Problem Set 6

How much firepower do context-free languages have? What are their limits? And just how awesome are PDAs? In this problem set, you'll get to find out!

In any question that asks for a proof, you **must** provide a rigorous mathematical proof. You cannot draw a picture or argue by intuition. You should, at the very least, state what type of proof you are using, and (if proceeding by contradiction, contrapositive, or induction) state exactly what it is that you are trying to show. If we specify that a proof must be done a certain way, you must use that particular proof technique; otherwise you may prove the result however you wish.

As always, please feel free to drop by office hours or send us emails if you have any questions. We'd be happy to help out.

This problem set has 125 possible points. It is weighted at 7% of your total grade. The earlier questions serve as a warm-up for the later problems, so do be aware that the difficulty of the problems does increase over the course of this problem set.

Good luck, and have fun!

Due Monday, February 25th at 12:50 PM

Problem One: Designing CFGs (32 Points)

Below are a list of alphabets and languages over those alphabets. For each language, design a context-free grammar that generates that language, then show derivations for the indicated strings.

i. For the alphabet $\Sigma = \{ \texttt{a}, \texttt{b}, \texttt{c} \}$, write a CFG for the language $L = \{ w \in \Sigma^* \mid w$ contains **aa** as a substring $\}$. Show a derivation of the strings **aa**, **abaabc**, and **ccaabb** using your grammar.

ii. For the alphabet $\Sigma = \{ \texttt{a}, \texttt{b}, \texttt{c} \}$, write a CFG for the language $L = \{ w \in \Sigma^* \mid w$ does not contain both **a** and **b**. $\}$ Show a derivation of the strings ε, **a**, **caac**, and **bbb**.

iii. For the alphabet $\Sigma = \{ \texttt{a}, \texttt{b}, \texttt{c} \}$, write a CFG for $L = \{ \texttt{a}^i\texttt{b}^j\texttt{c}^k \mid i, j, k \in \mathbb{N} \wedge (i = j \vee i = k) \}$. Show a derivation of the strings **aabcc**, **aabbc**, and **abc**.

iv. For the alphabet $\Sigma = \{\texttt{a}, \texttt{b}\}$, write a CFG for the language $L = \{ w \in \Sigma^* \mid w$ is ***not*** a palindrome $\}$. That is, w is not the same when read forwards and backwards. Thus **aab** $\in L$ and **baabab** $\in L$, but **aba** $\notin L$ and **bb** $\notin L$. Show a derivation of the string **aab** and the string **abbaba**.

v. For the alphabet $\Sigma = \{\texttt{a}, \texttt{b}\}$, write a CFG for the language $L = \{ w \in \Sigma^* \mid |w| \equiv_4 0$, and the first quarter of the characters in w contains at least one **b** $\}$ For example, **baaa** $\in L$, **bbbb** $\in L$, **abbbbbba** $\in L$, **bbbaaabbbaaa** $\in L$, **ababbbbbbbbb** $\in L$, but **abbb** $\notin L$, ε $\notin L$, **b** $\notin$ L, **aabbbbaa** $\notin L$, and **aaabbbbbbbbb** $\notin L$. (For simplicity, I've underlined the first quarter of the characters in each string). Show a derivation of **baaa**, **abaaaaaa**, and **baaaaaaa**.

Problem Two: The Complexity of Addition (16 Points)

On the previous problem set, we began addressing the question

How hard is it to add two numbers?

We will now directly answer that question.

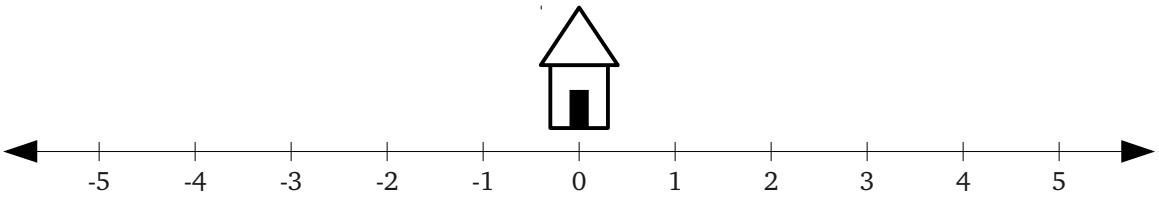
Consider the language $ADD = \{\texttt{1}^m\texttt{+}\texttt{1}^n\texttt{=}\texttt{1}^{m+n} \mid m, n \in \mathbb{N}\}$ over the alphabet $\Sigma = \{\texttt{1}, \texttt{+}, \texttt{=}\}$. That is, *ADD* consists of strings encoding two unary numbers and their sum. As you proved on the previous problem set, *ADD* is not regular.

i. Write a context-free grammar for *ADD*. Show a derivation of **1+1=11** and **+111=111** using your grammar. This proves that *ADD* is context-free. *(If you are hand-writing your grammar, please make sure your ones are easily distinguished from the vertical bars used to separate productions. Thanks!)*

ii. Design a **deterministic** PDA that recognizes *ADD* (recall that a DPDA is a PDA in which for any combination of a state, input symbol, and stack symbol there is at most one transition that can be followed, including ε-transitions). This proves that *ADD* is not only a context-free language, but also a deterministic context-free language.

Problem Three: There and Back Again (Part I) (16 Points)

Suppose that you live in a one-dimensional world with your house at position 0, as shown here:



One day, you decide to go for a walk by taking steps of size ±1 forward and backward. You begin at your house and, after completing your walk, end up back at your house.

Consider the alphabet $\Sigma = \{\texttt{L}, \texttt{R}\}$. A string in Σ^* describes a possible walk where at each step you either move a step left (**L**) or a step right (**R**). For example, the string "**LRRRLL**" means that you take a step left, then three steps right, then two steps left.

