Optimal Decision Making

Group Project

Eugénie Decaux - 346450
Ahmed Reda Seghrouchni - 297848
Valentin Vuillon - 315300

Phase 3:

Ph3-Task1:

Let's construct the payoff matrix $A \in \mathbb{R}^{m \times n}$ corresponding to Figure 1:

1. Actions of the two players:

- *Player 1 (TA)* inspects exactly one row or one column of the 6×6 board.

$$\text{Pure actions: } \{R_1, R_2, \dots, R_6\} \cup \{C_1, C_2, \dots, C_6\}, \quad m = 12.$$

- *Player 2 (Students)* hides the coin in one of the marked (red) cells. Numbering these from top-left in reading order gives

$$\begin{aligned} S_1 &= (1,2), \\ S_2 &= (3,1), \\ S_3 &= (3,4), \\ S_4 &= (3,6), \\ S_5 &= (4,2), \\ S_6 &= (5,2), \\ S_7 &= (5,3), \\ S_8 &= (5,5), \\ S_9 &= (6,1). \end{aligned}$$

2. Payoff definition: We define A_{ij} as the reward of Player 1 (TA) when the TA plays its i-th action and the students hide in cell S_j:

$$A_{ij} = \begin{cases} 1, & \text{if the inspected row/column } i \text{ contains cell } S_j, \\ 0, & \text{otherwise.} \end{cases}$$

Since it is a zero-sum game, the students' payoff is $-A_{ij}$.

3. The explicit matrix: When we index the TA's actions in the order $R_1, \dots, R_6, C_1, \dots, C_6$

and the student's in $S_1, \ldots, S_9$. Then:

$$A = \begin{array}{c|ccccccccc}
 & S_1 & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 & S_8 & S_9 \\ \hline
R_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
R_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
R_3 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
R_4 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
R_5 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
R_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline
C_1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
C_2 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
C_3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
C_4 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
C_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
C_6 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{array}$$

This 12×9 0–1 matrix completely describes the one-round payoff of the TA as a function of both players' pure moves.

Ph3-Task2:

Assuming the TA (Player 1) can find a mixed strategy $\hat{x} \in \Delta_m$ such that

$$\forall\, y \in \Delta_n, \quad \hat{x}^\top A\, y \;\geq\; \ell.$$

We will show that the value of the game,

$$p^\star \;=\; \max_{x \in \Delta_m} \min_{y \in \Delta_n} x^\top A\, y,$$

satisfies $p^\star \geq \ell$.

We proceed in four clear steps:

1) **Definition of the value:** By the minimax characterization of zero-sum games, the value $p^\star$ is
$$p^\star \;=\; \max_{x \in \Delta_m} \underbrace{\min_{y \in \Delta_n} \; x^\top A\, y}_{f(x)}\,. \tag{1}$$
Here $f(x)$ denotes the worst-case payoff Player 1 secures when Player 1 plays x.

2) **Understanding the hypothesis:** The hypothesis "$\forall\, y,\ \hat{x}^\top A\, y \geq \ell$" exactly means
$$f(\hat{x}) = \min_{y \in \Delta_n} \hat{x}^\top A\, y \;\geq\; \ell.$$
In words, even if the students respond adversarially by choosing the worst possible y, the TA still earns at least ℓ.

3) **Monotonicity of the maximum:** Since $p^\star$ is defined as the max of $f(x)$ over all $x \in \Delta_m$, it must satisfy

$$p^\star = \max_{x\in\Delta_m} f(x) \;\geq\; f(\hat{x}).$$

Indeed, the maximum of a set of real numbers is never smaller than any particular member of the set.

4) **Combining the two inequalities:** From step 2 we have $f(\hat{x}) \geq \ell$, and from step 3 that $p^\star \geq f(\hat{x})$. Hence by transitivity,

$$p^\star \;\geq\; f(\hat{x}) \;\geq\; \ell.$$

So Player 1's existence of a strategy guaranteeing ℓ implies the game value $p^\star$ cannot be below ℓ.

Ph3-Task3:

Let C be a line-cover: a set of rows and columns such that every marked cell is covered by at least one line in C. The TA's strategy is to choose a line $\ell \in C$ uniformly at random:

$$x_\ell = \frac{1}{|C|}, \quad \forall\, \ell \in C, \quad x_i = 0 \text{ for } i \notin C.$$

We will show this guarantees an expected payoff of at least $\frac{1}{|C|}$ per round, and conclude $p^\star \geq 1/|C|$:

1) **Expected payoff against an arbitrary student strategy:**
Fix any mixed hiding strategy $y \in \Delta_n$. Then the TA's expected payoff is

$$x^\top A\, y \;=\; \sum_{i=1}^{m} x_i\,(A_{i,:}\, y) \;=\; \sum_{\ell\in C} \frac{1}{|C|} \sum_{j=1}^{n} A_{\ell j}\, y_j.$$

Here $A_{\ell j} = 1$ precisely if line ℓ covers cell j.

