**T01**

*Can we solve* ***Min-Vertex-Cover in polynomial time*** *if all vertices in the input graph have* ***degrees not more than 2****? What if we want to solve* ***Min-Weight-Vertex-Cover*** *variant on that graph type?*

* Degree 0: isolated vertices 🡪 ignore as they have no edge associated with them.
* Degree 1: ignore vertex, but include connected / adjacent vertex as it covers the edge and possibly another edge (if degree = 2)
* Degree 2: either path (take every alternate vertex: k/2) or cycle (k/2 + mod(k, 2))

🡪 Count degree 1 vertices and alternate vertex in paths + vertices of cycles 🡪 Polynomial time

**Min-Weight:**

🡪 Degree 1: choose the cheapest vertex of this and the adjacent vertex

🡪 Path/Cycle: start at different positions and find minimal combinations through dynamic programming.

*In lecture, we have seen a* ***Dynamic Programming (DP) solution*** *to solve the Min-Vertex-Cover on a (Binary)* ***Tree****. How about the following* ***Greedy Algorithm*** *(solution for CLRS Ex 35.1-4): First, take* ***any leaf in the tree****, then add its* ***parent to the (minimal) vertex cover****, then* ***delete the leaf and parent and all associated edges****. Repeat this process* ***until there is no vertex remain*** *in the tree. Is this a correct Greedy Algorithm? Hint: Check https://visualgo.net/en/mvc, `MVC on Tree', the `Greedy MVC on Tree' option. Can it be used on the Min-Weight-Vertex-Cover variant?*

* Leaf nodes: vertices of degree 1 🡪 Selecting parent potentially covers other edges as well (more leaf nodes) 🡪 parent is the greedy choice
* Eliminating vertices reduces problem size as we can safely remove the covered edges and vertices.
* Iteration Results in a tree with no vertices 🡪 all edges covered.

Always results in optimal solutions as there are no cycles in a tree (more complex to cover edges optimally with cycles)

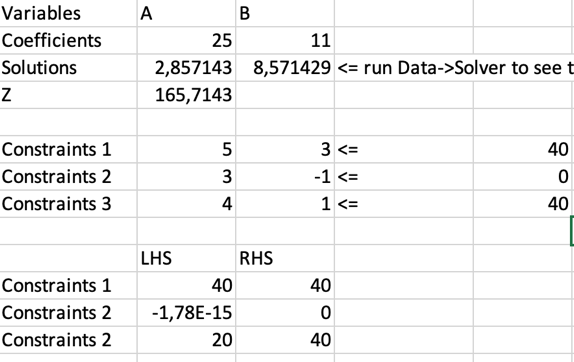
Optimal Substructure:

*A carpenter makes tables and chairs. Each table can be sold for a profit of 25 SGD and each chair for*

*a pro t of 11 SGD. The carpenter can afford to spend up to 40 hours per week working and takes 5 hours to make a table and 3 hours to make a chair. The owner requires that the carpenter makes at least 3 times as many chairs as tables (so that he can package it as dining table set). Tables take up 4 times as much storage space as chairs and there is room for at most 10 tables each week. Formulate this problem as a Linear Programming problem (write in standard form) and solve it (graphically, with Excel, with lp solve, or with a (Simplex? or Brute Force?) program :O). Now if the solution is a floating point numbers, please round them so that we have an Integer solution. Are you sure your Integer solution is the optimal one?*

* a = number of tables made per week
* b = number of chairs made per week
* Profit function: z = 25\*a + 11\*b
* Time Constraint: 5\*a + 3\*b <= 40
* Chair to Table ratio: 3\*a <= b
* Storage constraint: 4\*a + b <= 40

Solution: (A, B) = (2.86/ 8.57) 🡪 Z = 165.7

🡪 Integer: (2, 8), Z=138 🡪 rounding down generally ensures a feasible solution 🡪 but might not be optimal / not guaranteed

**PS1 – Debrief:**

* Keep track of local minimum path
* Model state as combination of fireworks and mountain id 🡪 apply Dijkstra
* Dijkstra on all possible transformations as states
* Dijkstra: initialize node = 0 with inf 🡪 rest 0 🡪 store minimum between current value and street capacity

Q4:

* Planar graph (Criterion: Kuratowski, Wagner) 🡪 K<= 4
* Test pair of edges O(m^2)
* Num of edges: 3n – 6

Q5:

* In any planar graph: you need at most 4 colors
* O(4^n) 🡪 try all possible 4 colors on each vertices