CISC 4090
Theory of Computation

Decidability

Professor Daniel Leeds
dleeds@fordham.edu
JMH 332

“Turing recognizable” vs. “Decidable”

Language is Turing recognizable if some Turing machine recognizes it
• Also called “recursively enumerable”

Machine that halts on all inputs is a decider. A decider that recognizes language L is said to decide language L

Language is Turing decidable, or just decidable, if some Turing machine decides it

Not all problems can be solved
• Good to know when you might not find an answer
• Get perspective on limits of computation

Decidable problems for regular languages
• Does DFA D accept string s?
• Is L(D) of DFA empty?
• Are two DFAs D1 and D2 equivalent?

Specify DFA on input TM, determine control algorithm to run DFA specified on tape
Arbitrary DFA $D$ accepts string $w$

Language: $A_{DFA} = \{(D, w) \mid D \text{ is DFA that accepts } w\}$

Theorem: $A_{DFA}$ is decidable

Proof idea:
- Define machine $M$ that simulates $D$ on $w$
- If simulation ends in an accept, accept; else, reject

Note: control states in $M$ cannot be states in $D$

$M$ needs to run arbitrary $D$

---

$A_{DFA}$ decider Proof Outline

DFA $D$ described as string: 5-tuple

Use marks on tape to track
- current state in simulated $D$
- current symbol read from $w$

Implement transition function of $D$ for current $D$ state and input $w$
- $D$'s transition $\delta$ is different from TM $M$'s transition $\delta$

---

Arbitrary DFA $D$ accepts no strings

$E_{DFA} = \{ D \mid D \text{ is DFA with } L(D) = \{\} \}$ is decidable language

Proof idea:
- Is there any way to reach accept from start?
- Think of graph-marking

Proof
- Mark start state of DFA $D$
- Repeat until no new states
  - Mark any state that past-marked states transition to
- If an accept state is marked, REJECT; else, accept

---

Two DFAs are equivalent

$E_{DFA} = \{ (A, B) \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B) \}$ is decidable language

Proof idea:
- Construct new DFA $C$ from $A$ and $B$; $C$ accepts only strings accepted by either $A$ or $B$, but not both
- Check if $C$'s language is empty (last slide)
A_{\text{CFG}} \text{ is decidable – Proof}

For CFG G and string w, determine if G generates w

Idea 1: Simulate G to go through all derivations
• May never terminate

Idea 2: Note |w|=n; 2n-1 steps from CNF rules to each string
  Produce all words of lengths n
• Breadth-first search of finite depth is fixed

B_{\text{CFG}} \text{ is a decidable language}

• For CFG G, determine if there is any terminal string generated by G

EQ_{\text{CFG}} \text{ is not a decidable language}

• Regular expressions closed under complement and intersection
• CFLs not closed under complement and intersection
• We will prove non-decidable languages later

The Halting Problem

Key theorem to theory of computation
Addressing unsolvable problems

Unsolvable: Software verification
• For arbitrary computer program P and precise specification of program's behavior S, determine if P fulfills S
Halting Problem specified

\[ A_{TM} = \{ (M,w) \mid M \text{ is a TM and } M \text{ accepts } w \} \]

- If \( M \) loops forever on \( w \), our TM for \( A_{TM} \) must reject \( w \)
- This problem is Turing recognizable, but not decidable!

Detour: Cantor diagonalization

Comparing sizes of two infinite sets

- What is larger: set of even positive integers or set of all strings in \((0\cup 1)^*\)?

Diagonalization: two sets have same size if each element of set A can be compared with one element of set B
From CISC 1400: Can you define bijection from set A to set B?