MIE 1603 / 1653 - Integer Programming
Winter 2019
Assignment #1

Due Date: January 23, Wednesday, no later than 11:59PM.

Solve the following problems. You may work in groups of *two* students (from the same course code) or individually. Turn in **one** solution set with both group members listed on it. Groups must work independently of each other. You *must* cite any references (texts, papers or websites) you have used to help you solve these problems.

Note that some questions will be only for MIE1603 or MIE1653. If so, this is indicated at the beginning of the question. Otherwise, the question is common for both groups. Although you are not responsible (and will not be getting points) from the questions targeting the other group, you are welcome to submit answers for such questions as well if you would like to get feedback. (Solutions, full or partial, will be posted for all questions.)

You can submit your solutions using Quercus (pdf file only) or give a hard-copy to the professor or TA in their office hours (or lectures).

1. [20pts] Your department is organizing a workshop for their graduate students and you volunteer to write an IP model to maximize the success of the workshop. The department has a budget of B dollars to spend in the workshop. There is a list of possible location $L = \{1, ..., m\}$ from which you have to choose only one. Each location $k \in L$ has a cost C_k (dollars) and a total time restriction T_k, i.e., you cannot schedule activities that in total take more than T_k units of time. Depending on the location that you choose is the set of activities that you can choose from, i.e., if you choose location k the set of available activities is $A_k = \{1, ..., n_k\}$ ($n_k > 1$, integer). Each activity $i \in A_k$ has a success factor f_{ik}, a duration d_{ik} and a cost of c_{ik} dollars.

   Each activity teaches one or more skills, where $S = \{1, ..., r\}$ is the set of all possible skills. For simplicity, consider p_{iks} as a known parameter such that $p_{iks} = 1$ if activity $i \in A_k$ of location $k \in L$ teaches skill $s \in S$, and $p_{iks} = 0$ otherwise. The department requires that, for each skill $s \in S$, there is at least one activity that teaches s. Moreover, the number of selected activities that teach skill $s \in S$ has to be strictly smaller than $U_s \geq 1$ (integer).

   Model this problem using (Mixed) Integer Linear Programming. Clearly state and explain all the components of your model (e.g., variables, constraints and objective function).

   Solution:
   Consider the following binary variables:

$$x_k = \begin{cases} 1, & \text{if location } k \text{ is picked} \\ 0, & \text{otherwise} \end{cases}, \qquad y_{ik} = \begin{cases} 1, & \text{if activity } i \text{ in location } k \text{ is picked} \\ 0, & \text{otherwise} \end{cases}$$

A possible IP model for this problem is as follows:

$$\max \sum_{k=1}^{m} \sum_{i=1}^{n_k} f_{ik} y_{ik} \tag{1a}$$

$$s.t. \sum_{k \in L} x_k = 1 \tag{1b}$$

$$\sum_{k \in L} C_k x_k + \sum_{k \in L} \sum_{i \in A_k} c_{ik} y_{ik} \leq B \tag{1c}$$

$$\sum_{i \in A_k} y_{ik} \leq n_k x_k \qquad \forall k \in L \tag{1d}$$

$$\sum_{i \in A_k} d_{ik} y_{ik} \leq T_k \qquad \forall k \in L \tag{1e}$$

$$\sum_{k \in L} \sum_{i \in A_k} p_{iks} y_{ik} \geq 1 \qquad \forall s \in S \tag{1f}$$

$$\sum_{k \in L} \sum_{i \in A_k} p_{iks} y_{ik} \leq U_s - 1 \qquad \forall s \in S \tag{1g}$$

$$x_k \in \{0, 1\} \qquad \forall k \in L \tag{1h}$$

$$y_{ik} \in \{0, 1\} \qquad \forall i \in A_k,\ k \in L \tag{1i}$$

The objective function (1a) maximizes the total success factor. Constraint (1b) guarantees that exactly one location will be picked. Constraint (1c) corresponds to the budget limit. Constraint (1d) limits the number of activities in the chosen location. This constraint also guarantees that only activities from the chosen location are picked. Constraint (1e) limits the total duration of the activities. Constraint (1f) and (1g) model the lower and upper bounds of the activities skills. The last two constraints define the variables domains.

