CS 383

Exam 2 Study Guide

Stephen Checkoway

Spring 2024

Exam topics

Broadly speaking: Everything through non-context-free languages (Sipser chapter 2)

- CFGs, both the mathematical definition as a 4-tuple $G = (V, \Sigma, R, S)$ and as lists of rules
- Converting a CFG to CNF
- PDAs, both the mathematical definition $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ and diagrams
- Closure properties of CFLs

Types of exam questions

The questions from the exam fall into these types

- True/false questions with explanation
- Constructions: Construct a CFG or PDA for a language
- Proofs: Proofs about operations on languages

Exam question break down

- Five true/false questions (4 points each)
- Two constructions (20 points each)
- Two proofs (20 points each)

No pumping lemma for context-free languages questions for this exam (but possibly on the final)

Function from symbols to strings

The text of this slide is from a problem on the exam

Let Σ and Γ be alphabets and let $f : \Sigma \to \Gamma^*$ be a function that maps a symbol from Σ to a string in Γ^*. (For example, if $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ and $\Gamma = \{\mathtt{1}, \mathtt{2}\}$, then we might have $f(\mathtt{a}) = \mathtt{21}$, and $f(\mathtt{b}) = \varepsilon$, and $f(\mathtt{c}) = \mathtt{1}$.)

Extend f to operate on strings in Σ^* by $f(\varepsilon) = \varepsilon$ and $f(x_1 \cdots x_n) = f(x_1) \cdots f(x_n)$. That is, to apply f to a string $w = x_1 x_2 \cdots x_n$ where each $x_i \in \Sigma$, apply f to each of the symbols individually and concatenate the result. (Continuing the example above, $f(\mathtt{abca}) = f(\mathtt{a})f(\mathtt{b})f(\mathtt{c})f(\mathtt{a}) = \mathtt{21}\varepsilon\mathtt{121} = \mathtt{21121}$.)

Fun fact about regular languages

If A is a regular language over the alphabet Σ and $f : \Sigma \to \Gamma^*$ is a function extended to strings as described in the previous slide (i.e., $f(\varepsilon) = \varepsilon$ and $f(xy) = f(x)f(y)$), then $B = \{f(w) \mid w \in A\}$ is regular.

How would we prove that? Two approaches

- Start with a DFA for A and construct an NFA for B
- Start with a regex for A and construct a regex for B

In both cases, you're going to need to use $f(t)$ for $t \in \Sigma$

Question 1

Every NFA can be converted to an equivalent PDA

Question 1

Every NFA can be converted to an equivalent PDA
True. Do not use the stack.

Question 2

Every PDA can be converted to an equivalent NFA

Question 2

Every PDA can be converted to an equivalent NFA

False. Nonregular, context-free languages cannot be recognized by an NFA but can by a PDA.

Question 3

Every CFG can be converted to an equivalent PDA

Question 3

Every CFG can be converted to an equivalent PDA

True. We have an explicit construction.

Question 4

Every PDA can be converted to an equivalent CFG

Question 4

Every PDA can be converted to an equivalent CFG

True. Proof is in the book.

Question 5

Which of the following statements is always true about a PDA's input alphabet Σ and stack alphabet Γ?

1. $\Sigma = \Gamma$
2. $\Sigma \neq \Gamma$
3. $\Sigma \subseteq \Gamma$
4. $\Sigma \subsetneq \Gamma$ ($\Sigma \subseteq \Gamma$ but $\Sigma \neq \Gamma$)
5. $\Gamma \subseteq \Sigma$
6. $\Gamma \subsetneq \Sigma$
7. Γ always contains a symbol that's not in Σ (e.g., \$)
8. There's no inherent relationship between Σ and Γ

Question 5

Which of the following statements is always true about a PDA's input alphabet Σ and stack alphabet Γ?

1. $\Sigma = \Gamma$
2. $\Sigma \neq \Gamma$
3. $\Sigma \subseteq \Gamma$
4. $\Sigma \subsetneq \Gamma$ ($\Sigma \subseteq \Gamma$ but $\Sigma \neq \Gamma$)
5. $\Gamma \subseteq \Sigma$
6. $\Gamma \subsetneq \Sigma$
7. Γ always contains a symbol that's not in Σ (e.g., \$)
8. There's no inherent relationship between Σ and Γ

No inherent relationship between them.

Question 6

Are context-free languages are always infinite

Question 6

Are context-free languages are always infinite

No. $\varnothing$ is a context-free language generated by the (silly) CFG $S \to S$ which derives no strings.

Question 7

Are Noncontext-free languages always infinite?

Question 7

Are Noncontext-free languages always infinite?

Yes. Finite languages are regular and regular languages are context-free

Question 8

Can a PDA's stack alphabet be infinite? (I.e., can it contain infinitely many symbols?)

Question 8

Can a PDA's stack alphabet be infinite? (I.e., can it contain infinitely many symbols?)

No. Alphabets are always finite.

Question 9

If A is context-free and B is regular, then is $A \cap B$ regular?

Question 9

If A is context-free and B is regular, then is $A \cap B$ regular?
It might be, but need not be, for example if A is not regular and B is Σ^*, then $A \cap B = A$.

Question 10

If A is regular, and B is context-free, then is $\overline{A} \cup B$ context-free?

Question 10

If A is regular, and B is context-free, then is $\overline{A} \cup B$ context-free?

Yes. Regular languages are closed under complement so $\overline{A}$ is regular and thus context-free. Context-free languages are closed under union so the result is context-free.

Question 11

If A is context-free and B is regular, then is $\overline{A} \cup B$ context-free?

Question 11

If A is context-free and B is regular, then is $\overline{A} \cup B$ context-free?

It might not be as context-free languages are not closed under complement.

Question 12

What does it mean for a CFG to be ambiguous?

Question 12

What does it mean for a CFG to be ambiguous?

Some string in the language generated by the grammar has (a) at least two left-most derivations; (b) at least two right-most derivations; and (c) at least two parse trees

Question 13

What does it mean for a CFG to be unambiguous?

Question 13

What does it mean for a CFG to be unambiguous?

Every string in the language generated by the grammar has (a) exactly one left-most derivation; (b) exactly one right-most derivation; and (c) exactly one parse tree

Question 14

If G is a CFG and $w \in L(G)$ has two different derivations, is G ambiguous?

Question 14

If G is a CFG and $w \in L(G)$ has two different derivations, is G ambiguous?

Not necessarily. If the two different derivations are both left-most or both right-most, then yes. Otherwise, there's not enough information to know.

Example constructions

1. Give a CFG that generates the language
   $A = \{w \mid w \in \{\mathtt{a}, \mathtt{b}\}^*$ contains at least 3 $\mathtt{a}$s$\}$
2. Give a CFG that generates the language $B = \{\mathtt{a}^m\mathtt{b}^n \mid n > 2m\}$
3. Give a PDA that recognizes the language
   $C = \{w \mid w \in \{\mathtt{a}, \mathtt{b}\}^*$ has odd length and the middle symbol is $\mathtt{b}\}$
4. Give a PDA that recognizes language B
5. Convert the CFG for language B to a PDA using the CFG to PDA construction
6. Convert the CFG for language B to CNF

Example proofs

1. Define a multi-push PDA (mPDA) as a PDA that can push 0 or more symbols on the stack in each move. Formally, the transition function is $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \to P(Q \times \Gamma^*)$. Prove that the class of languages recognized by an mPDA is the class of context-free languages. (Show how to simulate an mPDA using a normal PDA which uses additional states for each transition that pushes more than one symbol.)