an *ancestor*
(or *predecessor*) of v (or u *dominates* v) and in the
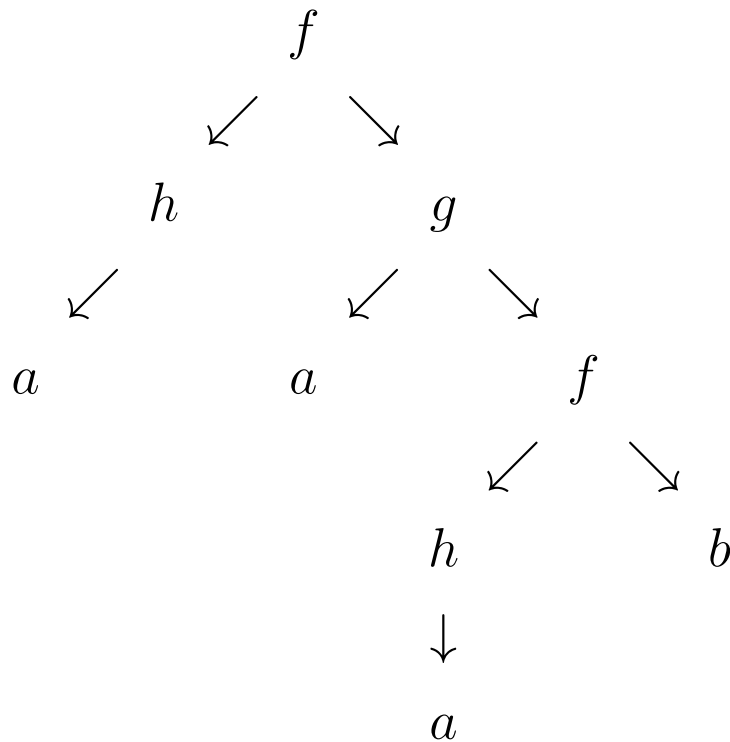second case, that u is *to the left* of v.

If $y = \epsilon$ and $z = \epsilon$, we say that xi is a *left brother* (or
left sibling) of xj, $(i < j)$. Two tree addresses u and v
are *independent* if u is not a prefix of v and v is not a
prefix of u.

Given a finite tree t, the *yield* of t is the string

$$t(u_1)t(u_2)\cdots t(u_k),$$

where $u_1, u_2, \ldots, u_k$ is the sequence of leaves of t in lexicographic order.

For example, the yield of the tree below is $aaab$:

$$
\begin{array}{c}
f \\
\swarrow \qquad \searrow \\
h \qquad\qquad\qquad g \\
\swarrow \qquad\quad \swarrow \qquad \searrow \\
a \qquad\quad a \qquad\qquad f \\
\swarrow \qquad \searrow \\
h \qquad\qquad b \\
\downarrow \\
a
\end{array}
$$

Given a finite tree t, the *depth* of t is the integer

$$d(t) = \max\{|u| \mid u \in dom(t)\}.$$

Given a tree t and a node u in $dom(t)$, the *subtree rooted at u* is the tree t/u, whose domain is the set

$$\{v \mid uv \in dom(t)\}$$

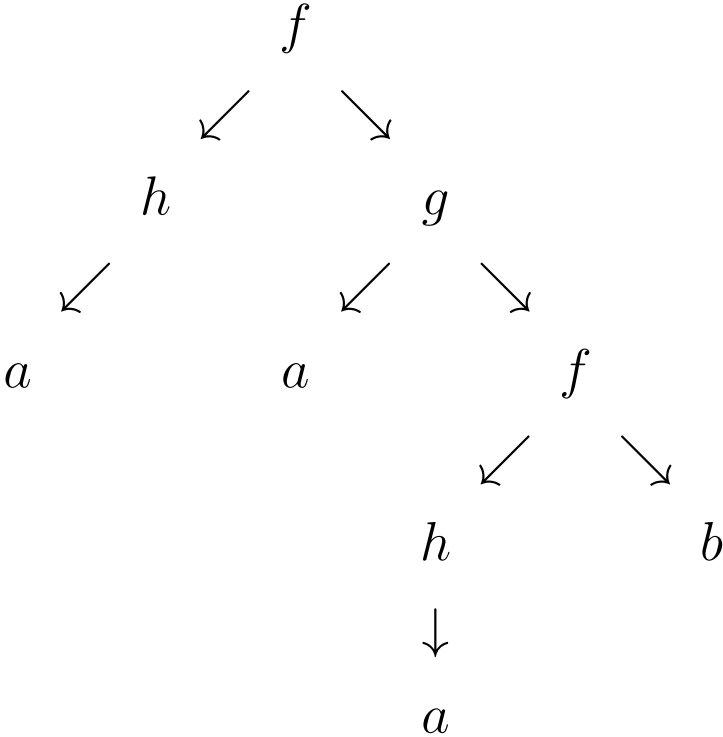and such that $t/u(v) = t(uv)$ for all v in $dom(t/u)$.

