BU CS 332 – Theory of Computation

Lecture 17:

- Midterm II review

Reading:

Sipser Ch 3.1-5.1, 5.3

Discussion sections:

- Answering questions about HW 6
- More practice problems

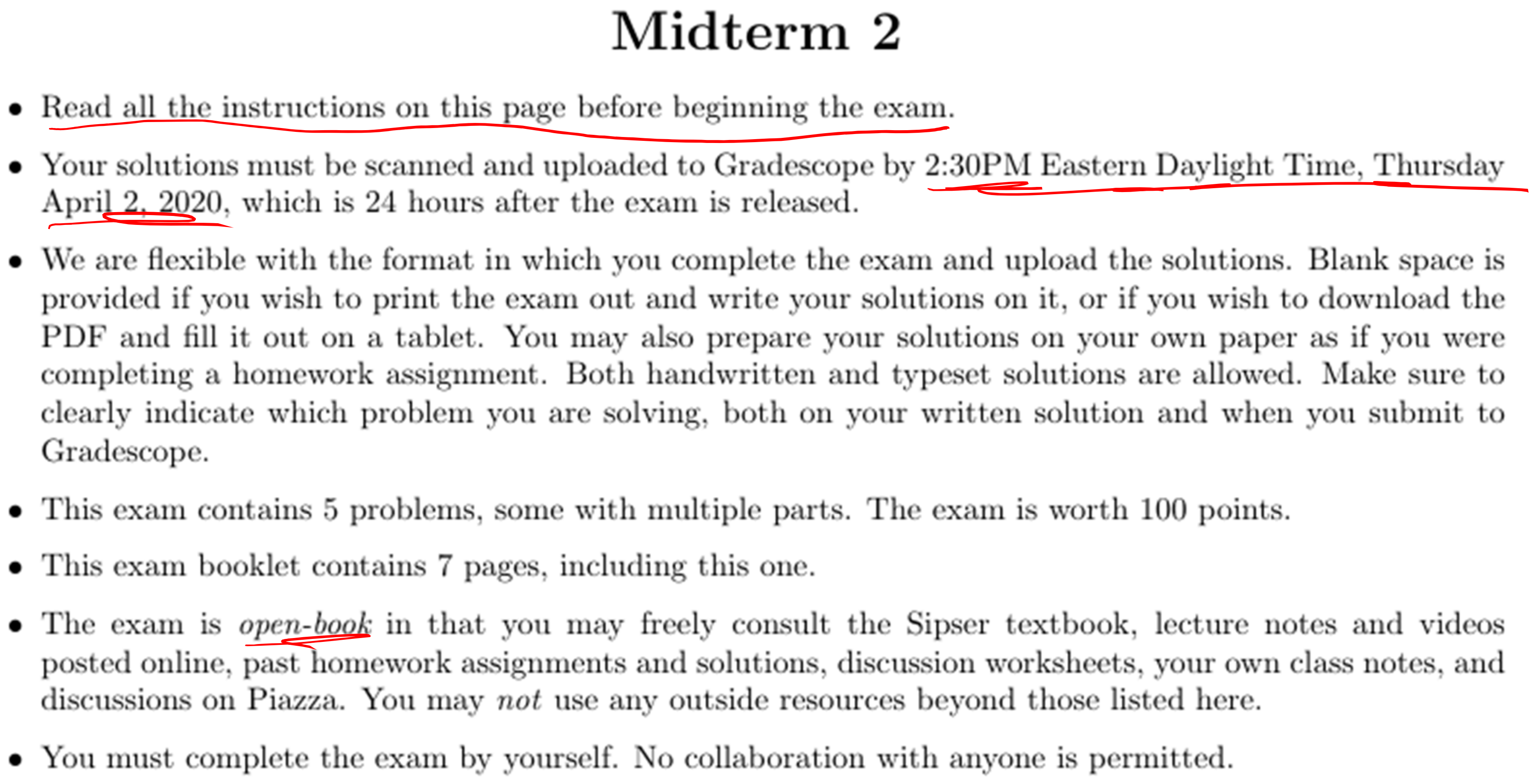
I'll have extra office hours Tuesday 3/31 5-6 pm

