CS 81 Assignment 10 Solutions for Mon., April 25

Hein p 52-54:

8.b L = {Λ, abb, b}, M = {bba, ab, a} = {a, ab, bba},
so ML = {a, ab, bba}{Λ} ∪ {a, ab, bba}{abb}∪ {a, ab, bba}{b}
= {a, ab, bba} ∪ {aabb, ababb, bbaabb } ∪ {ab, abb, bbab}
= {a, aabb, ab, ababb, abb, bba, bbaabb, bbab}

8.e L = {Λ, abb, b}
so L^2 = {Λ, abb, b}{Λ} ∪ {Λ, abb, b}{abb} ∪ {Λ, abb, b}{b}
= {Λ, abb, b} ∪ {abb, abbabb, babb } ∪ {b, abbb, bb}
= {Λ, abb, abbb, abbabb, b, babb, bb}

11.b {a, b}* - {b}* = The set of strings of a's and b's that include at least one a.

17.d Note that the author uses ⊂ to mean what we would notate as ⊆.
We want to show L(M∩N) ⊆ LM∩LN and (M∩N)L ⊆ ML∩NL.
Proof that L(M∩N) ⊆ LM∩LN:
We must show that any element of L(M∩N) is also an element of LM∩LN.
An element of L(M∩N) has the form xy where x ∈ L and y ∈ (M∩N).
Thus y ∈ M and y ∈ N.
Thus xy ∈ LM and xy ∈ LN.
Thus xy ∈ LM ∩ LN.

Proof that (M∩N)L ⊆ ML∩NL:
We must show that any element of (M∩N)L is also an element of ML∩NL.
An element of (M∩N)L has the form yx where x ∈ L and y ∈ (M∩N).
Thus y ∈ M and y ∈ N.
Thus yx ∈ ML and yx ∈ NL.
Thus yx ∈ ML ∩ NL.

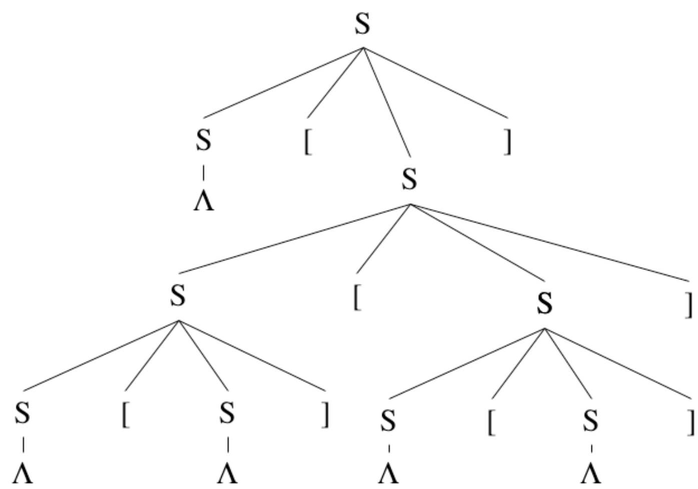
Hein p 188-190:

2.d S → S [S] | Λ   Derive [[ ] [[ ]]]:

Leftmost derivation: S ⇒ S [S] ⇒ Λ [S] ⇒ Λ [S [S]]
⇒ Λ [S [S] [S]] ⇒ Λ [Λ [S] [S]] ⇒ Λ [Λ [Λ] [S]]
⇒ Λ [Λ [Λ] [S [S]]] ⇒ Λ [Λ [Λ] [Λ [S]]] ⇒ Λ [Λ [Λ] [Λ [Λ]]] =
[[ ] [[ ]]] .

Rightmost derivation: S ⇒ S [S] ⇒ S [S [S]] ⇒ S [S [S [S]]]
⇒ S [S [S [Λ]]] ⇒ S [S [Λ [Λ]]] ⇒ S [S [S] [Λ [Λ]]]
⇒ S [S [Λ] [Λ [Λ]]] ⇒ S [Λ [Λ] [Λ [Λ]]] ⇒ Λ [Λ [Λ] [Λ [Λ]]] =
[[ ] [[ ]]]

Parse tree:



5.b Find a grammar for the odd palindromes over {a, b, c}:

S is the start symbol

$S \rightarrow aSa \mid bSb \mid cSc \mid a \mid b \mid c$

12 Find a grammar for the language of all strings over {a, b} that have the same number of a's and b's.

It is asserted that the grammar below, with start symbol S, has the desired property:

$S \rightarrow \Lambda \mid aB \mid bA$
$A \rightarrow aS \mid bAA$
$B \rightarrow bS \mid aBB$

The idea is that any non-empty string produced must start with an a or b. The auxiliary B represents a deficit of one b that must ultimately produced, while A represents a deficit of one a. At any point in a leftmost derivation, a deficit can be resolved by applying $A \rightarrow aS$ or $B \rightarrow bS$, or it can be maintained by applying $A \rightarrow bAA$ or $B \rightarrow aBB$.

To prove that the grammar is correct, we must show two things:

I. Every string generated has the desired form.

II. Every string of the desired form is generated.

Notation: For any string x and symbol σ, define

$\#_\sigma(x)$ = the number of σ's in x

$d(x) = \#_a(x) + \#_A(x) - \#_b(x) - \#_B(x)$

Proof of I: We claim the following:

$\forall x \in \{a, b, A, B, S\}^*\ S \Rightarrow^* x$ implies $d(x) = 0$.

