MA3K7 Assignment 2 Rubric

Cathal Lee (2106946)

1 Entry

I KNOW From the problem statement,

- Two players fill an $n \times n$ grid until all n^2 cells are filled.
- I can only place 0s (which annihilates any permutation term in the determinant) and my friend can only write the number 1.
- At the end, the grid is a 0–1 matrix $A \in \{0,1\}^{n\times n}$.
- I win if $\det(A) = 0$. Otherwise my friend wins.

NOTE From linear algebra, $\det(A) = 0$ means the rows (or columns) are linearly dependent. A useful way to look at $\det(A)$ for 0–1 matrices is via the permutation expansion. For this question, **I assume I move first** (I return to who starts first? in Review).

NOTATION For brevity, I use the following

- Cells are indexed by $\Omega = [n] \times [n]$.
- After the game, $A = (a_{ij})$ where

$$a_{ij} = \begin{cases} 0 & \text{if I played in cell } (i,j), \\ 1 & \text{if my friend played in cell } (i,j). \end{cases}$$

- $Z \subseteq \Omega$ is the set of my moves (the 0-cells).

I WANT

I want to understand whether I (the 0 player) can force $\det(A) = 0$ under optimal play, and how the answer depends on n. Concretely:

1. Solve small n exactly (to see what general idea of optimal play looks like).
2. Find a condition that must hold if my friend achieves $\det(A) \neq 0$.
3. Use (1) and (2) to motivate a general strategy or at least a rough theorem.

Before thinking of the general case, I checked $n = 1, 2$ by hand to see what winning should look like.

- $n = 1$. I win immediately if I move first.
- $n = 2$. Write $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ with $a, b, c, d \in \{0,1\}$ and $\det(A) = ad - bc$. If I can force two equal rows or two equal columns then $\det(A) = 0$ automatically.

TRY If I open at $a = 0$, then no matter where my friend plays a 1 next, I can play 0 on my second move to create either a zero row (e.g. if my friend plays in row 2, I complete row 1 with zeros), or a zero column (symmetrically) and then the determinant is forced to be 0 at the end. So $n = 2$ already suggests that the 0-player can often lock in singularity by creating a rigid linear dependence early.

2 Attack

Try

2.1 Cases for small n

I started with the first few sizes just to see what I am aiming for.

$n = 3$. Expanding a 3×3 determinant repeatedly is messy, and it is not clear what pattern I should aim for. So instead of computing determinants, I looked for a necessary condition for $\det(A) \neq 0$.

The determinant expansion is

$$\det(A) = \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i=1}^{n} a_{i,\pi(i)}.$$

I find

For a 0–1 matrix, each product $\prod_i a_{i,\pi(i)}$ is either 0 or 1.

Proposition 2.1. *If $\det(A) \neq 0$ for a 0–1 matrix A, then there exists a permutation $\pi \in S_n$ such that*

$$a_{i,\pi(i)} = 1 \quad \textit{for all } i \in [n].$$

Proof. Since $a_{i,\pi(i)} \in \{0,1\}$, the product $\prod_{i=1}^{n} a_{i,\pi(i)}$ equals 1 iff every factor equals 1. If $\det(A) \neq 0$, at least one summand in the expansion is nonzero, hence there is some π for which the product equals 1. $\square$

This suggests viewing rows as left vertices $R_1, \ldots, R_n$ and columns as right vertices $C_1, \ldots, C_n$, with an edge R_iC_j present whenever $a_{ij} = 1$. Then Proposition 2.1 says:

> If my friend wins, their 1-cells contain a perfect matching (one 1 in each row and each column).

$$\begin{array}{c|cccc}
 & C_1 & C_2 & C_3 & \cdots \\
\hline
R_1 & \bullet & & \bullet & \\
R_2 & & \bullet & & \\
R_3 & \bullet & & & \\
\vdots & & & &
\end{array}$$

Figure 1: Nonzero determinant implies a perfect matching of 1's.

Aha

So at minimum, I should be trying to block perfect matchings of 1's.

2.2 Main Idea

Small-n exact solving (below) suggests the 0 player is favored, so I am led to the following theorem. Importantly, this does not feel like a coincidence as same obstruction keeps reappearing whenever my friend tries to win, namely that they must preserve enough 1-cells to support a perfect matching.

Theorem 1. *If I move first, then for every $n \geq 4$, I have a strategy that forces* $\det(A) = 0$.