2) **Using the covering property:**
Because C covers *every* marked cell, for each $j = 1, \dots, n$ there is at least one $\ell \in C$ with $A_{\ell j} = 1$. Hence

$$\sum_{\ell\in C} A_{\ell j} \;\geq\; 1, \quad \forall\, j.$$

Multiply by $y_j \geq 0$ and sum over j:

$$\sum_{\ell\in C}\sum_{j=1}^{n} A_{\ell j}\, y_j = \sum_{j=1}^{n} \Big(\sum_{\ell\in C} A_{\ell j}\Big) y_j \;\geq\; \sum_{j=1}^{n} 1 \cdot y_j = 1.$$

3) **Lower-bounding the TA's payoff:**
Substituting back,

$$x^\top A\, y = \frac{1}{|C|} \sum_{\ell\in C}\sum_{j=1}^{n} A_{\ell j}\, y_j \;\geq\; \frac{1}{|C|} \times 1 = \frac{1}{|C|}.$$

Thus for *every* $y \in \Delta_n$, $x^\top A\, y \geq \frac{1}{|C|}$.

4) **Conclusion via Task 2:**
By the result of Ph3-Task2, any strategy x satisfying $\min_y x^\top A\,y \geq \ell$ forces $p^\star \geq \ell$. Here we have shown $\min_y x^\top A\,y \geq \frac{1}{|C|}$. Therefore

$$p^\star \;\geq\; \frac{1}{|C|}.$$

Ph3-Task4:

Suppose the students (Player 2) can find a mixed strategy $\hat{y} \in \Delta_n$ such that

$$\forall\, x \in \Delta_m, \quad x^\top A\,\hat{y} \;\leq\; u.$$

We will prove that the value of the game,

$$p^\star \;=\; \min_{y \in \Delta_n} \max_{x \in \Delta_m} x^\top A\,y,$$

satisfies $p^\star \leq u$.

Let's again break the argument into four clear steps:

1) **Alternative definition of the value:** By the minimax theorem (or by the symmetry of a zero-sum game), one may equally write

$$p^\star \;=\; \min_{y \in \Delta_n} \underbrace{\max_{x \in \Delta_m} x^\top A\,y}_{g(y)}.$$

Here $g(y)$ is the worst-case loss the students suffer if they commit to y: the TA will then pick the row/column x that maximizes $x^\top A\,y$.

2) **Interpretation of the assumption:** The assumption "$\forall\, x,\ x^\top A\,\hat{y} \leq u$" is precisely equivalent to:

$$\max_{x \in \Delta_m} x^\top A\,\hat{y} \;\leq\; u,$$

since the maximum of a set of numbers is at most u exactly when each member of the set is at most u.

3) **Monotonicity of the minimum:** Because $p^\star$ is the min of $g(y) = \max_x x^\top A\,y$ over all $y \in \Delta_n$, it follows that

$$p^\star = \min_{y \in \Delta_n} g(y) \;\leq\; g(\hat{y}).$$

Indeed, the minimum of a collection of real values cannot exceed the value at any particular choice $y = \hat{y}$.

4) **Combining the inequalities:** From step 2 we have $g(\hat{y}) \leq u$, and from step 3 that $p^\star \leq g(\hat{y})$. Hence by transitivity,

$$p^\star \;\leq\; g(\hat{y}) \;\leq\; u.$$

Thus the existence of a students' strategy $\hat{y}$ that keeps the catch-probability at most u forces the game value $p^\star$ to be no greater than u.

Ph3-Task5:

Let $M \subseteq \{\text{marked cells}\}$ be a *pairing*, i.e. no two cells in M lie in the same row or column. Suppose the students adopt the strategy

$$y_j \;=\; \begin{cases} \dfrac{1}{|M|}, & S_j \in M, \\ 0, & S_j \notin M. \end{cases}$$

We will show that for *any* TA strategy $x \in \Delta_m$, the expected catch-probability per round $\,x^\top A\,y$ is at most $\frac{1}{|M|}$. From there, we will conclude $p^\star \le 1/|M|$:

Let's proceed with the proof:

1) **Fix an arbitrary TA strategy:**
   Let $x = (x_1, \ldots, x_m) \in \Delta_m$ be any mixed strategy of the TA. By linearity of expectation,

$$\underbrace{x^\top A\,y}_{\text{[caught]}} \;=\; \sum_{i=1}^{m} x_i \left(A_{i,:}\,y\right) \;=\; \sum_{i=1}^{m} x_i \sum_{j=1}^{n} A_{ij}\,y_j.$$

   Here $A_{ij} = 1$ exactly when line i (row or column) would catch a coin hidden in cell j.

2) **Evaluate the inner sum for each pure action:**
   Since $y_j > 0$ only for $S_j \in M$, write

$$\sum_{j=1}^{n} A_{ij}\,y_j \;=\; \sum_{S_j \in M} A_{ij}\,\frac{1}{|M|} \;=\; \frac{1}{|M|}\sum_{S_j \in M} A_{ij}.$$

   But M is a pairing: no two cells in M lie in the same row or column. Hence each row or column i can cover *at most one* cell from M. Consequently

$$\sum_{S_j \in M} A_{ij} \;\le\; 1, \quad \text{for every } i.$$

   Therefore

$$\sum_{j=1}^{n} A_{ij}\,y_j \;\le\; \frac{1}{|M|}.$$

3) **Combine to bound the overall expectation:**
   Substituting back,

$$x^\top A\,y \;=\; \sum_{i=1}^{m} x_i \sum_{j=1}^{n} A_{ij}\,y_j \;\le\; \sum_{i=1}^{m} x_i\,\frac{1}{|M|} \;=\; \frac{1}{|M|}\sum_{i=1}^{m} x_i \;=\; \frac{1}{|M|},$$

   since $\sum_i x_i = 1$. This shows that no matter how the TA mixes its inspections, the catch-probability is at most $1/|M|$.