2. [20pts] Consider that you have solved the problem in Question 1 and the set of activities selected is $A = \{1, .., n\}$. Now your department wants to schedule these activities and asked for your help. Consider that each activity has a duration of 1 hour. You have a horizon of n hours, i.e, a set of n time slots $H = \{1, .., n\}$. Each activity $i \in A$ has a productivity factor P_{it} that depends on the time slot $t \in H$ that it is scheduled on. A valid schedule is one where each activity is present and each time slot has exactly one activity assigned to it. The goal is to find a schedule that maximizes the total productivity.

   Moreover, the set of activities is partitioned into three categories, i.e., $A = A_1 \cup A_2 \cup A_3$ and $A_i \cap A_j = \emptyset$ for $i \neq j$. Your schedule cannot have three consecutive activities from the same category. For example, if activities a_1, a_3 and a_5 are in A_2, then a schedule with activity a_1 in time slot 2, activity a_3 in time slot 3 and activity a_5 in time slot 4 is invalid. Lastly, consider that some activities have pre-requirements. Let R be the set of requirements such that $(i, i') \in R$ means that activity i needs to be scheduled before activity i' (not necessarily immediately before).

   Model this problem using (Mixed) Integer Linear Programming. Clearly state and explain all the components of your model (e.g., variables, constraints and objective function).

Solution:
Consider the following binary variables:

$$x_{it} = \begin{cases} 1, & \text{if activity } i \text{ is assigned to time slot } t \\ 0, & \text{otherwise} \end{cases}$$

An IP model for this problem is as follows:

$$\max \sum_{t=1}^{n} \sum_{i=1}^{n} P_{it} x_{it} \tag{2a}$$

$$s.t. \sum_{i \in A} x_{it} = 1 \qquad \forall t \in H \tag{2b}$$

$$\sum_{t \in H} x_{it} = 1 \qquad \forall i \in A \tag{2c}$$

$$\sum_{i \in A_k} x_{it} + \sum_{i \in A_k} x_{i,t+1} + \sum_{i \in A_k} x_{i,t+2} \leq 2 \qquad \forall k \in \{1,2,3\},\ t \in \{1,...,n-2\} \tag{2d}$$

$$\sum_{t \in H} t x_{it} \leq \sum_{t \in H} t x_{jt} - 1 \qquad \forall (i,j) \in R \tag{2e}$$

$$x_{it} \in \{0,1\} \qquad \forall i \in A,\ t \in H \tag{2f}$$

The objective function (2a) maximizes the total productivity. Constraints (2b) and (2c) guarantee that each activity is scheduled and that each time slot has exactly one activity, respectively. Constraint (2d) restricts the model to avoid scheduling three consecutive activities from the same category. Constraint (2e) model the precedene requierement between activities. The last constraint defines the domain of the variables.

3. [20pts] Consider an undirected graph $G = (N, E)$, where N is the set of nodes and E is the set of edges. Let $L = \{1, 2, ..., |E|\}$ be a set of natural numbers. The problem asks you to assign a number from L to each edge such that no incident edges (i.e., edges that have a common endpoint) have the same number. Consider that for each node $i \in N$, set $A(i)$ represents all the edges that are incident to the node i. The goal is to minimize the sum of the assigned numbers to the edges over all the edges. As an example, the graph in Figure 1 has $N = \{a, b, c, d, e, f\}$ and $E = \{(a, b), (a, c), (b, c), (b, d), (c, d), (b, e), (d, e), (d, f), (e, f)\}$. The numbers over the edges form a valid assignment with a total edge sum of 19.

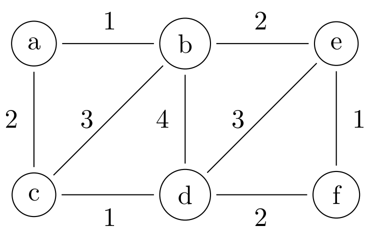


Figure 1: Example for Question 3 - A numbering of a graph

(a) Model the above problem using (Mixed) Integer Linear Programming. Clearly state and explain all the components of your model (e.g., variables, constraints and objective function).

