# DM545 – Linear and Integer Programming

## Answers to the Take-home Assignment, Winter 2025

In this assignmnent I will be using Tableau Prinitng function provided on github by Marco always including it as:

```
from util import tableau
```

I will also be using *numpy*, and *fractions* for matrix operations:

```
import numpy as np
from fractions import Fraction
```

Throughout the Excersise a website that was showed during lectures will also be used as solver/visualizer

# Task 1

## Subtask 1.a

In the following problem:

$$\begin{aligned}
\max\ & 4x_1 + 5x_2 - 7x_3 \\
& -x_1 - x_2 + x_3 \leq 2, \\
& -5x_1 + 10x_3 \leq 10, \\
& x_1 \in [0, 5], \\
& x_2 \in [-1, 1], \\
& x_3 \in [-2, 2].
\end{aligned}$$

By inspection of the objective function:

$$\begin{aligned}
4x_1 &\Rightarrow 4 \text{ is positive } \Rightarrow x_1 \text{ as large as possible,} \\
5x_2 &\Rightarrow 5 \text{ is positive } \Rightarrow x_2 \text{ as large as possible,} \\
-7x_3 &\Rightarrow -7 \text{ is negative } \Rightarrow x_3 \text{ as small as possible.}
\end{aligned}$$

Check potential solution formed by limits of interval bounds $[x_1, x_2, x_3] = [5, 1, -2]$ :

$$\begin{aligned}
\text{1st constraint: } & -x_1 - x_2 + x_3 \leq 2 \\
& -5 - 1 + (-2) \leq 2 \\
& -8 \leq 2 \quad \text{is feasible} \\
\text{2nd constraint: } & -5x_1 + 10x_3 \leq 10 \\
& -25 + 10(-2) \leq 10 \\
& -45 \leq 10 \quad \text{is feasible}
\end{aligned}$$

$$\begin{aligned}
&\text{Objecive function value:} \\
&4x_1 + 5x_2 - 7x_3 = \\
&20 + 5 + 14 = \\
&39
\end{aligned}$$

We conclude the optimal solution is $[x_1, x_2, x_3] = [5, 1, -2]$

### Subtask 1.b

$$\begin{aligned} \max\ & x_1 + x_2 \\ & sx_1 + tx_2 \leq 1 \\ & x_1, x_2 \geq 0 \end{aligned}$$

We can choose (s) and (t) to get different cases:

**I) Single Optimal Solution**

$$s = 2, \quad t = 1$$

- The slope of the only constraint is different from the slope of the objective function.
- Geometrically, the feasible region intersects the objective function at a single vertex.
- The vertex is our optimal solution

**II) Infinite Optimal Solutions**

$$s = 1, \quad t = 1$$

- The constraint line is parallel to the objective function.
- Objective function “placed” on the face gives us infinitely many optimal solutions.

**III/IV) Infeasible / Unbounded**

- If either $s$ or $t$ (or both) are negative, the problem becomes unbounded as each variable balances the other in the constraint allowing the objective function to grow indefinitely.

$$s = -1, \quad t = 1 \text{ is an example of unbounded}$$

- For any $s, t \geq 0$, setting $x_1 = \frac{1}{s}, x_2 = \frac{1}{t}$ gives a feasible solution (except if $s = 0$ or $t = 0$, in which case the value of $x_1$ or $x_2$ does not matter as it's multiplied by $0$).

It is **impossible** to set $s, t$ such that the problem becomes *infeasible*, we considered all posssible cases

# Task 2

## Subtask 2.a

First, we transform the problem into the standard form:

- Change a minimization into a maximization:
  min$(c^T x)$ into $-$ max $-(c^T x)$.
- Replace equalities with two inequalities.

- Convert all constraints to "$\leq$".

The result is:

$$\begin{aligned} \max -3x_1 - 2x_2 - 7x_3 \\ -x_1 + x_2 \leq 10, \\ x_1 - x_2 \leq -10, \\ -2x_1 + x_2 - x_3 \leq -10. \end{aligned}$$

