Lecture #8: Modelling Computation of Languages
Lecture Presentation

First Problem: Developing a Phrase-Structure Grammar

Let

$$\Sigma_D = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

and consider the language $L_{\text{unpad}} \subseteq \Sigma_D^\star$ of ***unpadded*** decimal representations of natural numbers — that is, the union of the set $\{0\}$ and the set of strings beginning with any digit except "0".

The first problem to be solved is to design a ***phrase-structure grammar*** for the language L_{unpad}. Recall that a phrase-structure grammar is a 4-tuple

$$G = (V, \Sigma, \Pi, S)$$

where

- V is a finite, nonempty set of ***variables***.
- Σ is an ***alphabet*** — such that $\Sigma \cap V = \emptyset$.
- Π is a ***string rewriting system*** over the alphabet $V \cup \Sigma$.
- S is a variable in S, called the ***start variable*** of G.

Since we wish to design a phrase-structure for a language over the alphabet Σ_D, we must set Σ to be Σ_D when developing a suitable phrase-structure grammar G. The set V of variables, the start variable S, and the rewrite rules in the string rewriting system Π will be developed by considering the language L_{unpad}

Let us start by setting V to be $\{S\}$ and Π to be $\emptyset$.

Rewrite Rules for Start Variable

The start variable, S, should be able to derive all the strings in L_{unpad} — and no other strings in $\Sigma_D^\star$. Since $0 \in L_{\text{unpad}}$ (and L_{unpad} is the union of $\{0\}$ and another set of strings with a well-defined structure) let us begin by adding the rewrite rule

$$S \to 0 \tag{1}$$

to the string rewriting system Π.

Let us introduce another variable, P to V — so that $V = \{S, P\}$ at this point. P will be used to derive all the unpadded decimal representations of ***positive integers*** — so that we should also add the rewrite rule

$$S \to P \tag{2}$$

— and then include rewrite rules, with P on the left-hand side, as needed.

Rewrite Rules for the Variable P

Recall that the unpadded binary representation of a positive integer begins with a ***nonzero digit*** and is followed by a sequence of zero or more ***digits***. Let us add two more variables, N, and D — so that (at this point) $V = \{S, P, N, D\}$. The variable N will be used to derive nonzero digits, and the variable D will be used to derive digits. Rewrite rules "for P" (that is, with the variable P on the left hand side) will have expressions that may include N, D or P on the right.

Rewrite Rules for N:

Rewrite Rules for D:

Rewrite Rules for P:

Example: A Derivation of “3802”:

Proving the Correctness of This Phrase-Structure Grammar

Second Problem: Developing a Markov Algorithm

Let $f_{+1} : \Sigma_D^\star \to \Sigma_D$ be the following total function.

- If $x \in L_{\text{unpad}}$, so that x is the unpadded decimal representation of some natural number n, then $f_{+1}(x)$ is the unpadded decimal representation of $n + 1$.
- If $x \notin L_{\text{unpad}}$ then $f_{+1}(x) = \lambda$, the empty string.

Computing This Function with a Turing Machine

Emulating This with a Markov Algorithm

Getting Started:

Figuring Out How To Proceed — and Handling an Empty Input:

Continuing When the Input is in L_{unpad}:

Continuing When the Input is not in L_{unpad}:

Summary: The Scheme That has Been Obtained:

Proving the Correctness of This Markov Algorithm