BU CS 332 – Theory of Computation

https://forms.gle/46bdU9t84D1FuEYx5

Lecture 15:

- Undecidability
- Reductions

Reading:

Sipser Ch 4.2, 5.1

Mark Bun

November 1, 2022

Where we are and where we're going

Church-Turing thesis: TMs capture all algorithms

Consequence: studying the limits of TMs reveals the limits of computation

Last time: Countability, uncountability, and diagonalization

Existential proof that there are undecidable and unrecognizable languages

All languages over $\{0,1\}$ is uncountable. All TMs is countable

Today: An explicit undecidable language

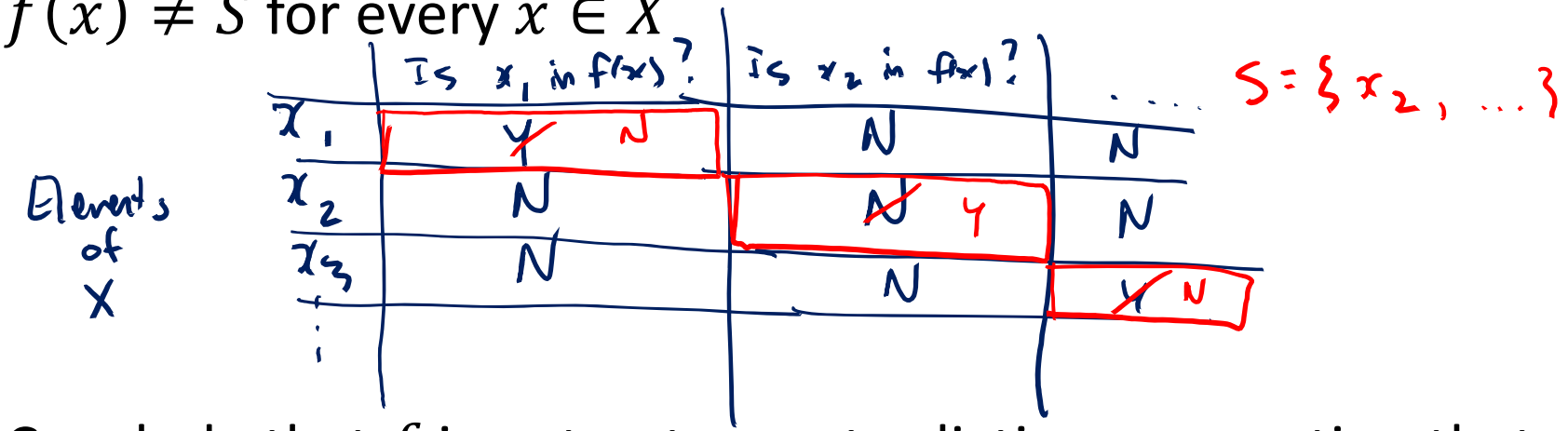
Reductions: Relate decidability / undecidability of different problems

An Explicit Undecidable Language

Last time:

Theorem: Let X be any set. Then the power set $P(X)$ does **not** have the same size as X.

1) Assume, for the sake of contradiction, that there is a bijection $f: X \to P(X)$
2) "Flip the diagonal" to construct a set $S \in P(X)$ such that $f(x) \neq S$ for every $x \in X$



3) Conclude that f is not onto, contradicting assumption that f is a bijection

Specializing the proof

Theorem: Let X be the set of all TM deciders. Then there exists an undecidable language in $P(\{0,1\}^*)$

1) Assume, for the sake of contradiction, that $L: X \rightarrow P(\{0,1\}^*)$ is onto $L(M)$ = language decided by TM M
2) "Flip the diagonal" to construct a language $UD \in P(\{0,1\}^*)$ such that $L(M) \neq UD$ for every $M \in X$
3) Conclude that L is not onto, a contradiction

An explicit undecidable language

TM M					
M_1					
M_2					
M_3					
M_4					
⋮					

Why is it possible to enumerate all TMs like this?

a) The set of all TMs is finite
b) The set of all TMs is countably infinite
c) The set of all TMs is uncountable

An explicit undecidable language

What happens when I run TM M_i on input $\langle M_i \rangle$
Run TM M_1 on input $\langle M_2 \rangle$
Record Y for accept N for reject or loop

TM M	$M(\langle M_1 \rangle)$?	$M(\langle M_2 \rangle)$?	$M(\langle M_3 \rangle)$?	$M(\langle M_4 \rangle)$?		$D(\langle D \rangle)$?
M_1	~~Y~~ N	N	Y	Y	...	
M_2	N	~~N~~ Y	Y	Y		
M_3	Y	Y	~~Y~~ N	N		
M_4	N	N	Y	N		
⋮					⋱	
D						~~Y~~ N N ~~Y~~

