and read a new character of the string to be translated. This process will work if and only if the following condition ("LR(1)") always holds in the grammar: If $X_1X_2 \cdots X_nX_{n+1}\omega_1$ is a sentential form followed by "$\dashv$" for which all characters of $X_{n+1}\omega_1$ are terminals or "$\dashv$", and if this string has a handle (n, p) ending at position n, then all 1-sentential forms $X_1X_2 \cdots X_nX_{n+1}\omega$ with $X_{n+1}\omega$ as above must have the same handle (n, p). The definition has been phrased carefully to account for the possibility that the handle is the empty string, which if inserted between X_n and X_{n+1} is regarded as having right boundary n.

This definition of an LR(k) grammar coincides with the intuitive notion of translation from left to right looking k characters ahead. Assume at some stage of translation we have made all possible reductions to the left of X_n ; by looking at the next k characters $X_{n+1} \cdots X_{n+k}$, we want to know if a reduction on $X_{r+1} \cdots X_n$ is to be made, *regardless* of what follows X_{n+k}. In an LR(k) grammar we are able to decide without hesitation whether or not such a reduction should be made. If a reduction is called for, we perform it and repeat the process; if none should be made, we move one character to the right.

An LR(k) grammar is clearly unambiguous, since the definition implies every derivation tree must have the same handle, and by induction there is only one possible tree. It is interesting to point out furthermore that nearly every grammar which is known to be unambiguous is either an LR(k) grammar, or (dually) is a right-to-left translatable grammar, or is some grammar which is translated using "both ends toward the middle." Thus, *the* LR(k) *condition may be regarded as the most powerful general test for nonambiguity that is now available.*

When k is given, we will show in Section II that it is possible to decide if a grammar is LR(k) or not. The essential reason behind this that *the possible configurations of a tree below its handle may be represented by a regular* (*finite automaton*) *language.*

Several related ideas have appeared in previous literature. Lynch (1963) considered special cases of LR(1) grammars, which he showed are unambiguous. Paul (1962) gave a general method to construct left-to-right parsers for certain very simple LR(1) languages. Floyd (1964a) and Irons (1964) independently developed the notion of *bounded context* grammars, which have the property that one knows whether or not to reduce any sentential form $\alpha\theta\omega$ using the production $A \rightarrow \theta$ by examining only a finite number of characters immediately to the left and right of θ. Eickel (1964) later developed an algorithm which would construct a