The Theory of Computation
Bernard M.E. Moret
Addison-Wesley-Longman, 1998

Solution Manual
Chapter 7, Additional Problem 27

Recall that a context-free grammar is given by a terminal alphabet, Σ, a set of non-terminal symbols, $\mathcal{N}$, a starting symbol $S \in \mathcal{N}$, and a set of productions, each of the form $A \to w$, where $A \in \mathcal{N}$ and $w \in \{\Sigma \cup \mathcal{N}\}^*$, i.e., w is a string made of terminal and non-terminal symbols or is the empty string, ε (in which case $A \to w$ is an ε production). A derivation in the grammar starts from the start symbol, S; applies to it any production whose left-hand side is S, replacing S by the right-hand side of the production, and continues by replacing any non-terminal symbol in the resulting string by any right-hand side of a production whose left-hand side is that non-terminal, until the string is composed only of terminal symbols. The strings of terminals (i.e., strings in Σ^*) that can thus be derived form the language defined by the grammar. A useless non-terminal is one which can never appear in any intermediate string derived from the start symbol (such a non-terminal is effectively "unreachable").

The problem is easily shown to be NL-hard by transforming *Digraph Reachability* to it. We make each vertex a non-terminal in the grammar; for each arc, (i, j), we add a production $i \to j$; in addition, there is a starting symbol, S, with productions $S \to i$ for each i (so that no nonterminal is inaccessible), productions $i \to a$ for some terminal a and each i (so that each nonterminal derives a string) and production $n \to a1$. This last production ensures that the language is infinite if and only if there is a path (a series of productions) from 1 to n. Note that the resulting grammar is in fact regular; together with the second half of the proof below, this show that the problem "is the language generated by a regular grammar infinite" is itself NL-complete.

To show that the problem is in NL, we need only show that there exists some nonterminal, A, such that there is a derivation $A \Rightarrow \alpha A \beta$, for some strings of terminals and nonterminals α and β (not both empty). Since no nonterminal is useless and there are no ε productions, nonemptiness of one of the two strings guarantees that $\alpha A \beta$ derives a terminal string and that that string is longer than that derived by using the same productions on A. The derivation from A to $\alpha A \beta$ need not be longer than n, where n

is the number of nonterminals of the grammar. Since we can count to n in logarithmic space, the solution is then as follows: the machine guesses the nonterminal, then the derivation, production by production; we can verify in logarithmic space if the original nonterminal appears on the right-hand side, if at least one of the productions used increased the length of the sentential form, and if each production can be applied to the current sentential form. (We need not store the sentential form: the machine can also guess which nonterminal need be stored, since we look only for a sequence of productions.)