Mark Bun

March 30, 2020

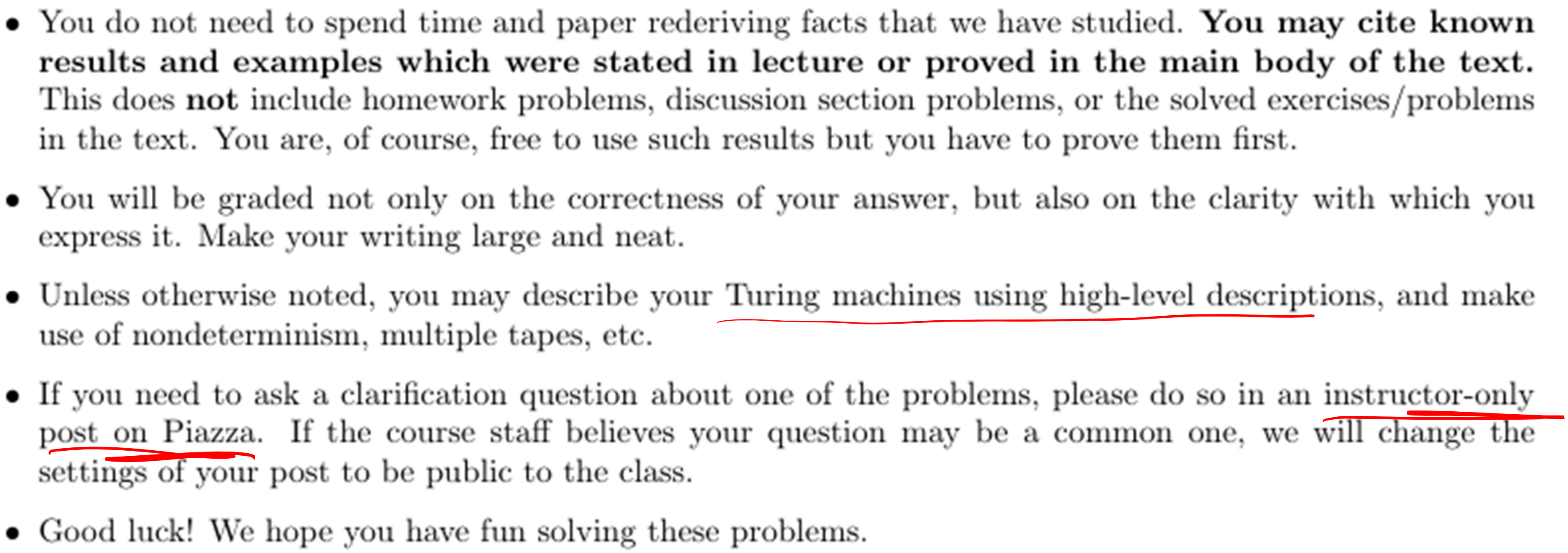
Format of the Exam

Release: 2:30 PM, Wed 4/1



Midterm 2

- Read all the instructions on this page before beginning the exam.
- Your solutions must be scanned and uploaded to Gradescope by 2:30PM Eastern Daylight Time, Thursday April 2, 2020, which is 24 hours after the exam is released.
- We are flexible with the format in which you complete the exam and upload the solutions. Blank space is provided if you wish to print the exam out and write your solutions on it, or if you wish to download the PDF and fill it out on a tablet. You may also prepare your solutions on your own paper as if you were completing a homework assignment. Both handwritten and typeset solutions are allowed. Make sure to clearly indicate which problem you are solving, both on your written solution and when you submit to Gradescope.
- This exam contains 5 problems, some with multiple parts. The exam is worth 100 points.
- This exam booklet contains 7 pages, including this one.
- The exam is *open-book* in that you may freely consult the Sipser textbook, lecture notes and videos posted online, past homework assignments and solutions, discussion worksheets, your own class notes, and discussions on Piazza. You may *not* use any outside resources beyond those listed here.
- You must complete the exam by yourself. No collaboration with anyone is permitted.



