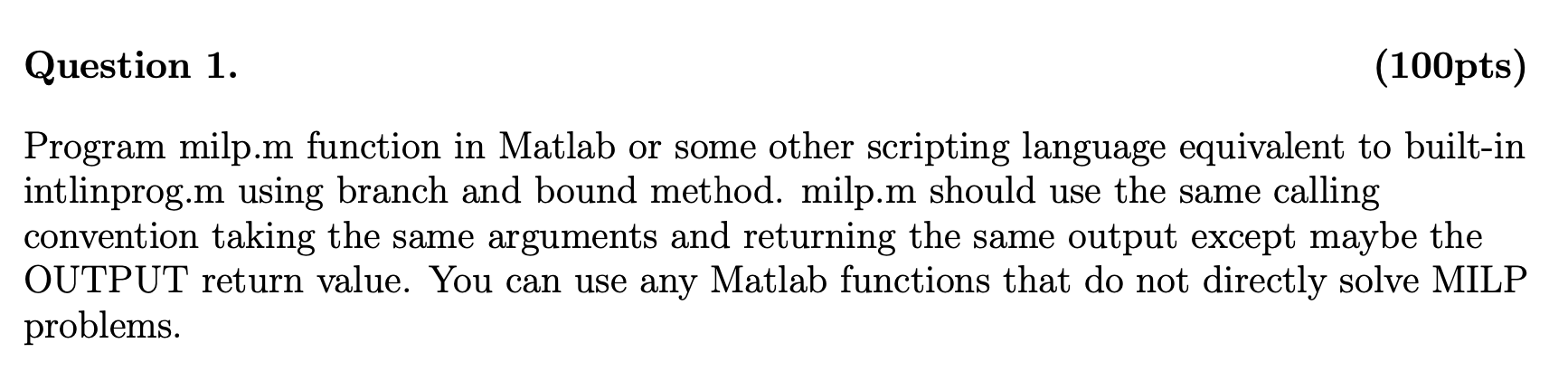
**18-879M: Optimization in Energy Networks Homework 6**

**Issued: Friday, April 17, 2020 Due: Friday, April 24, 2020, 11:59pm**

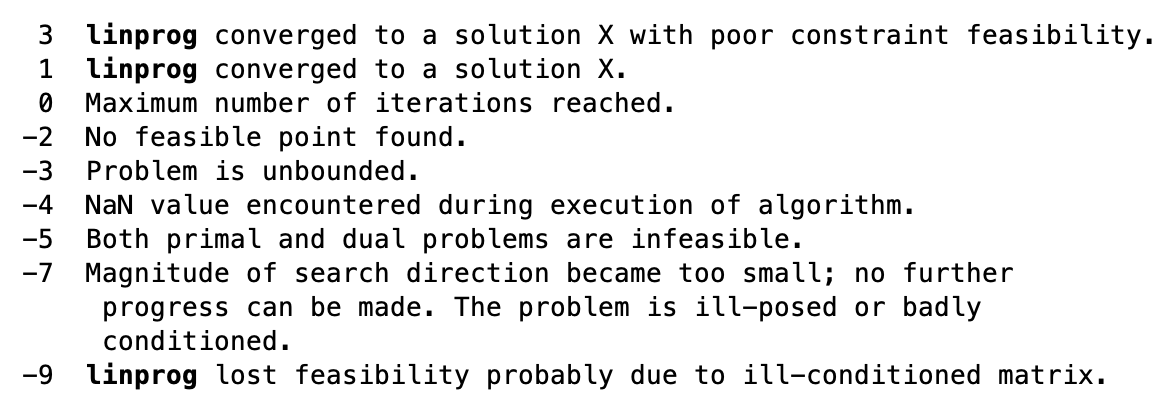


In this homework, we are supposed to write a MATLAB function for mixed-integer linear programming. I used branch and bound algorithm since it is simple. The algorithm is similar to a binary tree. There are two common ways to code a tree, depth first or breadth first. In this problem, I will code in breadth-first order, so I. use queue to record the order of nodes.

# Branching

To implement branch and bound algorithm, we need to figure out the branching conditions first. There are four possible branches in the problem.

**Branching 1**

****

Branching 1 is when no solution available. In MATLAB linprog document, the only two conditions we can get a solution is exitflag = 1 or 3. So, the first base condition is when linprog does not give us any solutions. We can find it by checking exitflag of linprog. In this condition, we will just set the value of current node to inf. The program is unnecessary go deeper for this node since the child nodes will have more constrains than current and make the current problem even more unsolvable.

