### Decidable Problems

#### Represent problem using language

**Problem:** Is \( w \) accepted by DFA \( B \)?

\[
A_{DFA} = \{ <B, w> \mid B \text{ is a DFA that accepts } w \}
\]

\(<>\) denotes encoding as string

**Th.** \( A_{DFA} \) is a decidable language.

**Proof idea:** Decider \( M \) will simulate \( D \) on input \( w \).
If \( D \) would accept \( w \), then \( M \) accepts; \( D \) rejects, \( M \) rejects.

<table>
<thead>
<tr>
<th>q0</th>
<th>q1</th>
<th>q2</th>
<th>q3</th>
</tr>
</thead>
</table>
| 011 | 1# | \( q_0 \) | | 0111 | \( q_0 \) | | | 1

### Decidable Problems

\[
A_{NFA} = \{ <B, w> \mid B \text{ is an NFA, } B \text{ accepts } w \}
\]

**Th.** \( A_{NFA} \) is decidable.

**Proof idea:** Define \( N \) to use TM \( M \) (last th) as subroutine.
1. Convert NFA \( B \) to DFA \( D \).
2. Run TM \( M \) on input \( <D, w> \)
3. If \( M \) accepts, then \( N \) accepts; \( M \) rejects, \( N \) rejects.

<table>
<thead>
<tr>
<th>q0</th>
<th>q1</th>
<th>q2</th>
<th>q3</th>
</tr>
</thead>
</table>
| 011 | 1# | \( q_0 \) | | 0111 | \( q_0 \) | | | 1

| \{\( q_0 \) \} \{\( q_0 \) \( q_1 \) \} \text{......} | <\text{det. table}> |
Decidable Problems

E_DFA = \{<B> | B is a DFA, L(B) is nonempty\}

Th. E_DFA is decidable.

Lemma: For DFA B, L(B) is nonempty if and only if B accepts a string of length at most |Q|.

Trivial.

Suppose B accepts some word. Let w be a shortest word accepted by B.

If |w| is < |Q|, then we are done. So suppose |w| > |Q|. Consider the sequence of states of B on w.

Decidable Problems

w = w_1 w_2 w_3 \ldots w_m, \quad m > |Q|\]

\begin{align*}
  &w_1 \rightarrow q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow \ldots \rightarrow q_m
\end{align*}

Since there are more than |Q| states in this sequence, there must be a repetition of states, by the pigeonhole principle. But then we can take a snippet z out of w, and get a shorter string that is accepted by B.

Contradiction. Therefore w is of length |Q|.

Back to Theorem:

Decider D will, on input <B>, simulate B on inputs of length 0 to |Q|. If B accepts any string of that length, D accepts. If no such string is found, D rejects. By the Lemma, D accepts E_DFA.
Decidable Problems

\[ \text{EQ}_{\text{DFA}} = \{ \langle A, B \rangle \mid A, B \text{ are DFA, and } L(A) = L(B) \} \]

Th: \( \text{EQ}_{\text{DFA}} \) is decidable.

Proof: We construct a DFA \( C \) with

\[
L(C) = (L(A) \mathbin{\mathbf{\setminus}} L(B)) \mathbin{\mathbf{\cup}} (L(A) \mathbin{\mathbf{\setminus}} L(B))
\]

(symmetric difference)

\[
\begin{array}{c}
L(A) \\
\cap \\
\downarrow \\
L(B)
\end{array}
\]

\( C \) can be constructed by TM \( M \) using algorithms we developed for closure under \( \mathbin{\mathbf{\cup}}, \mathbin{\mathbf{\setminus}}, \mathbin{\mathbf{\cap}} \).

Since \( L(C) \) is empty iff. \( L(A) = L(B) \), we can use previous theorem for deciding \( L(C) \) nonempty.

---

Decidable Problems for CFG’s

\[ A_{\text{CFG}} = \{ \langle G, w \rangle \mid G \text{ is a CFG, } G \text{ generates } w \} \]

Th. \( A_{\text{CFG}} \) is decidable.

Proof idea: TM could start with start var. \( S \) and try all derivations. If \( w \) is not in \( L(G) \), there will be none!

So trying all derivations to see if \( w \) would not yield decider.

