1.

You should first review what an instantaneous description is for Turing Machines (see 8.2.3, page 320 in the text).

Let reverse(ID) denote the reverse of an instantaneous description (same idea as reverse(x) for x a string). Then we define VALCOMPS(M,w) to be the set of strings of the form

$$ID_1 \rightarrow reverse(ID_2) \rightarrow \ldots \rightarrow ID_{n-1} \rightarrow reverse(ID_n)$$

where the string defines an accepting sequence of instantaneous descriptions of M on input w.

The key idea is that we can generate a grammar that produces $ID_i \rightarrow reverse(ID_{i+1})$, ie we can take a Turing Machine M and produce from its transition function a grammar that maintains a correct pair-wise relationship between adjacent IDs.

Once we have this, then we generate one grammar that makes sure the $ID_i \rightarrow reverse(ID_{i+1})$ transitions are correct, and another grammar that makes sure $ID_1$ is correct ($ID_1$ is simply $q_0w$), $ID_n$ is correct, and reverse($ID_i$) $\rightarrow$ $ID_{i+1}$ are correct. When we intersect these two languages, we get a cascading effect that produces exactly VALCOMPS(M,w).

First consider a grammar that generates $ID_i \rightarrow ID_{i+1}$.

$$A \rightarrow 0A0 \mid 1A1 \mid \epsilon$$

for $a, b, c \in \{0, 1\}$, $q \in Q$

- if $\delta(q,a) = (b, p, R)$ then $A \rightarrow qaApb \mid cqaAbcp$
- if $\delta(q,a) = (b, p, L)$ then $A \rightarrow cqaAbcp$
- if $\delta(q,\text{blank}) = (b, p, R)$ then $A \rightarrow q \rightarrow pb$
- if $\delta(q,\text{blank}) = (b, p, L)$ then $A \rightarrow cq \rightarrow bcp$

Then $G_1 = \{S, A\}, \{0, 1, \rightarrow\}, S, \{S \rightarrow S \leftarrow S \mid A\} + \text{ productions for } A$

Now let’s produce the second grammar described above.

$$G_2 = \{T, U, F, E\}, \{0, 1, \rightarrow\}, T, P$$ where P contains the productions

- $T \rightarrow q_0 x \rightarrow U \leftarrow F$
- $U \rightarrow U \leftarrow U \mid A$
- $F \rightarrow 0F \mid 1F \mid q_f \mid E$
- $E \rightarrow 0E \mid 1E \mid \epsilon$