Another important operation is the operation of tree replacement (or tree substitution).

Definition 7.9. Given two trees t_1 and t_2 and a tree address u in t_1, the *result of substituting* t_2 at u in t_1, denoted by $t_1[u \leftarrow t_2]$, is the function whose graph is the set of pairs
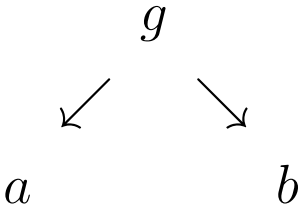
$$\{(v, t_1(v)) \mid v \in dom(t_1),\ u \text{ is not a prefix of } v\}$$
$$\cup\ \{(uv, t_2(v)) \mid v \in dom(t_2)\}.$$

Let t_1 and t_2 be the trees defined by the following diagrams:

Tree t_1

```
              f
            ↙   ↘
          h       g
        ↙       ↙   ↘
      a       a       f
                    ↙   ↘
                  h       b
                  ↓
                  a
```

Tree t_2

```
        g
      ↙   ↘
    a       b
```
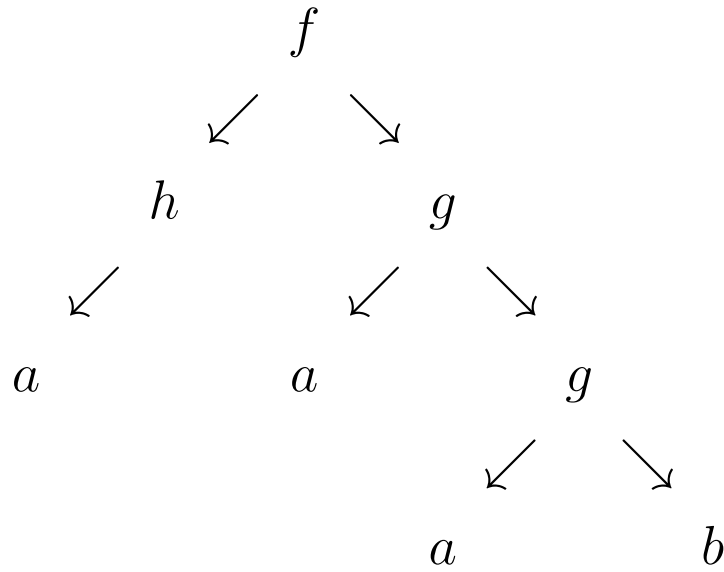
The tree $t_1[22 \leftarrow t_2]$ is defined by the following diagram:

We can now define derivation trees and relate derivations to derivation trees.

7.7 Derivations Trees

Definition 7.10. Given a context-free grammar $G = (V, \Sigma, P, S)$, for any $A \in N$, an *A-derivation tree for G* is a $(V \cup \{\epsilon\})$-tree t (a tree with set of labels $(V \cup \{\epsilon\})$) such that:

(1) $t(\epsilon) = A$;

(2) For every nonleaf node $u \in dom(t)$, if $u1, \ldots, uk$ are the successors of u, then either there is a production $B \to X_1 \cdots X_k$ in P such that $t(u) = B$ and $t(ui) = X_i$ for all i, $1 \leq i \leq k$, or $B \to \epsilon \in P$, $t(u) = B$ and $t(u1) = \epsilon$. A *complete derivation (or parse tree)* is an S-tree whose yield belongs to Σ^*.

Derivations trees are associated to derivations inductively as follows.

Definition 7.11. Given a context-free grammar $G = (V, \Sigma, P, S)$, for any $A \in N$, if $\pi : A \stackrel{n}{\Longrightarrow} \alpha$ is a derivation in G, we construct an *A-derivation tree t_π with yield α* as follows.

(1) If $n = 0$, then t_π is the one-node tree such that $dom(t_\pi) = \{\epsilon\}$ and $t_\pi(\epsilon) = A$.