Then the tableau looks like:

```
SLACK_COUNT = 3

A = np.array([[-1, -1, 0],
              [1,  -1, 0],
              [-2, 1, -1]], dtype=object)
C = np.array([-3, -2, -7], dtype=object)
B = np.array([10, -10,-10], dtype=object)

A = np.vectorize(Fraction)(A)
C = np.vectorize(Fraction)(C)
B = np.vectorize(Fraction)(B)

I = np.array([[Fraction(int(i == j)) for j in range(SLACK_COUNT)] \
for i in range(SLACK_COUNT)], dtype=object)
Z = np.array([Fraction(0) for _ in range(SLACK_COUNT)], dtype=object)

T = np.concatenate([A.T, I], axis=1)
T = np.column_stack((T, Z))
T = np.column_stack((T, B))
T = np.vstack((T, np.concatenate([C, np.full(SLACK_COUNT, Fraction(0), \
dtype=object), [Fraction(1), Fraction(0)]])))

print("\n \n") # for pdf formating

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+-------+-------+
|    x1 |    x2 |    x3 |    x4 |    x5 |    x6 |    -z |     b |
|-------+-------+-------+-------+-------+-------+-------+-------+
|    -1 |     1 |    -2 |     1 |     0 |     0 |     0 |    10 |
|    -1 |    -1 |     1 |     0 |     1 |     0 |     0 |   -10 |
|     0 |     0 |    -1 |     0 |     0 |     1 |     0 |   -10 |
|-------+-------+-------+-------+-------+-------+-------+-------+
|    -3 |    -2 |    -7 |     0 |     0 |     0 |     1 |     0 |
|-------+-------+-------+-------+-------+-------+-------+-------+
```

As we can see the tableau is **optimal** (no positive reduced costs) but **infeasible** (two of the $b_i \leq 0$), we could apply Dual Simplex to work towards feasibility

### Subtask 2.b

Tableau is given:

```
T = np.array([[0,0,0,1,1,0,0,0],
              [0,1,1,2,0,-1,0,30],
              [1,0,1,1,0,-1,0,20],
              [0,0,2,-7,0,5,1,-120]],dtype=object)

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+-------+--------+
|    x1 |    x2 |    x3 |    x4 |    x5 |    x6 |    -z |      b |
|-------+-------+-------+-------+-------+-------+-------+--------+
|     0 |     0 |     0 |     1 |     1 |     0 |     0 |      0 |
|     0 |     1 |     1 |     2 |     0 |    -1 |     0 |     30 |
|     1 |     0 |     1 |     1 |     0 |    -1 |     0 |     20 |
|-------+-------+-------+-------+-------+-------+-------+--------+
|     0 |     0 |     2 |    -7 |     0 |     5 |     1 |   -120 |
|-------+-------+-------+-------+-------+-------+-------+--------+
```

It is worth noting that this tableau is *unbounded* as all coefficients of $x_6$ in $A$ are negative, but $x_6$ has a positive reduced cost, however we can still technically perform a change of basis and see the results.

By largest coefficient $x_6$ would have to enter (and $x_3$ leave as it's constraint is tighter) and that would make the problem infeasible, and unoptimal

```
# III * -1
T[2] = T[2] * Fraction(-1, 1)

# II + III
T[1] = T[1] + T[2]

# IV - 5*III
T[3] = T[3] - 5*T[2]

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+-------+--------+
|    x1 |    x2 |    x3 |    x4 |    x5 |    x6 |    -z |      b |
|-------+-------+-------+-------+-------+-------+-------+--------+
|     0 |     0 |     0 |     1 |     1 |     0 |     0 |      0 |
|    -1 |     1 |     0 |     1 |     0 |     0 |     0 |     10 |
|    -1 |     0 |    -1 |    -1 |     0 |     1 |     0 |    -20 |
|-------+-------+-------+-------+-------+-------+-------+--------+
|     5 |     0 |     7 |    -2 |     0 |     0 |     1 |    -20 |
|-------+-------+-------+-------+-------+-------+-------+--------+
```

By ratio test $x_3$ enters (as $x_6$ has only negative coefficients in $A$) and $x_1$ leaves as $20/1 < 30/1$ (we ignore first line $0/0 = ?$).

Coincidentally by Bland's Rule $x_3$ enters and $x_1$ leaves (we take the lowest index)

We can follow with both of them:

```
T = np.array([[0,0,0,1,1,0,0,0],
              [0,1,1,2,0,-1,0,30],
              [1,0,1,1,0,-1,0,20],
              [0,0,2,-7,0,5,1,-120]],dtype=object)


# II - III
T[1] = T[1] - T[2]

# IV - 2*III
T[3] = T[3] - 2*T[2]

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+-------+-------+
|    x1 |    x2 |    x3 |    x4 |    x5 |    x6 |    -z |     b |
|-------+-------+-------+-------+-------+-------+-------+-------+
|     0 |     0 |     0 |     1 |     1 |     0 |     0 |     0 |
|    -1 |     1 |     0 |     1 |     0 |     0 |     0 |    10 |
|     1 |     0 |     1 |     1 |     0 |    -1 |     0 |    20 |
|-------+-------+-------+-------+-------+-------+-------+-------+
|    -2 |     0 |     0 |    -9 |     0 |     7 |     1 |  -160 |
|-------+-------+-------+-------+-------+-------+-------+-------+
```

