The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Solution Manual

Chapter 6, Problem 6.26

- Using a non-constructive proof is not a problem; after all, the classes P and NP themselves are defined non-constructively. That we would presumably be unable to construct the DTM is thus no obstacle. However, the big flaw is that the argument only applies for a *specific* yes instance: the NDTM, arriving at some decision point, will presumably take different branches depending on the instance, so that we would have, not one, but many DTMs corresponding to that one NDTM. So, the logic is correct for a single instance—but we knew that a single instance is easy to solve anyway; and the logic completely fails for an infinite collection of instances (we would have to guess which DTM to use at each point, bringing us right back to non-determinism).
- It is really quite simple to add unique labels to each transformed instance: a trivial example is just to repeat the original instance as a "postscript" to the transformed instance—this certainly ensures a one-to-one transformation! Indeed, for many encoding schemes, each code is a valid instance, which means that each instance is its own index. Hence we can derive, using the Schroeder-Bernstein theorem, a bijection between the two problems.
  The problem is that this bijection may not be a polynomial-time isomorphism, i.e., it may not be a valid transformation anymore. To be a valid transformation, it has to respect two conditions: (i) polynomial-time feasibility and (ii) mapping yes instances to yes instances and no instances to no instances. But the Schroeder-Bernstein theorem only states that, given two 1-1 functions (not transformations), there exists a bijection. We do not know that the bijection in question bears much relation to the two injections developped by padding; it may neither be computable in polynomial time nor respect the yes-yes, no-no dichotomy. So flaws may reside there. In fact, a flaw appears even earlier: our padding does result in an injective mapping, but does it respect the yes-yes, no-no dichotomy? This is not at all certain; it may not even produce legal instance descriptions.