Plan

The key point is that I do not actually need to block all perfect matchings. Instead, it suffices to force a linear dependence among a small, controlled set of rows (or columns). I will force

$$R_1 + R_2 = \mathbf{1} = R_3 + R_4,$$

which immediately implies the rows are linearly dependent and hence $\det(A) = 0$.

TRY My friend wants at least one perfect matching of 1's. A naive fear is that for large n there are $n!$ matchings, so maybe I cannot block them all.

Rather than trying to block matchings one-by-one, it would be enough to force some set of k rows to only connect (via 1-entries) to fewer than k columns, since then no perfect matching can exist. I did not see how to enforce this dynamically, so I switched to a different approach.

So I can think of the problem as trying to force some set of rows to have too few available 1-columns for any perfect matching to exist.

STUCK At this point I was stuck because I know what obstruction I want, but not how, since my friend chooses where the 1 edges appear.

MAYBE TRY A natural idea is a pairing/mirroring strategy where we pair cells so that whenever my friend claims one, I claim its partner and delete an edge pattern. I experimented with this on $n = 4$ by making pairings in rows/columns and checking whether it seems to annihilate all STUCK matchings. This did not immediately yield a clean general proof, so I instead did

- exact minimax for small n to confirm the small-n pattern.
- extracting structural lemmas (matching obstruction) that would guide a future general strategy.

3 Solution

3.1 Winner for $n \leq 4$ by minimax

INTRODUCE I implemented an exact recursion that explores *every* legal play sequence until the board is full. Minimax logic:

- On my turn, I win if there exists a move leading to a winning continuation.
- On my friend's turn, I only win if every move they can play still allows me to win.

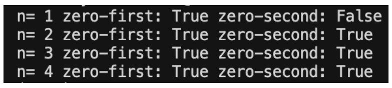
```python
import numpy as np
from functools import lru_cache

def zero_player_wins(n: int, zero_starts: bool) -> bool:
    @lru_cache(None)
    def solve(state, turn):
        if -1 not in state:
            A = np.array(state).reshape(n, n)
            return int(round(np.linalg.det(A))) == 0

        empties = [i for i, v in enumerate(state) if v == -1]

        if turn == 0:   # 0 player
            return any(solve(state[:k] + (0,) + state[k+1:], 1) for k
                in empties)
        else:           # 1 player
            return all(solve(state[:k] + (1,) + state[k+1:], 0) for k
                in empties)

    start_turn = 0 if zero_starts else 1
    return solve(tuple([-1] * (n*n)), start_turn)

for n in range(1, 5):
    print("n=", n,
          "zero-first:", zero_player_wins(n, True),
          "zero-second:", zero_player_wins(n, False))
```

Check

I was initially worried about floating-point issues in `np.linalg.det`. For $n \leq 4$ and 0–1 matrices, the determinant is an integer of modest magnitude, so rounding is reliable in practice; additionally, I checked random boards against an exact integer determinant routine and got agreement.

Claim 1. *For $n = 1, 2, 3, 4$, if the 0 player moves first then the 0 player has a winning strategy. Moreover, for $n = 3$ the 0 player also wins when moving second.*

Proof. The recursion is exhaustive and implements optimal play by construction (existential choice for the 0 player, universal choice for the 1 player). The printed outputs show `True` for the relevant cases. □



```
n= 1 zero-first: True zero-second: False
n= 2 zero-first: True zero-second: True
n= 3 zero-first: True zero-second: True
n= 4 zero-first: True zero-second: True
```

Figure 2: Output Result

3.2 A forcing pattern for 3×3

Idea

Case

In my hand analysis, a natural first move is the centre. It lies on four lines (row, column, and two diagonals), so it gives the most flexibility for creating "near-zero" lines.

Suppose my friend responds by placing a 1 in an edge cell (not a corner). A response I found effective is to place 0 in the opposite edge cell. This tends to set up two separate candidate lines: one horizontal/vertical and one diagonal. In many branches, each of these lines is then one move away from becoming an all-zero line, so my final move can aim to complete whichever line remains unblocked.

The key point is that on their next move my friend can only place a 1 to interfere with one of these threatened lines, so the other line is often still available to complete as a zero row/column/diagonal, which forces $\det(A) = 0$.

Corner responses behave similarly by symmetry, and a centre response is impossible since I occupy it first. Overall, this centre-opening repeatedly produced a "fork" between two zero-line threats in my analysis, which is consistent with the minimax result that the 0-player can win for $n = 3$.