As we can see the tableau is still unbounded (by the case of $x_6$)

# Task 3

## Subtask 3.a

Tableau given:

```
T = np.array([[1,0,1,-1,0,5],
              [0,1,-2,3,0,15],
              [0,0,-2,-2,1,-110]],dtype=object)

tableau(T)
```

```
|-------+-------+-------+-------+-------+-------+
|    x1 |    x2 |    x3 |    x4 |    -z |     b |
|-------+-------+-------+-------+-------+-------+
|     1 |     0 |     1 |    -1 |     0 |     5 |
|     0 |     1 |    -2 |     3 |     0 |    15 |
|-------+-------+-------+-------+-------+-------+
|     0 |     0 |    -2 |    -2 |     1 |  -110 |
|-------+-------+-------+-------+-------+-------+
```

From the tableau, we observe the following:

- The solution is $[x_1, x_2, x_3, x_4] = [5, 15, 0, 0]$ with objective value $110$.
- The reduced costs are $-2, -2$
- The values of dual variables are $2, 2$ (negative reduced costs).
- The shadow prices are the same as the dual variables: $2, 2$
- There is no over-capacity, as all the constraints are tight (no slacks in the basis).

# Task 4

## Subtask 4.a

We denote original problem as **P** and relaxed original problem as **PR** Let's start by considering the original problem with marked constraints by $\alpha, \beta, \gamma$

$$
\begin{aligned}
\min \ & \sum_{i=1}^{n} c_i y_i \\
& \sum_{j=1}^{m} a_{ij} x_{ij} \leq b_i y_i, \quad i = 1, \dots, n \quad (\alpha) \\
& \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, \dots, m \quad (\beta) \\
& y_i \leq 1, \quad i = 1, \dots, n \quad (\gamma) \\
\\
& y_i \geq 0, \quad x_{ij} \geq 0.
\end{aligned}
$$

We denote the potential dual variables:

$$
\begin{aligned}
\alpha &= [\alpha_1, \dots, \alpha_n] \\
\beta &= [\beta_1, \dots, \beta_m] \\
\gamma &= [\gamma_1, \dots, \gamma_n]
\end{aligned}
$$

Then measure the violation of constraints (by putting everything to one side and multiplying by corresponding dual variable):

$$
\begin{array}{ccc}
\alpha_1 \, (0 + b_1 y_1 - \sum_{j=1}^{m} a_{1j} x_{1j}) & \beta_1 \, (1 - \sum_{i=1}^{n} x_{i1}) & \gamma_1 \, (1 - y_1) \\
\alpha_2 \, (0 + b_2 y_2 - \sum_{j=1}^{m} a_{2j} x_{2j}) & \beta_2 \, (1 - \sum_{i=1}^{n} x_{i2}) & \gamma_2 \, (1 - y_2) \\
\vdots & \vdots & \vdots \\
\alpha_n \, (0 + b_n y_n - \sum_{j=1}^{m} a_{nj} x_{nj}) & \beta_m \, (1 - \sum_{i=1}^{n} x_{im}) & \gamma_n \, (1 - y_n)
\end{array}
$$

Then we denote **PR** relaxed problem:

$$
\mathrm{PR}(\alpha, \beta, \gamma) = \min_{\text{by all } y, x \geq 0} \{\vdots\} =
$$

$$= \min_{\text{by all } y,x\geq 0} \left\{ \begin{array}{r} c_1y_1 + \cdots + c_ny_n + \\ \alpha_1\left(b_1y_1 - \sum_{j=1}^{m} a_{1j}x_{1j}\right) + \\ \alpha_2\left(b_2y_2 - \sum_{j=1}^{m} a_{2j}x_{2j}\right) + \\ \vdots \\ \alpha_n\left(b_ny_n - \sum_{j=1}^{m} a_{nj}x_{nj}\right) + \\ \beta_1\left(1 - \sum_{i=1}^{n} x_{i1}\right) + \\ \beta_2\left(1 - \sum_{i=1}^{n} x_{i2}\right) + \\ \vdots \\ \beta_m\left(1 - \sum_{i=1}^{n} x_{im}\right) + \\ \gamma_1\left(1 - y_1\right) + \\ \gamma_2\left(1 - y_2\right) + \\ \vdots \\ \gamma_n\left(1 - y_n\right) \end{array} \right\} = \min_{\text{by all } y,x\geq 0} \left\{ \begin{array}{r} y_1\left(c_1 + \alpha_1 b - \gamma_1\right) + \\ y_2\left(c_2 + \alpha_2 b - \gamma_2\right) + \\ \vdots \\ y_1\left(c_n + \alpha_n b - \gamma_n\right) + \\ x_{1,1}\left(0 - \alpha_n a_1 - \beta_1\right) + \\ x_{2,1}\left(0 - \alpha_2 a_1 - \beta_2\right) + \\ \vdots \\ x_{n,1}\left(0 - \alpha_n a_1 - \beta_n\right) + \\ x_{1,2}\left(0 - \alpha_1 a_2 - \beta_1\right) + \\ x_{2,2}\left(0 - \alpha_2 a_2 - \beta_2\right) + \\ \vdots \\ x_{n,m}\left(0 - \alpha_n a_m - \beta_n\right) + \\ \sum_{j=1}^{m} \beta_j + \\ \sum_{i=1}^{n} \gamma_i + \end{array} \right\}$$