(2) If $A \stackrel{n-1}{\Longrightarrow} \lambda B\rho \Longrightarrow \lambda\gamma\rho = \alpha$, then if t_1 is the A-derivation tree with yield $\lambda B\rho$ associated with the derivation $A \stackrel{n-1}{\Longrightarrow} \lambda B\rho$, and if t_2 is the tree associated with the production $B \to \gamma$ (that is, if

$$\gamma = X_1 \cdots X_k,$$

then $dom(t_2) = \{\epsilon, 1, \ldots, k\}$, $t_2(\epsilon) = B$, and $t_2(i) = X_i$ for all i, $1 \leq i \leq k$, or if $\gamma = \epsilon$, then $dom(t_2) = \{\epsilon, 1\}$, $t_2(\epsilon) = B$, and $t_2(1) = \epsilon$), then

$$t_\pi = t_1[u \leftarrow t_2],$$

where u is the address of the leaf labeled B in t_1. The tree t_π is the *A-derivation tree associated with the derivation $A \stackrel{n}{\Longrightarrow} \alpha$.*

The following lemma is easily shown.

Lemma 7.6. *Let $G = (V, \Sigma, P, S)$ be a context-free grammar. For any derivation $A \stackrel{n}{\Longrightarrow} \alpha$, there is a unique A-derivation tree associated with this derivation, with yield α. Conversely, for any A-derivation tree t with yield α, there a unique leftmost derivation $A \stackrel{*}{\underset{lm}{\Longrightarrow}} \alpha$ in G having t as its associated derivation tree.*

We will now prove a strong version of the pumping lemma for context-free languages due to Bill Ogden (1968).

7.8   Ogden's Lemma

Ogden's lemma states some combinatorial properties of parse trees that are deep enough. The yield w of such a parse tree can be split into 5 substrings u, v, x, y, z such that

$$w = uvxyz,$$

where u, v, x, y, z satisfy certain conditions. It turns out that we get a more powerful version of the lemma if we allow ouselves to *mark* certain occurrences of symbols in w before invoking the lemma. We can imagine that *marked occurrences* in a nonempty string w are occurrences of symbols in w in boldface, or red, or any given color. For example, given $w = aaababbbaa$, we can mark the symbols of even index as follows:

$$a\mathbf{aa}b\mathbf{a}b\mathbf{bb}a\mathbf{a}.$$

A *marking* of a nonnull string $w\colon \{1, \ldots, n\} \to \Sigma$ is any function $m\colon \{1, \ldots, n\} \to \{0, 1\}$. Then, a letter w_i in w is a *marked occurrence* iff $m(i) = 1$, and an *unmarked occurrence* if $m(i) = 0$. The number of marked occurrences in w is

$$\sum_{i=1}^{n} m(i).$$

Lemma 7.7. *For every context-free grammar G, there is some integer $K > 1$ such that, for every string $w \in \Sigma^+$, for every marking of w, if $w \in L(G)$ and w contains at least K marked occurrences, then there exists some decomposition of w as $w = uvxyz$, and some $A \in N$, such that the following properties hold:*

(1) There are derivations $S \overset{+}{\Longrightarrow} uAz$, $A \overset{+}{\Longrightarrow} vAy$, and $A \overset{+}{\Longrightarrow} x$, so that

$$uv^n xy^n z \in L(G)$$

for all $n \geq 0$ (the pumping property);

(2) x contains some marked occurrence;

(3) Either (both u and v contain some marked occurrence), or (both y and z contain some marked occurrence);

(4) vxy contains less than K marked occurrences.

Proof. The general idea is to make sure that K is large enough so that parse trees with yield w contain enough repeated nonterminals along some path from the root to some marked leaf. Let $r = |N|$, and let

$$p = \max\{2, \ \max\{|\alpha| \mid (A \to \alpha) \in P\}\}.$$

We claim that $K = p^{2r+3}$ does the job.

The condition $p \geq 2$ is needed to ensure that $p^m \leq p^n$ implies that $m \leq n$.

The key concept of the proof is the notion of a B-node. Given a parse tree, a *B-node* is a node with at least two immediate successors u_1, u_2, such that for $i = 1, 2$, either u_i is a marked leaf, or u_i has some marked leaf as a descendant.

We construct a path from the root to some marked leaf, so that for every B-node, we pick the leftmost successor with the maximum number of marked leaves as descendants. $\square$

The "standard pumping lemma" due to Bar-Hillel, Perles, and Shamir, is obtained by letting all occurrences be marked in $w \in L(G)$.