**Branching 2**

Branching 2 is when we find a feasible solution and the solution is better (objective value is less than the best solution). So, we will record the current solution sets and update the overall bound. The program is unnecessary to go deeper for this node since child nodes will have more constraints and make the objective value bigger than the current.

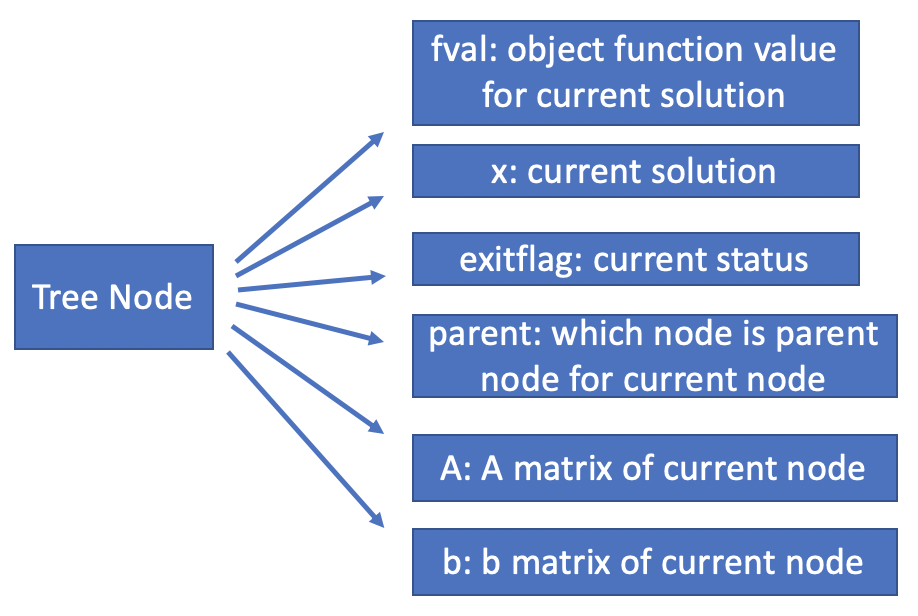
**Branching 3**

Branching 3 is when we find a solution and the solution is worse than the best solution. Going deeper only makes the objective value bigger than the current, so we just set the value of current node to current and the program will not go deeper for this node.

**Branching 4**

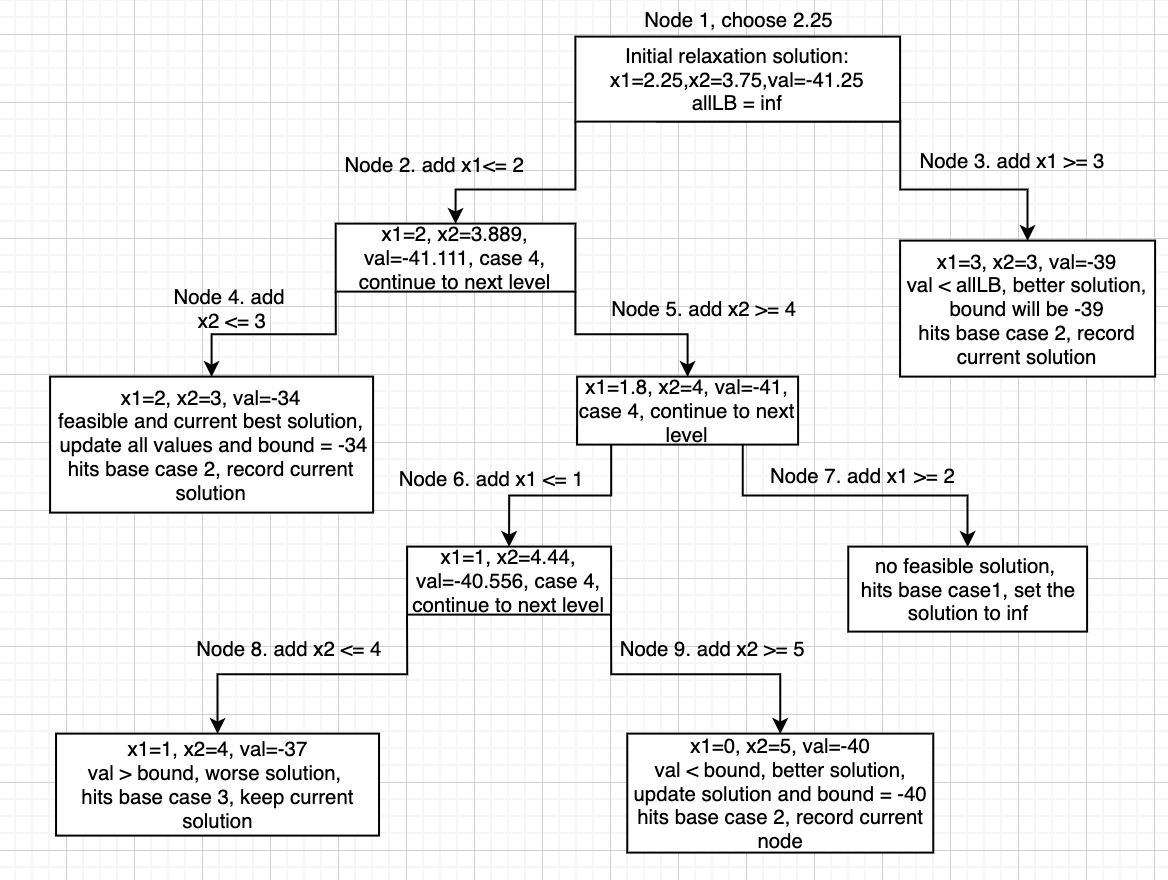
Branching 4 is when any integer constrains violations in our relaxation solution but potentially has better solution. In my program, it will choose the first violated variables and break each one into two subproblems. For example, if the violation variable is 3.2, the first subproblem will be added x <= 3 as a new condition and the second subproblem will be added x >= 4 as new condition of parent node and solve them respectively. We can create child nodes and add more constrains to child nodes in this branching condition. We also need add new node to queue here.