To avoid useless lower bounds we set all parts multiplied by $y, x \geq 0$, (otherwise corresponding variable, for instance $x_{i,j} -> \infty$ and $\min_{\text{by all } y,x\geq 0} = -\infty$)
That's how we get to the dual constraints:

$$\begin{array}{r} (c_1 + \alpha_1 b - \gamma_1) \geq 0 \\ (c_2 + \alpha_2 b - \gamma_2) \geq 0 \\ \vdots \\ (c_n + \alpha_n b - \gamma_n) \geq 0 \\ (0 - \alpha_n a_1 - \beta_1) \geq 0 \\ (0 - \alpha_2 a_1 - \beta_2) \geq 0 \\ \vdots \\ (0 - \alpha_n a_1 - \beta_n) \geq 0 \\ (0 - \alpha_1 a_2 - \beta_1) \geq 0 \\ (0 - \alpha_2 a_2 - \beta_2) \geq 0 \\ \vdots \\ (0 - \alpha_n a_m - \beta_n) \geq 0 \end{array} \Rightarrow \begin{array}{r} \alpha_1 b - \gamma_1 \geq -c_1 \\ \alpha_2 b - \gamma_2 \geq -c_2 \\ \vdots \\ -\alpha_n b - \gamma_n \geq -c_n \\ -\alpha_n a_1 - \beta_1 \geq 0 \\ -\alpha_2 a_1 - \beta_2 \geq 0 \\ \vdots \\ -\alpha_n a_1 - \beta_n \geq 0 \\ -\alpha_1 a_2 - \beta_1 \geq 0 \\ -\alpha_2 a_2 - \beta_2 \geq 0 \\ \vdots \\ -\alpha_n a_m - \beta_n \geq 0 \end{array}$$

To get new Objective function we take the sums uncorrelated with $y, x$ in front of the $\min_{\text{by all } y,x} \{\vdots\}$. Now we want to:

$$\max_{\alpha,\beta,\gamma}\left\{\text{PR}(\alpha,\beta,\gamma)=\sum_{j=1}^{m}\beta_j+\sum_{i=1}^{n}\gamma_i+\min_{\text{by all } y,x}\{:\}\right\}.$$

However to keep the:

$$\text{opt}(\text{PR}(\alpha,\beta,\gamma)) \ \leq \ \text{opt}(P)$$

We need to penalize breaking the constraints.
That will cause the found $\min\limits_{\text{by all } y,x\geq 0}$ to not break any constraint.
We have to make it impossible to achieve minimum when breaking the constraint.
And to do that each dual variable must be chosen so that **violating a original constraint increases the value of the objective**.

For the constraints of type $\alpha \Rightarrow \sum_{j=1}^{m} a_{ij}x_{ij} \leq b_i y_i$

- The violation measure is
  $b_i y_i - \sum_{j=1}^{m} a_{ij}x_{ij}$.
- When **broken**, this becomes **negative**.
- To penalize the objective it requires
  $\alpha_i \leq 0$.

For the constraints of type $\gamma \Rightarrow y_i \leq 1$:

- The violation measure is
  $1 - y_i$.
- When **breaking** this becomes **negative**.
- To ensure the penalty, we need
  $\gamma_i \leq 0$.

For the constraints of type $\beta \Rightarrow \sum_{i=1}^{n} x_{ij} = 1$:

- Violations can happen **in both ways**.
- Therefore we set: $\beta_j \in \mathbb{R}$.