- You do not need to spend time and paper rederiving facts that we have studied. **You may cite known results and examples which were stated in lecture or proved in the main body of the text.** This does **not** include homework problems, discussion section problems, or the solved exercises/problems in the text. You are, of course, free to use such results but you have to prove them first.
- You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Make your writing large and neat.
- Unless otherwise noted, you may describe your Turing machines using high-level descriptions, and make use of nondeterminism, multiple tapes, etc.
- If you need to ask a clarification question about one of the problems, please do so in an instructor-only post on Piazza. If the course staff believes your question may be a common one, we will change the settings of your post to be public to the class.
- Good luck! We hope you have fun solving these problems.

Midterm II Topics

Turing Machines (3.1, 3.3)

- Know the three different “levels of abstraction” for defining Turing machines and how to convert between them: Formal/state diagram, implementation-level, and high-level
- Know the definition of a configuration of a TM and the formal definition of how a TM computes
- Know how to “program” Turing machines by giving implementation-level descriptions
- Understand the Church-Turing Thesis

TM Variants (3.2)

- Understand the following TM variants: Multi-tape TMs, Nondeterministic TMs, Enumerators
- Know how to give a simulation argument (implementation-level description) to compare the power of TM variants
- Understand the specific simulation arguments we've seen: multi-tape TM by basic TM, nondeterministic TM by basic TM, enumerator by basic TM and basic TM by enumerator.

Decidability (4.1)

- Understand how to use a TM to simulate another machine (DFA, another TM)
- Know the specific decidable languages from language theory that we've discussed, and how to decide them: A_{DFA}, E_{DFA}, EQ_{DFA}, A_{CFG}, E_{CFG}, etc.
- Know how to use a reduction to one of these languages to show that a new language is decidable
- (You don't need to know details of what Chomsky Normal Form is, but understand how it is used to prove decidability of A_{CFG})

Undecidability (4.2)

- Know the definitions of countable and uncountable sets and how to prove countability and uncountability
- Understand how diagonalization is used to prove the existence of explicit undecidable languages (A_{TM} in the book, or $\overline{SA_{TM}}$ from lecture)
- Know that a language is decidable iff it is recognizable and co-recognizable, and understand the proof

A is co-recognizable iff $\overline{A}$ is recognizable

Reducibility (5.1)

- Understand how to use a reduction (contradiction argument) to prove that a language is undecidable
- Know the reductions showing that $HALT_{TM}$, $E_{TM}, REGULAR_{TM}, CFL_{TM}, EQ_{TM}$ are undecidable

- You are not responsible for understanding the computation history method. However, you should know that the language ALL_{CFG} is undecidable, and reducing from it might be useful.

Mapping Reducibility (5.3)

- Understand the definition of a computable function
- Understand the definition of a mapping reduction
- Know how to use mapping reductions to prove decidability, undecidability, recognizability, and unrecognizability

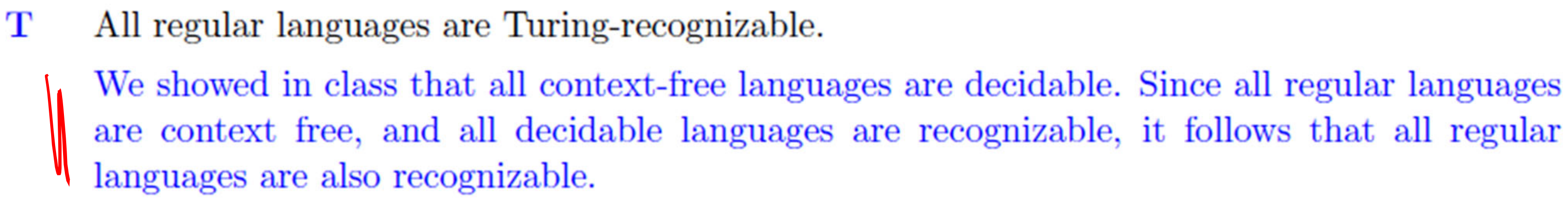
$A \leq_m B$, B recognizable $\Rightarrow$ A recognizable

$\overline{A_{TM}}$ "reduces" to A_{TM}

Tips for Preparing Exam Solutions

True or False

- It’s all about the justification!
- The logic of the argument has to be clear
- Restating the question is not justification; we’re looking for some additional insight