To define a decider, we first put \( G \) in Chomsky normal form.

Then we can bound the number of steps in a derivation TM needs to try to \( 2 |w| - 1 \).

QU: WHY?

The decider \( D \) will then list all derivations of length 0 to \( 2 |w| - 1 \). If any derivation generates \( w \), \( D \) accepts; \( \omega w \), reject.
### Decidable Problems on CFG's

\[ E_{\text{CFG}} = \{<G> \mid G \text{ is a CFG, } L(G) \text{ is nonempty}\} \]

**Th.** \( E_{\text{CFG}} \) is decidable.

Proof idea: Can’t try all strings (won’t be decidable).
Consider any parse tree for a string \( w \).

\[
\begin{array}{c}
S \\
\Downarrow \\
\text{If parse tree has path of length } |V|,
\end{array}
\]

then there is a repetition of variables along that path, and we can produce a shorter parse tree. We can repeat this process so that there is a parse tree with no path of length \( |V| \). Therefore, if \( G \) generates any string, it generates a string whose parse tree has no path \( |V| \).

---

### Decidable Problems on CFG's

**Th.** \( E_{\text{CFG}} \) is decidable. **(continued)**

A decidер \( D \) for this problem will generate a collection \( C \) of parse trees on its tape. \( C \) will initially contain \( S \).

\( D \) repeatedly adds to \( C \) any tree that can be obtained from one already in \( C \) by applying a single rule, such that

1. the new tree is not in \( C \)
2. the new tree does not have any path of length \( |V| \)

Since there are only a finite number of trees of fixed length, the TM \( D \) will eventually complete \( C \). \( L(G) \) is nonempty iff at least one tree in \( C \) has only terminals as leaves.
Decidable Problems for CFG’s

\[ \text{EQ}_{\text{CFG}} = \{ \langle G, H \rangle \mid G, H \text{ are CFG, and } L(G) = L(H) \} \]

\[ \text{EQ}_{\text{CFG}} \text{ decidable??} \]

Can’t use same idea we used for DFA’s (symmetric diff.) because CFL’s are NOT closed under complement.

Turns out \( \text{EQ}_{\text{CFG}} \) is NOT decidable. We can’t prove it yet.

Th. Every CFL is decidable. (HW 3)

Undecidable Problems

Showed these problems for FA and CFG’s decidable: Acceptance, Emptiness, Equivalence

Qu: What about for TM?

\[ A_{\text{TM}} = \{ \langle M, w \rangle \mid M \text{ is a TM that accepts } w \} \]

Will show that \( A_{\text{TM}} \) is not decidable (ie, undecidable)

What’s wrong with:

Define TM \( U \) (Universal TM) on input \( \langle M, w \rangle \)

1. Simulate \( M \) on input \( w \)
2. If \( M \) accepts, \( U \) accepts; if \( M \) rejects, \( U \) rejects.

\( U \) can act like any TM on any input

Is \( U \) a decider????
**Universal TM U is not a decider**

*U has no way to determine if M halts on input w.*

**Lemma:** $A_{TM}$ is Turing-recognizable.

**Proof:** $U$ is the recognizer.

**Halting Problem:** $A_{HALT} = \{ <B,w> \mid B \text{ halts on input } w \}$

$A_{HALT}$ is decidable $\iff$ $A_{TM}$ is decidable

We show $A_{TM}$ is undecidable (then $A_{HALT}$)

Doesn’t mean that we CAN’T determine for a particular $<B,w>$ whether $B$ halts (accepts) on input $w$.

Does mean that there is no algorithm that works for all input $<B,w>$. Gives fundamental limitation of Algorithms/TM.

---

**Diagonalization for TM**

*Suppose there was a super-diagonalizer TM D on input $<M>$:*

1. Simulate $M$ on input $<M>$
2. If $M$ accepts, $D$ rejects; if $M$ rejects, $D$ accepts.
   
   If $M$ never halts, $D$ accepts.

*Note $D$ has power to decide if $M$ halts or not on $<M>$.*

*QU: Can $D$ be in the list (ie = $M_i$ for some $i$)?*

$$ M_1 \quad M_2 \quad M_3 \quad \ldots $$

$<M_1>$

$<M_2>$

$<M_3>$

$\vdots$