This ensures that always:

$$\text{opt}(\text{PR}(\alpha,\beta,\gamma)) \leq \text{opt}(P)$$

Combinig everything togheter we are left with:

$$
\begin{aligned}
\max \ & \sum_{j=1}^{m} \beta_j + \sum_{i=1}^{n} \gamma_i \\
& \alpha_1 b - \gamma_1 \geq -c_1 \\
& \alpha_2 b - \gamma_2 \geq -c_2 \\
& \vdots \\
& -\alpha_n b - \gamma_n \geq 0 \\
& -\alpha_n a_1 - \beta_1 \geq 0 \\
& -\alpha_2 a_1 - \beta_2 \geq 0 \\
& \vdots \\
& -\alpha_n a_1 - \beta_n \geq 0 \\
& -\alpha_1 a_2 - \beta_1 \geq 0 \\
& -\alpha_2 a_2 - \beta_2 \geq 0 \\
& \vdots \\
& -\alpha_n a_m - \beta_n \geq 0 \\
\\
& \alpha_i \leq 0. \quad i = 1, \dots, n \\
& \beta_j \in \mathbb{R}. \quad j = 1, \dots, m \\
& \gamma_i \leq 0. \quad i = 1, \dots, n
\end{aligned}
$$

$\Rightarrow$
organized as

$$
\begin{aligned}
\max \ & \sum_{j=1}^{m} \beta_j + \sum_{i=1}^{n} \gamma_i \\
\\
& -b\alpha_i + \gamma_i \leq c_i \quad i = 1, \dots, n \\
& \alpha_i a_j + \beta_j \leq 0 \quad i = 1, \dots, n \qquad j = 1, \dots, m \\
\\
& \alpha_i \leq 0. \quad i = 1, \dots, n \\
& \beta_j \in \mathbb{R}. \quad j = 1, \dots, m \\
& \gamma_i \leq 0. \quad i = 1, \dots, n
\end{aligned}
$$

Q.E.D.

# Task 5

## Subtask 5.a

First write original problem with changed $\min() \Rightarrow -\max -()$ (and with slacks):

$$c = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad A = \begin{bmatrix} -3 & 7 & 1 & 0 \\ 1 & -4 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} -21 \\ 4 \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

So far $B = 3, 4$ Let's prepare for Revised Simplex:

$$B = 1, 2$$
$$A_B = \begin{bmatrix} -3 & 7 \\ 1 & -4 \end{bmatrix}$$
$$A_B^{-1} = \frac{1}{det(A_B)} \begin{bmatrix} -4 & -7 \\ -1 & -3 \end{bmatrix} = \begin{bmatrix} -\frac{4}{5} & -\frac{7}{5} \\ -\frac{1}{5} & -\frac{3}{5} \end{bmatrix}$$

*(by using the formula for the inverse of 2x2 matrix, and det() of 2x2 matrix)*

$$A_N = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$
$$c_B^T = \begin{bmatrix} -1 & -1 \end{bmatrix}$$
$$c_N^T = \begin{bmatrix} 0 & 0 \end{bmatrix}$$

To check whether or not the solution with this basis is optimal we just need to compute $c_n^T - c_B^T A_B^{-1} A_N$ (new reduced costs) which in our case can be easily done by hand:

$$\begin{bmatrix} 0 & 0 \end{bmatrix} - \begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} -\frac{4}{5} & -\frac{7}{5} \\ -\frac{1}{5} & -\frac{3}{5} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} =$$
$$\begin{bmatrix} 0 & 0 \end{bmatrix} - \begin{bmatrix} (\frac{4}{5} + \frac{1}{5}) & (\frac{7}{5} + \frac{3}{5}) \end{bmatrix} =$$
$$\begin{bmatrix} 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 2 \end{bmatrix} =$$
$$\begin{bmatrix} -1 & -2 \end{bmatrix}$$

As we can see all reduced costs $\leq 0$, therefore solution is optimal

### Subtask 5.b

Firstly we need to calculate the values $x_1, x_2, s_1, s_2$ in the optimal solution this can be done by calculatng $A_B^{-1}b$:

$$A_B^{-1}b = \begin{bmatrix} -\frac{4}{5} & -\frac{7}{5} \\ -\frac{1}{5} & -\frac{3}{5} \end{bmatrix} \begin{bmatrix} -21 \\ 4 \end{bmatrix} = \begin{bmatrix} \frac{56}{5} \\ \frac{9}{5} \end{bmatrix}, \quad \text{optimal basis was } B = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

Therefore $x_1 = 11\frac{1}{5}$ and $x_2 = 1\frac{4}{5}$. for convinience not that $s_1 = x_3$ , $s_2 = x_4$ Let's calculate new non basic $A$

$$A_{N_{new}} = A_B^{-1}A_N = \begin{bmatrix} -\frac{4}{5} & -\frac{7}{5} \\ -\frac{1}{5} & -\frac{3}{5} \end{bmatrix}$$