$UD = \{\langle M \rangle \mid M \text{ is a TM that does not accept on input } \langle M \rangle\}$

Claim: UD is undecidable

For this table $UD = \{\langle M_2 \rangle, \langle M_4 \rangle \ldots\}$

Assume for a contradiction that D decides UD

Case 1: $D(\langle D \rangle)$ accepts $\Rightarrow \langle D \rangle \notin UD$ by construction of UD
(contradicts assumption that D behaves correctly on $\langle D \rangle$)

Case 2: $D(\langle D \rangle)$ does not accept $\Rightarrow \langle D \rangle \in UD$ by construction
again contradicts correctness of D on $\langle D \rangle \in$

An explicit undecidable language

Theorem: $UD = \{\langle M \rangle \mid M \text{ is a TM that does not accept on input } \langle M \rangle\}$ is undecidable

Proof: Suppose for contradiction, that TM D decides UD

Examine what happens when we run $D(\langle D \rangle)$.

Case 1: $D(\langle D \rangle)$ accepts $\Rightarrow \langle D \rangle \notin UD$ by def. of UD

$\Rightarrow D$ produces wrong answer on $\langle D \rangle$ [accepts something it should reject]

Contradicts assumption that D decides UD $\lightning$.

Case 2: $D(\langle D \rangle)$ rejects $\Rightarrow \langle D \rangle \in UD$ by def. of UD

$\Rightarrow D$ rejects $\langle D \rangle$ when it's supposed to accept

Contradicts D being a decider for UD. $\lightning$.

A more useful undecidable language

$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input } w\}$

Theorem: A_{TM} is undecidable

Proof: Assume for the sake of contradiction that TM H decides A_{TM}:

$$H(\langle M, w\rangle) = \begin{cases} \text{accept} & \text{if } M \text{ accepts } w \\ \text{reject} & \text{if } M \text{ does not accept } w \end{cases}$$

Idea: Show that H can be used to construct a decider for the (undecidable) language UD -- a contradiction.

A more useful undecidable language

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

Proof (continued):

Suppose, for contradiction, that H decides A_{TM}

Consider the following TM D:

> "On input $\langle M\rangle$ where M is a TM:
>
> 1. Run H on input $\langle M, \langle M\rangle\rangle$
> 2. If H accepts, reject. If H rejects, accept."

Claim: D decides $UD = \{\langle M\rangle \mid \text{TM } M \text{ does not accept } \langle M\rangle\}$

1) $\langle M\rangle \in UD \Rightarrow M$ does not accept $\langle M\rangle \Rightarrow \langle M, \langle M\rangle\rangle \notin A_{TM}$
$\Rightarrow H(\langle M, \langle M\rangle\rangle)$ rejects $\Rightarrow D$ accepts ✓

2) $\langle M\rangle \notin UD \Rightarrow M$ accepts on input $\langle M\rangle \Rightarrow \langle M, \langle M\rangle\rangle \in A_{TM}$
$\Rightarrow H(\langle M, \langle M\rangle\rangle)$ accepts $\Rightarrow D$ rejects ✓

…but this language is undecidable $\Rightarrow D$ decides UD

Unrecognizable Languages

Theorem: A language L is decidable if and only if L and $\overline{L}$ are both Turing-recognizable.

Corollary: $\overline{A_{\text{TM}}}$ is unrecognizable

$\{\langle M, w\rangle \mid$ TM M does not accept input $w\}$

Proof: A_{TM} is undecidable $\Rightarrow$ either A_{TM} or $\overline{A_{TM}}$ is unrecognizable

We know A_{TM} is recognizable by UTM $\Rightarrow$ $\overline{A_{TM}}$ is unrecognizable.

Proof of Theorem: $\Rightarrow$

Let L be decidable. Then L is recognizable.

↳ $\overline{L}$ is decidable $\Rightarrow$ $\overline{L}$ is recognizable

(decidable langs are closed under complement)

Unrecognizable Languages

Theorem: A language L is decidable if and only if L and $\overline{L}$ are both Turing-recognizable.

Proof continued: $\Leftarrow$

Let L and $\overline{L}$ both be recognizable.
Let M recognize L and M' recognize $\overline{L}$.
Goal: construct N deciding L.

