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

Solution Manual
Chapter 6, Problem 6.20

We already know $\text{P} \subseteq \text{NP}$ and $\text{Exp} \subseteq \text{NExp}$. So our additional hypothesis is $\text{NP} \subseteq \text{P}$ and what we need to prove is $\text{NExp} \subseteq \text{Exp}$. In order to prove it, we start with some set $S \in \text{NExp}$ and proceed to pad its elements so that the set of padded instances, call it S_p, now belongs to NP; we then use the hypothesis $\text{NP} \subseteq \text{P}$ to conclude that there exists some deterministic algorithm A that decides S_p in polynomial time; finally, we show that A (or a trivial modification of it) also decides the original S, albeit in exponential time, since the elements of S are much more concise than those of S_p. This sequence of implications proves that any $S \in \text{NExp}$ is also part of Exp, so that we have, as desired, $\text{NExp} \subseteq \text{Exp}$.

The critical step is obviously the padding. Without loss of generality, let the alphabet be binary. Given $x \in S$, we generate $x_p \in S_p$ by setting $x_p = x\#0^{2^{|x|}}$, where # is a new symbol—we can keep the binary alphabet by the simple device of recoding 0 as 00, 1 as 01, and coding the new # as 11. The new string is exponentially longer than the old one. The deterministic checking algorithm for S can be trivially modified to read its input only up to the code for # and decode the 2-digit code back into its "native" alphabet and thus will serve as a checker for S_p; since the checker ran in time $O(2^{p(|x|)})$ on input x, it runs in the same amount of time on the new input (at worst, there is some small—linear—additive term due to the check for # and the decoding), but now that time is polynomial in $|x_p|$. It follows that S_p is in NP and thus, by hypothesis, in P. Hence there exists some deterministic polynomial-time algorithm A that solves S_p. Now, note that, whatever A looks like, there is no need for it to scan its input past the marker #; that is, there exists some similar algorithm A', that acts just like A except that it never reads its input tape past the marker. A trivial modification to A' allows it to run on elements of S as well—just replace the check for the marker # by a check for a blank. Hence this new algorithm, call it A'', solves S deterministically; its running time is some polynomial function of $|x_p|$, say $q(|x_p|)$; however, running on $x \in S$, the running time becomes $q(2^{p(|x|)})$ and thus exponential in $|x|$. Hence $S \in \text{Exp}$, as desired.