Both of the solution $x_1, x_2$ fractional, we provide Gomory cuts for both:

$$\text{row } u = 1, 2$$

$$\sum_{j \in N_{new}} (\bar{a}_{u,j} - \lfloor \bar{a}_{u,j} \rfloor) x_j \ \geq \ \bar{b}_u - \lfloor \bar{b}_u \rfloor$$

Gomory cut for $u = 1$

$$(-\frac{4}{5} - \lfloor -\frac{4}{5} \rfloor)s_1 + (-\frac{9}{5} - \lfloor -\frac{9}{5} \rfloor)s_1 \geq \frac{56}{5} - \lfloor \frac{56}{5} \rfloor$$

$$\frac{1}{5}s_1 + \frac{3}{5}s_2 \geq \frac{1}{5}$$

$$s_1 + 3s_2 \geq 1$$

Gomory cut for $u = 2$

$$(-\frac{1}{5} - \lfloor -\frac{1}{5} \rfloor)s_1 + (-\frac{3}{5} - \lfloor -\frac{3}{5} \rfloor)s_1 \geq \frac{9}{5} - \lfloor \frac{9}{5} \rfloor$$

$$\frac{4}{5}s_1 + \frac{2}{5}s_2 \geq \frac{4}{5}$$

$$4s_1 + 2s_2 \geq 4$$

We can express them in terms of the original variables by substitution, using the equalities from standard form:

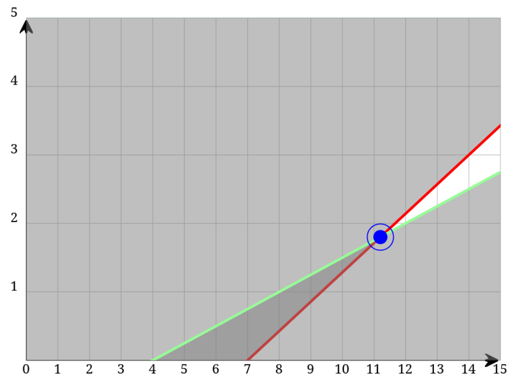
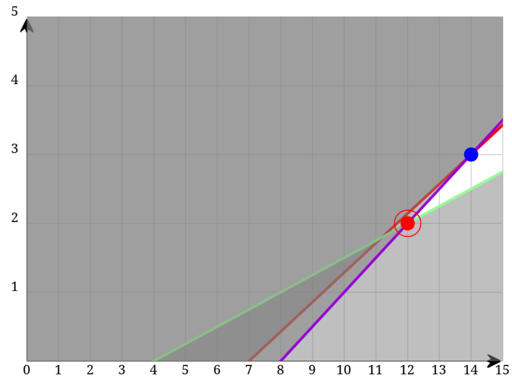
$$-3x_1 + 7x_2 + s_1 = -21$$
$$x_1 - 4x_2 + s_2 = 4$$

$$s_1 = -21 + 3x_1 - 7x_2$$
$$s_2 = 4 - x_1 + 4x_2$$

| | |
|---|---|
| Gomory cut for $u = 1$ | Gomory cut for $u = 2$ |
| $s_1 + 3s_2 \geq 1$ | $4s_1 + 2s_2 \geq 4$ |
| $-21 + 3x_1 - 7x_2$<br>$+12 - 3x_1 + 12x_2 \geq 1$ | $-84 + 12x_1 - 28x_2$<br>$+8 - 2x_1 + 8x_2 \geq 4$ |
| $-9 + 5x_2 \geq 1$ | $-76 + 10x_1 - 20x_2 \geq 4$ |
| $x_2 \geq -2$ | $x_1 - 2x_2 \geq 8$ |

## Subtask 5.c

Website was used as a visualizer
$x_2 \geq -2$ is not visible as it's weaker than base constraints $x_1, x_2 \geq 0$

