Assignment 2

Marcus Aurén

February 11, 2019

1

As discussed in the fourth lecture, the optimal solution to a Minimum Vertex Cover is integral if the matrix A in the equality $Ax = B$ is totally unimodular (TUM). As discussed in the lecture a matrix is TUM if
(1) The entries are all {-1, 0 1}
(2) There are at most 2 non-zero entries in any column
(3) The rows can be partitioned into two subsets so that the entries in a column are in different subsets if they have the same sign or in the same subset if they have different signs.

The A matrix for the problem is

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

which has only one 1 entry in every column that is a 1 or less hence the matrix is TUM. Given that this matrix i TUM, the optimal solution to the problem is integral.

2

(a) The solution that I got from the modulation 7 with the two nodes with weights 2 and the node with weight 4 are chosen.

(b) The LP relaxation gives the following solution.
Optimal Solution is:
7.0
With values:
x1 = 0.4847027530392067 (This is the top left node)
x2 = 0.5152972405931519 (This is the top right node)

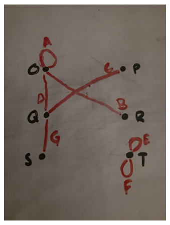
x3 = 0.51529724053O5474 (This is the middle left node)
x4 = 0.9999999956149173 (This is the middle right node)
x5 = 0.4847027535346086 (This is the bottom left node)
x6 = -3.6797789551761404e-10 (This is the bottom right node)

I solved the problem using the CVXOPT modeling.op function. The solution is rather similar to the solution for ILP. The same nodes from (a) are the ones that are preferred, as those nodes are the ones that are given the highest value, and the same optimal solution is given. Here one of the values is extremely small but negative, my guess is that the program is having float-point errors as one of the constraints does not allow it to go below zero.

(c) When applying the rounding rule I got the following solution.
Rounded Solution is:
x1 = 0
x2 = 1
x3 = 1
x4 = 1
x5 = 0
x6 = 0
Value of rounded solution is: 7
The solution given my the rounding rule gave the same solution as the ILP solution.

3

(a) A hypergraph can be seen as a set of vertices and we can view E as a set of hyperedges. In this case we only have hyperedges connecting 1 or 2 vertices so we can formulate the hyperedge as a graph like this. From here we can formulate the problem the same way we would for a graph, making sure not to forget the edges that are from a node to the same node. For the given problem we could formulate a graph such as this.



If we would have a hyperedge between 3 nodes, let's say x1, x2, x3 we would only need to pick on out of the three vertices to have the edge covered. Hence, we would get a vertex cover if we picked vertices so that we have one vertex in every subset.

(b) We get a constraint from each of the hyperedges. Here we get one hyperedge for each of the languages which connects all of the translators that speak that language.
The solution I got from cvxopt.glpk.ilp was to hire the translator O, Q and T with an optimal solution of 5.

The LP relaxation was implemented using glpk.lp and gave the that exact same solution as the ILP namely:
OPTIMAL LP SOLUTION FOUND
$[1.00e + 00]$ (This is translator O)
$[0.00e + 00]$ (This is translator P)
$[1.00e + 00]$ (This is translator Q)
$[0.00e + 00]$ (This is translator R)
$[0.00e + 00]$ (This is translator S)
$[1.00e + 00]$ (This is translator T)

(d) The optimal solution is an integer solution so using LP relaxation gives the same solution. If the optimal solution is an integer solution LP relaxation will still return it.

4

(a) The solution that I got from cvxopt.glpk.ilp was the following:

INTEGER OPTIMAL SOLUTION FOUND
70.0
(b) Here I got a strange and pretty useless result.
OPTIMAL LP SOLUTION FOUND
50.0
ROUNDED OPTIMAL SOLUTION GIVES SOLUTION:
100.0

The solution that the LP relaxation gave was that 0.5 of each node should be picked giving a answer vector that was filled with only 0.5 entries, when rounding all of the vertices get picked which gives a vertex cover, however far from the minimum one.

(d) Solution to the given algorithm
93.0
The given solution is a vertex cover because the algorithm guarantees that at least one vertex on either side of an edge is included in S and hence each edge will be covered in the vertex cover. It also won't cost anymore than the rounding rule as it picks the most valuable of each of the vertexes according to the LP solution. It does not guarantee finding an optimal solution and it did not

for me.
The cost of this solution is a lot worse than the optimal solution, most likely because my implementation starts from the first node and works from there. If the nodes are suitable order in the correct order it will give the optimal solution however the way that the vertices are looked at in my implementation it seems like it unfortunately picks some bad nodes which gives a worse solution. Choosing a random order for the edges to be looked at might give a better or worse solution than my implementation depending on chance.