Let $L = \{ w \in \Sigma^* \mid w$ describes a series of steps in which you arrive at the same place at which you started $\}$.

i. Write a CFG that generates L. Show a derivation of **LLRR**, **RLRL**, and **LLRRRRLL** using your grammar.

ii. Design a **deterministic** PDA that recognizes L (recall that a DPDA is a PDA in which for any combination of a state, input symbol, and stack symbol there is at most one transition that can be followed, including ε-transitions). This proves that L is not only a context-free language, but also a deterministic context-free language.

Problem Four: Shrinking PDAs (16 Points)

As you saw in lecture, any CFG can be converted into a PDA with just three states, meaning that the full power of the context-free languages can be expressed by three-state PDAs. This question asks you to see just how few states PDAs can have while retaining their expressive power.

i. Show how to modify the construction from lecture that turns CFGs into PDAs so that the generated PDA has only *two* states instead of three. You should briefly describe how your construction works, but you don't need to formally prove that it is correct.

ii. Using your result from (i), prove that for any PDA P, there is a PDA P' such that $\mathscr{L}(P) = \mathscr{L}(P')$ and P' has only two states.

iii. Your result from (ii) shows that any PDA can be converted into an equivalent PDA with just two states. However, it is not always possible to convert a PDA into an equivalent PDA with just one state. Find a context-free language that does not have a one-state PDA, prove it is context-free, then prove that it does not have a one-state PDA.

Problem Five: There and Back Again (Part II) (20 Points)

Suppose that you live in a *two*-dimensional world with your house at position (0, 0). One day, you decide to go for a walk by taking steps of size ±1 up, down, left, and right. You begin at your house and, after completing your walk, end up back at your house.

Consider the alphabet $\Sigma = \{$`L`, `R`, `U`, `D`$\}$. A string in Σ^* describes a possible walk where at each step you either move a step left (`L`), a step right (`R`), a step up (`U`), or a step down (`D`). For example, the string "`ULULDDRR`" means that you take a step up, then a step left, then a step up, then a step left, then two steps down, then two steps right.

Let $L = \{ w \in \Sigma^* \mid w$ describes a series of steps in a two-dimensional grid in which you arrive at the same place at which you started $\}$. Prove that L is not context-free.

Problem Six: The Complexity of String Searching (20 Points)

On the previous problem set, we began addressing the question

How hard is it to search a string for a substring?

Given a string to search for (called the **pattern**) and a string in which the search should be conducted (called the **text**), we want to determine whether the pattern appears in the text. To encode this as a language problem, we let $\Sigma = \{$`0`, `1`, `?`$\}$ and encoded questions of the form "does pattern string p appear in text t" as the string p`?`t. For example:

"Does `0110` appear in `1110110` ?"	would be encoded as	`0110?1110110`
"Does `11` appear in `0001` ?"	would be encoded as	`11?0001`
"Does ε appear in `1100` ?"	would be encoded as	`?1100`

Let the language $SEARCH = \{ p\texttt{?}t \mid p, t \in \{\texttt{0}, \texttt{1}\}^*$ and p is a substring of $t \}$. On the last problem set, you proved that *SEARCH* is not regular using the pumping lemma for *regular* languages. Now, using the pumping lemma for context-free languages, prove that *SEARCH* is not context-free either.

As a hint, the pattern and text string you show cannot be pumped must use both `0`s and `1`s. In fact, if you restrict the pattern and text strings to strings consisting solely of `0`s, you get the language

$$LE = \{ \texttt{0}^n\texttt{?}\texttt{0}^m \mid n, m \in \mathbb{N} \wedge n \leq m \}$$

which *is* context-free.

Problem Seven: Course Feedback (5 Points)

We want this course to be as good as it can be, and we'd really appreciate your feedback on how we're doing. For a free five points, please answer the following questions. We'll give you full credit no matter what you write (as long as you write something!), but we'd appreciate it if you're honest.

i. How hard did you find this problem set? How long did it take you to finish? Does that seem unreasonably difficult or time-consuming for a five-unit class?

ii. Did you attend a recitation section? If so, did you find it useful?

iii. How is the pace of this course so far? Too slow? Too fast? Just right?

iv. Is there anything in particular we could do better? Is there anything in particular that you think we're doing well?

Submission Instructions

There are three ways to submit this assignment:

1. Hand in a physical copy of your answers at the start of class. This is probably the easiest way to submit if you are on campus.
2. Submit a physical copy of your answers in the filing cabinet in the open space near the handout hangout in the Gates building. If you haven't been there before, it's right inside the entrance labeled "Stanford Engineering Venture Fund Laboratories." There will be a clearly-labeled filing cabinet where you can submit your solutions.
3. Send an email with an electronic copy of your answers to the submission mailing list (cs103-win1213-submissions@lists.stanford.edu) with the string "[PS6]" somewhere in the subject line. If you do submit electronically, please submit your assignment as a single PDF if at all possible. Sending multiple image files makes it hard to print and grade your submission.

Extra Credit Problem: Non-Repeating Strings (5 Points Extra Credit)

Let $\Sigma = \{\mathtt{a}, \mathtt{b}\}$ and let $L = \{ wx \mid w \in \Sigma^*, x \in \Sigma^*, |w| = |x|, \text{ and } w \neq x \}$. Intuitively, L consists of all strings of even length whose first half is not equal to its second half. For example, $\mathtt{aaba} \in L$, $\mathtt{aaababbb} \in L$, $\mathtt{ab} \in L$, and $\mathtt{bbbbba} \in L$.

Write a CFG for L. Give derivations for $\mathtt{ab}$, $\mathtt{aaabaaba}$, and $\mathtt{aabaaabbab}$.