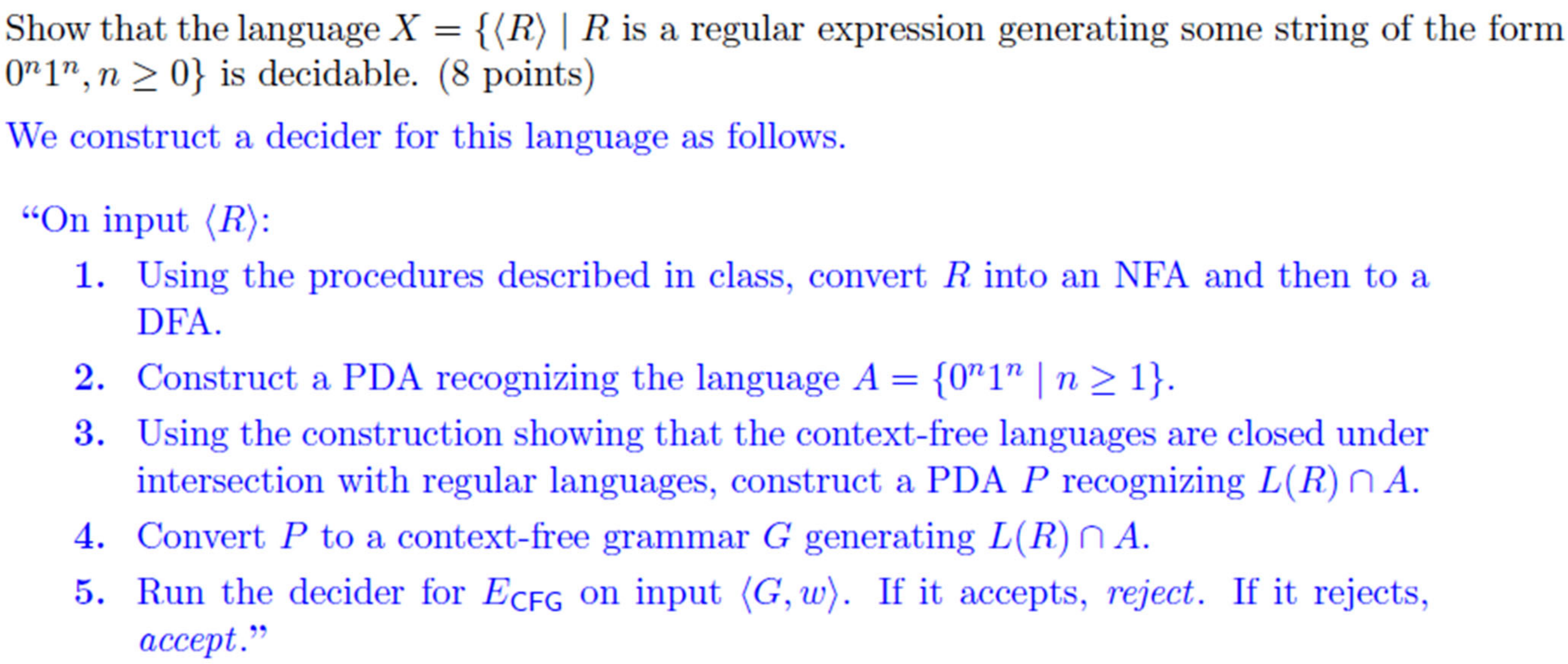
T All regular languages are Turing-recognizable.

We showed in class that all context-free languages are decidable. Since all regular languages are context free, and all decidable languages are recognizable, it follows that all regular languages are also recognizable.

Regular $\Rightarrow$ CFL Decidable $\Rightarrow$ Recognizable

CFL $\Rightarrow$ Decidable

Simulation arguments, constructing deciders



Show that the language $X = \{\langle R \rangle \mid R$ is a regular expression generating some string of the form $0^n1^n, n \geq 0\}$ is decidable. (8 points)

