CS109B Notes for Lecture 5/8/95

Expressive Power of Languages

There are several different schemes we have seen for describing languages:

1. DFA, NFA, RE, for defining "regular sets."

2. Grammars for *context-free languages.*

There is a rough tradeoff: Regular sets are a proper subset of the languages that grammars can define, but it is easier to build recognizers and processors for regular sets than for context-free languages.

- Practical consequence: compilers and similar text-processing software generally have two components:

    1. A *lexical analyzer* for aspects of the input that can be described by regular sets (e.g., form of identifiers).

    2. A *parser* for aspects that need the power of a grammar, e.g., nested statements, expressions.

Grammars Can Simulate RE's

Structural induction on the expression tree for a RE that there is a grammar one of whose SC's has the language of the subexpression dangling from a node.

Basis: Leaf: If labeled a, then

$$< S > \rightarrow a$$

works.

- For ϵ, the same with ϵ in place of a.

- For $\emptyset$, just $< S >$ with no production.

Induction: Suppose we have grammars for subexpressions R_1 and R_2.

- Assume these grammars have no SC's in common (rename if necessary). Let S_1, S_2 be their "starting" SC's, respectively.

For $R_1 \mid R_2$ add production

$$< S > \rightarrow < S_1 > \mid < S_2 >$$

For $R_1 R_2$ add

$$< S > \rightarrow < S_1 > < S_2 >$$

For ${R_1}^*$ add

$$< S > \rightarrow < S_1 > < S > \mid \epsilon$$

Class Problem

For the extended operators $R_1?$ and ${R_1}^+$ what productions would you use?

Fooling Arguments: Showing a Language has no RE

If a language has an RE it has a DFA.

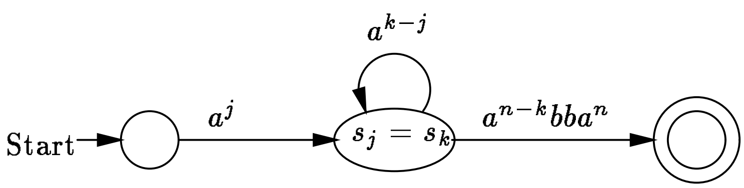
- The DFA has n states for some n. We don't know n, but we know it exists.
- Consider some string longer than n in the language and argue that at two times, the DFA must be in the same state.
- Use this observation to show the existence of a path leading to acceptance, with a label that is not in the language.

Example: Palindromes (even-length only) with symbols a and b. Grammar:

$$< pal > \rightarrow a < pal > a$$
$$< pal > \rightarrow b < pal > b$$
$$< pal > \rightarrow \epsilon$$

- Suppose $L(<pal>)$ had a DFA D. Let D have n states.
- Consider the behavior of D on input $a^n bba^n$.
  - ☐ Remember x^i is shorthand for the string of i x's.

- This string is a palindrome of even length, so D accepts.
- Let s_i be the state D is in after reading a^i.
  - Not all of $\{a_0, a_1, \ldots, a_n\}$ can be different (pidgeonhole principle!).
- Thus, there are integers j and k such that $0 \leq j < k \leq n$ for which $s_j = s_k$.
- As a result, $a^{n+k-j}bba^n$ also leads to acceptance.
  - Go around loop twice in diagram.



- But that string is not in the language. Thus, D does not accept $L($<*pal*>$)$ as claimed. Since we assumed nothing special about D, we have proved that no DFA accepts this language.

Class Problem

The language consisting of all strings of 0's whose length is a perfect square, i.e., $\{0, 0^4, 0^9, 0^{16}, \ldots\}$, is not a regular set.

- It isn't a context-free language either, but the proof is much harder.

Use a "fooling argument" to show that this language has no DFA.

- Important trick: the squares are very sparse. After n^2 the next square is $(n+1)^2$, which is $2n+1$ greater than n^2. Given a hypothetical DFA D, we can see what it does on some very large (compared with the number of states of D) square.