Solution:

Consider the following binary variables:

$$x_{ek} = \begin{cases} 1, & \text{if edge } e \text{ is given number } k \\ 0, & \text{otherwise} \end{cases}$$

An IP model for this problem is as follows:

$$\min \sum_{k \in L} \sum_{e \in E} k x_{ek} \tag{3a}$$

$$s.t. \sum_{k \in L} x_{ek} = 1 \qquad \forall e \in E \tag{3b}$$

$$\sum_{e \in A(i)} x_{ek} \leq 1 \qquad \forall i \in N,\ k \in L \tag{3c}$$

$$x_{ek} \in \{0,1\} \qquad \forall e \in E,\ k \in L \tag{3d}$$

The objective function (3a) minimizes the sum of all labels assigned to the edges. Constraint (3b) guarantees that each edge is assigned to exactly one label. Constraint (3c) models the restriction that no two adjacent edges have the same label. The last constraint corresponds to the variable domain.

(b) Modify your model in part (a) to consider the following objective: minimize the largest number assigned to any edge. For example, for the graph in Figure 1, the new objective value of the provided example solution is 4. Explain any new variables and/or constraints that you use.

Solution:

For this question we define a new variable y_max that represents the maximum number assing to any edge.

A possible IP model for this problem is as follows:

$$\min y_{max} \tag{4a}$$

$$s.t. \sum_{k \in L} x_{ek} = 1 \qquad \forall e \in E$$

$$\sum_{e \in A(i)} x_{ek} \leq 1 \qquad \forall i \in N,\ k \in L$$

$$k \cdot x_{ek} \leq y_{max} \qquad \forall e \in E,\ k \in K \tag{4b}$$

$$y_{max} \geq 1 \tag{4c}$$

$$x_{ek} \in \{0,1\} \qquad \forall e \in E,\ k \in L$$

The new objective (4a) minimizes the maximum number assigned to all edges. Constraint (4b) computes the maximum number assigned to all edges. Constraint (4c) defines the domain for the new variable.

(c) (**MIE1603**) Modify your model in part (a) to consider the following change in the objective function. If any two edges are assigned the same number, then you need to pay an additional cost equal to the square of that number. The goal is to minimize the total cost, i.e., edge sum plus the total pairwise duplicate edge-number cost. As an example, the objective value in the about graph is $19 + (3 \cdot 1^2 + 3 \cdot 2^2 + 3^2) = 43$. Explain any new variables and/or constraints that you use.

Solution:

Define $\mathcal{E}$ as the set of all edge pairs such that for all edges $e, e' \in E$ ($e \neq e'$) either $(e, e') \in \mathcal{E}$ or $(e', e) \in \mathcal{E}$. We will use this set to define a new binary variable that will represent if any two edges have the same value:

$$z_{ee'k} = \begin{cases} 1, & \text{if edge } e \text{ and } e' \text{ are assigned to the number } k \\ 0, & \text{otherwise} \end{cases}$$

A possible IP model for this problem is as follows:

$$\begin{aligned}
\min \ & \sum_{k \in L} \sum_{e \in E} k x_{ek} + \sum_{k \in L} \sum_{(e,e') \in \mathcal{E}} k^2 z_{ee'k} && && \text{(5a)} \\
s.t. \ & \sum_{k \in L} x_{ek} = 1 && \forall e \in E && \\
& \sum_{e \in A(i)} x_{ek} \leq 1 && \forall i \in N,\ k \in L && \\
& x_{ek} + x_{e'k} \leq z_{ee'k} + 1 && \forall (e, e') \in \mathcal{E},\ k \in L && \text{(5b)} \\
& z_{ee'k} \in \{0, 1\} && \forall (e, e') \in \mathcal{E},\ k \in L && \text{(5c)} \\
& x_{ek} \in \{0, 1\} && \forall e \in E,\ k \in L &&
\end{aligned}$$

The new objective (5a) minimizes the total cost. Constraint (5b) relates variables x and z. Constraint (5c) defines the domain of the z variable.

