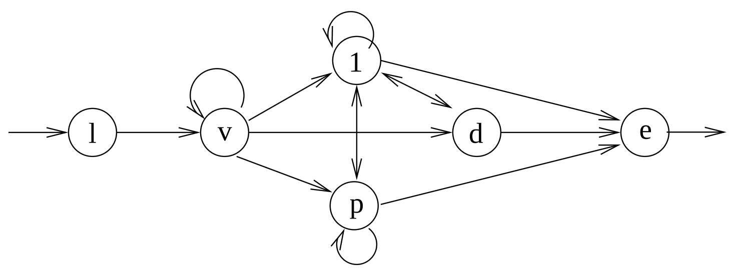
Tik-61.231 Principles of Pattern Recognition

Answers to exercise 8: 15.11.2000

1. A grammar $G = (V_T, V_N, P, S)$ consists of the following entities: V_T, a set of terminal symbols
   V_N, a set of nonterminal symbols
   P, a set of production rules
   S, a set of starting symbols

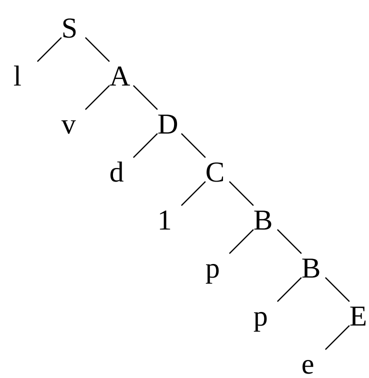
   Lets use the notations l (locomotive), v (luggage van), p (passenger coach), 1 (first class coach), d (dining car, interpreted as a special case of a passenger coach) and e (the end of the train, an empty symbol). Now we can draw a syntax diagram for the acceptable train structures: The BNF (Backus-

   Naur-Form or Backus normal form) uses the symbols := as $\rightarrow$, | for 'or' and $< \cdots >$ for nonterminal symbols. The following grammar describes all possible trains:

$$V_T = \{l, v, p, 1, d, e\}$$
$$V_N = \{S, A, B, C, D, E\}$$
$$P = \left\{ \begin{array}{l} <S> := l <A> \\ <A> := v(<A> \mid <B> \mid <C> \mid <D>) \\ <B> := p(<B> \mid <C> \mid <E>) \\ <C> := 1(<B> \mid <C> \mid <D> \mid <E>) \\ <D> := d(<C> \mid <D> \mid <E>) \\ <E> := e \end{array} \right\}$$

   Now the parse tree for the example train $\{l, v, d, 1, p, p\}$ is

2. Now to check if the grammar accepts a train of the form $\{l, v, v, d, p, 1\}$. From the syntax diagram in the previous question we can instantly see, that the symbol d cannot be followed by p. Thus the grammar does not accept the train.

3. The grammar is

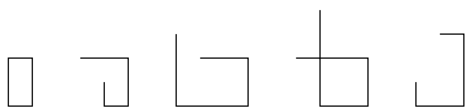
$$
\begin{aligned}
&V_T = \{o, a, \neg, +\} \\
&V_N = \{Square, Side1, Side2, Side3, Side4\} \\
&P = \left\{ \begin{array}{l}
Square \to Side1 + Side2 + Side3 + Side4 \\
\quad Side1 \to o | Side1 + o \\
\quad Side2 \to a | Side2 + a \\
\quad Side3 \to \neg o | Side3 + \neg o \\
\quad Side4 \to \neg a | Side4 + \neg a
\end{array} \right\} \\
&S = \{Square\}
\end{aligned}
$$

$$\Rightarrow L = \{(o|o^{n_1}) + (a|a^{n_2}) + (\neg o|(\neg o)^{n_3} + (\neg a|(\neg a)^{n_4})\}$$

Where $n_1, n_2, n_3, n_4 \geq 1$, $|$ stands for 'or' and $a^n = a + a + a + \cdots + a$ (n a's).

a) Yes, the grammar produces squares if $n_1 = n_2 = n_3 = n_4$.

b) The grammar can produce several other structures, since the only constrained factor is the order of the turns; for example (when $n_1 = n_2 = n_3 = n_4$ doesn't hold)



