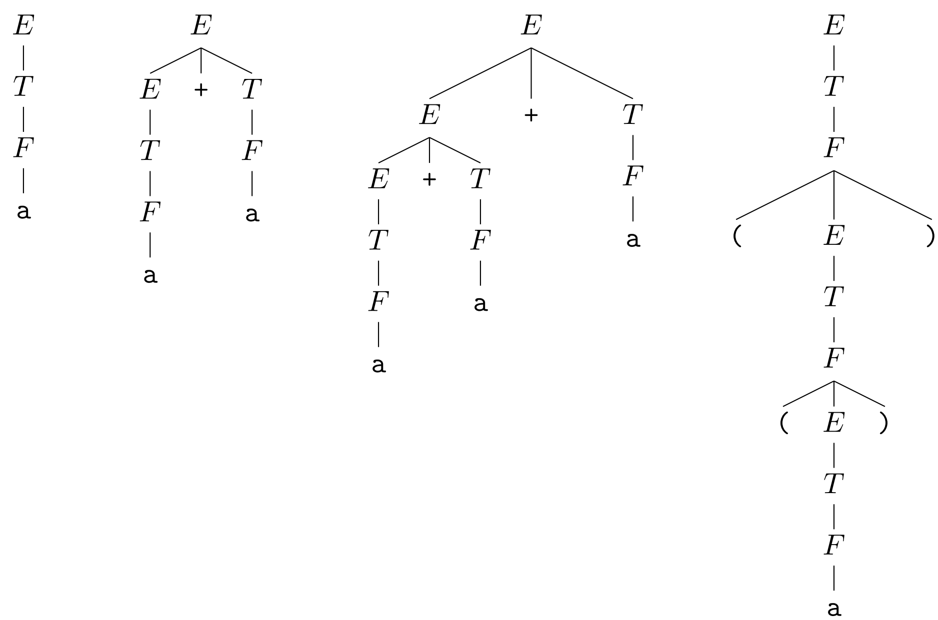
# SOLUTIONS TO HOMEWORK 6

## Sipser 2.1

The derivation trees for parts (a)–(d) are as follows.



## Sipser 2.6(d)

$$S \to S' \mid X\#S' \mid S'\#X \mid X\#S'\#X$$
$$S' \to aS'a \mid bS'b \mid a \mid b \mid \varepsilon \mid \# \mid \#X\#$$
$$X \to aX \mid bX \mid \#X \mid \varepsilon$$

Observe that this grammar allows $i = j$, which amounts to saying that some $x_i$ is a palindrome.

## Sipser 2.19

The grammar generates precisely those strings that are *not* of the form $a^n b^n$ for any $n \geq 0$. The complementary language is therefore $\{a^n b^n : n \geq 0\}$, with grammar $S \to aSb \mid \varepsilon$.

## Sipser 2.27

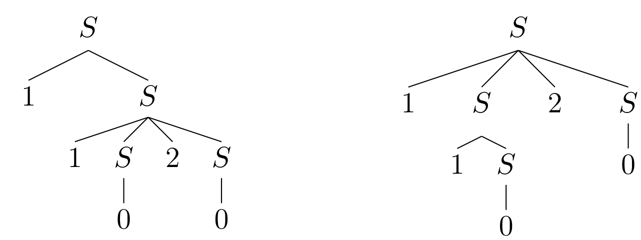
The grammar in the problem statement is quite cluttered. To clean things up, we adopt the following abbreviations:

| | | |
|---|---|---|
| 0 | stands for | `a:=1` |
| 1 | stands for | `if condition then` |
| 2 | stands for | `else` |

With these changes, the grammar becomes *vastly* easier to read:

$$S \to 0 \mid 1S \mid 1S2S.$$

This grammar is ambiguous because 11020 has at least two parse trees:



An equivalent unambiguous grammar is

$$S \to 0 \mid 1S \mid 1T2S$$
$$T \to 0 \mid 1T2T$$

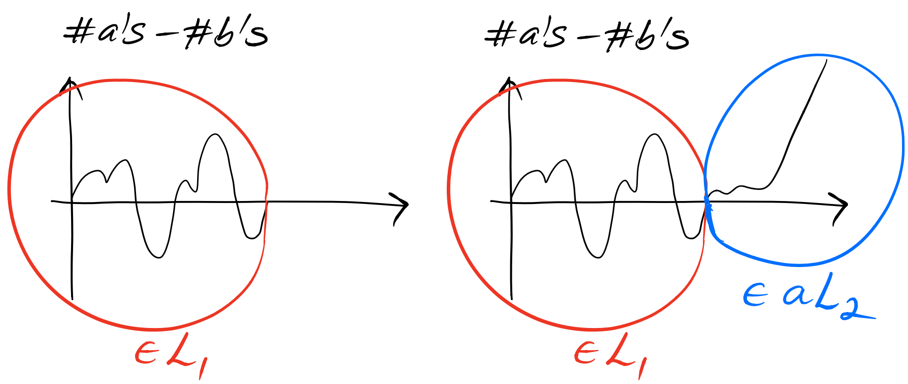
## Sipser 2.28(c)

Consider the following languages:

$$L_1 = \{w : w \text{ contains equally many } a\text{'s and } b\text{'s}\},$$
$$L_2 = \{w : \text{every prefix of } w \text{ contains at least as many } a\text{'s as } b\text{'s}\}.$$

In class, we constructed unambiguous grammars for $L_1$ and $L_2$. Let $S_1$ and $S_2$ denote their corresponding start variables.

Now consider the language $L$ of strings with at least as many $a$'s as $b$'s. Every $w \in L$ can be written in a unique way as $w = uv$, where $u \in L_1$ and $v \in \varepsilon \cup aL_2$. This is easiest to see by studying the graphs below:




Therefore, $L$ is computed by the unambiguous grammar $S \to S_1 \mid S_1 a S_2$.

## Problem 6

Let $L \subseteq \{a,b\}^*$ denote the language of strings with equally many $a$'s and $b$'s. To prove that the grammar generates every string in $L$, we argue by induction on string length $n$. The base case $n = 0$ corresponds to the empty string $\varepsilon$, which the grammar clearly generates.

For the inductive step, fix $n \geq 1$ be arbitrarily and assume that the grammar generates all strings in $L$ of length less than $n$. Fix an arbitrary string $w \in L$ of length $n$. Consider the *smallest* positive integer $i$ such that the prefix $w_1 w_2 \dots w_i$ contains as many $a$'s as $b$'s (at least one such integer exists, namely, $n$). Then the remainder of the string also contains as many $a$'s as $b$'s:

$$w_{i+1} \dots w_n \in L.$$

The key observation is that

$$w_1 \neq w_i.$$

Indeed, if $w_1 = a$, then the minimality of $i$ implies that the prefixes $w_1$, $w_1 w_2$, $w_1 w_2 \dots w_{i-1}$ all contain more $a$'s than $b$'s, which in turn forces $w_i = b$. The argument for $w_1 = b$ is analogous. In conclusion,

$$w = a \underbrace{w_2 \dots w_{i-1}}_{\in L} b \underbrace{w_{i+1} \dots w_n}_{\in L} \qquad \text{or} \qquad w = b \underbrace{w_2 \dots w_{i-1}}_{\in L} a \underbrace{w_{i+1} \dots w_n}_{\in L}.$$

The substrings $w_2 \dots w_{i-1}$ and $w_{i+1} \dots w_n$ have length less than $n$ and, by the inductive hypothesis, are generated by the grammar. Thus, $w$ itself can be generated, by starting with the rule $S \to aSbS$ or $S \to bSaS$ and then expanding each $S$ on the right-hand side into the corresponding substring.

## Problem 7

Fix an arbitrary regular language $L$, and let $(Q, \Sigma, \delta, q_0, F)$ be a DFA for $L$. To obtain a context-free grammar for $L$, we create a variable $V_q$ for each state $q \in Q$. The start variable will be $V_{q_0}$. Our grammar uses the alphabet $\Sigma$ and the following substitution rules:

$$\begin{aligned} &V_q \to \sigma V_{\delta(q,\sigma)} && \text{for all } q \in Q \text{ and } \sigma \in \Sigma, \\ &V_q \to \varepsilon && \text{for all } q \in F. \end{aligned}$$