CSE 105
Theory of Computation

Professor Jeanne Ferrante
Today’s Agenda

• Final Review

Reminders and announcements:
• Final Exam Review: Tonight, starts 7 pm, Peterson 108
• Final Exam: Sat Jun 4, 11:30 am - 2:29 pm in WLH 2001
• BRING
  • Your ID
  • Your Seat Assignment
  • One 5 in by 8 in index card of notes (both sides)
Test Tips

• Get a good night’s sleep night before
• Practice writing out solutions beforehand

At the Test
• Remember to breathe!
• Read through all the problems before starting, decide which are harder
• Recommended strategy:
  Start harder problem but don’t get stuck there!
  Only spend a few minutes
  Jump to easier
  Go back to harder ...
Map we used for the quarter

What is theory of Computation?
What is “Theory of Computation”?

• **What is Computation?**
  
  • **Computational Models**, successively more powerful, leading up to general model, the Turing Machine
  
  • **Computability**: What languages are recognized by each model? What problems (suitably encoded) can be solved by each model? Which cannot be?
  
  • **Complexity**: How easy or hard is a given language to recognize, or problem to solve?
    
    • Decidability, Time Complexity, P, NP
MODEL:
Understand, Design, Description of Language

1. Finite Automata & Regular Expressions
2. Push Down Automata & CFG’s
3. Turing Machines (May Not Always Halt)
   • If Always Halt

LANGUAGE CLASS:
Closure Properties, Language Problems, NOT in class

1. Regular Languages
   Show NOT in Class: Pumping Lemma
2. Context-Free Languages
3. Turing Recognizable Languages (TR)
   • Decidable
     • P
     • NP
4. Undecidable, Non TR
   (Diagonalization, reduction)
Problem: Show $L$ is regular.

Solution:

1. Construct a
   • DFA, or
   • NFA, or
   • Regular Expression

2. Show that it is correct for $L$:
   $$w \in L \iff w \text{ is accepted by ...}$$

Ex 1: $L = \{w \in \{0,1\}^* \mid w \text{ has an odd number of 1's}\}$

DFA:

Reg Exp:
Ex 2. \( L = \{w \mid w \text{ has an odd number of } 1\text{'s OR starts with a } 0\} \)

- NFA:

- Regular Expression:

**Qu:** To Show a language \( A \) is **Not** Regular, we can:
A. Show there is a CFG generating \( A \)
B. Use the Pumping Lemma for Regular Languages
C. Show \( A \) is undecidable
D. None or more than one of the above
To show a language $L$ is:

Decidable:
1. *Show that there is a TM $D$ that always halts and accepts exactly $L$
2. Show that you can reduce $L$ to a decidable problem
3. *Use closure properties

Recognizable:
1. *Show that there is a TM $R$ that accepts exactly $L$
2. Show that you can reduce $L$ to a recognizable problem
3. *Use closure properties

* Most frequently used

Not Decidable:
1. Use Diagonalization or
2. *Use Reduction (e.g. from $A_{TM}$ to $L$, alternatively via reduction of other undecidable language to $L$)

Not Recognizable:
1. If $L$ is undecidable, and show its complement is recognizable, then $L$ not recognizable. (Th 4.22)
Many Undecidability proofs follow a common pattern....

- Always a proof by contradiction
  - Assume $T$ is decidable by TM $M_T$
    - $T$ checks for condition $P$, and always halts with accept or reject
- Use $M_T$ to construct TM $M_{ATM}(M, w)$ to decide $A_{TM}$

- Within $M_{ATM}$, construct special TM $X$ such that
  1. If $M$ accepts $w$, then $L(X)$ has property $P$
  2. If $M$ does not accept $w$, then $L(X)$ has property not $P$

Run $M_T$ with input $<X>$ to distinguish between $P$ or not $P$ for $L(X)$, to decide if $M$ accepts $w$

- Show that $M_{ATM}$ decides $A_{TM}$ for the contradiction

Note: sometimes easier to build $X$ so that $X$ has $P$ iff $w$ not in $L(M)$
Show $T = \{ <M> \mid M \text{ is a TM, and } |L(M)| = 1 \}$ is undecidable

• Proof by contradiction: ($A_{TM}$ reduces to $T$)
• Assume $T$ is decidable by TM $M_T$. Use $M_T$ to construct TM $D_{ATM}$ that decides $A_{TM}$.
• $D_{ATM} =$ “On input $<M,w>$:
  1. Construct TM $Z =$ “ On input $x$:
     a) If $x \neq 105$ then reject.
     b) If $x = 105$, Run $M$ on input $w$. If it accepts, accept. If it rejects, reject.”
  2. Run $M_T$ on $<Z>$. If it accepts then accept, otherwise reject.
  • Correctness: ....
• But $A_{TM}$ is undecidable, a contradiction. So the assumption is false and $T$ is undecidable. QED.