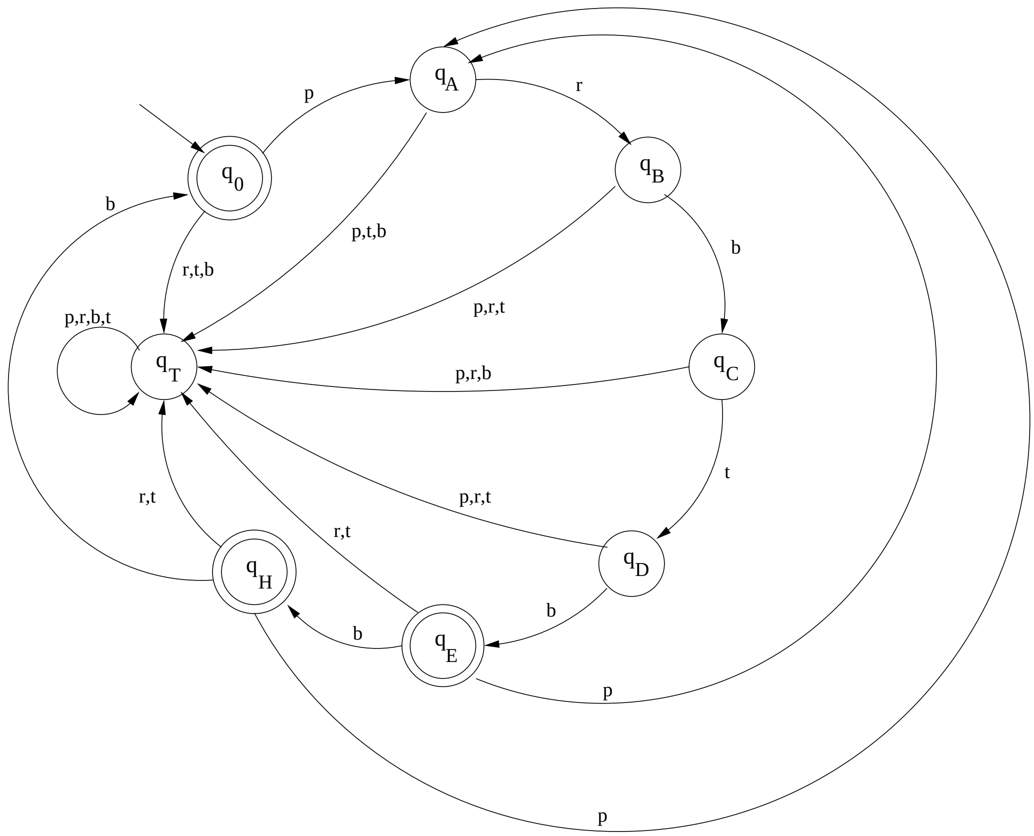
c) The grammar can be made to produce only squares by constraining the length of each vertice to be equal,

$$L = \{o^n + a^n + (\neg o)^n + (\neg a)^n | n \geq 1\}$$

Thus the grammar is

$$
\begin{aligned}
&V_T = \{o, a, \neg, +\} \\
&V_N = \{N, A, V, Y\} \\
&P = \left\{ \begin{array}{l}
\quad N \to o + N + A + V + Y | o + A + V + Y \\
Y + A \to A + Y \text{ (correct order)} \\
V + A \to A + V \text{ (correct order)} \\
Y + V \to V + Y \text{ (correct order)} \\
o + A \to o + a \text{ (upper right corner)} \\
a + V \to a + \neg o \text{ (lower right corner)} \\
\neg o + Y \to \neg o + \neg a \text{ (lower left corner)} \\
a + A \to a + a \\
\neg o + V \to \neg o + \neg o \\
\neg a + Y \to \neg a + \neg a
\end{array} \right\} \\
&S = \{N\}
\end{aligned}
$$

4. a) The deterministic automaton thus becomes



b) With the symbol sequence *prbtbprbtbbb*... the resulting state sequence is

$$q_0 \rightarrow q_A \rightarrow q_C \rightarrow q_D \rightarrow q_E \rightarrow q_A \rightarrow q_B \rightarrow q_C \rightarrow q_D \rightarrow q_E \rightarrow q_H \rightarrow q_0$$

Now when the sequence starts again, the same route is traversed. As alarms are a result of the trap state q_T, which is not entered, no alarm is produced with this sequence.