(d) **(BONUS - 8pts)** Consider that the value of a node $i \in N$ is given by the sum of the number assigned to its incident edges. For example, the value of node a is 3, while the value of node c is 6 . Make the following changes to your model in part (a). Make sure that your model remains linear. Explain any new variables and/or constraints that you use.

i. Change the objective value such that it minimizes the total node value.

Solution:

New model:

$$\begin{aligned}
\min \ & \sum_{i \in N} \sum_{e \in A(i)} \sum_{k \in L} k \cdot x_{ek} && && \text{(6a)} \\
s.t. \ & \sum_{k \in L} x_{ek} = 1 && \forall e \in E && \\
& \sum_{e \in A(i)} x_{ek} \leq 1 && \forall i \in N,\ k \in L && \\
& x_{ek} \in \{0, 1\} && \forall e \in E,\ k \in L &&
\end{aligned}$$

The new objective minimize the total sum of each node.

ii. Enforce the restriction that two adjacent nodes cannot have the same value.
Solution:
Consider $U = \{1, ..., |E| \cdot |E|\}$ as the set of possible values that a node can take. We define a new binary variable:

$$w_{iu} = \begin{cases} 1, & \text{if node } i \text{ has value } u \\ 0, & \text{otherwise} \end{cases}$$

A possible IP model for this problem is as follows:

$$\begin{aligned}
\min \ & \sum_{i \in N} \sum_{u \in L} u \cdot w_{iu} & & \text{(7a)} \\
s.t. \ & \sum_{k \in L} x_{ek} = 1 & \forall e \in E & \\
& \sum_{e \in A(i)} x_{ek} \leq 1 & \forall i \in N, \ k \in L & \\
& \sum_{e \in A(i)} \sum_{k \in L} k x_{ek} \leq u w_{iu} & \forall i \in N, \ u \in U & \quad \text{(7b)} \\
& w_{iu} + w_{ju} \leq 1 & \forall (i,j) \in E & \quad \text{(7c)} \\
& x_{ek} \in \{0,1\} & \forall e \in E, \ k \in L & \\
& w_{iu} \in \{0,1\} & \forall e \in E, \ u \in U & \quad \text{(7d)}
\end{aligned}$$

The objective function minimizes the total node cost (equivalent to the one in the previous question). Constraint (7b) links variables x and w to define the value of a node. Constraint (7c) makes sure that any two adjacent nodes have a different value. The last constraint represents the domain of the new variable.

iii. Consider the graph in Figure 1. Enforce the restriction that any solution where node a has value 3 and node b has value 7 (simultaneously) is invalid.
Solution:

$$w_{a3} + w_{b7} \leq 1$$

4. [5pts] Consider an undirected graph $G = (N, E)$ where each edge $e \in E$ has a cost $c_e \geq 0$. Given two vertices $s, t \in N$ $(s \neq t)$, the *Shortest Path* (SP) problem asks for a minimum-cost path (i.e., no cycles) that starts at s and ends at t. As an example, the graph in Figure 2 highlights a path from s to t with cost equal to 26.

This is a well-known problem in the Operation Research community that can be efficiently solved, e.g., using Dijkstra's Algorithm. A closely related problem is the *Bounded Shortest Path* (BSP) problem which asks for the shortest path from s to t with a cost greater than or equal to b.

Given any graph $G = (N, E)$ and two vertices $s, t \in N$ $(s \neq t)$, prove that the optimal value of the SP problem is a valid lower bound for the BSP problem for any $b \in \mathbb{R}$. (Note that if the BSP problem is infeasible for a given b, then its optimal objective value is assumed to be $+\infty$.)

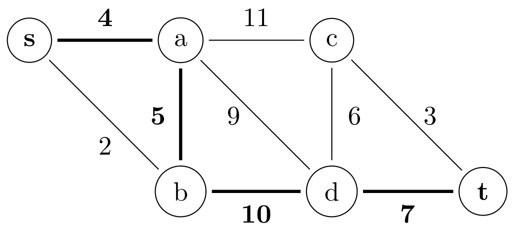


Figure 2: Example for Questions 4 and 5 - A shortest path instance

Solution:
Given a graph $G = (N, E)$ and two vertices $s, t \in N$ $(s \neq t)$, consider F and F_B as the set of feasible solution of the SP and BSP problem, respectivebly. We first need to show that $F_B \subseteq F$. Consider a solution $x \in F_B$, by definition x is a path from s to t over G. Therefore, x is also feasible for the SP problem, i.e., $x \in F$.

