Homework 6

You may collaborate with *one or two* other students on the graded homework if you wish, or work alone. Collaboration must be true collaboration however, which means that the work put into each problem should be roughly equal and all parties should come away understanding the solution.

The ungraded homework may be done in whatever way works best for you. There are no rules regarding collaboration. The point of ungraded homework is to develop your abilities and prepare you for the quiz. Solutions will be provided, but they should be consulted only when you need a hint and/or afterward to compare and contrast your solution with mine.

Graded Homework

Completion of these tasks by 11:59pm Sunday April 10 is worth approximately 1-2% of your course grade. No late homework will be accepted.

1. I will soon place a "Homework quiz" on Canvas for you to complete. It will be untimed and you can take it as many times as you wish. You will be able to see your score after each submission, and the highest will be kept as your score.
2. I will soon place a programming problem on Mimir for you to complete.

Ungraded Homework

1. Design both a CFG and a PDA for each of the following languages over alphabet {0,1}.
    1. { w | w has at least three 1s}
    2. { w | the length of w is odd and the middle symbol is 0}
    3. { 0^m1^n | m ≠ n }
2. It is rather easy to tell if a sequence of parentheses is legal (ie, properly balanced). When looking at the parentheses from left to right, an opening parenthesis is always legal, but a closing one is only legal if more opening parentheses than closing parentheses have occurred before it. So, after `(()(` either an opening or closing parentheses is legal, but after `(()())' only an opening parentheses is legal. Also, a legal sequence of parentheses has the same number of opening and closing parentheses. Design a PDA that recognizes legal sequences of parentheses.
3. Design a context-free grammar that generates all legal sequences of parentheses.
4. Here is an unambiguous context-free grammar that handles left-to-right associativity and operator precedence. If you build a parse tree for an expression, you will see that the order of operations has earlier ones down by the leaves and later ones up by the root.

    S → E
    E → E + T | T
    T → T x F | F
    F → (E) | a

    Draw both a parse tree and a leftmost derivation for each of the following: (i) a + a x a, (ii) (a+a) x a.
5. The following context-free grammar generates the same language. Demonstrate that it is ambiguous by finding a string in its language that has two different parse trees.

    S → E
    E → E + E | E x E | (E) | a
6. Following the context-free grammar to PDA conversion process seen in class convert your grammar from Problem 3 into a PDA.
7. (Ignore until taught) Use the pumping lemma to argue that the language L = {$a^ib^jc^k$ | i ≤ j ≤ k} is not context-free.

Ungraded homework solutions

Study these only after completing the homework or after struggling with it for a while.

Ungraded solutions