Attempt 1:

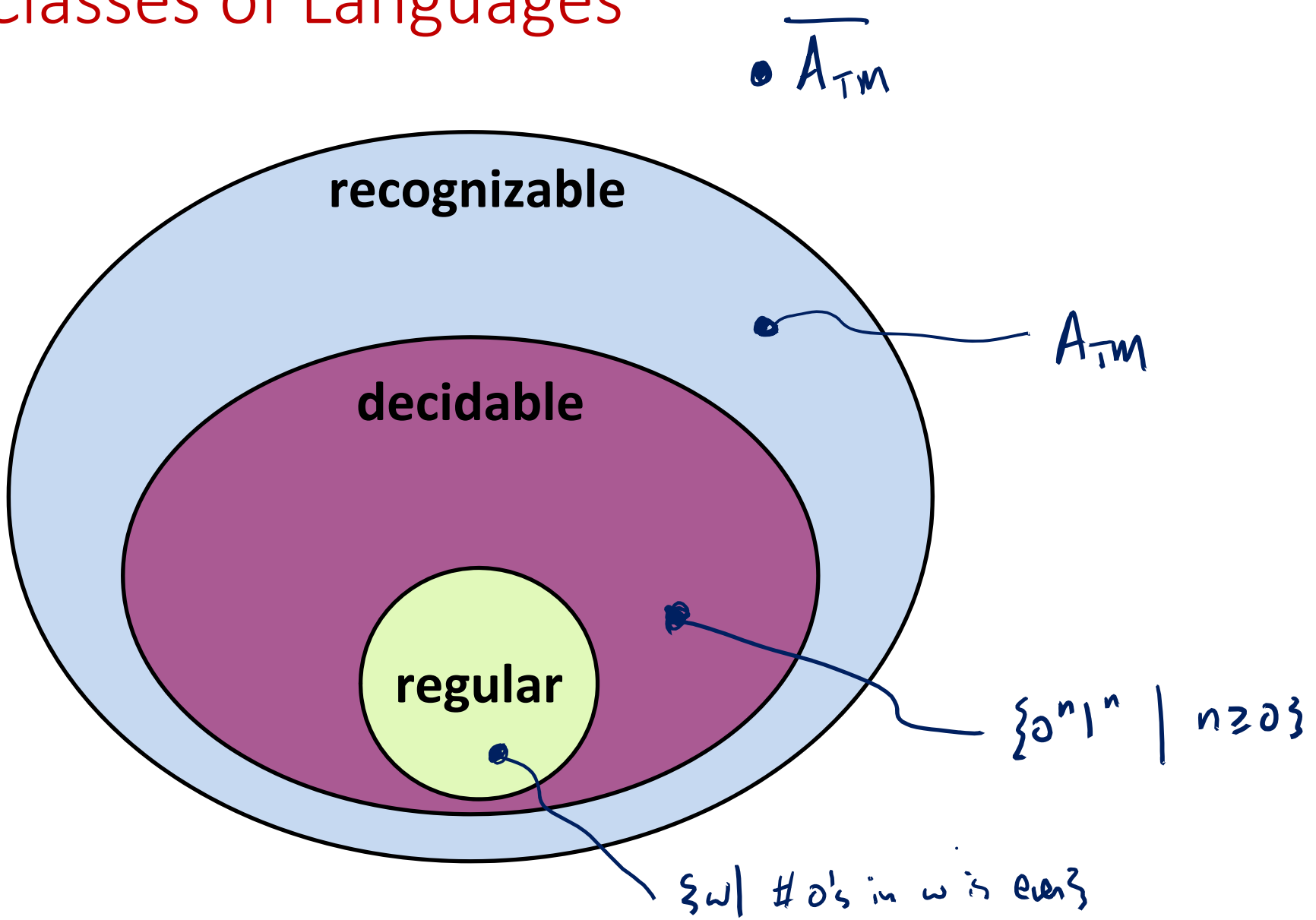
On input w:
1) Run M on w. If accepts, accept
2) Run M' on w. If accepts, reject.

Problem: If $w \notin L$ and M loops on w, then N loops on w

Attempt 2:

On input w:
1) Repeat forever:
   a) Run M for one step on w. If accepts, accept
   b) Run M' for one step on w. If accepts, reject.

Classes of Languages



Reductions

Scientists vs. Engineers

A computer scientist and an engineer are stranded on a desert island. They find two palm trees with one coconut on each. The engineer climbs a tree, picks a coconut and eats.

