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

Solution Manual
Chapter 7, Problem 7.42

Obviously, what we need is a proof by induction. In order to examine what the induction step involves, let us simply look at going from NP to Σ_2^p. If we can work it out, we can then repeat the idea to go from Σ_k^p to Σ_{k+1}^p and thus prove the result (since it is clearly sufficient to show it for one type of class). In order to establish that problems in $\Sigma_2^p = \mathrm{NP}^{\mathrm{NP}}$ can be solved in $O(n^{\log^k n})$ time for some constant k, we need to "push" a problem from that class down to NP, use the known solution algorithm for NP problems, then recover a running time for the original problem. What normally makes $\mathrm{NP}^{\mathrm{NP}}$ different from NP is that the oracle can handle both yes and no answers, not just yes answers, preventing us from guessing a polynomial number of certificates (one for each oracle call). However, we know now that the oracle, which simply answers a question in NP, can be replaced by a certificate that uses $O(n^{\log n})$ bits—because the certificate can simply be the entire sequence of steps taken by the solution algorithm known to exist for NP problems. Thus we can solve a problem in $\mathrm{NP}^{\mathrm{NP}}$ non-deterministically by guessing the polynomial number of oracles answers in the form of a polynomial number of certificates, each of size $O(n^{\log n})$, and by otherwise doing whatever the original nondeterministic machine did. In other words, our problem of size n in $\mathrm{NP}^{\mathrm{NP}}$ corresponds to a problem of size $O(p(n)\cdot n^{\log n})$ in NP, where $p(n)$ is some polynomial. But we have a deterministic solution algorithm for problems in NP; on a problem of size $S = O(p(n)\cdot n^{\log n})$, this algorithm will take $O(S^{\log S}) = O\left((p(n)\cdot n^{\log n})^{(}\,\log(p(n)\cdot n^{\log n}))\right)$ time. Simplifying this last expression takes some work; note that $\log(p(n)\cdot n^{\log n})$ is $O(\log^2 n)$ and that $p(n)\cdot n^{\log n}$ is $O(n^{k+\log n})$ for some constant $k > 0$, which in turn is $O(n^{\log^2 n})$, so that the entire expression is $O(n^{\log^4 n})$, proving this first step. The very same line of reasoning proves the general step, so we are done.