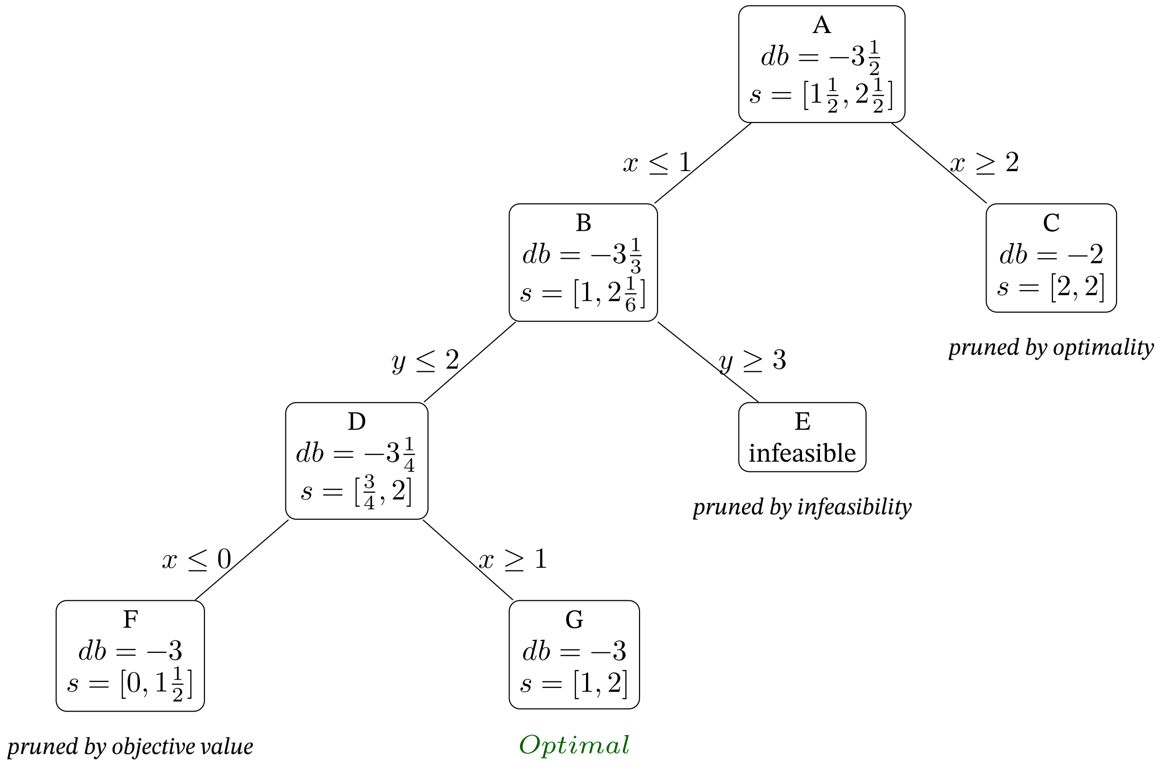
| without Gomory cuts | with Gomory cuts |
|---|---|
|   |   |
| (56/5, 9/5) Minimum | (14, 3) (12, 2) Minimum |

# Task 6

## Subtask 6.a

$$
\begin{aligned}
\min\ & x - 2y \\
\text{s.t.}\ & -4x + 6y \leq 9 \\
& x + y \leq 4 \\
& x, y \in \mathbb{N}
\end{aligned}
$$

As the problem has only 2 variables website was used as a visualizer and solver.
Nodes were opened in alphabetical order (i.e. A $\rightarrow$ B $\rightarrow$ C ...)



# Task 7

## Subtask 7.a

Firstly let us note *Hamming Distance* $H_d$ between two words $w_i, w_j$ of lenght $d$
i.e. $w_i, w_j \in \{\mathbf{0}, \mathbf{1}\}^d$

$$H_d(w_i, w_j) = \sum_{k=1}^{d} \mathbf{1}(w_i[k] \neq w_j[k])$$

$k$ being the index of a word being checked and **1** indicator function

Our problem can then be expressed as:

$$\max \left\{ \min \left\{ H_d(w_i, w_j) \right\} \right\} \text{ for } i, i \in \{1 \cdots N\}$$

across all $w_i \neq w_j$ of $|w_i| = |w_j|$

We transform it into *Mathematical Programming form* by first denoting:

$$M_d = \min \left\{ H_d(w_i, w_j) \right\} \text{ for } i, i \in \{1 \cdots N\}$$

And subsequently transforming the formulation into:

$$\begin{aligned} &\max M_d \\ &\sum_{k=1}^{d} \mathbf{1}(w_i[k] \neq w_j[k]) \geq M_d \\ &\text{for } 1 \leq i < j \leq N \end{aligned}$$

To make the constraints more workable:

$$\begin{aligned} &\max M_d \\ &\sum_{k=1}^{d} |w_i[k] - w_j[k]| \geq M_d \\ &\text{for } 1 \leq i < j \leq N \\ &\underset{j,k,i}{\forall} \; w_i[k], w_j[k] \in \{\mathbf{0}, \mathbf{1}\} \end{aligned}$$