4) **Conclusion:**
   By the symmetry of the minimax theorem, the value of the zero-sum game can also be

written $p^\star = \min_y \max_x x^\top A\, y$. The above argument exhibits one particular y (namely the uniform distribution over M) for which $\max_x x^\top A\, y \leq 1/|M|$. Hence

$$p^\star \;=\; \min_y \max_x x^\top A\, y \;\leq\; \max_x x^\top A\,\bigl(y \text{ on } M\bigr) \;\leq\; \frac{1}{|M|}.$$

We therefore proved that $p^\star \leq 1/|M|$.

Ph3-Task6:

We will first find a minimum-size line–cover and a maximum-size pairing, and thereby pin down $p^\star$. Then compute the expected loss over 100 rounds and interpret.

1. First recall the board's marked cells:

$$\{(1,2),\ (3,1),(3,4),(3,6),\ (4,2),\ (5,2),(5,3),(5,5),\ (6,1)\}.$$

2. Minimum-size line–cover:
From Figure 2 we see the TA can cover all marked cells with the four lines

$$C_{\min} = \{\, R_3,\ R_5,\ C_1,\ C_2\}.$$

One checks by inspection that:

- R_3 covers $(3,1),(3,4),(3,6)$.
- R_5 covers $(5,2),(5,3),(5,5)$.
- C_1 covers $(3,1),(6,1)$.
- C_2 covers $(1,2),(4,2),(5,2)$.

No cover of size 3 exists, since each line covers at most 3 cells and there are 9 total. Thus

$$|C_{\min}| = 4.$$

3. Maximum-size pairing:
From Figure 3 we exhibit a pairing of size 4:

$$M_{\max} = \{(1,2),\ (3,4),\ (5,3),\ (6,1)\}.$$

Each lies in a distinct row and distinct column. By the pigeonhole constraint, one cannot find 5 such cells on a 6×6 grid with these marked positions. Hence

$$|M_{\max}| = 4.$$

4. Exact value:
By Ph3-Task3 we have $p^\star \geq 1/|C_{\min}| = 1/4$. By Ph3-Task5 we have $p^\star \leq 1/|M_{\max}| = 1/4$. Combining gives

$$\boxed{p^\star = \frac{1}{4}.}$$

5. Now let's move to the expected total loss over 100 rounds:
Since in each round the students lose on average $p^\star = 1/4$ coins, over 100 independent rounds the total expected loss is:

$$100 \times \frac{1}{4} \;=\; 25 \quad \text{coins (CHF).}$$

This exactly matches the 25 CHF the students were given at the start.

This concludes that the game is *fair*: if both sides play optimally, the students lose 25 CHF in expectation, so they end with zero net gain or loss. The TA recovers exactly the vanished 25 CHF, and neither side has an advantage beyond the prescribed budget.

Phase 4:

Ph4-Task1:

Let the bipartite graph have vertex set $V = X \cup Y = \{v_1, \ldots, v_m\}$ and edge set $E = \{e_1, \ldots, e_n\}$.

Variables

Let $x_j \in \{0, 1\}$ for each edge $e_j \in E$, where:

$$x_j = \begin{cases} 1, & \text{if edge } e_j \text{ is in the matching} \\ 0, & \text{if edge } e_j \text{ is not in the matching} \end{cases}$$

Objective Function

We want to maximize the number of edges in the matching. The objective function is thus:

$$\text{Maximize} \sum_{e_j \in E} x_j$$

Constraints

For each vertex $v_i \in X \cup Y$, we ensure that at most one edge incident to v_i is in the matching. Let $\delta(v_i)$ be the set of edges incident to vertex v_i. The constraint for each vertex is:

$$\sum_{e_j \in \delta(v_i)} x_j \leq 1, \quad \forall v_i \in X \cup Y$$

The variable type constraint is given by:

$$x_j \in \{0, 1\}, \quad \forall e_j \in E$$

Integer Program

The complete Integer Program is:

$$\begin{aligned} \text{Maximize} & \sum_{e_j \in E} x_j \\ \text{Subject to} & \sum_{e_j \in \delta(v_i)} x_j \leq 1, \quad \forall v_i \in X \cup Y \\ & x_j \in \{0, 1\}, \quad \forall e_j \in E \end{aligned}$$

Explanation of Constraints

The constraint $\sum_{e_j \in \delta(v_i)} x_j \leq 1$ for each vertex v_i ensures that no more than one edge in the matching is connected to any single vertex.

- $\delta(v_i)$ represents all the edges that touch vertex v_i.
- $\sum_{e_j \in \delta(v_i)} x_j$ sums the variables x_j for all these edges. If an edge is in the matching, its corresponding x_j is 1; otherwise, it's 0.

- By requiring this sum to be less than or equal to 1, we allow either zero or one edge to be chosen from all the edges touching v_i. This directly enforces the definition of a matching, where no two edges can share a vertex.

Relaxation

In the context of the maximum matching problem, the integrality constraint $x_j \in \{0, 1\}$ can actually be relaxed to $x_j \in [0, 1]$ or even $x_j \in \mathbb{Z}$ without affecting the nature of the optimal solution. This is because the constraints of the problem naturally force the variables to take integer values. For instance, if a variable x_j were to take a fractional value like 0.5, it would mean that "half" of an edge is included in the matching, which doesn't make sense—an edge is either in the matching or it's not. So, the structure of the problem ensures that only 0 or 1 are valid. Moreover, if the linear program has an optimal solution, it will have one that is integral. This is mainly due to the fact that, in the case of the maximum matching problem, the constraint matrix is totally unimodular, which guarantees that the solution to the linear relaxation will always be integral.

