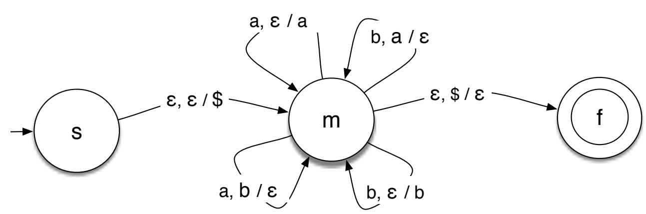
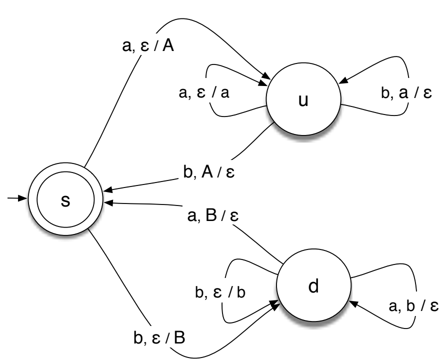
The first three questions on this assignment should be completed as group work. One copy of the solution to each problem should be submitted by noon on Wednesday or Thursday 4/15 or 4/16 depending on your group's meeting time.

The final question on this assignment should be submitted independently by each student by Tuesday 4/14. You may discuss approaches to the question with members of your working group, but the final writeup you submit should represent your own work in the same sense that was expected for all homework submissions during the first half of the semester.

1. In the lectures, we saw several examples of pushdown automata designed to recognize the language of strings containing equal numbers of a's and b's. At one extreme, we considered a machine that exploited nondeterminism extensively:



At the other extreme we saw an example of a deterministic machine for the same language.



(The diagrams presented here are identical to those seen in the lecture notes and slides except the nodes are now labeled with single letter names to simplify the presentation of your answer to the questions below.)

In the text, as part of a proof that any language recognized by a pushdown automaton must be context-free, Sipser presents an algorithm to produce a context-free grammar for the language of a given pushdown automaton (Lemma 2.27 starting on page 121). The grammar produced by this algorithm has non-terminal symbols of the form A_{pq} for each pair of states p and q in the PDA being considered. For example, for the first machine with states s, m and f, the non-terminals would be $A_{ss}, A_{sm}, A_{sf}, A_{ms}, A_{mm}, A_{mf}, A_{fs}, A_{fm}$ and A_{ff}.

At the top of page 122 in the text, Sipser provides three descriptions of subsets that together form all of the productions/rules that should be included in the grammar formed for a given PDA.

(a) Show all the rules required by item 1 of Sipser's list of rule types for the first of the two machines shown above (the machine with states s, f and m). These would be all the rules formed using the template $A_{pq} \rightarrow aA_{rs}b$.

(b) Show all the rules required by item 1 of Sipser's list of rule types for the second of the two machines shown above (the machine with states s, u and d).

(c) Since there are three states in each of these machines, the third collection of rules Sipser describes would include $3^3 = 27$ rules. That is a lot of rules! Most of them are unnecessary! A few of them are essential.

To see how they are essential, consider the grammar formed from the first PDA by combining the rules you included in part (a) of your answer with the rules described in Sipser's second step ($A_{ss} \rightarrow \epsilon, A_{mm} \rightarrow \epsilon$ and $A_{ff} \rightarrow \epsilon$) while not including any of the rules of the third type. Show an example of a string that belongs to the language of the PDA that could not be derived using the grammar if only these rules were included. Justify the claim that the string you identified cannot be derived using the grammar.

(d) Now, to appreciate how many of the rules of Sipser's third type are not essential, consider the second PDA. You already know that a grammar for this machine's language can be formed by combining the rules you listed in your answer to part (b), the rules added by Sipser's second step ($A_{ss} \rightarrow \epsilon, A_{uu} \rightarrow \epsilon$ and $A_{dd} \rightarrow \epsilon$) and the 27 rules described in Sipser's third set. Identify as small a subset as possible of the 27 third step rules that will correctly complete the grammar. Briefly justify the claim that these rules are sufficient.

2. We have seen that context-free languages are closed under union and intersection but not under complement. We have also observed that the subset

of the set of context-free languages recognized by deterministic pushdown automata is closed under complement since we can simply interchange the final and non-final states of a deterministic machine to obtain a machine that recognizes the complement of the original machine's language. For this question, I would like you to consider whether the set of languages accepted by deterministic pushdown automata is closed under union and intersection. We have not covered most of the material in the text related to such languages. I don't expect (or want) you to try to cover this material on your own to answer this question. Instead, your argument should be based on the one fact about deterministic context-free languages stated above (that they are closed under complement) and general properties of context-free languages in general.

3. Given an alphabet Σ and a string $w \in \Sigma^*$ and a symbol $x \in \Sigma$, let $\text{occurs}(x, w)$ equal the number of times the symbol x appears in w. We can define an infinite collection of languages

$$L_{uni-\Sigma} = \{w \mid \text{ for all } x, y \in \Sigma, \text{occurs}(x, w) = \text{occurs}(y, w)\}$$

That is $L_{uni-\Sigma}$ is the set of strings over a particular alphabet Σ in which all of the symbols in the alphabet occur equally often.

Consider the special case $L_{uni-\{a,b,c\}}$ of strings containing equals numbers of a's, b's and c's. This language is a simple example of a language that is known not be be context-free. Using JFlap (http://jflap.org), design a Turing machine that decides this language. Submit a PDF including both a copy of the machine's state diagram produced using JFlap and a brief, informal explanation of how the machine works.

4. Let $b(n)$ denote the binary representation of n without leading 0s (e.g., $b(10) = 1010$). Consider the language $A = \{b(n)\#b(n+1) \mid n \geq 1\}$.

   (a) Show that this language is not context-free using the pumping lemma. That is, show that for any value of p there exists a string w of length p or greater that cannot be pumped.

   (b) Just for the fun of it, also show that for any sufficiently large value of p there exists a string w of length p or greater that can be pumped.