18.404 Midterm Review

18.404 Course Staff

MIT

Regular Languages

How do we show a language is regular?

- Use closure properties
- Construct a regex, NFA, or DFA

How do we prove that it's not?

- Pumping Lemma (up or down)

Context Free Languages

How do we prove that a language is context-free?

- Use closure under union or intersection with regular languages
- Construct PDAs or context-free grammars

Examples from class

- $\{0^k 1^k\}$
- $\{0^k 1^{2k}\}$
- $\{0^a 1^b | b \leq a \leq 2b\}$

How do we prove that it's not?

- Pumping Lemma (up or down)
- Closure properties then pumping lemma

T-Recognizable Languages

How do we show that a language is recognizable?

- Construct a TM that recognizes it
- Construct an enumerator
- Show that it's mapping reducible to A_{TM} (or other recognizable language)

How do we show that a language is co-recognizable?

- Prove that the complement is recognizable
- Show that it's mapping reducible to $\overline{A_{\mathrm{TM}}}$

How do we show a language is not recognizable?

- Show that it's co-recognizable and undecidable
- Show that A_{TM} is mapping reducible to its complement.
- Try to obtain a contradiction in other ways (for example using the recursion theorem, or using diagonal argument like with A_{TM})

T-Decidable Languages

How do we show that a language is decidable?

- Construct a TM that decides it
- Prove that it's recognizable and co-recognizable

How do we show that a language is not decidable?

- Reduce A_{TM} to the problem
- Use the computation history method (especially if the problem is about existence in other computational models)
- Sometimes you can use the recursion theorem to obtain simpler proofs

Example: Proving Regular

Let $L = \{w$ containing exactly one `a` or exactly one `b`$\}$.
Show that L is regular by explicitly giving:

1. A regular expression
2. An NFA or DFA

Example: Proving not regular

Let $L = \{w \text{ contains only } \texttt{0}\text{'s and } |w| = k^2 \text{ for some } k \in \mathbb{N}\}$.
Show that L is not regular.

Example: Lots of 1's

Call a language *major* if it includes some string s with with strictly more 1's than 0's.
Is $L = \{\langle M \rangle \mid M \text{ is a DFA and } L(M) \text{ is major}\}$ decidable?

Example: CFLs

Is $L = \{w\#x$ such that w^R is a substring of $x\}$ context-free?

Example: Closure Properties

Suppose A is context free, B is not context free. Which of the following are possible:

1. $B = A \cap C$ for C regular?
2. $B = A \cap C$ for C context free?
3. $B = A \cup C$ for C regular?
4. $B = A \cup C$ for C context free?

Example: The Butterfly's Head

Recall: A_{TM} is T-recognizable, but $\overline{A_{\mathrm{TM}}}$ is not.

1. True or false? $\overline{A_{\mathrm{TM}}} \leq_{\mathrm{m}} A_{\mathrm{TM}}$.
2. Let $J = \{bx : b = 0 \text{ if } x \in A_{\mathrm{TM}} \text{ otherwise } 1\}$. Prove $A_{\mathrm{TM}} \leq_{\mathrm{m}} J$ and $\overline{A_{\mathrm{TM}}} \leq_{\mathrm{m}} J$.
3. Prove $J \leq_{\mathrm{m}} \overline{J}$.
4. Is $\overline{E_{\mathrm{LBA}}} \leq_{\mathrm{m}} J$? **Hint:** What if you find a mapping reduction to either A_{TM} or $\overline{A_{\mathrm{TM}}}$?

Example: Recognize Recognizability

Let $SH = \{\langle M \rangle : M$ halts on some inputs$\}$. Is SH T-recognizable? What about $\overline{SH}$?

Hints:

1. To show SH is T-recognizable, recall how we showed $\overline{E_{\text{TM}}}$ is T-recognizable.
2. Use a mapping reduction to show $\overline{SH}$ is T-unrecognizable.

Example: Recognize Recognizability

Let $AH = \{\langle M \rangle : M$ halts on all inputs$\}$. Is AH T-recognizable? What about $\overline{AH}$?

Hints:

1. Use a mapping reduction to show $\overline{AH}$ is T-unrecognizable.
2. AH is also T-unrecognizable. To construct the mapping in $\overline{A_{\mathrm{TM}}} \leq_{\mathrm{m}} \overline{AH}$, map $\langle M, w \rangle$ to a Turing machine that uses its input as a parameter to its simulation of M on w.

Mapping Reduction Edge Case

True or false? Whenever A is decidable and B is regular: $A \leq_{\mathrm{m}} B$.

Double The Tapes Is Double The Fun!

Definition: A ***2-tape DFA*** has two inputs, with a read-only, left-to-right head on each. The transition function is a mapping $Q \times (\Sigma \times \Sigma) \to Q \times \{\texttt{Head 1}, \texttt{Head 2}\}$, to indicate the new state and which head to step right. It accepts if it moves a head off the end of the tape while in an accepting state.

Show that the following language is decidable:

$$ALL_{\text{2-tape DFA}} = \{\langle M \rangle \mid \text{2-tape DFA } M \text{ accepts every } (x, y)\}.$$

Hints:

1. Try to construct a PDA P with alphabet $\Sigma \cup \{\#\}$, such that $L(P)$ is empty iff M accepts every (x, y). Then use the E_{CFG} decider.

Once More, With Nondeterminism!

Definition: A ***2-tape NFA*** has transition function $Q \times (\Sigma \times \Sigma) \to \mathcal{P}(Q \times \{\texttt{Head 1}, \texttt{Head 2}\})$. It accepts if on some branch the NFA moves a head off the end of the tape while in an accepting state.

Show that the following language is *undecidable*:

$$ALL_{\text{2-tape NFA}} = \{\langle M \rangle |\ \text{2-tape NFA } M \text{ accepts every } (x, y)\}.$$

Hints:

1. Construct a 2-tape NFA that accepts strings that are *not* an accepting computation history.