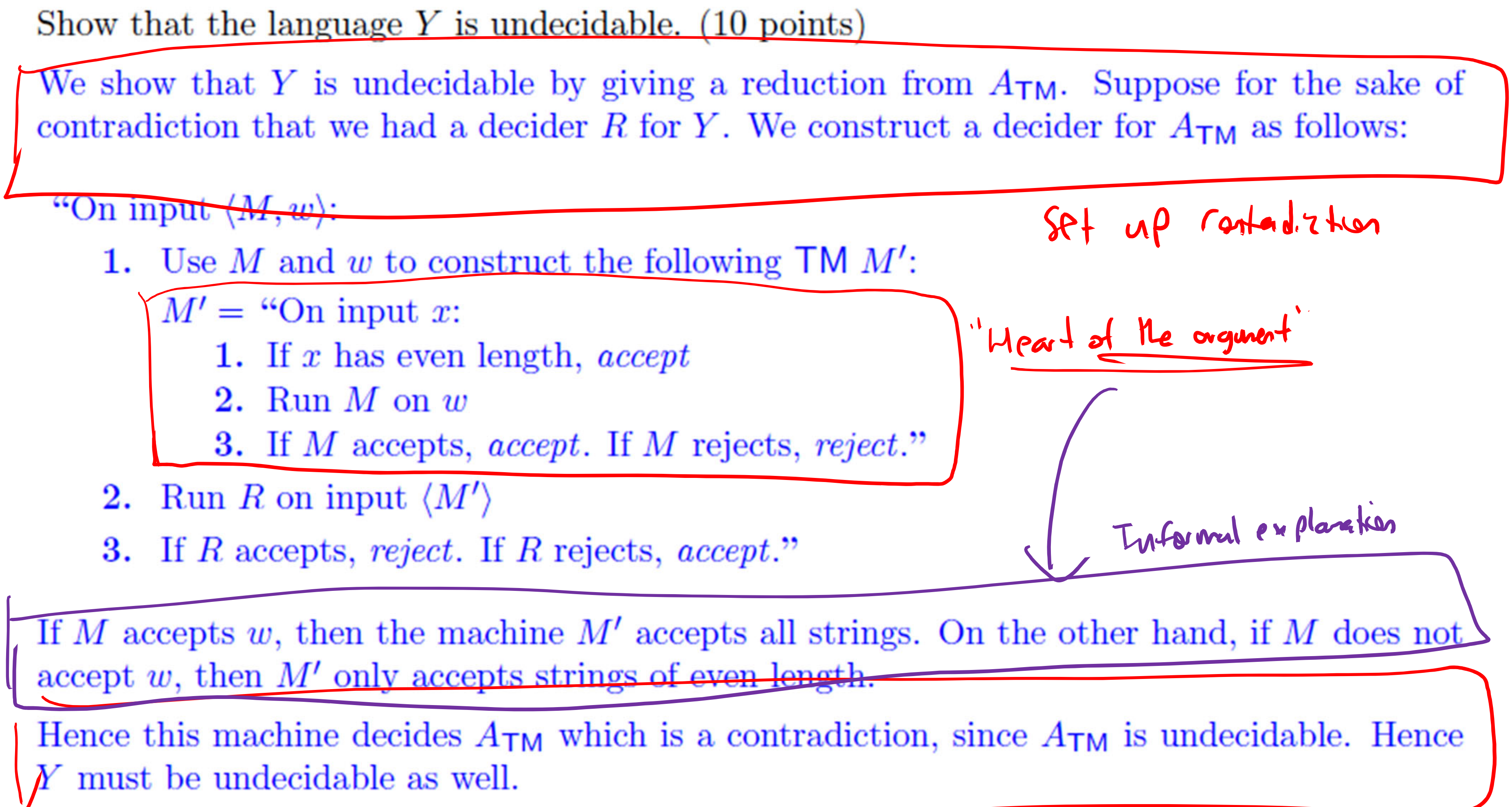
We construct a decider for this language as follows.

"On input $\langle R \rangle$:

1. Using the procedures described in class, convert R into an NFA and then to a DFA.
2. Construct a PDA recognizing the language $A = \{0^n1^n \mid n \geq 1\}$.
3. Using the construction showing that the context-free languages are closed under intersection with regular languages, construct a PDA P recognizing $L(R) \cap A$.
4. Convert P to a context-free grammar G generating $L(R) \cap A$.
5. Run the decider for E_{CFG} on input $\langle G, w \rangle$. If it accepts, *reject*. If it rejects, *accept*."

