Homework 12

Adam Karl

November 10, 2020

1 Problem 37 (6 points)

1.1 Motivation

In order to be valid, the first trip of the boat in the FGAB (fox, goose, and beans) problem must remove vertices that disjoint the entire graph leaving no remaining edges. Since this seems very similar to the VertexCover problem, the first instinct to solve VertexCover(G,k) using an algorithm for FGAB is to simply run FGAB(G,k) and return what it returns.

This solution works for some solutions, including the original problem (1 fox, 1 goose, 1 bag of beans):

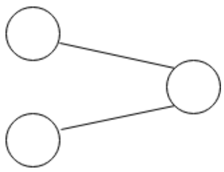

figure 1: min vertex cover = 1; min boat size = 1

But does not work for some other solutions, and will return a "false negative" here: (by "false negative," I mean FGAB would return false on k=1 when VertexCover should return true on k=1)

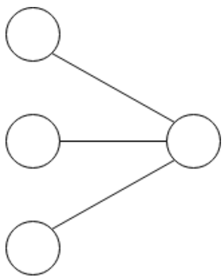

figure 2: min vertex cover = 1; min boat size = 2

The minimum boat size needed is lower bounded by minVertexCover. In order to leave the shore on the first boat trip and not immediately lose, the remaining vertices must not have any edges between them. this is only possible is we remove (at least) a set of vertices that cover all edges.

The minimum boat size needed is upper bounded by minVertexCover + 1. An algorithm to solve any FGAB problem using a boat of size minVertexCover + 1 would simply be: add the vertices from minVertexCover to the boat, then transport the rest of the vertices 1 by 1 while never removing the minVertexCover from the boat until everything else is across.

Thus, we may write $minVertexCover \leq minFGAB \leq minVertexCover + 1$.

It's as though we sometimes have an "off by one error" when trying to use FGAB to solve VertexCover. If VertexCover of G should return true for a value of k, it's possible that FGAB(G,k) returns true, but it's also possible that FGAB(G,k) returns false and FGAB(G,k+1) returns true.

However, we cannot simply run FGAB(G,k+1) and return what it returns, because the "off by one error" only occurs in certain scenarios, and FGAB(G,k+1) returning true may actually mean a minimum VertexCover of k+1 vertices (in which case VertexCover(G,k) should return false).

In order to bypass this case, there is an extremely elegant solution: we double the problem. Take G and add a complete duplicate of G with all the vertices and edges of G, but with no connections to the original graph. Let this graph be 2G. Now, FGAB(2G, 2k+1) returning true indicates a vertex cover of size k exists on G, without the possibility that the minimum vertex cover is actually of size k+1.

- If k is the minimum VertexCover for G, then FGAB(2G,2k) may or may not return true, but FGAB(2G,2k+1) will always return true.
- If k+1 is the minimum VertexCover for G, then FGAB(2G,2k+2) may or may not return true, but FGAB(2G,2k+3) will always return true.
- Note there is now no overlap between the end of the range for k (2k and 2k+1) and the beginning of the range for k=1 (2k+2 and 2k+3)

Therefore, running FGAB(2G, 2k+1) will return true if and only if VertexCover(G,k) should return true.

1.2 Algorithm Description

To solve VertexCover(G,k) using an algorithm for FGAB, do:

- Take G and add a duplicate of G including all vertices and edges, but with no edges to the original graph. Let this new graph be 2G.
- Run FGAB(2G, 2k+1) and return what it returns.

1.3 Conclusion

Since FGAB(2G, 2k+1) returns true if and only if VertexCover(G,k) should return true, and the input/output transformations are polytime, we have shown VertexCover is polytime reducible to FGAB. Since VertexCover is a known NP-hard problem, FGAB must also be NP-hard.