Now consider z^* and w^* are the optimal solutions of the SP and the BSP, respectively. Let $c(\cdot)$ a function that represents the cost of a path. Since $F_B \subseteq F$, we have that $w^* \in F$ and so $c(z^*) = \min\{c(x) : \ x \in F\} \leq c(w^*)$. Therefore, the optimal solution of the SP problem is a lower bound for the BSP. $\square$

5. [25pts](**MIE1603**) Consider the BSP problem described above and answer the following questions.

   (a) Design a branch-and-bound algorithm to solve the BSP problem, where the lower bound at each node of the search is given by the value of the SP (i.e, ignoring the bound constraint at that node). In other words, the subproblem to be solved at each node of the branch-and-bound tree should be an SP problem in *a certain graph*. Clearly define the branching rule in your branch-and-bound algorithm, and describe how the SP problem is defined at each child node given your branching rule.

   Hints: (1) The SP solution may violate the bound constraint for some nodes: in this case the optimal path cannot possibly contain all the edges in the SP solution, so at least one of them cannot be in the optimal solution. (2) Deleting edges from the graph just gives another graph, a subgraph of the original; the subproblem to be solved at each node of the branch-and-bound tree should be an SP problem in some subgraph.

   Solution:
   Input: Problem $(G = (N, E),\ s, t \in N,\ b \in \mathbb{R})$, $k \in \mathbb{R}$ and initial upper bound (optional);
   Branching rule: Consider a subgraph $G' = (V, E')$ of G and its shortest path P'. For each $e \in P'$, create a new subproblem with $G'' = (V, E' \backslash \{e\})$. In other words, we create a new subproblem for each edge that part of the shortest path P'. Each subproblem solves the SP problem in a subgraph with one edge less than the parent subproblem.
   Algorithm notation:

   - P^* = shortest path for the BSP.
   - UB = the global upper bound for the BSP. Initally $UB = \infty$ or any upper bound given as input (i.e., k).

- LB = the global lower bound for the BSP. Initially $LB = -\infty$.
- $Branch(G')$ = branching algorithm that uses the branching rule describe above.
- $SubproblemList$ = list of subproblems to evaluate in the branch-and-bound algorithm.

The alogrithm is as follows:

```
Define an initial node with graph $G' = G$, and add it to $SubproblemList$.
while $LB < UB$ & !$SubproblemList$.empty() do
    Take a node from $SubproblemList$ and let $G'$ denote its associated graph;
    if $G'$ disconnected then
        Subproblem is prune by infeasibility;
    end if
    Calculate $P'$, the SP of $G'$;
    if $P'$ is feasible for BSP then
        If $c(P') < UB$, then $UB := c(P')$ and $P^* := P'$;
        Prune the subproblem;
    else
        if $c(P') \geq UB$ then
            Node prune by bound;
        else
            Branch($G'$): create the child subproblems and add them to $SubproblemList$;
        end if
    end if
    Update $LB$ and remove the current subproblem from $SubproblemList$;
end while
return  $P^*$ and $UB$;
```

(b) Discuss your branching rule: does it partition the solution space, or not? Explain, and/or illustrate with an example.