- Full credit for a clear and correct description of the new machine
- Still a good idea to provide an explanation
  (partial credit, clarifying ambiguity)

Undecidability proofs



Show that the language Y is undecidable. (10 points)

We show that Y is undecidable by giving a reduction from A_{TM}. Suppose for the sake of contradiction that we had a decider R for Y. We construct a decider for A_{TM} as follows:

Set up contradiction

```
"On input ⟨M, w⟩:
  1. Use M and w to construct the following TM M':
       M' = "On input x:
         1. If x has even length, accept
         2. Run M on w
         3. If M accepts, accept. If M rejects, reject."
  2. Run R on input ⟨M'⟩
  3. If R accepts, reject. If R rejects, accept."
```

"Heart of the argument"

Informal explanation

If M accepts w, then the machine M' accepts all strings. On the other hand, if M does not accept w, then M' only accepts strings of even length.

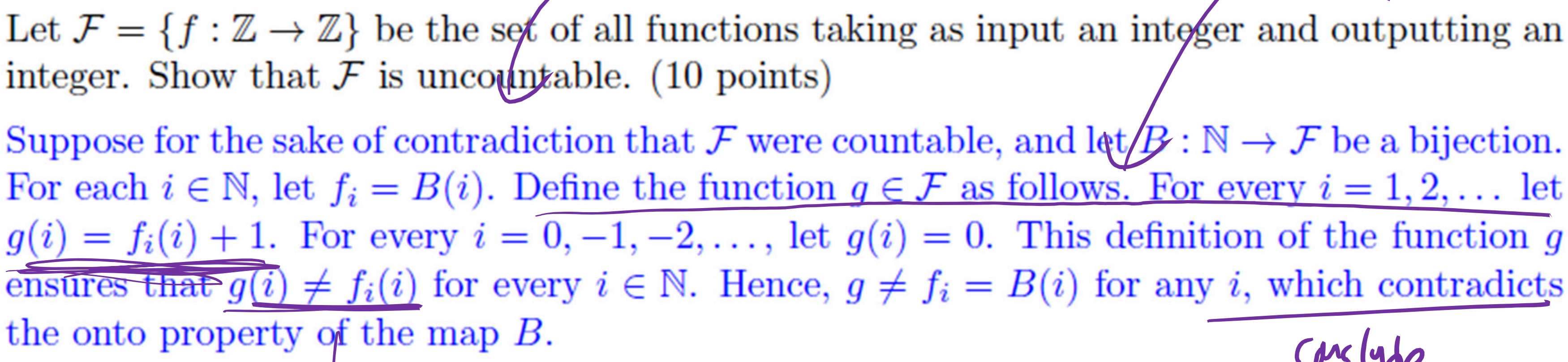
Hence this machine decides A_{TM} which is a contradiction, since A_{TM} is undecidable. Hence Y must be undecidable as well.

Conclude contradiction

Uncountability proofs

set up contradiction

Define element of $\mathcal{F}$ not in table



Let $\mathcal{F} = \{f : \mathbb{Z} \to \mathbb{Z}\}$ be the set of all functions taking as input an integer and outputting an integer. Show that $\mathcal{F}$ is uncountable. (10 points)

Suppose for the sake of contradiction that $\mathcal{F}$ were countable, and let $B : \mathbb{N} \to \mathcal{F}$ be a bijection. For each $i \in \mathbb{N}$, let $f_i = B(i)$. Define the function $g \in \mathcal{F}$ as follows. For every $i = 1, 2, \ldots$ let $g(i) = f_i(i) + 1$. For every $i = 0, -1, -2, \ldots$, let $g(i) = 0$. This definition of the function g ensures that $g(i) \neq f_i(i)$ for every $i \in \mathbb{N}$. Hence, $g \neq f_i = B(i)$ for any i, which contradicts the onto property of the map B.

conclude

Explain why it's not in the table

- The 2-D table is useful for thinking about diagonalization, but is not essential to the argument
- The essential part of the proof is the construction of the "inverted diagonal" element, and the proof that it works

Practice Problems

Decidability and Recognizability

Let $A = \{\langle D\rangle \mid D \text{ is a DFA that does not accept any string containing an odd number of } 1's\}$

Show that A is decidable

Prove that $\overline{E_{\mathrm{TM}}}$ is recognizable

Prove that if A and B are decidable, then so is $A \setminus B$

Countable and Uncountable Sets

Show that the set of all valid (i.e., compile without errors) C++ programs is countable

A Celebrity Twitter Feed is an infinite sequence of ASCII strings, each with at most 140 characters. Show that the set of Celebrity Twitter Feeds is uncountable.