What is $L(Z)$?
A. Empty set
B. $\{105\}$
C. $\{105\}$ if $M$ accepts $w$, and Empty set if $M$ does not accept $w$
D. Empty set if $M$ accepts $w$, and $\{105\}$ if $M$ does not accept $w
TH. 5.2: \( E_{TM} = \{<M> \mid M \text{ is a TM and } L(M) = \emptyset\} \) is undecidable

We show \( A_{TM} \) reduces to \( E_{TM} \)

Proof: Assume \( E_{TM} \) is decidable, with TM \( R \). We show then that \( A_{TM} \) is decidable, a contradiction.

- Using \( R \), we construct a TM \( M_{ATM} \) that decides \( A_{TM} \):

\[
M_{ATM} = \begin{cases} 
\text{Correctness: } M_{ATM} \text{ is a decider since } R \text{ is, and accepts } <M,w> \text{ iff } L(X) \text{ is nonempty iff } M \text{ accepts } w. \\
\text{But } A_{TM} \text{ is undecidable, a contradiction. So the assumption is false and } E_{TM} \text{ is undecidable.}
\end{cases}
\]

What is \( L(X) \) if \( M \) accepts \( w \)?

A. \( \{w\} \)
B. \( w \)
C. \( \emptyset \)
D. None above
## Countable and Uncountable

### Countable
- Show there is a 1-1 correspondence from \( \mathbb{N} \)

### Examples
- Any language over \( \Sigma \)
- Even numbers
- Integers
- Rational numbers
- All regular languages over \( \Sigma \)

### Uncountable
- Diagonalization (proof by contradiction)

### Examples
- Set of all subsets of \( \Sigma^* \)
- Real numbers between 0 and 1
- Real Numbers
- Set of all infinite binary sequences
If L is (in a language class) then show L’ is also in the same (language class):

Given: L is in (Regular languages or CFL or Decidable or Recognizable…)
so it has a (DFA/NFA or CFG or TM decider or TM recognizer…)

Want to show: L’ has a (DFA/NFA or CFG or TM decider or TM recognizer)

Proof Method: Direct Construction OR Use Closure Properties of the Class
Closure Properties of Language Classes

The Regular Languages are closed under:
- Union
- Intersection
- Complement
- Star
- Concatenation

The CF Languages are closed under:
- Union
- Intersection
- Complement
- Star
- Concatenation

The Decidable Languages are closed under:
- Union
- Intersection
- Complement
- Star
- Concatenation

The Turing-Recognizable Languages are closed under:
- Union
- Intersection
- Complement
- Star
- Concatenation
Tips for Writing Closure Proofs

• A closure proof provides an answer to the question, "If I have a class of languages, and do [blah] to a language in it, is the new language still in the class?"

• **GIVEN:** Write down what is known and give names to each of them and their component parts so you can use them later.

• **WANT TO SHOW:**
  • Announce what you will prove and your plan for it..

• **CONSTRUCTION:**
  • Let $M' = ...$, where ...
  • The construction will depend on the problem.
  • Though *not part of the proof*, a description in English of what you are trying to do is often useful.

• **CORRECTNESS:**
  • Here you prove that you construction actually works.

• **CONCLUSION:**
  • Finish by stating what you have proved.
Prove that the class of Turing-recognizable languages is closed under Concatenation

• Given: Two Turing-recognizable languages A and B, and TM’s that recognize them, $M_A$ and $M_B$.
• Want to Show: There is a TM $M$ that recognizes $A \cdot B$.
• Construction: 
  
  $M =$ “On input $w$: 
  
  1. Nondeterministically split $w$ into $x$ and $y$, $w = xy$, and for each such split:
    
    a. Simulate running $M_A$ on input $x$
      
      a. If it rejects, reject. If it accepts, go to step b:
    
    b. Simulate running $M_B$ on input $y$
      
      a. If it accepts, accept. If it rejects, reject.”

• Correctness: $w$ is in $A \cdot B$ if and only if $M$ accepts $w$:
  
  If $w$ is in $A \cdot B$ then $w = xy$ where $x$ is in $A$ and $y$ is in $B$. When we run $M_A$ on $x$ and $M_B$ on $y$ both halt and accept, so $M$ accepts $w$.
  
  If $w$ is not in $A \cdot B$ then for any split of $w = xy$, either $x$ is not in $A$ or $y$ is not in $B$. So either $x$ is not accepted by $M_A$ or $y$ is not accepted by $M_B$, and either one (or both) will reject or fail to halt on their input. Then $M$ will either reject $w$ or fail to halt on $w$, so $M$ does not accept $w$.