Lemma 7.8. *For every context-free grammar G (without ϵ-rules), there is some integer $K > 1$ such that, for every string $w \in \Sigma^+$, if $w \in L(G)$ and $|w| \geq K$, then there exists some decomposition of w as $w = uvxyz$, and some $A \in N$, such that the following properties hold:*

(1) There are derivations $S \overset{+}{\Longrightarrow} uAz$, $A \overset{+}{\Longrightarrow} vAy$, and $A \overset{+}{\Longrightarrow} x$, so that

$$uv^n xy^n z \in L(G)$$

for all $n \geq 0$ (the pumping property);

(2) $x \neq \epsilon$;

(3) Either $v \neq \epsilon$ or $y \neq \epsilon$;

(4) $|vxy| \leq K$.

A stronger version could be stated, and we are just following tradition in stating this standard version of the pumping lemma.

The pumping lemma or Ogden's lemma can be used to show that certain languages are not context-free.

The method is to proceed by contradiction, i.e., to assume (contrary to what we wish to prove) that a language L is indeed context-free, and derive a contradiction of Ogden's lemma (or of the pumping lemma).

Thus, as in the case of the regular languages, it would be helpful to see what the negation of Ogden's lemma is, and for this, we first state Ogden's lemma as a logical formula.

For any nonnull string $w\colon \{1,\ldots,n\} \to \Sigma$, for any marking $m\colon \{1,\ldots,n\} \to \{0,1\}$ of w, for any substring y of w, where $w = xyz$, with $|x| = h$ and $k = |y|$, the number of marked occurrences in y, denoted as $|m(y)|$, is defined as

$$|m(y)| = \sum_{i=h+1}^{i=h+k} m(i).$$

We will also use the following abbreviations:

$$
\begin{aligned}
nat &= \{0,1,2,\ldots\}, \\
nat32 &= \{32,33,\ldots\}, \\
A &\equiv w = uvxyz, \\
B &\equiv |m(x)| \geq 1, \\
C &\equiv (|m(u)| \geq 1 \wedge |m(v)| \geq 1) \\
&\qquad \vee\, (|m(y)| \geq 1 \wedge |m(z)| \geq 1), \\
D &\equiv |m(vxy)| < K, \\
P &\equiv \forall n\colon nat\ (uv^n xy^n z \in L(D)).
\end{aligned}
$$

Ogden's lemma can then be stated as

$$\forall G\colon \text{CFG}\ \exists K\colon nat32\ \forall w\colon \Sigma^*\ \forall m\colon \text{marking}$$
$$\Bigg((w \in L(D) \wedge |m(w)| \geq K)$$
$$\supset (\exists u, v, x, y, z\colon \Sigma^*\ A \wedge B \wedge C \wedge D \wedge P)\Bigg).$$

Recalling that

$$\neg(P \wedge Q) \equiv \neg P \vee \neg Q \equiv P \supset \neg Q$$

and

$$\neg(P \supset Q) \equiv P \wedge \neg Q,$$

the negation of Ogden's lemma can be stated as

$$\exists G\colon \text{CFG}\ \forall K\colon nat32\ \exists w\colon \Sigma^*\ \exists m\colon \text{marking}$$
$$\Bigg((w \in L(D) \wedge |m(w)| \geq K)$$
$$\wedge\ (\forall u, v, x, y, z\colon \Sigma^*\ (A \wedge B \wedge C \wedge D) \supset \neg P)\Bigg).$$

Since

$$\neg P \equiv \exists n \colon nat \ (uv^n x y^n z \notin L(D)),$$

in order to show that Ogden's lemma is contradicted, one needs to show that for some context-free grammar G, for every $K \geq 2$, there is some string $w \in L(D)$ and some marking m of w with at least K marked occurrences in w, such that for every possible decomposition $w = uvxyz$ satisfying the constraints $A \wedge B \wedge C \wedge D$, there is some $n \geq 0$ such that $uv^n x y^n z \notin L(D)$.

When proceeding by contradiction, we have a language L that we are (wrongly) assuming to be context-free and we can use any CFG grammar G generating L.

The creative part of the argument is to pick the right $w \in L$ and the right marking of w (not making any assumption on K).

As an illustration, we show that the language

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

is not context-free. Since L is infinite, we can use Ogden's lemma.

We can also show that the language

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

is not context-free.

Ogden's lemma can also be used to show that the language

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

is inherently ambiguous. The proof is quite involved.

Another corollary of the pumping lemma is that it is decidable whether a context-free grammar generates an infinite language.

Lemma 7.9. *Given any context-free grammar, G, if K is the constant of Ogden's lemma, then the following equivalence holds:*

$L(G)$ is infinite iff there is some $w \in L(G)$ such that $K \leq |w| < 2K$.

In particular, if G is in Chomsky Normal Form, it can be shown that we just have to consider derivations of length at most $4K - 3$.