Proof of Rice's theorem

Theorem

Let P be a non-trivial property of Turing recognizable languages. Let $\mathcal{L}_P = \{M \mid L(M) \in P\}$. Then $\mathcal{L}_P$ is undecidable.

Design of N

Let M_1 be the TM s.t. $L(M_1)$ has Property P.

Let $L(M_2)$ be the TM s.t. $L(M_2) = \emptyset$.

we assume that $\emptyset$ does not have property P

```
on input x
{
    if M accepts w
    then if M1 accepts x
        then accept
}
```

Claim: $w \in L(M)$ if and only if $\langle N \rangle \in \mathcal{L}_P$

if $w \in L(M)$ then $L(N) = L(M_1)$.

if $w \notin L(M)$ then $L(N) = \emptyset$.

Getting rid of the assumption on P

We now show how to get around the assumption.

Suppose $\varnothing$ has property P.

Consider $\overline{P}$.

Now $\varnothing$ does not have property $\overline{P}$.

Use Rice's theorem on $\mathcal{L}_{\overline{P}}$ to prove undecidibility.

Conclude undecidibility of $\mathcal{L}_P$.

Universality of CFLs

Lemma

$ALL_{CFL} = \{\langle M\rangle \mid M$ is a PDA and $L(M) = \Sigma^*\}$ is undecidable.

Proof Strategy

For a TM M and input w we create a PDA $N_{M,w}$ such that

$N_{M,w}$ accepts all string (i.e. accepts Σ^*) if M accepts w, and

$N_{M,w}$ rejects at least one string if M does not accept w.

Formally,

$$\text{Input } (M, w) \longrightarrow N_{M,w}$$

$$\text{if } w \in L(M) \longrightarrow \exists x \in \Sigma^*, \text{ s.t. } x \notin L(N_{M,w})$$

$$\text{if } w \notin L(M) \longrightarrow L(N_{M,w}) = \Sigma^*$$

Filling in the details

The following two details need to be addressed.

Q_1 How should we design $N_{M,w}$?

Q_2 If such an $N_{M,w}$ is designed then why have we proved that ALL_{CFL} is undecidable?

Details for Q_2

Q_2 If such an $N_{M,w}$ is designed then why have we proved that ALL_{CFL} is undecidable?

Input (M, w)	$\longrightarrow$	$N_{M,w}$
if $w \in L(M)$	$\longrightarrow$	$\exists x \in \Sigma^*$, s.t. $x \notin L(N_{M,w})$
if $w \notin L(M)$	$\longrightarrow$	$L(N_{M,w}) = \Sigma^*$

Assume that ALL_{CFL} is decidable.
C be the TM deciding it.

Design A as follows:

For an M, w pair, create $N_{M,w}$.

Feed $\langle N_{M,w} \rangle$ to C.

If C accepts then reject;

else accept.

Details for Q_1: reduction via computation history

Q_1 How should we design $N_{M,w}$?

Main idea

Use computational history of M on w.

Accepting computation history is a sequece of configutations: $C_1, C_2, \ldots, C_\ell$ such that

- C_1 is a start configutation.
- C_ℓ is an accepting configuration.
- for each $1 \leq i \leq \ell$, C_i yields C_{i+1}.

Rejecting computation history is a sequece of configutations: $C_1, C_2, \ldots, C_\ell$ such that

- C_1 is a start configutation.
- C_ℓ is a rejecting configuration.
- for each $1 \leq i \leq \ell$, C_i yields C_{i+1}.

Details for Q_1: reduction via computation history

Interprete input x to $N_{M,w}$ as a computational history of M on w.

Design $N_{M,w}$ s.t. it accepts x if any of the following conditions holds:

x does not have the pattern of a computational history of x OR

x is a computational history, but C_1 is not a start configuration OR

x is a computational history, C_1 is a start configuration, but C_ℓ is not an accepting configutation OR

x is a computational history, C_1 is a start configuration, C_ℓ is an accepting configutation, but there exists an i s.t. $1 \le i \le \ell - 1$ and C_i does not yield C_{i+1}.

If M accepts w, let $\tilde{x}$ be a accepting computation history of M on w.

$N_{M,w}$ will reject $\tilde{x}$, i.e. $\tilde{x} \notin L(N_{M,w})$.

If M does not accept w, then

no matter what x is, $N_{M,w}$ will accept x, i.e. $L(N_{M,w}) = \Sigma^*$.