Ph4-Task2:

Variables

For each vertex $v_i \in X \cup Y$, we define a binary variable $y_i \in \{0, 1\}$ as follows:

$$y_i = \begin{cases} 1, & \text{if vertex } v_i \text{ is in the vertex cover} \\ 0, & \text{if vertex } v_i \text{ is not in the vertex cover} \end{cases}$$

Objective Function

We want to minimize the number of vertices in the vertex cover. The objective function is thus:

$$\text{Minimize} \sum_{v_i \in X \cup Y} y_i$$

Constraints

For each edge $e_j = \{v_i, v_k\} \in E$, we ensure that at least one of its endpoints is in the vertex cover:

$$y_i + y_k \geq 1, \quad \forall e_j = \{v_i, v_k\} \in E$$

The variable type constraint is given by:

$$y_i \in \{0, 1\}, \quad \forall v_i \in X \cup Y$$

Integer Program

The complete Integer Program is:

$$\begin{aligned} \text{Minimize} \; & \sum_{v_i \in X \cup Y} y_i \\ \text{Subject to} \; & y_i + y_k \geq 1, \quad \forall \{v_i, v_k\} \in E \\ & y_i \in \{0, 1\}, \quad \forall v_i \in X \cup Y \end{aligned}$$

Explanation of Constraints

The constraint $y_i + y_k \geq 1$ for each edge $\{v_i, v_k\} \in E$ enforces the vertex cover condition. For any edge, it requires that the variable corresponding to at least one of its endpoints is equal to 1, meaning at least one of the endpoints is included in the vertex cover. This ensures that every edge is "covered" by at least one vertex in the chosen set.

Relaxation

In the minimum vertex cover problem, the integrality constraint $y_i \in \{0,1\}$ can be relaxed to $y_i \in [0,1]$ or even $y_i \in \mathbb{Z}$ without changing the nature of the optimal solution.

While it seems like relaxing to $y_i \in [0,1]$ might allow for fractional solutions, the structure of the constraints ensures that in an optimal solution, the variables will still take on integer values (0 or 1). If a variable were to take a fractional value (e.g., 0.5), it wouldn't make sense in the context of choosing a whole vertex. The constraints push the variables towards integer solutions. In some cases, the LP relaxation of the vertex cover problem has an integer optimal solution.

Ph4-Task3

We will demonstrate that the LP relaxations of the Maximum Matching and Minimum Vertex Cover problems form a primal-dual pair, and by strong duality, their optimal values are equal.

0. Standard Primal–Dual LP Template

We recall the classical primal–dual pair of linear programs as commonly presented in duality theory:

$$\begin{aligned} \textbf{Primal :} &\quad \min\left\{\mathbf{c}^T\mathbf{x} \;\middle|\; A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq 0\right\} \\ \textbf{Dual :} &\quad \max\left\{\mathbf{b}^T\mathbf{p} \;\middle|\; A^T\mathbf{p} \leq \mathbf{c},\ \mathbf{p} \in \mathbb{R}^m\right\} \end{aligned}$$

This standard form assumes equality constraints in the primal and unrestricted signs in the dual variables. In our problem setting, we instead use inequalities in the primal — hence we slightly depart from this canonical form, but the duality correspondence remains valid with appropriate adjustments.

1. LP Relaxation Formulations

Primal: Minimum Vertex Cover LP Relaxation

$$\begin{aligned} \text{Minimize:} &\quad \sum_{v_i \in X \cup Y} y_i \\ \text{Subject to:} &\quad y_i + y_k \geq 1, \quad \forall \{v_i, v_k\} = e_j \in E \\ &\quad y_i \geq 0, \quad \forall v_i \in X \cup Y \end{aligned}$$

Here:

- y_i indicates whether vertex v_i is selected in the vertex cover.

Dual: Maximum Matching LP Relaxation

$$\begin{aligned}
\text{Maximize: } & \sum_{e_j \in E} x_j \\
\text{Subject to: } & \sum_{e_j \in \delta(v_i)} x_j \leq 1, \quad \forall v_i \in X \cup Y \\
& x_j \geq 0, \quad \forall e_j \in E
\end{aligned}$$

Here:

- x_j represents whether edge e_j is selected in the matching.
- $\delta(v_i)$ is the set of edges incident to vertex v_i.

2. Matrix Form Representation

Let $A \in \{0,1\}^{m \times n}$ be the vertex-edge incidence matrix of the graph $G = (X \cup Y, E)$. Then:

Primal LP in Matrix Form

$$\begin{aligned}
\text{Minimize: } & \mathbf{1}^T \mathbf{y} \\
\text{Subject to: } & A^T \mathbf{y} \geq \mathbf{1} \\
& \mathbf{y} \geq \mathbf{0}
\end{aligned}$$

Dual LP in Matrix Form

$$\begin{aligned}
\text{Maximize: } & \mathbf{1}^T \mathbf{x} \\
\text{Subject to: } & A\mathbf{x} \leq \mathbf{1} \\
& \mathbf{x} \geq \mathbf{0}
\end{aligned}$$

3. Duality Correspondence

We observe the standard primal-dual mapping:

- Primal objective coefficients $\mathbf{c} = \mathbf{1}$ become the right-hand side $\mathbf{b}$ of the dual constraints.
- Primal RHS $\mathbf{b} = \mathbf{1}$ becomes the objective coefficients $\mathbf{c}$ of the dual.
- The matrix A in the primal becomes transposed A^T in the dual.
- Variables $\mathbf{x}$ and $\mathbf{y}$ are distinct.