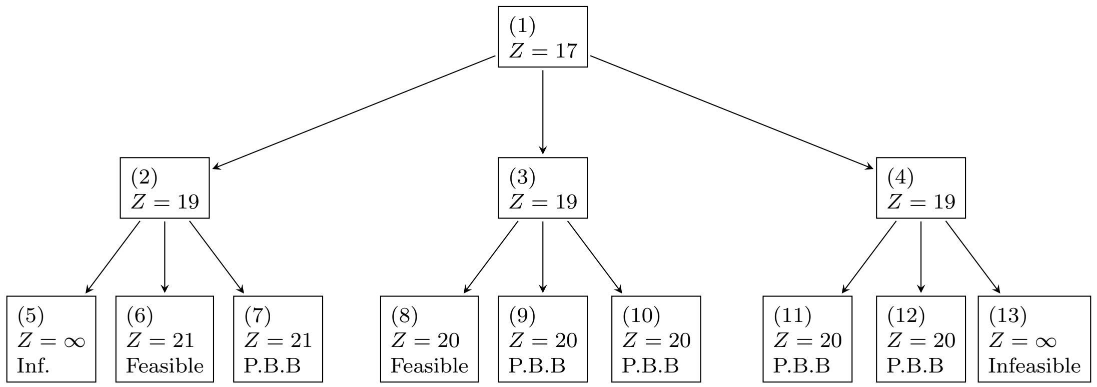
Solution:
The branching rule guarantees that all the search space will be explore. However, **it does not partition the search space**, i.e., the subproblems are nos disjoint. For example, consider an initial graph $G = (V, E)$ and the following partially tree search, where each node indicated the subgraph that we are solving. Notice that subproblem 5 and 6 have the same subgraph, i.e., the subproblems are the same.

$$\begin{tikzcd}
 & & & \boxed{(1),\ G=(V,E)} \arrow[dll] \arrow[drr] & & & \\
 & \boxed{(2),\ G'=(V,E\backslash\{e_1\})} \arrow[dl] \arrow[dr] & & & & \boxed{(3),\ G'=(V,E\backslash\{e_2\})} \arrow[dl] \arrow[dr] & \\
\boxed{(4),\ G'=(V,E\backslash\{e_1,e_3\})} & & \boxed{(5),\ G'=(V,E\backslash\{e_1,e_2\})} & & \boxed{(6),\ G'=(V,E\backslash\{e_2,e_1\})} & & \boxed{(7),\ G'=(V,E\backslash\{e_2,e_4\})}
\end{tikzcd}$$

(c) Illustrate your branch-and-bound method on the graph shown in Figure 2, with $b = 20$. You may use an initial upper bound of 26, which may be updated once your branch-and-bound search yields a feasible solution. Use the best-bound tree search strategy and label all branch-and-bound nodes in the order in which you created them. Clearly indicate where the branch-and-bound tree was pruned, which branch-and-bound nodes are leaf nodes, and why. Which is the shortest path with cost greater or equal to $b = 20$, and at which node of the branch-and-bound tree was it found?

Solution:

Below is the branch-and-bound tree. Nodes 5 to 10 are leaf nodes, where Inf. means *infeasible* and P.B.B means *prune by bound.*

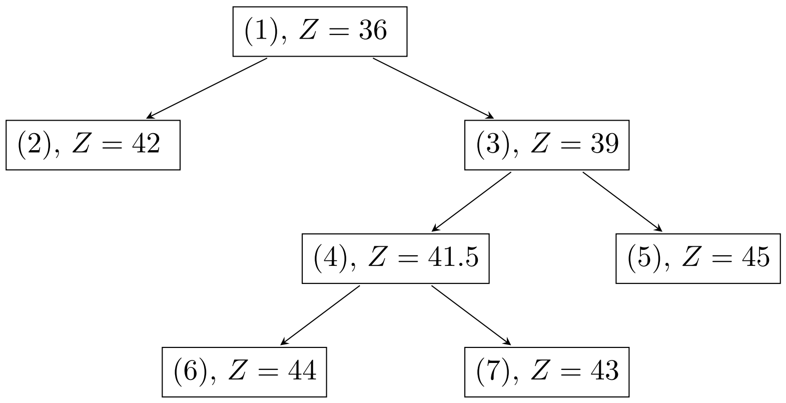


The sub-graph for each subproblem are:

$$\begin{aligned}
(1) &: G = (N, E) \\
(2) &: G' = (N, E'), \quad E' = E \setminus \{\{s,a\}\} \\
(3) &: G' = (N, E'), \quad E' = E \setminus \{\{a,c\}\} \\
(4) &: G' = (N, E'), \quad E' = E \setminus \{\{c,t\}\} \\
(5) &: G' = (N, E'), \quad E' = E \setminus \{\{s,a\},\{s,b\}\} \\
(6) &: G' = (N, E'), \quad E' = E \setminus \{\{s,a\},\{b,d\}\} \\
(7) &: G' = (N, E'), \quad E' = E \setminus \{\{s,a\},\{d,t\}\} \\
(8) &: G' = (N, E'), \quad E' = E \setminus \{\{a,c\},\{s,b\}\} \\
(9) &: G' = (N, E'), \quad E' = E \setminus \{\{a,c\},\{b,d\}\} \\
(10) &: G' = (N, E'), \quad E' = E \setminus \{\{a,c\},\{d,t\}\} \\
(11) &: G' = (N, E'), \quad E' = E \setminus \{\{c,t\},\{s,b\}\} \\
(12) &: G' = (N, E'), \quad E' = E \setminus \{\{c,t\},\{b,d\}\} \\
(13) &: G' = (N, E'), \quad E' = E \setminus \{\{c,t\},\{d,t\}\}
\end{aligned}$$

The optimal solution is path $P^* = (\{s,a\},\{a,d\},\{d,t\})$ with cost equal to 20. It was first found in node 8.

6. [10pts] (**MIE1653**) The following diagram shows a branch-and-bound tree search for an integer programming **minimization** problem. Each node has a number in parenthesis representing the search order and a value Z indicating the optimal LP relaxation value in that node. Consider that nodes 5 and 6 yield integer solutions (i.e., their optimal LP relaxation solution turned out to be an integer solution).



(a) Which is the incumbent (i.e., the best solution) for the problem? What is the global lower bound? What are the absolute optimality gap and the relative optimality gap?

Solution:
The incumbent is 44, found at node 6.
The global lower bound is given by $\min\{42, 43, 44, 45\} = 42$.
The optimallity gap is 44-42 = 2.

(b) Now consider that we branch on node 2 and create two new nodes: 8 and 9. Node 9 is infeasible and node 8 has $Z = 43$. What can you say about the optimal solution of the original IP? What would you do next?

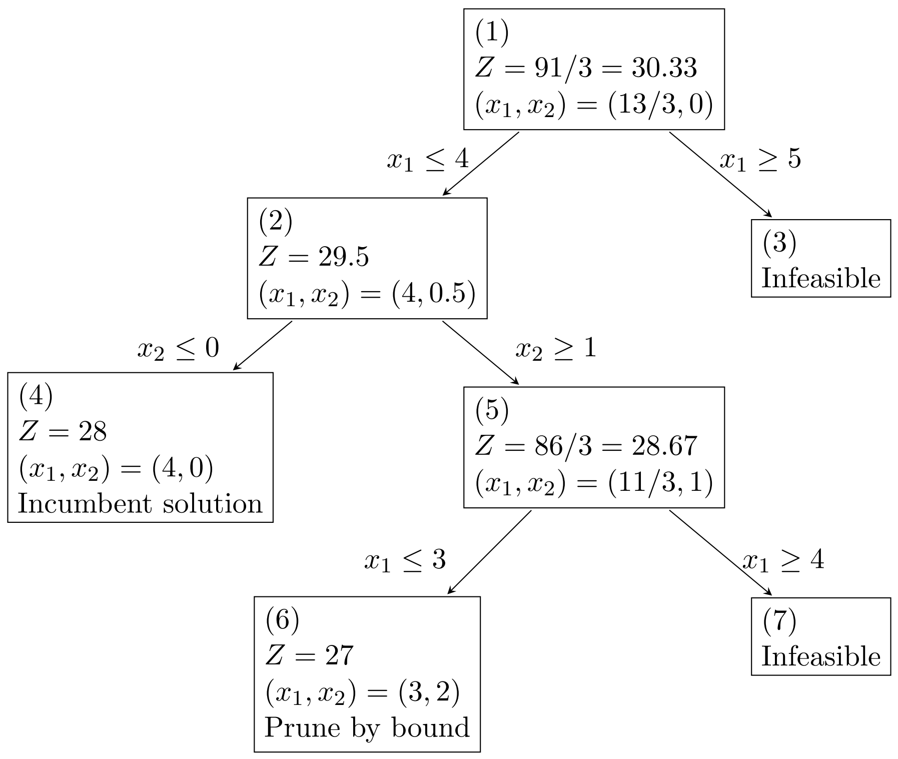
Solution:
The new lower bound would be 43, so the new gap is 1. We need to branch on node 8 to either find a better incumbent or prove optimality.

