COL352 Lecture 14

Contents

1 Recap

Discussion in previous class about Myhill-Nerode theorem and starting of context free languaages.

2 Definitions

3 Content

3.1 Context-Free Languages

Motivating example:

Example 1

Inductively define L as the smallest language satisfying the following:

1. $\epsilon \in L$.
2. If $x, y \in L$ then $x \cdot y \in L$.
3. If $x \in L$, then $0x1 \in L$.

Note 1

This is something like balanced parenthesized expressions. We needed the smallest language thing because any superset of L also works.

Claim 0.1

$x \in L \iff$ the number of 0s in x = number of 1s in x, and for all y (prefixes of x), the number of 0s in y is at least the number of 1s in y.

Proof. Exercise. □

Claim 0.2

This language is not regular.

Proof. Use the pumping lemma on 0^n1^n or give another proof using the Myhill-Nerode theorem. □

This is something like a grammar.
In what follows, S is the initial non-terminal.

1. $S \to \epsilon$

2. $S \to SS$

3. $S \to 0S1$

$0, 1$ are called terminals, S is a non-terminal, the above three rules are called the production rules for the grammar.
We can make parse trees using this.

Example 2

We want to make a primitive calculator to add and multiply non-negative integers. $\Sigma = \{0, \cdots, 9, +, *, (,)\}$

$E \to E + T|T$

$T \to T * F|F$

$F \to (E)|N$

$N \to 0N|1N|\cdots|9N|0|1|\cdots|9$

Here E stands for an expression, T for a term, F for a factor, N for a number.

$X \to Y|Z$ is shorthand for the two rules $X \to Y$ and $X \to Z$.

Definition 1

A grammar (context-free grammar) G is a $4-$tuple of the form (N, Σ, R, S) where

1. N = a finite nonempty set of "nonterminals" (a.k.a. "variables"),
2. Σ = a finite nonempty alphabet (set of terminals)
3. $R \subseteq N \times ((N \cup \Sigma)^*)$ = the set of production rules
4. $S \in N$ = initial non-terminal

Note that $(X, w) \in R$ is also written as $X \to_G w$ or $X \to w$ if G is clear from context.

Note that this is analogous to regular expressions (i.e., this is not a machine).
We shall make the 'expansion' of a non-terminal more formal in the following definition.

Definition 2

Let $G = (N, \Sigma, R, S)$ be a grammar, and $x, y \in (N \cup \Sigma)^*$. We say v produces y and write $x \implies_G y$ if $\exists X \in N, u, v, w \in (N \cup \Sigma)^*$ such that $x = uXv, y = uwv, X \to_G w \in R$.

Definition 3

We say "x derives y", and write $x \overset{*}{\implies}_G y$ if $\exists$ a finite sequence of strings $z_0, \ldots, z_n \in (N \cup \Sigma)^*$ such that $x = z_0, y = z_n$ and $\forall i, z \implies z_{i+1}$.

Note that the derivation relation is the reflexive transitive closure of the production relation.

Definition 4

The language of grammar $G = (N, \Sigma, R, S)$, written as $\mathcal{L}(G)$ is the set of all $x \in \Sigma^*$ such that $S \overset{*}{\implies}_G x$.

Example 3

$E \implies E+T \implies T+T \implies T+T*F \implies \cdots \implies F+F*F \implies \cdots \implies 1+2*3$ - the derivation of $1 + 2 * 3$.

Example 4

Using the first example we did, we can do $S \implies 0S1 \implies 0SS1 \overset{*}{\implies} 00S10S11 \overset{*}{\implies} 001011$.

The parse tree's yield is what you get from reading the parse tree's leaves in left to right order (parse tree doesn't need to have leaves as terminals).

Definition 5

Let $G = (N, \Sigma, R, S)$ be a grammar. The set of parse trees of G and the root and yield of each parse tree is recursively defined as follows:

1. If $A \in N \cup \Sigma$, then A is a parse tree. Its root is A and the yield is A.
2. If $X \to \epsilon \in R$, then $X - \epsilon$ is a parse tree. Its root is X, yield is E.
3. If $X \to X_1 X_2 \cdots X_n \in R$, and $R_1, \cdots, T_n$ are parse trees with roots $X_1, \cdots, X_n$, and yield $y_1, \cdots, y_n$, then $X - T_1, X - T_2, \cdots X - T_n$ is a parse tree with root x and yield $y_1 \cdots y_n$.

Definition 6

The definition of the language of a grammar can be equivalently stated as: $\mathcal{L}(G) = \{x \in \Sigma^* \mid x$ is the yield of a parse tree of

Note 2

Note that the yield of a parse tree is a string in $(N \cup \Sigma)^*$.

Definition 7

L is a context-free language (CFL) if there exists a grammar G such that $L = \mathcal{L}(G)$.

Example 5

Denote $S \to \epsilon|0S1|SS$. Then $S \implies SS \implies S0S1 \implies S01$ proves that the yield is neither the pre-order traversal or the bfs traversal.
Further, consider the string 0101.
$S \implies SS \implies 0S1S \implies 0S10S1 \implies 010S1 \implies 0101$
and
$S \implies SS \implies S0S1 \implies 0S10S1 \implies 010S1 \implies 0101$
and so on, all give the same parse tree.
So this shows that there can be multiple derivations for the same parse tree (consider labelling the expanded node by expansion time for uniqueness).

Definition 8

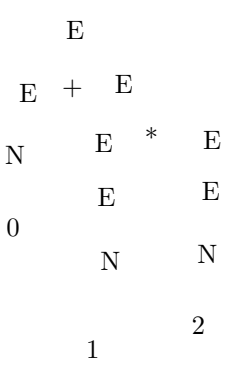
$Z_0 \implies X_1 \implies \cdots \implies Z_n$ is called a leftmost derivation if in each production $Z_i \implies Z_{i+1}$ a production rule is applies to the leftmost non-terminal of Z_i (essentially a pre-order traversal).

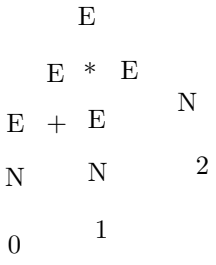
Claim 0.3

There is a one-one correspondence between parse trees yielding strings in $\mathcal{L}(G)$ and the leftmost derivations of strings in $\mathcal{L}(G)$.

Proof. Exercise – should be intuitively clear. □

We show that there can be multiple parse trees for the same string:
Consider $E \to E + E|E * E|(E)|N$.





Definition 9

Grammar G is said to be ambiguous if there is some string $x \in \mathcal{L}(G)$ which has two leftmost derivations (i.e., two parse trees).