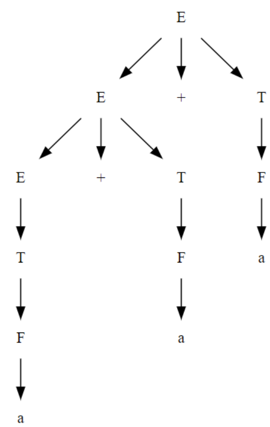
Anthony Morrell
CSIS 616

Problem (a)

(a)

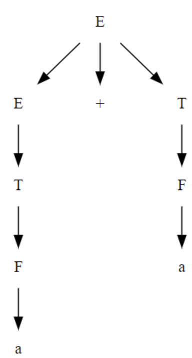
$E => T$
$=> F$
$=> a$

(c)



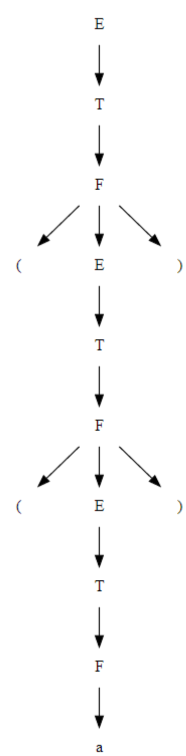
$E => E + T$
$=> E + T + T$
$=> T + T + T$
$=> F + T + T$
$=> a + T + T$
$=> a + F + T$
$=> a + a + T$
$=> a + a + F$
$=> a + a + a$

(b)



$E => E + T$
$=> T + T$
$=> F + T$
$=> a + T$
$=> a + F$
$=> a + a$

(d)



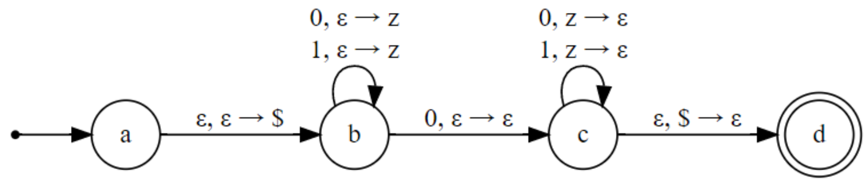
$E => T => F$
$=> (E)$
$=> (T)$
$=> (F)$
$=> \big((E)\big)$
$=> \big((T)\big)$
$=> \big((F)\big)$
$=> ((a))$

Problem (b)

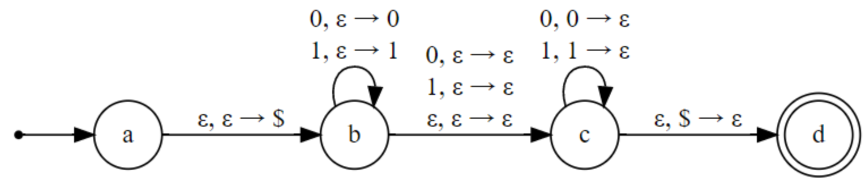
(d)	(e)	(f)
$E \to ABA$ $A \to 0 \mid 1$ $B \to E \mid 0$	$E \to 0E0 \mid 1E1 \mid 0 \mid 1 \mid \epsilon$	(no productions)

Problem (c)

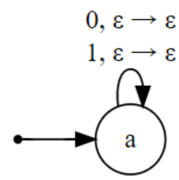
(d) This PDA begins by pushing a \$ onto the stack. For every 0 or 1 it reads, it pushes a z onto the stack. Halfway through the input (decided non-deterministically), it reads the 0 at the center of the word, and then begins popping off one z for every remaining letter. The word is accepted if there are no remaining letters in the input and the stack contains only the initial \$.



