Decidable problems?

- **Acceptance** problem:
  - decide whether an automaton accepts a string
- **Equivalence** problem:
  - Decide whether two automata are equivalent, i.e. accept the same language
- **Emptiness testing** problem:
  - Decide whether the language of an automaton is empty
- Can be applied to
  - DFA, NFA, REX, PDA, CFG, TM, ...

The acceptance problem for DFAs

\[ A_{DFA} = \{ \langle B, w \rangle \mid B \text{ is a DFA that accepts input string } w \} \]

Theorem

\( A_{DFA} \) is a decidable language

Proof

\[ M = \text{"On input } \langle B, w \rangle, \text{ where } B \text{ is a DFA and } w \text{ is a string:} \]
\[ 1. \text{ Simulate } B \text{ on input } w. \]
\[ 2. \text{ If the simulation ends in an accept state, accept. If it ends in a nonaccepting state, reject."} \]
**The acceptance problem for NFAs**

\[ A_{\text{NFA}} = \{ \langle B, w \rangle \mid B \text{ is an NFA that accepts input string } w \} \]

**Theorem**

\[ A_{\text{NFA}} \text{ is a decidable language} \]

**Proof**

---

**The emptiness testing problem for DFAs**

\[ E_{\text{DFA}} = \{ \langle A \rangle \mid A \text{ is DFA for which } L(A) = \emptyset \} \]

**Theorem**

\[ E_{\text{DFA}} \text{ is a decidable language} \]

**Proof**

---

**The acceptance problem for Regular Expressions**

\[ A_{\text{REX}} = \{ \langle R, w \rangle \mid R \text{ is a regular expression that generates input string } w \} \]

**Theorem**

\[ A_{\text{REX}} \text{ is a decidable language} \]

**Proof**

---

**The equivalence problem for DFAs**

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

**Theorem**

\[ EQ_{\text{DFA}} \text{ is a decidable language} \]

**Proof**

---
Decidable problems for CF languages

$A_{\text{CFG}} = \{(G, w) | G \text{ is a CFG that generates input string } w\}$

**Theorem**

$A_{\text{CFG}}$ is a decidable language

**Proof**

Relies on the following property:

If $G$ is in Chomsky Normal Form, then any derivation of $w$ has length at most $2|w|-1$

There are only finitely many derivations of length less than $n$

**Theorem**

$E_{\text{CFG}} = \{(G) | G \text{ is CFG for which } L(G) = \emptyset\}$

**Proof**

Determine for each variable whether that variable is capable of generating a string of terminals

---

$E_{\text{QCFG}} = \{(G, H) | G \text{ and } H \text{ are CFLs and } L(G) = L(H)\}$

**Theorem**

$E_{\text{QCFG}}$ is not decidable

**Proof**

Follows later

The problem with adapting the proof for DFAs is that the class of context free languages is not closed under complementation or intersection!

---

**Theorem**

Every context free language is decidable

**Proof**

Let $G$ be a CFG for $A$ and design a TM $M_G$ that decides $A$. We build a copy of $G$ into $M_G$. It works as follows.

1. Run TM $S$ on input $(G, w)$
2. If this machine accepts, accept; if it rejects, reject.
The halting problem

• There is a specific problem that is algorithmically unsolvable (undecidable), e.g. the halting problem

• Philosophical implications: computers are fundamentally limited

Diagonalization

• Georg Cantor 1873

• Measure the size of (infinite) sets
  
  Consider function \( f : A \rightarrow B \)
  
  - \( f \) is one-to-one if \( f(a) \neq f(b) \) whenever \( a \neq b \)
  
  - \( f \) is onto if for every \( b \in B \) there is an \( a \in A : f(a) = b \)
  
  - \( f \) is a correspondence if it is onto and one-to-one

Example: \( f : \mathbb{N} \) (natural numbers) \( \rightarrow \) \( E \) (even nat.)

\( f(n) = 2n \) is a correspondence

Both sets have the same size! A set is countable if it has the same size as \( \mathbb{Q} \)

\[ \mathbb{Q} = \left\{ \frac{m}{n} \mid m, n \in \mathbb{N} \right\} \]

Theorem

\( \mathbb{Q} \) is countable

Proof idea

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

Theorem

\( A_{TM} \) is Turing recognizable

Proof

Consider \( U: (\text{Universal Turing Machine}) \)

On input \( \langle M, w \rangle \), where \( M \) is a TM and \( w \) a string

1. Simulate \( M \) on \( w \)

2. If \( M \) ever enters its accept state, \textit{accept,}
   if \( M \) ever enters its reject state, \textit{reject}

\( U \) loops when \( M \) does the halting problem:

Theorem \( A_{TM} \) is undecidable

shows that recognizers are more powerful than deciders

requires quite involved proof
The set of real numbers (have a decimal representation)

**Theorem**
The set is uncountable

**Proof** idea
We prove (by contradiction) that there is no correspondence between $\mathbb{R}$ and $\mathbb{N}$
Assume that there were a correspondence $f$
We now construct an $x \in \mathbb{R}$ that is not paired with any element of $\mathbb{N}$
Choose the $i$-th fractional digit of $x$ different from the $i$-th frac. digit of $f(i)$

*Example:

<table>
<thead>
<tr>
<th>$n$</th>
<th>$f(n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.1414..</td>
</tr>
<tr>
<td>2</td>
<td>5.857..</td>
</tr>
<tr>
<td>3</td>
<td>0.888888..</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

So, $x \neq f(n)$ for all $n$

---

**Theorem**
Some languages are not Turing recognizable

**Proof**
There is a countable number of Turing Machines
(Each Turing Machine can be encoded in a string; the set of all strings over a finite alphabet is countable; not all strings need to encode legal TMs)

The set of all languages is uncountable

Therefore there is no correspondence between the set of all TMs and the set of all languages.
A language is co-Turing recognizable if it is the complement of a language that is Turing recognizable.

**Theorem**
A language is decidable if and only if it is both Turing-recognizable and co-Turing recognizable.

**Proof**
1. If $A$ is decidable then $A$ and $\overline{A}$ Turing recognizable
   Trivial
2. If $A$ and $\overline{A}$ Turing recognizable then $A$ is decidable
   Let $M_1$ and $M_2$ be TMs for $A$ and $\overline{A}$
   Define $M$:
   On input $w$
   1. Run both $M_1$ and $M_2$ on $w$ in parallel
   2. If $M_1$ accepts, then accept;
      If $M_2$ accepts, then reject;
   $M$ decides $A$
   all strings are either in $A$ or $\overline{A}$
   either $M_1$ or $M_2$ must accept any given string
   $M$ always terminates with correct answer

**Theorem**
$A_{TM}$ is not Turing-recognizable

**Proof**
$A_{TM}$ is Turing-recognizable
If $A_{TM}$ were also Turing-recognizable
Then $A_{TM}$ would be decidable.