Ex. ("How can mirrors be real if our eyes aren't", "LOL", "I'm out of creativity", . . .) ⟶ F_i

String of length ≤ 140

Index into sequence

$F_i^2 = $ "LOL"

Idea:

Twitter feeds

	1	2	3	4	. . .
F_1	F_1^1	F_1^2	F_1^3		. . .
F_2		F_2^2			
F_3			F_3^3		
⋮				⋱	

Assume for contradiction that {CTFs} is countable, let $F_1, F_2, F_3 ..$ be an enumeration of CTFs. Construct a new feed $G = (G^1, G^2, G^3, \dots)$

$$G^i = \begin{cases} \varepsilon & \text{if } F_i^i \text{ is nonempty} \\ a & F_i^i \text{ is empty} \end{cases}$$

- By definition $G^i \neq F_i^i$ for every i ⇒ $G \neq F_i$ for every i
- G is a CTF (every G^i has length ≤ 140) Thus G is a CTF that does not appear in the table ✗

Undecidability and Unrecognizability

Prove or disprove: If A and B are recognizable, then so is $A \setminus B$

Prove that the language $ALL_{\mathrm{TM}} = \{\langle M\rangle | M \text{ is a TM and } L(M) = \Sigma^*\}$ is undecidable

Give a nonregular language A such that $\underbrace{A \le_m}_{\text{"mapping reduction"}} L(0^*1^*)$ or prove that none exists

Idea: Construct A nonregular, but decidable

Ex: $A = \{0^n 1^n \mid n \ge 0\}$ (context free, but not regular)

$\Downarrow$

decidable

Mapping reduction showing $A \le_m L(0^*1^*)$:

TM implement function f

```
On input $x$:
  1) Check if $x$ is of the form $0^n 1^n$ $(n \ge 0)$
  2) If yes: output $x$
     If no: output $10$
```

If $x \in A$, $f(x) = x \in L(0^*1^*)$ $\Rightarrow$ f is a mapping reduction from A to $L(0^*1^*)$

If $x \notin A$, $f(x) = 10 \notin L(0^*1^*)$

$\boxed{A \le_m B,\ B \text{ decidable} \Rightarrow A \text{ decidable}}$ // $A \le_m B,\ A$ undecidable $\Rightarrow B$ undecidable

Give an undecidable language A such that $A \le_m L(0^*1^*)$ or prove that none exists

Claim: There is no undecidable A s.t. $A \le_m L(0^* 1^*)$

Proof. Assume (for contradiction) $\exists$ undecidable A s.t. $A \le_m L(0^*1^*)$

(No decider for A)

$\Longrightarrow L(0^*1^*)$ undecidable $\Big\{$ but $L(0^*1^*)$ is regular $\Rightarrow$ context free $\Rightarrow$ decidable $\Big\}$ ✗

Proof (v.2) $L(0^*1^*)$ decidable $\Longrightarrow A$ decidable

$\Longrightarrow$ impossible for A to be undecidable

If $x \in L(0^*1^*) \Rightarrow f(x) \in A_{TM}$ | If $x \notin L(0^*1^*) \Rightarrow f(x) \notin A_{TM}$

$= \langle M, w \rangle$

Give an undecidable language A such that $\underbrace{L(0^*1^*)}_{\text{regular, hence decidable}} \leq_m A$ or prove that none exists

$A = A_{TM} = \{ \langle M, w \rangle \mid M$ is a TM, w is an input to M, M accepts when run on $w \}$

Mapping reduction f from $L(0^*1^*)$ to A:

Idea: Just solve $L(0^*1^*)$ and create an instance to A_{TM} depending on solution

On input x: [x input to $L(0^*1^*)$]

1) If x is of the form 0^*1^*:
   Output $\langle M, w \rangle$ where M = "Ignore input and accept"
   $w = \epsilon$

2) Otherwise if x is not of the form 0^*1^*:
   Output $\langle M, w \rangle$ where M = "Ignore input and reject"
   $w = \epsilon$