Thus, the matching LP is the primal, and the vertex cover LP is its dual.

4. Strong Duality Application

By the Strong Duality Theorem of linear programming: if both the primal and dual have feasible solutions, then both have optimal solutions, and the corresponding optimal objective values are equal.

Conclusion: Since both LPs (matching and vertex cover relaxations) are feasible and bounded, their optimal values coincide:

$$\text{Optimal Matching Size} = \text{Minimum Vertex Cover Size}$$

This duality underpins the equality observed in König's Theorem and supports our previous game-theoretic conclusions.

Ph4-Task4:

Let A be a totally unimodular matrix, and let $\bar{A}$ be the matrix obtained by appending a unit vector e_i as a new last column. We want to show that $\bar{A}$ is also totally unimodular.

By definition, a matrix is totally unimodular if every square submatrix of that matrix has a determinant in $\{0, \pm 1\}$.

Let B be any square submatrix of $\bar{A}$. We consider two cases:

1. **Case 1:** If B does not contain the appended column e_i, then B is a square submatrix of A. Since A is totally unimodular, we have $\det(B) \in \{0, \pm 1\}$.
2. **Case 2:** If B contains the appended column e_i, then B can be written as $B = [B', e_i]$, where B' is a submatrix of A with one fewer column than B. To compute the determinant of B, we can perform a cofactor expansion along the column e_i. Since e_i is a unit vector, it has only one non-zero entry, which is 1 at the i-th position. Therefore, the determinant of B is equal to ± 1 times the determinant of a square submatrix of A'. Since A is totally unimodular, this determinant is in $\{0, \pm 1\}$, and thus $\det(B) \in \{0, \pm 1\}$.

In both cases, the determinant of any square submatrix B of $\bar{A}$ is in $\{0, \pm 1\}$. Therefore, $\bar{A}$ is also totally unimodular.

Ph4-Task5:

Considering the linear program

$$\begin{aligned} \max_{x \in \mathbb{R}^n} \ & c^\top x \\ \text{s.t. } Ax \ &\le \ b, \\ x \ &\ge \ 0, \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$ is totally unimodular and $b \in \mathbb{Z}^m$. We will show that if this LP attains an optimum, then there exists an optimal solution $x^* \in \mathbb{Z}^n$. (We will use the hint about basic feasible solutions (BFS))

1. Existence of an optimal BFS: A fundamental result in linear programming (the Fundamental Theorem of LP) says that if an LP is feasible and its objective is bounded, then there exists an optimal basic feasible solution. Concretely, a BFS is obtained by:

- Selecting a set of $k \le m$ linearly independent constraints from among the m inequalities $A_i x \le b_i$ or the nonnegativity constraints $x_j \ge 0$.

- Tightening exactly n of these inequalities to equalities so that we have a square system of n linear equations in the n unknowns $x_1, \dots, x_n$.
- Solving that system to obtain a basic solution, and then checking that it satisfies all the original inequalities (hence is feasible).

At an optimum, we may choose such a BFS that also maximizes $c^\top x$. Hence it suffices to show that every BFS is integral.

2. Structure of the system defining a BFS: Label the constraints $1, 2, \dots, m$ for $A\,x \le b$, and the n nonnegativity constraints $x_j \ge 0$. A BFS picks exactly n of these constraints to hold as equalities. Without loss of generality, suppose we pick m of the row-constraints and $n - m$ nonnegativity constraints (the argument is identical if some nonnegativity constraints are tight in place of some non-row ones).

Reorder and partition $x = (x_B, x_N)$ so that the m basic variables $x_B \in \mathbb{R}^m$ are those indexed by the chosen row-constraints, and the remaining $n - m$ nonbasic variables $x_N \in \mathbb{R}^{n-m}$ are set to zero. Then the active row-constraints become

$$A_B\,x_B + A_N\,x_N \;=\; b_B, \quad x_N = 0,$$

where $A_B \in \mathbb{R}^{m\times m}$ is the square submatrix of A formed by the columns corresponding to the basic variables, and $b_B \in \mathbb{Z}^m$ the corresponding entries of b. Hence the BFS is obtained by solving

$$A_B\,x_B = b_B, \qquad x_N = 0.$$

3. Total unimodularity implies integer inverse: By hypothesis, A is totally unimodular, so every square submatrix of A has determinant in $\{0, \pm 1\}$. In particular, our chosen basis matrix A_B is unimodular: $\det(A_B) = \pm 1$. Since the LP is assumed feasible and bounded, A_B must be nonsingular, so $\det(A_B) = \pm 1 \neq 0$. Hence A_B is invertible over the integers, and by Cramer's rule its inverse has integer entries:

$$A_B^{-1} \;=\; \frac{\operatorname{adj}(A_B)}{\det(A_B)} \quad\Longrightarrow\quad A_B^{-1} \in \mathbb{Z}^{m\times m}.$$

Because $b_B \in \mathbb{Z}^m$, we conclude

$$x_B \;=\; A_B^{-1}\,b_B \;\in\; \mathbb{Z}^m.$$

Together with $x_N = 0 \in \mathbb{Z}^{n-m}$, the full basic solution $x = (x_B, x_N)$ is integral.

