The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Solution Manual
Chapter 3, Problem 3.33

Let L be our language and SUB(L) the set of subsequences of strings in L; denote by L' the complement of SUB(L). Note that, by definition of minimal elements, no minimal element may be a subsequence of any other minimal element. We want to show that the set of minimal elements for a language is always finite.

Since this is the hard part of the proof, let us start with the relatively simple case where the alphabet has two symbols (the case of a one-symbol alphabet is of course trivial). Let the alphabet be $\Sigma = \{0, 1\}$. Let x be any string in Σ^* of length $n = |x|$. It is easily seen that x must be a subsequence of $(01)^n$ (or $(10)^n$), since the latter has n copies of each alphabet symbol. Let y be a string over Σ^* such that x is not a subsequence of y; then it follows immediately that neither $(01)^n$ nor $(10)^n$ can be a subsequence of y (because the relation "is a subsequence of" is transitive). Now define the family of languages $\{L_i\}$, $i \in \mathbb{N}$, with $L_{2n} = (0^*1^*)^n$ and $L_{2n+1} = L_{2n}0^*$. Thus L_i is defined on the basis of the largest possible number of alternations between 0 and 1 in any string ($i - 1$ as per our definition). Note that any string of length n must belong to L_{2n}—again, we can choose any symbol for each position.

We claim that, if z belongs to L_k for some k and S is a subset of L_k in which any two strings are incomparable, then S contains only a finite number of strings y such that z is not a subsequence of y. We prove our claim by induction on k. The basis ($k = 0$) is trivially true. Assume that the result holds for all strings z in L_i, $0 \leq i < k$, and consider a string z in L_k; in particular, z has at most $k - 1$ alternations. Note that our inductive hypothesis implies that S contains only a finite number of strings in L_{k-1} (as can be seen by choosing $z = \varepsilon$). Thus we need only prove that S contains finitely many strings y with exactly $k - 1$ alternations such that z (a string with at most $k - 1$ alternations) is not a subsequence of y. But S cannot contain more than a finite number of strings y with exactly $k - 1$ alternations in any case—regardless of z: in fact, it cannot contain more than k such strings, one for each run in the series of alternations (you can imagine each such string as having $k - 1$ short runs and one long run; each has its long run in a different place in the series of alternations, so that none can be a subsequence of the other; but any additional string with $k-1$ alternations is either a subsequence or a supersequence of one of the existing k strings.)

Now we are almost done, at least for languages over a two-symbol alphabet: we simply need to conclude that the set of minimal elements for any language over the alphabet $\Sigma = \{0, 1\}$ is finite. Our previous result indicates that any set of S of minimal elements can only have a finite number of members with at most k alternations, for any value of k (because we can pick z to be the empty string). Moreover, our earlier results indicate that any string with at least $2n$ alternations includes as a subsequence any string of length n. Thus suppose that our set S includes at least one string of length n; then it cannot have any strings in it with more than $2n-1$ alternations, since such strings would not be minimal. Hence the maximum number of alternations is bounded by some (unknown) constant, which is at most twice the length of the shortest string in S; such being the case, the number of strings in S must be finite.

Now we need to extend this result to alphabets of more than two elements. The details are simple and mechanical (we can base our reasoning on alternations again, but this time have to consider a larger number of alternations), but rather tedious, so we omit them here. With this result in mind we have proved that the set of minimal elements for L', the complement of SUB(L), is finite. Using this result, we can easily recognize L' with an NFA: the NFA will guess a subsequence of the input string and verify that it is one of the finitely many minimal elements of L'—because a string belongs to L' if and only if it contains a minimal element as a subsequence. (The "only if" part is trivial. The "if" part follows from the definition of SUB(L): if the string were in fact in SUB(L), then all of its subsequences would be too, yet one of these subsequences is a minimal element of L', a contradiction.) Since the complement of SUB(L) is regular, so is SUB(L): we have thus obtained the desired result.