Exam 3 will be cumulative, but it will emphasize material covered since the second exam. Thus while any material that’s been covered in class or in the assigned reading might be included, questions on earlier topics are more likely to involve fundamental concepts like the differences between regular and context-free languages or the models of computation we’ve seen than they are to require you to carry out cumbersome constructions such as conversion from an NFA-ε to a DFA.

As with the first two exams, Exam 3 is open-textbook/open-notes.

Topics

The new topics for this exam include:

1 Turing machines

- Formal (tuple) specification and transition (state) diagrams
- Using sequences of configurations (a.k.a., instantaneous descriptions) to show the moves of the TM
- Relations between variants of Turing machines, e.g., with a tape infinite in one direction or two, single-tape or multi-tape, deterministic vs nondeterministic
- Given a Turing machine’s state diagram or specification, be able to describe the machine’s behavior and language.
- Design a Turing machine that recognizes a specific language and/or behaves in a specific way. Describe it using a state diagram – for very simple problems only! – or pseudo-code.
- Definition of Turing-recognizable (RE) and Turing-decidable (R), and especially how they differ.
- The universal Turing machine $U$.

2 Decidable and undecidable problems

- Know the definitions of standard decision problems involving Turing machines, e.g., $A_{TM}$, $E_{TM}$.
- Be able to classify each of these standard problems as decidable (R), recognizable (RE), co-recognizable (co-RE), or none of these.
- For the decidable or recognizable problems, sketch an algorithm that decides or recognizes it.

3 Proving languages undecidable (somewhat difficult parts)

- Know that a language is decidable if and only if the language and its complement are both recognizable.
• Know that decidable languages are closed under complement, but recognizable languages aren’t.

• Prove that a language is undecidable or not recognizable, using a simple proof by contradiction that relies on closure properties or similar basic facts.

4 Proving languages undecidable (more difficult parts)

• Write a simple diagonalization proof, e.g., show that the reals are uncountable, show that there are more languages than Turing machines\(^2\).

• Understand the diagonalization proof that \(A_{\text{TM}}\) is undecidable (even if you aren’t sure you fully believe in it!).

\(^2\) See Theorem 4.18
Problem 1

Provided below is a Turing machine for the language \( \{a^n b^n \mid n \in \mathbb{N}_0\} \). The input alphabet is \( \{a, b\} \).

Modify the above to create a TM for the language \( \{a^n b^{2n} \mid n \in \mathbb{N}_0\} \), and draw a diagram for the resulting TM.
Problem 2

For each proof, explain clearly why the provided proof does not satisfactorily prove the given claim.

a. CLAIM The language described by the regular expression $a^*b$ is undecidable.

PROOF By contradiction; assume $a^*b$ is decidable. Let $D$ be a decider for it. Consider what happens when we run $D$ on a string of infinitely many as followed by a $b$ and on a string of infinitely many as. Let's call this first string $x$ and the second string $y$. Since $D$ is a decider, it halts on all inputs, and therefore cannot run for an infinitely long time. Therefore, $D$ must halt before reading the last character of $x$ and the last character of $y$. Because $x$ and $y$ are the same except for their last character, we see that $D$ must have the same behavior when run on $x$ and when run on $y$. If $D$ accepts $x$, then $D$ also accepts $y$, but $y$ is not in the language $a^*b$. Otherwise, $D$ rejects $x$, but $x$ is in the language $a^*b$. Both cases contradict the fact that $D$ is a decider for $a^*b$. We have reached a contradiction, so our assumption must have been wrong. Thus $a^*b$ is undecidable. ■

b. CLAIM The set of languages that can be recognized by a deterministic pushdown automaton is equivalent to the set of languages that can be recognized by a Turing machine.

PROOF We prove the conjecture by showing how to simulate any DPDA with a TM... (unnecessary details elided) ... Thus, we can simulate a DPDA with a TM. This proves a DPDA is equivalent to a TM.
Problem 3

Consider the Turing machine $M = (\{q_0, q_1, q_{\text{accept}}, q_{\text{reject}}\}, \{0, 1\}, \{0, 1, \square\}, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$, where $\delta$ consists of the following rules:

\[
\begin{align*}
\delta(q_0, \square) &= (q_{\text{reject}}, \square, R) \\
\delta(q_0, 0) &= (q_0, \square, R) \\
\delta(q_0, 1) &= (q_1, \square, R) \\
\delta(q_1, \square) &= (q_{\text{accept}}, \square, R) \\
\delta(q_1, 0) &= (q_{\text{reject}}, \square, R) \\
\delta(q_1, 1) &= (q_1, \square, R)
\end{align*}
\]

a. Informally but clearly describe the language $L(M)$.

b. Give a string in this language that is of length at least 4, and show the sequence of moves of this TM when this string is the input to $M$. 
Problem 4

Answer each of the following as indicated.

a. Indicate whether the following is true or false and fully justify your answer: If language $A$ is a subset of language $B$, and $A$ is Turing-recognizable, then $B$ is Turing-recognizable.

b. Indicate whether the following is true or false and fully justify your answer: A nondeterministic TM accepts a string $w$ if and only if every possible computation on the input leads to an accepting state.

c. Indicate whether the following is true or false and fully justify your answer: The computational path of a TM on an input $w$ always either halts accepting, halts rejecting, or enters a never-halting loop.
d. Circle the number to the left of each of the languages below that is Turing-decidable but not context-free:

1. \( \{ww^R \mid w \in \{a, b\}^*\} \)
2. \( \{wcw \mid w \in \{a, b\}^*\} \)
3. \( \{a^{2n}b^n c^{2n} \mid n \in \mathbb{N}_0\} \)

e. Circle the number to the left of each of the languages below that is Turing-recognizable but not Turing-decidable:

1. \( \{\langle M, w \rangle \mid \text{TM } M \text{ accepts input } w\} \)
2. \( \{\langle M \rangle \mid L(M) \neq \emptyset\} \)
3. \( \{\langle M_1, M_2 \rangle \mid L(M_1) = L(M_2)\} \)

f. Circle the number to the left of each of the languages below that is not Turing-recognizable:

1. \( \{\langle M, w \rangle \mid \text{TM } M \text{ halts on input } w\} \)
2. \( \{\langle M \rangle \mid L(M) \neq \emptyset\} \)
3. \( \{\langle M_1, M_2 \rangle \mid L(M_1) \cap L(M_2) = \emptyset\} \)
Problem 5

Determine whether the following problems are decidable or not:

a. Is a number $m$ prime?

b. Given a regular language $L$ and a string $w$, is $w \in L$?
Problem 6

Prove that the language $L = \{ \langle M \rangle \mid M$ is a TM and $M$ accepts string $w$ whenever it accepts $w^R \}$ is undecidable.  

*Hint:* Do a proof by contradiction. Show, at least informally, how you could solve a known undecidable problem if $L$ were decidable.