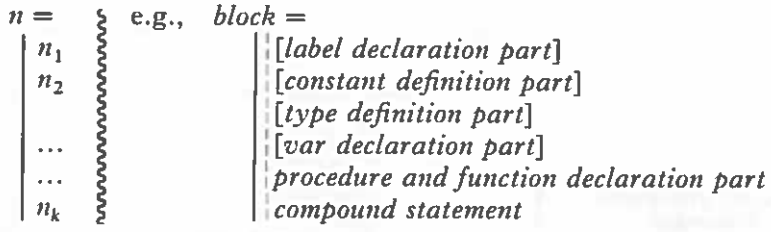

where all the n_j are right next to the references vertical and have no $\backslash n$ character appearing to its left.

4.1. The predicate properly-laid-out

For example, we say that a string named *rptst* produced by the non-terminal repeat statement is properly laid out at m if (1) the reserved word *repeat* is the first word on that line starting at a margin of m, (2) the statements of the loop body obey the rules of Figure 2 recursively, (3) the reserved word *until* is the first word on that line starting at margin m and (4) the expression after *until* obeys the rules recursively. More formally, if the instance *rptst* we are considering had two statements, say *st*1 and *st*2, in its body and *exp* as its expression, and *w*1, *w*2 are white spaces, i.e.

$$rptst = w1 \mid \text{"}repeat\text{"} \mid st1 \mid \text{";"} \mid st2 \mid w2 \mid \text{"}until\text{"} \mid exp$$

then the logical conjunction given by the diagram is:

PLOT(*repeat statement*, *rptst*, *m*) =
PLOT(*nt*REPEAT, *w*1 | "*repeat*", *m*) & NEWL(*w*1 | "*repeat*")
& PLOT(*statement*, *st*1, *m* + UOI)
& PLOT(*ny*SEMICOLON, ";", *m* + UOI)
& PLOT(*statement*, *st*2, *m* + UOI) & NEWL(*st*2)
& PLOT(*nt*UNTIL, *w*2 | "*until*", *m*) & NEWL(*w*2 | "*until*")
& PLOT(*expression*, *exp*, *m* + UOI)

where UOI stands for the unit of indentation. We now define PLOT and NEWL more precisely.

Definition of PLOT

PLOT is a predicate on triplets consisting of a non-terminal, a string and a margin width.

1. PLOT(n, $s \mid c$, m) ::= PLOT(n, s, m), where c is either % or $\backslash e$. Thus we assume below that s has no trailing white space.
2. PLOT(n, s, m) ::= **false**, if n does not produce s. Thus we further assume below that $n \rightarrow^* s$.
3. PLOT(*empty*, " ", m) ::= **true**, for all m.
4. PLOT(t, s, m) ::= ISAT(s, m), where t is a (non-terminal) token.
5. Let $n = n_1 n_2 \ldots n_k$ be a syntax rule of the language. Let $s, s_1, s_2, \ldots, s_k$ be corresponding strings generated from the non-terminals n and the n_i. Then