Chapter 18: Decidability

I. Theory of Automata

→ II. Theory of Formal Languages

III. Theory of Turing Machines …

Examples of undecidable problems:

- Are the languages generated by two context-free grammars the same language?
- Is a particular context-free grammar ambiguous?
- Given a context-free grammar that is ambiguous, is there another context-free grammar that generates the same language that is not ambiguous?
- Is the complement of a context-free language also context-free?
- Is the intersection of two context-free languages also context-free?
- Is the intersection of two context-free languages empty?
- Given a context-free grammar, are there any words that are not in the language generated by the grammar?

Some Decidable Problems

1. Given a context-free grammar, is the language that it generates empty? **Emptiness** problem.
2. Given a context-free grammar, is the language that it generates finite or infinite? **Finiteness** problem.
3. Given a context-free grammar and a word w, is w a word that can be generated by the grammar? **Membership** problem.

<u>Theorem:</u> Given a context-free grammar, there is an algorithm to determine if the language it generates is empty.

<u>Proof.</u>

- $S \Rightarrow^* \Lambda$? (by the constructive algorithm in Chapter 13 for deciding if a nonterminal is nullable)
- If not, convert the grammar to Chomsky normal form.
- Is there a production of the form: $S \rightarrow$ terminal?
- If not, repeat the steps:
  1. Choose a nonterminal N such that: $N \rightarrow t$ (sequence of terminals). Replace N on the right sides of productions with t. Remove all other productions for N.
  2. Stop if there is a production $S \rightarrow t$ or there are no more nonterminals to choose.

An algorithm that can determine if a nonterminal X can generate a sequence of terminals

- Reverse S and X and use the algorithm from the previous theorem.

If X cannot generate a sequence of terminals, X is unproductive.

<u>Theorem:</u> There is an algorithm to decide whether or not a given nonterminal X in a context-free grammar can ever be used in generating words.

Proof.

- Determine if X is unproductive.
- If not,
  1. Find all unproductive nonterminals.
  2. Remove all productions containing unproductive nonterminals.
  3. Paint all X's blue (on the right and left in the grammar).
  4. If any nonterminal on the right of a production is blue, paint the nonterminal on the left blue. Then paint blue all occurrences of this nonterminal in the grammar.
  5. Repeat step 4 until no new nonterminals are painted blue.
  6. If S is blue, there is a derivation that uses X.

If X cannot be used in a derivation, X is useless.

An algorithm that can decide if a nonterminal X is self-embedded.

- Replace X's on the left in productions with *X*.
- Paint all X's blue, and use the blue paint algorithm (steps 3 to 5 of the previous algorithm).
- If *X* is blue, then X is self-embedded.

Theorem: There is an algorithm to decide if a context-free grammar generates a language that is finite or infinite.

1. Find all useless nonterminals.
2. Test all nonterminals to see if any is self-embedded using the above algorithm.
3. If step 2 found a self-embedded nonterminal, then the language is infinite. Otherwise it is finite.

Membership problem

Given a context-free grammar G and a word $w = w_1w_2\ldots w_n$, is w a word that can be generated by the grammar?

CYK Algorithm (Cocke, Kasami(1965),Younger(1967)

- Transform G into a Chomski Normal Form.
- At each step i of the algorithm, $1 \leq i \leq n$, find all sub words of w with length i that can be generated stating from any of the non terminals.