This formulation has exactly $\frac{N(N-1)}{2}$ constraints (all unique pairs) and $N \cdot d$ variables ($N$ words of lenght $d$)
We would like however to get rid of not explicitily linear absolute value in the constraints. We can do that by introductiong another variable that will replace it, let's call it $\gamma$:

$$\gamma_{i,j,k} \approx |w_i[k] - w_j[k]|$$

We have to bound it so that in all possible situations it is forced to hold a correct value:

$$\begin{aligned}
\gamma_{i,j,k} &\leq 2 - (w_i[k] + w_j[k]) \\
\gamma_{i,j,k} &\leq w_i[k] + w_j[k] \\
\gamma_{i,j,k} &\geq w_i[k] - w_j[k] \\
\gamma_{i,j,k} &\geq w_j[k] - w_i[k]
\end{aligned}$$

We can se that using those we can replace absolute value in all cases:

$$\begin{array}{ll}
w_i[k], w_j[k] = (0,0) & w_i[k], w_j[k] = (1,1) \\
& \\
\gamma_{i,j,k} \leq 2 & \gamma_{i,j,k} \leq 0 \\
\gamma_{i,j,k} \leq 0 & \gamma_{i,j,k} \leq 2 \\
\gamma_{i,j,k} \geq 0 & \gamma_{i,j,k} \geq 0 \\
\gamma_{i,j,k} \geq 0 & \gamma_{i,j,k} \geq 0 \\
\Rightarrow \gamma_{i,j,k} = 0 & \Rightarrow \gamma_{i,j,k} = 0 \\
w_i[k], w_j[k] = (0,1) & w_i[k], w_j[k] = (1,0) \\
& \\
\gamma_{i,j,k} \leq 1 & \gamma_{i,j,k} \leq 1 \\
\gamma_{i,j,k} \leq 1 & \gamma_{i,j,k} \leq 1 \\
\gamma_{i,j,k} \geq -1 & \gamma_{i,j,k} \geq 1 \\
\gamma_{i,j,k} \geq 1 & \gamma_{i,j,k} \geq -1 \\
\Rightarrow \gamma_{i,j,k} = 1 & \Rightarrow \gamma_{i,j,k} = 1
\end{array}$$

Combinig that our new formulation looks as follows:

$$\begin{aligned}
&\max M_d \\
&\left\{ \sum_{k=1}^{d} \gamma_{i,j,k} \geq M_d \right\} \quad 1 \leq i < j \leq N \\
&\left\{ \begin{aligned}
\gamma_{i,j,k} &\leq 2 - (w_i[k] + w_j[k]) \\
\gamma_{i,j,k} &\leq w_i[k] + w_j[k] \\
\gamma_{i,j,k} &\geq w_i[k] - w_j[k] \\
\gamma_{i,j,k} &\geq w_j[k] - w_i[k]
\end{aligned} \right\} \begin{array}{l} 1 \leq i < j \leq N \\ \\ 1 \leq k \leq d \end{array} \\
&\underset{j,k,i}{\forall} \; \gamma_{i,j,k} \in \mathbb{R} \\
&\underset{j,k,i}{\forall} \; w_i[k], w_j[k] \in \{\mathbf{0}, \mathbf{1}\}
\end{aligned}$$

That yields a total of ($\frac{N(N-1)}{2} + 4d\frac{N(N-1)}{2}$) constraints and ($N \cdot d + d\frac{N(N-1)}{2}$) variables ($w_i[k] + \gamma_{i,j,k}$)

### Subtask 7.b

$$\max M_d$$

$$\begin{aligned}
&\sum_{k=1}^{d} \gamma_{i,j,k} \geq M_d && 1 \leq i < j \leq N \\
&\gamma_{i,j,k} \leq 2 - (w_i[k] + w_j[k]) && \\
&\gamma_{i,j,k} \leq w_i[k] + w_j[k] && 1 \leq i < j \leq N \\
&\gamma_{i,j,k} \geq w_i[k] - w_j[k] && 1 \leq k \leq d \\
&\gamma_{i,j,k} \geq w_j[k] - w_i[k] &&
\end{aligned}$$

$$\begin{aligned}
&\underset{j,k,i}{\forall}\ \gamma_{i,j,k} \in \mathbb{R} \\
&\underset{j,k,i}{\forall}\ w_i[k], w_j[k] \in \{\mathbf{0}, \mathbf{1}\}
\end{aligned}$$

In the final formulation the problem belongs to the **Mixed Integer Lineary Programming** family as its constraints are all linear and some of the variables ($\gamma$) can be real, the other ($w_i[k]$) integer. It's worth noting however that all integer variables here are **binary** therefore it's more of a *Mixed Binary Lineary Programming*, which *might* be a bit easier to solve than regular *Integer* programming.

# Task 8

## Subtask 8.a

### Subtask 8.b

### Subtask 8.c

**Subtask 8.d**

**Subtask 8.e**

**Subtask 8.f**