7. [15pts] (**MIE1653**) Consider the following IP:

$$\begin{aligned} \max\ & 7x_1 + 3x_2 \\ \text{s.t. } & 2x_1 + x_2 \leq 9 \\ & 3x_1 + 2x_2 \leq 13 \\ & x_1, x_2 \geq 0, \text{integer} \end{aligned}$$

Solve (by hand) the problem using branch-and-bound. Branch on the most fractional variable. Write down each subproblem, its solution (objective value and variable values) and draw the branch-and-bound search tree. You can use any method to solve the LP relaxations, e.g., an LP solver or a graphical solution.

Solution:
Bellow is the branch-and-bound tree for the problem.



8. [10pts] Consider the following polyhedra:

$$A = \left\{(x, y) \in \mathbb{R}^2 : x \geq 1,\ x + 2y \leq 9,\ -x + 2y \geq 1\right\}$$
$$B = \left\{(x, y) \in \mathbb{R}^2 : 3x + y \geq 13,\ -3x + 4y \geq -5,\ -x + 5y \leq 12\right\}$$

(a) Let $P = A \cup B$. For each of the following sets, draw the set and its convex hull:

i. $S_1 = P$

ii. $S_2 = P \cap \{(x, y) \in \mathbb{Z} \times \mathbb{R}\}$

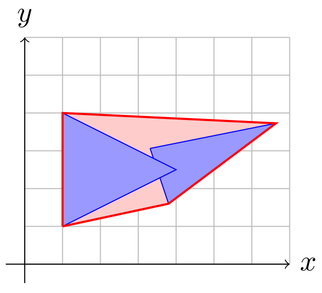
iii. $S_3 = P \cap \{(x, y) \in \mathbb{Z}^2\}$

Solution:
The extreme points of $conv(S_1)$ are $(1, 1), (1, 4), (6.636, 3727)$ and $(3.8, 1.6)$. The extreme points of $conv(S_2)$ are $(1, 1), (1, 4), (6, 3.6), (6, 3.25)$, and $(4.1.75)$. The extreme points of $conv(S_3)$ are $(1, 1), (1, 4), (5, 3)$, and $(4, 2)$.

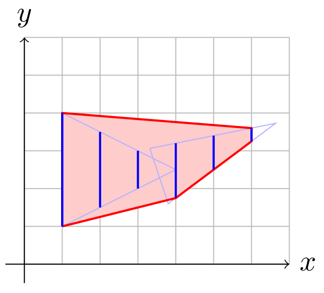
(b) Consider the following problem:

$$\begin{aligned} &\max x + y \\ &\text{s.t. } (x, y) \in conv(S_2) \\ &\qquad (x, y) \in \mathbb{Z}^2 \end{aligned}$$

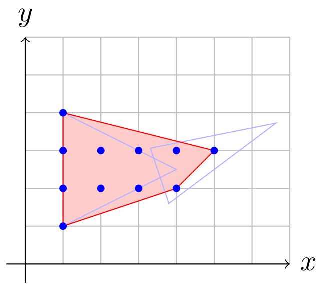
i. Find and report the optimal IP solution and the optimal LP relaxation solution, and their objective values, for the problem. What happens if you round down the LP solution? What happen if you round up the LP solution?



(a) S_1-blue, $conv(S_1)$-red



(b) S_2-blue, $conv(S_2)$-red



(c) S_3-blue, $conv(S_3)$-red

ii. Write down an LP model that solves the above IP, i.e., the set of integer solutions of your LP should be the same as the above IP and the optimal solution (and value) should be the same.

Solution:
The optimal IP solution is $(x, y) = (5, 3)$ with cost equal to 8. The optimal LP solution is $(x, y) = (6, 3.6)$ with cost equal to 9.6. If we round down or up the LP soltion we end up with an infeasible solution.

The LP model asked is:

$$\begin{aligned} \max\ & x + y \\ \text{s.t. } & (x, y) \in conv(S_3) \\ & (x, y) \in \mathbb{R}^2 \end{aligned}$$