# Node Attributes

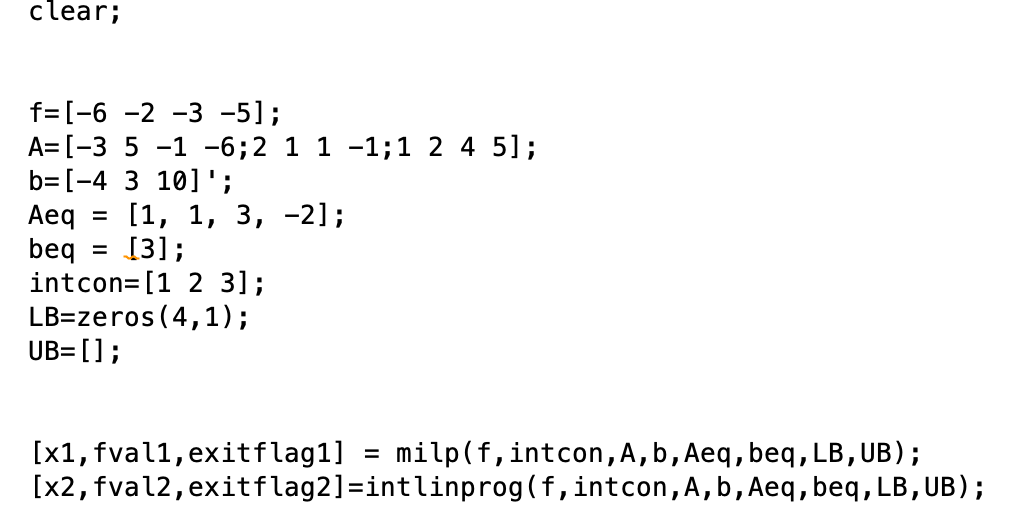


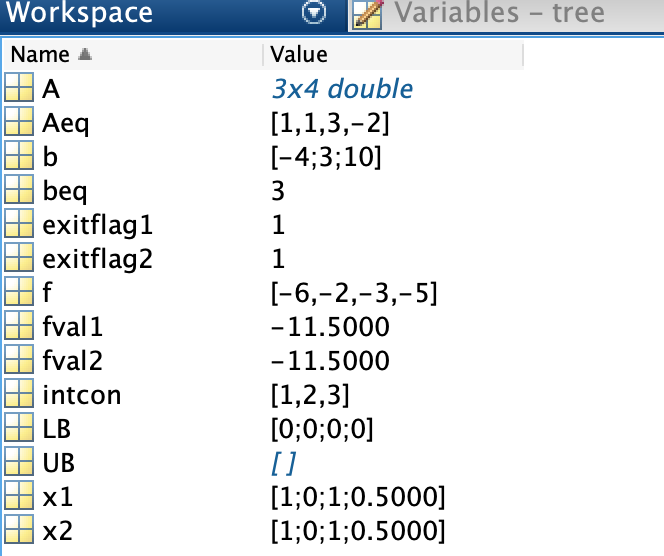
# Flowchart

Next, I will use a real example to introduce how is my algorithm works.

****

# Function Testing





My answer agree with MATLAB.