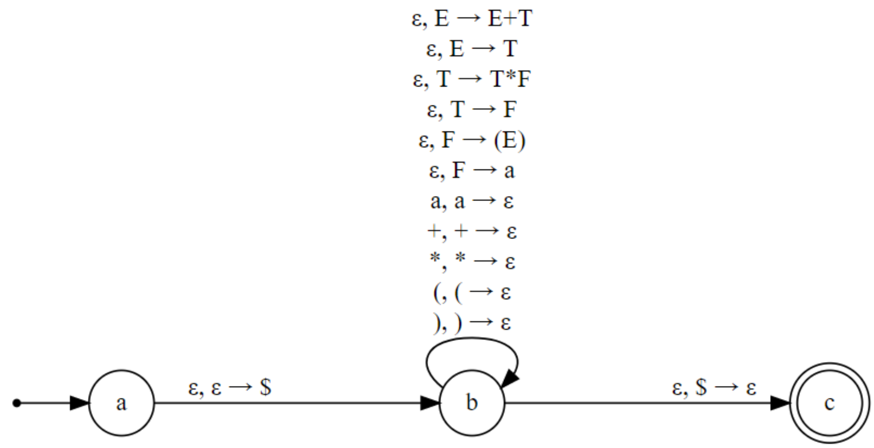
(e) This PDA begins by pushing a \$ onto the stack. For every 0 or 1 it reads, it pushes that letter onto the stack. Halfway through the input (decided non-deterministically), it reads the middle letter, if any, and then begins popping letters off the stack if they match each letter in the word. The word is accepted if there are no remaining letters in the input and the stack contains only the initial \$.



(f) This PDA reads any number of 0's and 1's, and does not utilize its stack. It has no accepting states.



Problem (d)



This state transition diagram uses Sipster's shorthand notation for pushing strings (instead of single letters) onto the stack (p. 119).

Problem (e)

(a) $L(G)$ is the language consisting of all strings of 2 or more 0's containing 1 or 2 pound signs. If a word contains a single pound sign, it is 1/3 of the way into the word. If a word contains two pound signs, they can be anywhere.

(b) Assume that $L(G)$ is regular. Let $s = 0^p\#0^{2p}$, where p is the pumping length given by the pumping lemma. Because $s \in L(G)$ and $|s| > p$, the pumping lemma guarantees that s can be split into three parts, $s = xyz$, where:

(1) $xy^iz \in L(G)\ \forall\ i \geq 0$,

(2) $|y| > 0$, and

(3) $|xy| \leq p$.

Case 1: *y* contains the #. Then *xyyyz* would contain 3 pound signs, breaking condition (1).

Case 2: *y* does not contain the #, meaning it contains only 0's on the left or on the right of the pound sign. Then *xyyz* would not contain the correct ratio of 0's on the left and right, breaking condition (1).

A contradiction is unavoidable, so $L(G)$ must not be regular.

Problem (f)

Original grammar:

$A \to BAB \mid B \mid \epsilon$

$B \to 00 \mid \epsilon$

1. Add a new start variable:

$\boldsymbol{S \to A}$

$A \to BAB \mid B \mid \epsilon$

$B \to 00 \mid \epsilon$

2. Remove ϵ-rules:

$S \to A \mid \boldsymbol{\epsilon}$

$A \to BAB \mid B \mid \boldsymbol{AB} \mid \boldsymbol{BA} \mid \boldsymbol{BB}$

$B \to 00$

3. Remove unit rules:

$S \to \boldsymbol{BAB} \mid \mathbf{00} \mid \boldsymbol{AB} \mid \boldsymbol{BA} \mid \boldsymbol{BB} \mid \epsilon$

$A \to BAB \mid \mathbf{00} \mid AB \mid BA \mid BB$

$B \to 00$

4. Convert remaining rules:

$S \to \boldsymbol{BC} \mid \boldsymbol{UU} \mid AB \mid BA \mid BB \mid \epsilon$

$A \to \boldsymbol{BC} \mid \boldsymbol{UU} \mid AB \mid BA \mid BB$

$B \to \boldsymbol{UU}$

$\boldsymbol{C} \to \boldsymbol{AB}$

$\boldsymbol{U} \to \mathbf{0}$