CSE 431 - Theory of Computation
Lecture 5: April 15
Lecturer: James Lee
Scribed by: Jake Utley and Sam Courts

Recap

- A Turing machine is a simple robust model of computation. Robust meaning that reasonable modifications to the model do not increase the computational power.
- TMs express algorithms, so we don't need to focus on the underlying machine.
- Church-Turing thesis: Any reasonable model of computation is equivalent to the Turing machine model
  - Reasonable means a finite amount of work per step
- TMs can take the description of other TMs as input
  - <M> is the description of a TM M
  - Once we have the description of a TM as a string, we can pass that description as input to other TMs

Undecidable Languages

There are languages that are undecidable, meaning that there are languages that no Turing machine can decide. This can be seen by counting both the total number of languages and the total number of TMs.

Fix $\Sigma = \{0, 1\}$
Recall that a language L is a subset $L \subseteq \Sigma^*$

How many languages are there? 2^{Σ^*}
- Σ^* is countable
- 2^{Σ^*} is uncountable

How many Turing machines are there?
- Countably many. Any TM can be encoded into a string, so the set of all TMs is a subset of Σ^*. This means there are countably many TMs.

By the counting argument, we can see that the vast majority of languages are undecidable.

Example

$$A_{TM} = \{< M, w >:\ M\ is\ a\ valid\ TM, and\ w\ is\ an\ input, M\ accepts\ w\}$$

Fact: A_{TM} is Turing recognizable

$R(<M, w>)$ = " 1. Check that $<M>$ is a valid TM description
2. Simulate M on w
3. If M accepts w, ACCEPT
4. If M rejects w, REJECT"

R recognizes A_{TM}, but does not decide since the simulation may loop forever.

Theorem: A_{TM} is Turing-undecidable

Intuition:

The set of all TMs is countable, so we can list them M_1, M_2, M_3, ... We can run M_i on $<M_j>$. This seems strange, but is a valid thing to do and might sometimes do reasonable things. Running all TMs on the description of all TMs, we get the following table:

	$<M_1>$	$<M_2>$	$<M_3>$	$\cdots$
M_1	A	N	N	$\cdots$
M_2	N	N	N	$\cdots$
M_3	A	A	A	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

A: accept, N: doesn't accept

We create a TM S such that on input $<M_i>$, S does the opposite of M_i on $<M_i>$ (flips the diagonal). Because S is a TM, it must be in the table. So what would S show in the diagonal?

$S(<S>)$ results in A $\Rightarrow$ $S(<S>)$ results in N
$S(<S>)$ results in N $\Rightarrow$ $S(<S>)$ results in A

Clearly, this assumption results in a contradiction, so S cannot be a TM. We will prove that if A_{TM} is decidable, we could build an implementation of S.

Proof: Suppose (for the sake of contradiction) that A_{TM} is decidable.

$D(<M>)$ = " 1. Check that $<M>$ is a valid TM description, if not then REJECT
2. Use the decider for A_{TM} to decide whether M accepts $<M>$
3. If M accepts $<M>$, REJECT
4. If M rejects $<M>$, ACCEPT"

Run $D(<D>)$, this results in a contradiction:

$D(<D>)$ accepts $\Rightarrow$ $D(<D>)$ rejects
$D(<D>)$ rejects $\Rightarrow$ $D(<D>)$ accepts

This implies that D cannot exist, so our assumption was false, and A_{TM} must be undecidable.

Unrecognizable Languages

Definition: A language L is <u>co-Turing-recognizable</u> if $\overline{L} = \{x \in \Sigma^* : x \notin L\}$ is Turing-recognizable.

Lemma: A language L is decidable iff it's both recognizable and co-recognizable.

Proof:

1) L is decidable $\Rightarrow$ L is recognizable and co-recognizable
   Using the decider for L, we can recognize L.
   Negating the decider for L, we can recognize $\overline{L}$.
   Therefore, L is both recognizable and co-recognizable.

2) L is recognizable and co-recognizable $\Rightarrow$ L is decidable
   Let M recognize L and N recognize $\overline{L}$
   We can define our decider R as:
   R = " On input w, simulate:
         M on w and
         N on w in parallel
      If M accepts w, ACCEPT
      If N accepts w, REJECT"
   If $w \in L$, then M accepts $w \Rightarrow R$ accepts w.
   If $w \notin L$, then N accepts $w \Rightarrow R$ rejects w.

Theorem: $\overline{A_{TM}}$ is not Turing-recognizable.

Proof:

1) A_{TM} is not decidable.
2) A_{TM} is recognizable.
3) From 1 and 2, A_{TM} is not recognizable.
4) $\overline{A_{TM}}$ is not recognizable.