
Section 4.2 – Undecidability

CSC 320

Countable Sets

A function from set A to B is **1-1** if it never maps two elements of A to the same element of B . It is **onto** if for every $b \in B$ there is an $a \in A$ such that $f(a) = b$.

A set A is **countable** if it is finite or if there is a 1-1 and onto function $f : A \rightarrow \mathbb{N}$

Note: If B is countable, and there is a 1-1 and onto $g : A \rightarrow B$, then A is countable (why?)

Examples of Countable Sets

- Even numbers: define $f(m) = \frac{m}{2}$.
 - **1-1:** Suppose m, m' are even numbers, say $m = 2n$ and $m' = 2n'$ where $n, n' \in \mathbb{N}$. If $m \neq m'$ then clearly $n \neq n'$ so $f(m) \neq f(m')$.
 - **Onto:** For any $n \in \mathbb{N}$, $2n$ is even and $f(2n) = n$
- $\{0,1\}^*$ – define $f: \{0,1\}^* \rightarrow \mathbb{N}$ by $f(w) = n$ where $n \in \mathbb{N}$ has binary representation $1w$. This is a bijection since every natural number has a unique binary representation (why do we use $1w$ instead of w ?)

The Rational Numbers are Countable

Let's just do this for \mathbb{Q}^+ (positive rationals). Any $q \in \mathbb{Q}^+$ has a unique representation $\frac{m}{n}$ where $\gcd(m, n) = 1$. Define $f : \mathbb{Q}^+ \rightarrow \mathbb{N}$ by $f\left(\frac{m}{n}\right) = 2^n 3^m$. Then f is 1-1 by the Fundamental Theorem of Arithmetic.

We can use a similar argument to show that if Σ is finite, then Σ^* is countable, i.e., if $\Sigma = \{a_1, \dots, a_n\}$ map $w = a_{i_1} a_{i_2} \dots a_{i_k}$ to $2^{i_1} 3^{i_2} \dots p_k^{i_k}$ where p_i is the i th prime number.

Are there sets that are not countable?

Paradoxes

- **The Paradox of the Liar:** "This sentence is not true."
- **The Barber's Paradox:** The barber cuts the hair of everyone in the town who doesn't cut his or her own hair.

Cantor's Theorem

- The proof is an example of a **diagonalization argument**.
- Let S be a countably infinite set, say $S = \{x_1, x_2, \dots\}$. $\mathcal{P}(S)$ is the set of all subsets of S . This set is infinite but not countably infinite, i.e., it is *uncountably infinite*.
- **Proof:** Suppose to the contrary that $\mathcal{P}(S)$ is countable. Let f be a 1-1 and onto function from \mathbb{N} to $\mathcal{P}(S)$. Define the set $T = \{x_i \mid x_i \notin f(i)\}$.
- Since T is in $\mathcal{P}(S)$ there must be some j such that $f(j) = T$.
- Is x_j in T ?
- Neither Yes nor No. Hence our assumption, that $\mathcal{P}(S)$ was countable is wrong.

Application to TMs

- Theorem: There are languages which are not Turing- recognizable.
- Say $\Sigma = \{0,1\}$. The set of all possible languages over Σ is just $\mathcal{P}(\{0,1\}^*)$. By Cantor's Theorem, this set is uncountably infinite.
- The set of TM's is countable because a TM can be described by a finite string over a finite alphabet
- Since each Turing machine accepts one language, there are only countably infinite Turing-recognizable languages.
- Hence, since there are an uncountable number of languages, there are languages which are not recognized by any TM.
- Can we show an *explicit* language which is not Turing recognizable? We will start by showing a language that is not decidable.

The Acceptance Problem is Undecidable

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

Proof: Assume it's decidable and show a contradiction. Let H be a TM which decides A_{TM} , i.e., H is halting, and H accepts input $\langle M, w \rangle$ iff M accepts w .

We construct a new TM D which uses H as a subroutine. D takes as input any TM description $\langle M \rangle$ and simulates H on $\langle M, \langle M \rangle \rangle$. When H halts, D enters the opposite final state. So

- D accepts $\langle M \rangle$ if M rejects $\langle M \rangle$
- D rejects $\langle M \rangle$ if M accepts $\langle M \rangle$

Now, what happens when D is given $\langle D \rangle$ as input? D accepts $\langle D \rangle$ iff D rejects $\langle D \rangle$!

This is a contradiction. Therefore D and H can't exist.

Can view as a diagonalization argument in a table.

A_{TM} is Turing-Recognizable

Recall that we are assuming a standard way of encoding the pair $\langle M, w \rangle$ as a string

Given this input, we want to *simulate* the computation of M . Use a 4-tape TM, which we call U .

- Tape 1 stores the string encoding $\langle M, w \rangle$, (the input to U)
- Tape 2 stores the simulated tape of M .
- Tape 3 stores the state of M
- Tape 4 is scratch.

Steps of the Simulation

- Examine the code to make sure it's for a legitimate TM. If not, halt without accepting.
- Initialize the second tape by putting w on it
- Place the start state 1 on tape 3. Move the head of the second tape to the leftmost simulated cell.
- To simulate a move:
 - Based on the state on tape 3, and symbol scanned on tape 2, search through the description of M on tape 1 until we find the appropriate transition.
 - Update the contents of tape 2, and the state on tape 3, based on this transition
- If M has no transition that matches the symbol being read, U halts.
- If M enters an accepting state, U accepts.

A Turing Unrecognizable Language

A language is *co-Turing-recognizable* if its complement is Turing recognizable.

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

Proof Suppose M_1 recognizes L and M_2 recognizes \bar{L} . Define M as follows: on input w , run M_1 and M_2 on w , in parallel. Exactly one will accept w (why?). If M_1 accepts, then M accepts, if M_2 accepts, then M rejects. So M decides L .

Corollary The complement of A_{TM} is not Turing recognizable.

Decidable Languages and their Complements

From the preceding slides, we see that there are languages which are Turing-recognizable, but whose complements are not Turing recognizable. What about decidable languages?

Theorem: If a language L is decidable, so is its complement.

Proof: Let M be the halting TM which accepts L .

- Change accepting states to nonaccepting states.
- Make a new accepting state and add a transition to it from every (old) nonaccepting state labeled with every tape symbol such that there was no transition out of that state with that label (in the old machine).
- Make old nonaccepting states ordinary.
- Make new accepting state transition back to itself on every symbol.