The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Detailed Solutions
Chapter 8, Problem 10, part 2

We give three successive solutions, improving and simplifying in each case. The last solution is particularly simple and takes care of *VC* and *Maximum Cut* at once, while also helping in general with other similar degree-reduction problems.

A solution to the degree reduction in *Maximum Cut* can be developed along much the same lines as for *VC*, but needs a small amendment. To replace a vertex of degree $d > 3$, we can try to use a simple cycle of length $2d$, where odd vertices serve as attaching points, and increase the bound by $2d$, thereby effectively requiring that all gadget edges be cut. This cut can be accomplished by coloring all attaching points with the same color and all others with the opposite color. Thus a solution to the original problem immediately implies a solution to the transformed problem. Unfortunately, there could be unforeseen cuts that still work. If the original problem has a vertex of degree $2k$ where k of the neighbors are placed on one side of the partition and the other k on the other side and the k neighbors on the same side are next to each other in the cycle, we could partition the gadget to cut all $2k$ edges leading out of it to the neighbors and still manage to cut $4k - 2$ edges inside the gadget, leading to a cut of $6k - 2$ edges, whereas the original graph had only k cut edges, a difference of $5k - 2$—exceeding the $4k$ increase in the bound and thus invalidating our transformation for $k > 2$. But what works once works twice: we can connect the gadget vertices of degree 2 in a cycle of their own by adding another d vertices of degree 2, thereby creating a second, "inner" cycle; the bound is correspondingly increased by an additional $2d$ (this turns the vertices of degree 2 into vertices of degree 4, higher than allowed; but our gadget works for degree 4 vertices without modification, so we can use it later to take care of these vertices). Now a vertex of degree $2k$ in the same circumstances gives rise to a gadget where we can cut all outside edges and all gadget edges, except for 2 per cycle, for a total cut of $8k - 4 + 2k$ (not accounting for the contribution of the gadgets used to replace the vertices of degree 4 introduced by the construction); since the original had a cut of k, the difference is $9k - 4$, which exceeds the difference in bounds of $4d = 8k$ when $k > 4$. At this point, we can add a third inner cycle in the same style; in general, a simple computation shows that a vertex of degree $2k$ or $2k+1$ will require $n = \lceil k/2 \rceil$ nested cycles, with a corresponding bound increase of $4nk$ or $2n(2k+1)$, which remains safely polynomial. Once the nested cycles have been introduced, all vertices of degree 4 are replaced by our "first-level" gadget.

We can avoid the nesting altogether and use a simple cycle of $2d$ vertices as originally intended if it is used in a context where all edges connected to the original vertex of degree d must be cut—in which case the problem described above cannot arise; such a context is precisely that arising out of our original transformation from *NAE3SAT* to *MaxCut*: the clauses vertices all have degree 3 and thus need not be replaced, while and all edges incident upon the truth-setting vertices must be cut.

In both *VC* and *MaxCut*, it is possible that the transformed instance admits "weird" solutions, where the gadget is not treated as described above. Those solutions arise whenever the original instance had a loose bound—where many fewer than B vertices would constitute a cover or where many more than B edges could be cut. In those cases, however, we can easily "migrate" elements of the solution into the gadgets to obtain a new solution where the gadgets are used as intended. In both *VC* and *MaxCut*, the latter problem can be avoided by using a transformation from a *SAT* problem; the simplest such (even simpler than above) is to use, in both cases, an extension of the truth-setting component to provide separate attaching points for each clause literal. The original truth-setting component in each transformation was a single edge per variable; for variable x_i, use instead $2k_i + 1$ edges in a line, where k_i is the larger of the number of occurrences of x_i and the number of occurrences of $\overline{x}_i$, and increase the bound for *VC* by k and the bound for *MaxCut* by $2k$. The resulting instances have exactly the same structure as those produced in the original transformation, but the degree of every vertex is at most 3, so that no gadget at all is required!