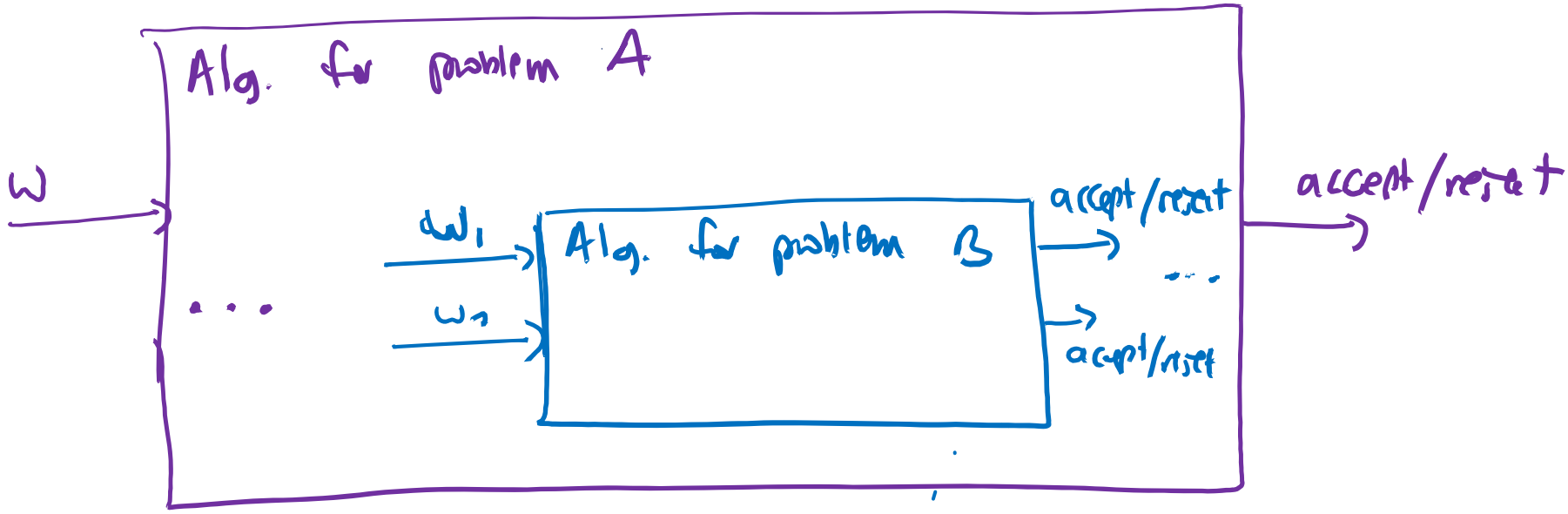
The computer scientist climbs the second tree, picks a coconut, climbs down, climbs up the first tree and places it there, declaring success.

"Now we've reduced the problem to one we've already solved." (Please laugh)

Reductions

A reduction from problem A to problem B is an algorithm solving problem A which uses an algorithm solving problem B as a subroutine

If such a reduction exists, we say "A reduces to B"



Reductions

A reduction from problem A to problem B is an algorithm solving problem A which uses an algorithm solving problem B as a subroutine

If such a reduction exists, we say "A reduces to B"

If A reduces to B, and B is decidable, what can we say about A?

a) A is decidable

b) A is undecidable

c) A might be either decidable or undecidable

Two uses of reductions

Positive uses: If A reduces to B and B is decidable, then A is also decidable

$E_{DFA} = \{\langle D \rangle \mid L(D) = \emptyset\}$

$EQ_{\text{DFA}} = \{\langle D_1, D_2 \rangle \mid D_1, D_2 \text{ are DFAs and } L(D_1) = L(D_2)\}$

Theorem: EQ_{DFA} is decidable

Proof: The following TM decides EQ_{DFA}

"Reduction"

On input $\langle D_1, D_2 \rangle$, where $\langle D_1, D_2 \rangle$ are DFAs:

1. Construct a DFA D that recognizes the symmetric difference $L(D_1) \triangle L(D_2)$
2. Run the decider for E_{DFA} on $\langle D \rangle$ and return its output

Two uses of reductions

Negative uses: If A reduces to B and A is undecidable, then B is also undecidable

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

Suppose H decides A_{TM}

Reduction from UD to A_{TM}

Consider the following TM D.

On input $\langle M\rangle$ where M is a TM:

1. Run H on input $\langle M, \langle M\rangle\rangle$
2. If H accepts, reject. If H rejects, accept.

Claim: D decides

$UD = \{\langle M\rangle \mid M \text{ is a TM that does not accept input } \langle M\rangle\}$

Two uses of reductions

Negative uses: If A reduces to B and A is undecidable, then B is also undecidable

Template for undecidability proof by reduction:

1. Suppose to the contrary that B is decidable
2. Using a decider for B as a subroutine, construct an algorithm deciding A
3. But A is undecidable. Contradiction!