Undecidable Problems About Languages

Sipser: Section 5.1 pages 215 - 226

Clique and Independent Set

CLIQUE = \{<G,k> | G is a graph with a k-clique\}

INDEPENDENT = \{<G,k> | G is a graph containing an independent set of size k\}

CLIQUE reduces to INDEPENDENT
Certified Impossible

\[ \text{Theorem.} \quad A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ accepts } w \} \text{ is undecidable.} \]

\[ \text{Definition.} \quad HALT_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ halts on input } w \} \]

The Halting Problem (Again!)

\[ \text{Theorem.} \quad HALT_{TM} \text{ is undecidable.} \]

\[ \text{Proof Idea.} \quad \text{We know } A_{TM} \text{ is undecidable. We need to reduce one of } HALT_{TM} \text{ or } A_{TM} \text{ to the other.} \]

Which way to go?

HALT_{TM} is undecidable

\[ \text{Proof.} \quad \text{Suppose } R \text{ decides } HALT_{TM}. \text{ Define} \]

\[ S = \text{"On input } \langle M, w \rangle, \text{ where } M \text{ is a TM and } w \text{ a string:} \]

1. Run TM \( R \) on input \( \langle M, w \rangle \).
2. If \( R \) rejects, then reject.
3. If \( R \) accepts, simulate \( M \) on input \( w \) until it halts.
4. If \( M \) enters its accept state, accept. If \( M \) enters its reject state, reject."
Proof. Given an input $<M, w>$ we construct a machine $M_w$ as follows:

$M_w = "\text{On input } x:\n1. \text{ If } x \neq w, \text{ reject.}\n2. \text{ If } x = w, \text{ run } M \text{ on input } w \text{ and accept if } M \text{ does.}"$

to be continued...

---

The Proof Continues

Proof continued. Suppose TM $R$ decides $E_{TM}$. Define

$S = "\text{On input } <M, w>:\n1. \text{ Use the description of } M \text{ and } w \text{ to construct } M_w.\n2. \text{ Run } R \text{ on input } <M_w>.\n3. \text{ If } R \text{ accepts, reject. If } R \text{ rejects, accept.}"

---

With Power Comes Uncertainty

$M$ accepts $w$ $\quad L(M) = \varnothing$ $\quad L(M_1) = L(M_2)$

Rice's Theorem. Any nontrivial property of the languages recognized by Turing machines is undecidable.
For Example

**Definition.** \( \text{REGULAR}_{TM} = \{ <M> | M \text{ is a TM and } L(M) \text{ is regular} \} \).

**Theorem.** \( \text{REGULAR}_{TM} \) is undecidable.

---

**REGULAR\(_{TM}\) is undecidable**

**Proof.** Let \( R \) be a TM that decides \( \text{REGULAR}_{TM} \). Define \( S = \text{"On input } <M, w>\text{:} \)

1. Construct TM \( M_2 = \text{"On input } x\text{:} \)
   1. If \( x \) has the form \( 0^n1^n \), accept.
   2. Otherwise, run \( M \) on input \( w \) and accept if \( M \) accepts \( w \).
2. Run \( R \) on input \( <M_2> \).
3. If \( R \) accepts, accept. Otherwise, if \( R \) rejects, reject."