To conclude, every optimal BFS is integral. Since an optimal solution exists and can be chosen to be basic, we obtain an $x^* \in \mathbb{Z}^n$ attaining the maximum of $c^\top x$.
Hence if the LP has any optimum, there is an integral optimum $x^* \in \mathbb{Z}^n$.

Ph4-Task6 (Optional):

Let $G = (X \cup Y, E)$ be a bipartite graph with incidence matrix $A \in \{0,1\}^{(|X|+|Y|)\times|E|}$, where each column of A has exactly two ones (one in a row-vertex, one in a column-vertex). We will prove that A is totally unimodular: every square submatrix has determinant in $\{0, \pm 1\}$.
As suggested, the proof will be by induction, on the size k of a square submatrix.

Base case ($k = 1$): Any 1×1 submatrix of A is either 0 or 1. Its determinant is 0 or 1, so it lies in $\{0, \pm 1\}$.

Inductive step: Assume every $(k-1)\times(k-1)$ submatrix of A has determinant in $\{0,\pm1\}$. Let B be an arbitrary $k\times k$ submatrix of A. We analyze two cases by looking at the columns of B, each of which comes from an edge of G.

1. **Case 1:** Some column of B has at most one nonzero entry.
   - If the column is all zeros, $\det(B)=0$.
   - If it has exactly one entry 1 in row r, expand $\det(B)$ by cofactor along that column:

     $$\det(B) \;=\; \pm1 \;\times\; \det\bigl(B'\bigr),$$

     where B' is the $(k-1)\times(k-1)$ submatrix obtained by deleting row r and that column. By the induction hypothesis $\det(B')\in\{0,\pm1\}$, so $\det(B)\in\{0,\pm1\}$.

2. **Case 2:** Every column of B has exactly two ones (since A is an incidence matrix of a simple bipartite graph).
   - Partition the k selected rows of B into two disjoint subsets $R_X\subseteq X$ and $R_Y\subseteq Y$.
   - Form the row-vector $w=\mathbf{1}_{R_X}-\mathbf{1}_{R_Y}\in\mathbb{R}^k$, where $\mathbf{1}_{R_X}$ is 1 on rows in R_X and 0 elsewhere, and similarly for $\mathbf{1}_{R_Y}$.
   - Multiply $w^\top$ by B. Since each column of B has exactly one 1 in R_X and one in R_Y,

     $$w^\top B = (\underbrace{1-1}_{=0},\ \underbrace{1-1}_{=0},\ \dots) = (0,\dots,0).$$

     Thus $w^\top$ is a nonzero left-nullvector of B, so the rows of B are linearly dependent and $\det(B)=0$.

In both cases, we conclude $\det(B)\in\{0,\pm1\}$. So by induction, every square submatrix of A has determinant in $\{0,\pm1\}$. Therefore A is totally unimodular, as claimed.

Ph4-Task7:

We will now combine all previous steps to prove König's Theorem: in any bipartite graph G, the size of a maximum matching equals the size of a minimum vertex cover:

Let $G=(X\cup Y,E)$ be a bipartite graph, and let A be its vertex–edge incidence matrix. Recall:

1) **LP formulations and duality (Ph4-Tasks 1–3):**

   - The minimum vertex cover problem relaxes to the primal LP:

     $$\min\,\{\mathbf{1}^\top y : A^\top y\ge\mathbf{1},\ y\ge0\}.$$

   - The maximum matching problem relaxes to the dual LP:

     $$\max\,\{\mathbf{1}^\top x : A\,x\le\mathbf{1},\ x\ge0\}.$$

   - By strong duality, the two LPs have the same optimal objective value:

     $$\max\{\mathbf{1}^\top x : A\,x\le\mathbf{1}\} \;=\; \min\{\mathbf{1}^\top y : A^\top y\ge\mathbf{1}\}.$$

2) **Total unimodularity (Ph4-Task6):**
We showed that the incidence matrix A of a bipartite graph is totally unimodular.

3) **Integrality of LP optima (Ph4-Task5):**
Whenever an LP has a totally unimodular constraint matrix and an integral right-hand side, any optimal solution can be chosen integral. Applying this to both the matching LP and the cover LP (whose right-hand sides are all ones), we deduce:

$$\exists\, x^* \in \{0,1\}^{|E|} \text{ optimal for the matching LP}, \quad \exists\, y^* \in \{0,1\}^{|X|+|Y|} \text{ optimal for the cover LP.}$$

In other words, the LP relaxations automatically return integral solutions.

4) **Conclusion (König's Theorem):**

- The integral vector x^* corresponds to a matching of size $\mathbf{1}^\top x^*$.
- The integral vector y^* corresponds to a vertex cover of size $\mathbf{1}^\top y^*$.
- By duality, these two sizes are equal.

Hence the maximum number of edges in a matching equals the minimum number of vertices in a cover:

$$\max\{|M| : M \text{ a matching}\} \;=\; \min\{|C| : C \text{ a vertex cover}\},$$

which is exactly König's Theorem.

