2.14

1st step: add start node

$S_0 \rightarrow A$

$A \rightarrow BAB \mid B \mid \varepsilon$

$B \rightarrow 00 \mid \varepsilon$

2nd step: remove ε in B

$S_0 \rightarrow A$

$A \rightarrow BAB \mid BA \mid AB \mid A \mid \beta \mid \varepsilon$

$B \rightarrow 00$

3rd step: remove ε in A

$S_0 \rightarrow A \mid \varepsilon$

$A \rightarrow BAB \mid BA \mid AB \mid A \mid \beta \mid BB$

$B \rightarrow 00$

4st step : remove unit rule :

$S_0 \rightarrow A \mid \varepsilon$

$A \rightarrow BAB \mid BA \mid AB \mid 00 \mid BB$

$B \rightarrow 00$

$S_0 \rightarrow BAB \mid BA \mid AB \mid 00 \mid BB \mid \varepsilon$

$A \rightarrow BAB \mid BA \mid AB \mid 00 \mid BB$

$B \rightarrow 00$

5st step : replace 0s

Let $U = 0$

$S_0 \rightarrow BAB \mid BA \mid AB \mid UU \mid BB \mid \varepsilon$

$A \rightarrow BAB \mid BA \mid AB \mid 00 \mid BB$

$B \rightarrow UU$

6 st step : Shorten the grammar

Let $A_1 = AB$

$S_0 \rightarrow BA_1 | BA | AB | UU | BB | \varepsilon$

$A \rightarrow BA_1 | BA | AB | 00 | BB$

$B \rightarrow UU$

$U \rightarrow 0$

$A_1 \rightarrow AB$

2.16

Union:

To Prove $S \rightarrow S_1 \cup S_2 \in CFL$

Let L_1 generated by S_1

L_2 generated by S_2

$L_1 \in CFL$ and $L_2 \subset CFL$

By definition of union. if both $L_1, L_2 \in CFL$

$L_1 \cup L_2 \in CFL$

Concatenation

To Prove $S \rightarrow S_1 S_2 \in CFL$

Let L_1 generated by S_1

L_2 generated by S_2

$L_1 \in CFL$ and $L_2 \subset CFL$

$\{w_1 w_2 : w_1 \in L_1 \wedge w_2 \in L_2\} \in CFL$

So $S_1 S_2 \in CFL$

Star:

To prove S_1 is closed under star.

$S^* \rightarrow S_1 S \mid \varepsilon$

Because $S_1 \in CFL$ and $\varepsilon \in CFL$

there will only be S_1 and ε in S^*

So $S^* \subset CFL$

3.6

The defect of this forward direction proof is on the stage 2. If the M gets in a loop on an input, E will not be able to check any input after.

3.7

For step 1, to try all possible settings of $X_1 \cdots X_k$ will require an infinite memory.

For step 2, since we have infinite number of instances X_i, It take infinite time to process P on these values.

Therefore, step 3 will not be executed.

4.2

let D be DFS and R be regular expressions that $L(D) = L(R)$

Convert R to a DFA D_R with Kleene's Theorem.

Put (D, D_R) into a turing machine decider

If F is true, accept the language

otherwise, reject the language

4.15

The first step is to check w satisfy L(R) and L(S), if not, reject w.

Then Convert R into DFA D_R and convert S into DFA D_S that $L(R) = L(D_R)$ and $L(S) = L(D_S)$

Run Turing machine T with D_R, D_S to find F value, if F accepts, accept the language if F rejects, reject the language.