CS21
Decidability and Tractability
Lecture 14
February 5, 2021

Outline
• undecidable problems
  – computation histories
  – surprising contrasts between decidable/undecidable
• Rice’s Theorem
• Post Correspondence Problem

Dec. and undec. problems
• two problems regarding Context-Free Grammars:
  – does a CFG generate all strings:
    \( \text{ALL}_{\text{CFG}} = \{ <G> : G \text{ is a CFG and } L(G) = \Sigma^* \} \)
  – CFG emptiness:
    \( \text{E}_{\text{CFG}} = \{ <G> : G \text{ is a CFG and } L(G) = \emptyset \} \)

Dec. and undec. problems

**Theorem:** \( \text{E}_{\text{CFG}} \) is decidable.

**Proof:**
– observation: for each nonterminal \( A \), the set 
  \( S_A = \{ w : A \Rightarrow^* w \} \)
  is non-empty iff there is some rule:
  \( A \rightarrow x \)
  and for all non-terminals \( B \) in string \( x \), \( S_B \neq \emptyset \)

Dec. and undec. problems

**Theorem:** \( \text{ALL}_{\text{CFG}} \) is undecidable.

**Proof:**
– reduce from co-\( A_{\text{TM}} \) (i.e. show co-\( A_{\text{TM}} \leq_m \text{ALL}_{\text{CFG}} \))
– what should \( f(<M, w>) \) produce?
– Idea:
  • produce CFG \( G \) that generates all strings that are not accepting computation histories of \( M \) on \( w \)

Dec. and undec. problems

**Proof:**
– on input <\( G \)>
– mark all terminals in \( G \)
– repeat until no new non-terminals get marked:
  • if there is a production \( A \rightarrow x_1x_2...x_n \)
  • and each symbol \( x_1, x_2, ..., x_n \) has been marked
  • then mark \( A \)
  – if \( S \) marked, reject (\( G \notin \text{E}_{\text{CFG}} \)), else accept (\( G \in \text{E}_{\text{CFG}} \)).
  – terminates? correct?
Dec. and undec. problems

Proof:
– build a NPDA, then convert to CFG
– want to accept strings not of this form,
  \#C₁\#C₂\#C₃\#...\#Cₖ\#
  plus strings of this form but where
  • C₁ is not the start config. of M on input w, or
  • Cₖ is not an accept. config. of M on input w, or
  • Cᵢ does not yield in one step Cᵢ₊₁ for some i

Dec. and undec. problems

Proof:
– our NPDA nondeterministically checks one of:
  • C₁ is not the start config. of M on input w, or
  • Cₖ is not an accept. config. of M on input w, or
  • Cᵢ does not yield in one step Cᵢ₊₁ for some i
  • input has fewer than two #'s
  – details of first two?
  – to check third condition:
    • nondeterministically guess Cᵢ starting position
    • how to check that Cᵢ doesn’t yield in 1 step Cᵢ₊₁?

Dec. and undec. problems

Proof:
– checking:
  • Cᵢ does not yield in one step Cᵢ₊₁, for some i
  – push Cᵢ onto stack
  – at #, start popping Cᵢ and compare to Cᵢ₊₁
    • accept if mismatch away from head location, or
    • symbols around head changed in a way inconsistent with M’s transition function.
  – is everything described possible with NPDA?

Dec. and undec. problems

Proof:
– f(<M, w>) = <G> equiv. to NPDA below:
  on input x, accept if not of form:
  \#C₁\#C₂\#C₃\#...\#Cₖ\#
  • is f computable?
  • YES maps to YES?
  • <M, w> ∈ co-\text{A_{TM}} ⇒ f(M, w) ∈ \text{ALL}_{CFG}
  • NO maps to NO?
  • <M, w> ∈ co-\text{A_{TM}} ⇒ f(M, w) ∈ \text{ALL}_{CFG}

Dec. and undec. problems

Rice’s Theorem

• We have seen that the following properties of TM’s are undecidable:
  – TM accepts string w
  – TM halts on input w
  – TM accepts the empty language
  – TM accepts a regular language
• Can we describe a single generic reduction for all these proofs?
  • Yes. Every property of TMs undecidable!
Rice’s Theorem

- A **TM property** is a language $P$ for which
  - if $L(M_1) = L(M_2)$ then $<M_1> \in P$ iff $<M_2> \in P$
- TM property $P$ is **nontrivial** if
  - there exists a TM $M_1$ for which $<M_1> \in P$, and
  - there exists a TM $M_2$ for which $<M_2> \notin P$.

**Rice’s Theorem**: Every nontrivial TM property is undecidable.

---

Rice’s Theorem

Proof:

- reduce from $A_{TM}$ (i.e. show $A_{TM} \leq_m P$)
- what should $f(<M, w>)$ produce?
- $f(<M, w>) = <M’>$ described below:
  - on input $x$,
    - accept iff $M$ accepts $w$ and $M_1$ accepts $x$
    - (intersection of two RE languages)

  - $f$ computable?
  - YES maps to YES?
  - $<M, w> \in A_{TM}$ ⇒ $L(f(M, w)) = L(M_1)$ ⇒ $f(M, w) \in P$

---

Post Correspondence Problem

- **Post Correspondence Problem**
  - many undecidable problems unrelated to TMs and automata
  - classic example: Post Correspondence Problem
    $$PCP = \{<x_1, y_1>, (x_2, y_2), \ldots, (x_k, y_k)> : x_i, y_i \in \Sigma^* \text{ and there exists } (a_1, a_2, \ldots, a_n) \text{ for which } x_1x_2\cdots x_n = y_1y_2\cdots y_n\}$$

---

Post Correspondence Problem

$$PCP = \{<x_1, y_1>, (x_2, y_2), \ldots, (x_k, y_k)> : x_i, y_i \in \Sigma^* \text{ and there exists } (a_1, a_2, \ldots, a_n) \text{ for which } x_1x_2\cdots x_n = y_1y_2\cdots y_n\}$$

- $x_1, x_2, x_3, y_1, y_2, y_3$ are the "tiles"
- $y_1y_2y_3y_4y_5$ is the "match"