The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Solution Manual
Chapter 7, Additional Problem 13

That *Kernel* is in NP is clear. To show that it is NP-complete, we transform *3SAT* to it. As usual, the transformation involves a truth-setting component, a satisfaction-testing component, and appropriate connections between the two. For each variable in *3SAT*, we put 2 vertices in the graph, connected as a 2-cycle; those vertices correspond to the uncomplemented and the complemented literals for the variable. Those are the truth setting components: at most one vertex from a 2-cycle can be in a kernel and we shall make sure that at least one vertex must be selected by not having any arc terminate at a vertex from the 2-cycle. Placing a vertex in the kernel will correspond to setting its associated literal to true. For each clause in *3SAT*, we place 3 vertices in the graph, connected in a 3-cycle; the vertices correspond to the three literals of the clause. This is the satisfaction testing component; it is a sink, just like the 2-cycle is a source; it will also have at most one vertex in a kernel, yet one vertex alone does not cover the 3-cycle, so that there must be at least one arc coming from a kernel vertex to one of the other vertices of the cycle; this arc will "point" to the literal that makes the clause true. Finally, we add arcs from the literals in the truth-setting components to their matching part in the satisfaction-testing components. It should now be obvious that we have a solution to *3SAT* if and only if we have a kernel for the graph.