• Conclusion: We have constructed a TM that recognizes $A \cdot B$, therefore $A \cdot B$ is Turing-recognizable, and Turing-recognizable languages are closed under concatenation. QED.
Show that the decidable languages are closed under the property of Reversal, that is, if $L$ is decidable, then $L^R = \{w \mid w^r \text{ is in } L\}$ is decidable.

**Proof:**

**Given:** $L$ is decidable, so there is a TM $D$ that decides $L$.

**Want to show:** There is a TM $R$ that decides $L^R$.

**We give a high-level description of $R$:**

$R =$ “On input $w$:
Show that the CFL’s are closed under the property of Reversal, that is if \( L \) is CF, then \( L^R = \{w \mid w^r \text{ is in } L\} \) is CF.

Example: \( S \rightarrow 0 \ S \ 1 \ | \ \varepsilon \)

\( L(S) = \)

How can we get the reverse language?

•Proof: Let \( L \) be a CFL with CFG \( G = (V, \Sigma, R, S) \). We construct a CFG \( G^R = (V, \Sigma, R', S) \). for \( L^R \) as follows:

•Correctness:
• Show that the regular languages over \{0,1,2\} are closed under the operation Mirror, where

\[ \text{Mirror}(L) = \{ w \mid w \text{ is obtained from some } w' \text{ in } L \text{ by changing each 0 to 2 and 2 to 0} \} \]

• Proof: Suppose L is regular with DFA \( D = (Q, \Sigma, \delta, q_0, F) \)
Which is the best description of the language of the given PDA?

A. \{ w \mid \text{number of b's in } w \geq \text{number of a's in } w\}

B. \{ w \mid w = a^n b^{n+1} \text{ for some } n \geq 0\}

C. \{ w \mid w = a^n b^{n+2} \text{ for some } n \geq 0\}

D. \{ w \mid w = a^n b^{2n} \text{ for some } n \geq 0\}

E. \{ w \mid w = 0a^n b^{2n} 0 \text{ for some } n \geq 0\}
What is the language of this CFG?

S → aSb | bY | Ya
Y → bY | Ya | ε
Pumping Lemma Practice

- Thm. \( L = \{ww_r \mid w_r \text{ is the reverse of } w \text{ in } \{0,1\}^*\} \) is not regular.
- Proof (by contradiction):
- Assume (towards contradiction) that \( L \) is regular. Then the pumping lemma applies to \( L \). Let \( p \) be the pumping length. Choose \( s \) to be the string \( \underline{\qquad} \). The pumping lemma guarantees \( s \) can be divided into parts \( xyz \) s.t. for any \( i \geq 0 \), \( xy^iz \) is in \( L \), and that \( |y| > 0 \) and \( |xy| \leq p \). But if we let \( i = \underline{\qquad} \), we get the string \( XXXX \), which is \textit{not} in \( L \), a contradiction. Therefore the assumption is false, and \( L \) is not regular.
- Q.E.D.

A. \( s = 000000111111, \ i=6 \)
B. \( s = 0^p0^p, \ i=2 \)
C. \( s = 0^p110^p, \ i = 2 \)
D. None or more than one of the above
Running times of decider TM’s

**Deterministic**

- $f(n) = \text{MAX number of steps with input length } n$
- Transition from $q_0$ to $q_{\text{rej}}$

**Nondeterministic**

- $f(n) = \text{MAX number of steps on any branch with input length } n$
- Transition from $q_0$ to $q_{\text{rej}}$, $q_{\text{acc}}$, $q_{\text{acc}}$
P and NP

P is the class of languages that can be decided in polynomial time on a **deterministic**, single-tape TM

- Contains all realistically solvable problems
- Examples: PATH, Simple arithmetic, CFL’s,…

NP: (Equivalent to) the class of languages that can be decided in polynomial time on a **non-deterministic** single-tape TM

- \( P \subseteq NP \), but we don’t know if \( P = NP \)
- Examples include Travelling Salesperson, HAMPATH, Satisfiability of Boolean Expressions, CLIQUE
- For problems in NP not known to be in P, best deterministic algorithms take exponential time
Computational Language Hierarchy

- Regular
- Context-Free
- Decidable
- P
- NP
- Turing-Recognizable
More Problems (Study guide)

• Design NFA for $C = A \cup B$ over $\{0,1\}$, where $A = \{w \mid |w| \text{ is odd}\}$ and $B = \{w \mid w \text{ starts with } 1 \text{ and } |w| \text{ is even}\}$.

Use no more than 5 states.

Show that $L = \{x \in \{0,1\}^* \mid \text{for all } k = 0,1,...,|x|, \text{ the first } k \text{ symbols of } x \text{ contain at least as many } 0\text{'s as } 1\text{'s}\}$ is not regular.