Consequently, for $x \in \{a, b\}^*$, we have the desired property. The claim is proved by induction on the length of the derivation.

Basis: The derivation is length 0, i.e. $S \Rightarrow^* S$. Then clearly $d(S) = 0$.

Induction step: Suppose $S \Rightarrow^* u\alpha w \Rightarrow uvw = x$ is a length n+1 derivation, where $\alpha \in \{S, A, B\}$. By the induction hypothesis, $d(u\alpha w) = 0$.
If $\alpha = S$, then $v \in \{\Lambda, aB, bA\}$. So in each case, $d(\alpha) = d(v)$, and thus $d(uvw) = d(u\alpha w) = 0$, because S, aB, and bA each contribute nothing to d.
If $\alpha = A$, then $v \in \{aS, bAA\}$. Again $d(uvw) = d(u\alpha w) = 0$, because the A is balanced with either an a, or bAA.
If $\alpha = B$, then $v \in \{bS, aBB\}$. Similarly $d(uvw) = d(u\alpha w) = 0$.
Therefore $d(uvw) = 0$ regardless of which production is applied.

Proof of II: We need to show:

$\forall x \in \{a, b\}^*\ d(x) = 0$ implies $S \Rightarrow^* x$.

In order to carry this out, we again need to show something more general, from which the above statement follows directly:

$\forall x \in \{a, b\}^*\ P(x)$ where

P(x) is ($d(x) = 0$ implies $S \Rightarrow^* x$
$\wedge\, d(x) = 1$ implies $A \Rightarrow^* x$
$\wedge\, d(x) = -1$ implies $B \Rightarrow^* x$)

This is proved is by strong induction of the length of x.

Basis: If x is of length 0, then $x = \Lambda$, and clearly $S \Rightarrow^* \Lambda$ by the production $S \rightarrow \Lambda$. In this case $d(x) = 0$ and thus p(x).

Induction step: Suppose that $x \in \{a, b\}^*$ and for all y shorter than x, P(y).

The case where $d(x) = 0$:

If x begins with a, then $x = ay$, where $d(y) = -1$. From the induction hypothesis, we have $B \Rightarrow^* y$, so from the productions, $S \Rightarrow aB \Rightarrow^* ay = x$.

Similarly, if x begins with b, then $x = by$, where $d(y) = 1$. From the induction hypothesis, $A \Rightarrow^* y$, so from the productions, $S \Rightarrow bA \Rightarrow^* by = x$.

The case where $d(x) = 1$:

If x begins with a, then $x = ay$, where $d(y) = 0$. From the induction hypothesis, we have $S \Rightarrow^* y$, so from the productions, $A \Rightarrow aS \Rightarrow^* ay = x$.

If x begins with b, then $x = byz$, where $d(y) = d(z) = 1$. From the induction hypothesis, we have $A \Rightarrow^* y$, and $A \Rightarrow^* z$, so from the productions, $A \Rightarrow bAA \Rightarrow^* byz = x$.

The case where $d(x) = -1$:

If x begins with b, then $x = by$, where $d(y) = 0$. From the induction hypothesis, we have $S \Rightarrow^* y$, so from the productions, $B \Rightarrow bS \Rightarrow^* by = x$.

If x begins with a, then $x = ayz$, where $d(y) = d(z) = -1$. From the induction hypothesis, we have $B \Rightarrow^* y$, and $B \Rightarrow^* z$, so from the productions, $B \Rightarrow aBB \Rightarrow^* ayz = x$.

14.b Find a grammar equivalent to the following, with no occurrence of Λ on the right-hand side of any rule:

$S \rightarrow AcAB$
$A \rightarrow aA \mid \Lambda$
$B \rightarrow bB \mid b$

Auxiliary A produces exactly the elements of $\{a\}^*$.
We can replace the A production with

$A \rightarrow aA \mid a$

provided that we make up for the loss of Λ in productions with A on the right-hand side. Thus we change $S \rightarrow AcAB$ to

$S \rightarrow AcAB \mid cAB \mid AcB \mid cB$

The resulting grammar is:

S → AcAB | cAB | AcB | cB
A → aA | a
B → bB | b

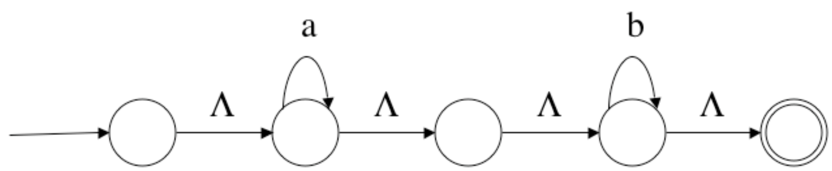
Hein p 640-642:

2.h Find a regular expression to describe $\{a^{2k} \mid k \in N\} \cup \{b^{2k+1} \mid k \in N\}$:

(aa)* + (bb)*b

Hein p 664

6.b Use 11.4 to construct an NFA for a*b*



6.d Use 11.4 to construct an NFA for a* + b*.