Phase 5:

Verification of our implementation of the simplex algorithm

We verify the implementation of our simplex algorithm using the example LP in `simplex_solver.py`. The example LP is as follows:

$$\begin{aligned} \text{Minimize} \quad & c^T x \\ \text{subject to} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

With:

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 1 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} 30 \\ 60 \end{bmatrix},$$

The constraint $Ax = b$ implies that x needs to be at the intersection of two planes in $\mathbb{R}^3$. The equation of this line is as follows:

$$\mathbf{x} = \begin{bmatrix} 30 \\ 0 \\ 0 \end{bmatrix} + \lambda \begin{bmatrix} -2 \\ 1 \\ 1 \end{bmatrix} \tag{2}$$

Furthermore, since we have the constraint $x \geq 0$ it implies that $\lambda \in [0, 15]$.

• First, we test our simplex algorithm with $c = \begin{bmatrix} 2 & 1 & 3 \end{bmatrix}^{\mathrm{T}}$, it is the default value in the `simplex_solver.py` code. In this case we can see that $c \cdot \begin{bmatrix} -2 & 1 & 1 \end{bmatrix}^{\mathrm{T}} = 0$ and so we expect that any x on the line 2 is an optimal solution. Here are the results we obtain with Bland's rule:

Optimal solution	[30, 0, 0]
Optimal value	60.0
Optimal basis	[0, 2]
CVXPY optimal solution	[11.8947633, 9.05261835, 9.05261835]
CVXPY optimal value	59.99999999999999

Table 1: Results for $c = \begin{bmatrix} 2 & 1 & 3 \end{bmatrix}^{\mathrm{T}}$ using Bland's rule

We get different optimal solutions between our solver and CVXPY but it is expected because any x on the line 2 is an optimal solution and CVXPY is not using the simplex algorithm it typically uses interior-point methods like ECOS or SCS. We find that both solutions have the same optimal value. Our solver indeed found a solution on the line 2, it is the x we obtain with $\lambda = 0$. So our solver is correct.

In the table 2 the results with Dantzig's rule are shown.
The solver is again correct, the solution has the correct optimal value and is located on the line 2, with $\lambda = 15$.

The points $\begin{bmatrix} 30 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$ and $\begin{bmatrix} 0 & 15 & 15 \end{bmatrix}^{\mathrm{T}}$ are extreme points of the feasible region, obtained with $\lambda = 0$ or 15. The choice of pivoting rule decides which of these extreme points will be given as a solution by the simplex solver.

Optimal solution	[0, 15, 15]
Optimal value	60.0
Optimal basis	[1, 2]
CVXPY optimal solution	[11.8947633, 9.05261835, 9.05261835]
CVXPY optimal value	59.99999999999999

Table 2: Results for $c = \begin{bmatrix} 2 & 1 & 3 \end{bmatrix}^{\mathrm{T}}$ using Dantzig's rule

• We now test our solver in situations where there is only one optimal solution. For instance let's take $c = \begin{bmatrix} 4 & 1 & 3 \end{bmatrix}^{\mathrm{T}}$

Here are the results with Bland's rule:

Optimal solution	[0, 15, 15]
Optimal value	60.0
Optimal basis	[1, 2]
CVXPY optimal solution	$[4.62266256 \times 10^{-8},\ 15.0,\ 15.0]$
CVXPY optimal value	60.00000009245326

Table 3: Results for $c = \begin{bmatrix} 4 & 1 & 3 \end{bmatrix}^{\mathrm{T}}$ using Bland's rule

We can see that the results of our solver and CVXPY's are the same. The only difference is due to CVXPY being less precise because it doesn't use the simplex algorithm.

Here are the results with Dantzig's rule:

Optimal solution	[0, 15, 15]
Optimal value	60.0
Optimal basis	[1, 2]
CVXPY optimal solution	$[4.62266256 \times 10^{-8},\ 15.0,\ 15.0]$
CVXPY optimal value	60.00000009245326

Table 4: Results for $c = \begin{bmatrix} 4 & 1 & 3 \end{bmatrix}^{\mathrm{T}}$ using Dantzig's rule

The solver is again correct.

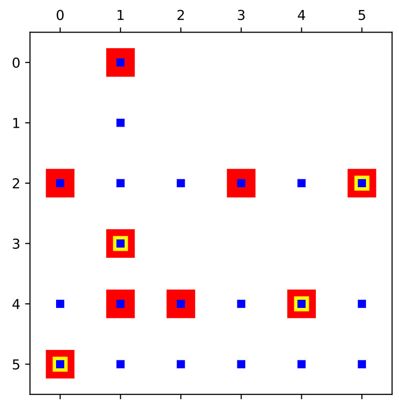
We tried other possibilities for the vector c, each time making sure that there is only one optimal solution and the results between CVXPY and our solver where the same. So we can conclude that our solver works correctly.

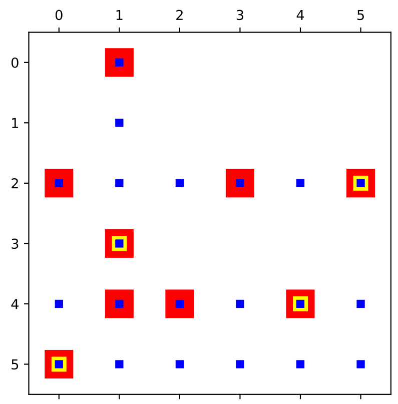
Optimal stategy for the game

In the the figure 1 and 2 we show the optimal solutions for the 6x6 and 100x100 boards. We show the optimal solutions obtained by our simplex solver using Bland's and Dantzig's rules. We also show for comparison the optimal solutions found using GLPK's simplex solver (using CVXPY). Remark: now we are using CVXPY with a simplex solver which was not the case in the previous part. This allows us to precisely assess the performances of our solver.

In the tables 5 and 6 we show the obtained optimal values, for the 6x6 and 100x100 boards, with our simplex solver with Bland's and Dantzig's rules. We also show the optimal values found using GLPK's simplex solver .

Note: the code used to solve the problem with CVXPY is in `run_coin_game_cvxpy.py`.


(a) Custom solver, Bland's rule


(b) Custom solver, Dantzig's rule

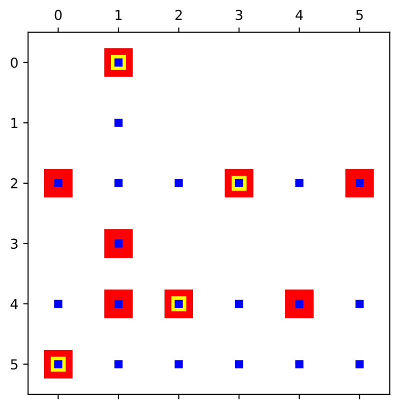
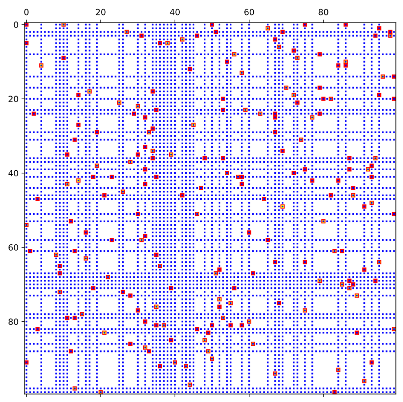
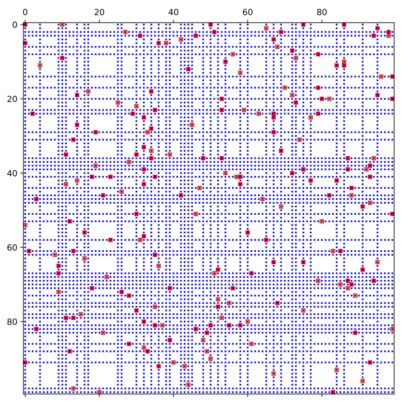

(c) GLPK's simplex solver

Figure 1: Optimal solutions for the 6x6 board


(a) Custom solver, Bland's rule


(b) Custom solver, Dantzig's rule

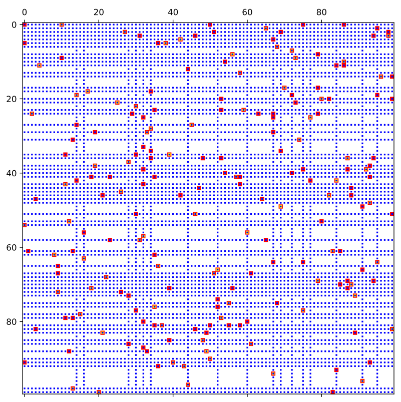

(c) GLPK's simplex solver

Figure 2: Optimal solutions for the 100x100 board

	optimal matching value	optimal vertex cover value
Custom solver, Bland's rule	4.0	4.0
Custom solver, Dantzig's rule	4.0	4.0
GPLK's simplex solver	4.0	4.0

Table 5: Optimal values for the 6x6 board

	optimal matching value	optimal vertex cover value
Custom solver, Bland's rule	80.0	80.0
Custom solver, Dantzig's rule	80.0	80.0
GPLK's simplex solver	80.0	80.0

Table 6: Optimal values for the 100x100 board

Analysis of the results for the 6x6 board

We can see that the optimal line cover found by our solver (using Bland's or Dantzig's) and GLPK is the same. The optimal pairing found by our solver with Bland's and Dantzig's rule is the same. However, it is different form the optimal pairing found by GLPK's solver.

We can see that the optimal matching value and optimal vertex cover value found by all the methods are the same and equal to 4.

Analysis of the results for the 100x100 board

We can clearly see that the optimal line cover found by our solver with Bland's rule, with Dantzig's rule and GLPK's solver are different from each other.

The optimal pairings found with our solver with Bland's rule and Dantzig's rule are the same. We were able to verify this by using the code `equality_test.py` which tests the equality between two matrices. We verified that the two matching matrices are the same.

It can be seen that the optimal pairings found with our solver with Bland's rule is different from the one found by GLPK's solver but they are very similar.

It can also be seen that the optimal pairings found with our solver with Dantzig's rule is different from the one found by GLPK's solver but they are similar.

The optimal matching value and optimal vertex cover value found by all the methods are the same and equal to 80.

Analysis of the overall results

We are in a situation where multiple optimal solutions exist. The solutions obtained with our simplex solver (using Bland's rule and Dantzig's rule) and GLPK's solver are all optimal and have the same optimal value. This optimal value is 4 for the 6x6 board and 80 for the 100x100 board.

The difference of optimal solutions while using Bland's rule and Dantzig's rule can be explained by the fact that these pivoting rules make the simplex algorithm choose different extreme points of the feasible region as optimal solutions. We saw this situation already in the previous part about the verification of our implementation of the simplex algorithm.