In many branches this creates two distinct zero-line threats, each one move away from completion. Since my friend can only interfere with one such line per move, I can usually complete the other on my final move.

3.3 A winning strategy for all $n \geq 4$

Goal

I now give a direct turn-by-turn strategy for the 0-player that forces $\det(A) = 0$ for every $n \geq 4$.

Setup

Fix the first four rows. Pair them as $(1, 2)$ and $(3, 4)$. For each column $j \in [n]$, define two *paired cells*:

$$(1, j) \leftrightarrow (2, j), \qquad (3, j) \leftrightarrow (4, j).$$

The rest of the board (rows 5 to n) I call the *outside region.*

Invariant

I will aim to finish the game with the property that for every column j:

$$a_{1j} + a_{2j} = 1 \quad \text{and} \quad a_{3j} + a_{4j} = 1.$$

Equivalently,

$$R_1 + R_2 = \mathbf{1} \quad \text{and} \quad R_3 + R_4 = \mathbf{1}.$$

Then $(R_1 + R_2) - (R_3 + R_4) = 0$, so the rows are linearly dependent and hence $\det(A) = 0$.

Strategy

I describe my moves in two phases.

Phase 1: while there exists an empty cell in the outside region.

- If my friend plays a 1 in one of the four special rows (rows 1–4), say at $(1, j)$, I immediately respond by playing 0 at the paired cell $(2, j)$. Similarly, if they play at $(2, j)$ I respond at $(1, j)$; if they play at $(3, j)$ I respond at $(4, j)$; and if they play at $(4, j)$ I respond at $(3, j)$.
- If my friend plays a 1 in the outside region, I respond by playing 0 in *any* empty outside cell.

Note

Since moves are alternating, whenever my friend places a 1 in a paired cell, the partner cell is still empty and I can immediately occupy it with 0. In particular, once a 1 appears in a paired cell, the partner is immediately fixed as 0, so a pair can never end up with two 1's.

Why

In Phase 1, I avoid placing 0's in the special rows unless it is forced by the pairing response. So the outside region acts as slack that lets me postpone difficult decisions until the end.

The outside region has $(n-4)n$ cells, so it provides a buffer of moves. In any case, once the outside region is full, all remaining moves lie in rows 1–4, so Phase 2 describes how I finish consistently.

Phase 2: after the outside region is completely filled. At this point, all remaining empty cells lie in rows 1–4. From now on, I ensure the following invariant:

> At any time in Phase 2, there is *at most one* paired column-cell (among the pairs $(1, j) \leftrightarrow (2, j)$ and $(3, j) \leftrightarrow (4, j)$) of the form one entry is 0 and the other entry is still empty. All other pairs are either completely empty or already completed as $(0, 1)$ or $(1, 0)$.

I implement this as follows:

- If my friend plays a 1 in a pair whose partner is empty, I respond by playing 0 in that partner, completing the pair as $(1, 0)$.
- Otherwise (i.e. my friend plays a 1 in the empty partner of the *unique* half-finished pair), then that pair becomes completed. I respond by starting a *new* half-finished pair: choose any completely empty pair and play 0 in one of its two cells.

Check

This maintains the invariant: exactly one half-finished pair exists until the very end, where it gets completed by my friend's final 1.

Endgame structure

Proof of Theorem 1. I prove that following the above strategy forces $\det(A) = 0$.

At the end of the game, every paired column-cell in the special rows must be completed (the board is full). By construction of my responses, a completed pair in rows $(1, 2)$ has the form $(1, 0)$ or $(0, 1)$. So for every column j,

$$a_{1j} + a_{2j} = 1.$$

Similarly, every completed pair in rows $(3, 4)$ has one 1 and one 0, hence

$$a_{3j} + a_{4j} = 1.$$

Therefore,

$$R_1 + R_2 = \mathbf{1} = R_3 + R_4.$$

Conclusion

Then $(R_1 + R_2) - (R_3 + R_4) = 0$, so the four rows R_1, R_2, R_3, R_4 are linearly dependent. Hence the full matrix A has linearly dependent rows, so $\det(A) = 0$. Thus the 0-player has a winning strategy for every $n \geq 4$. □

3.4 Necessary condition for my friend (what I must block in my turn)

I FIND From the Attack section we can package the matching obstruction neatly.

Proposition 3.1. *If my friend wins (so* $\det(A) \neq 0$*), then the set of* 1*-cells contains a perfect matching.*

Proof. Proposition 2.1 gives a permutation π with $a_{i,\pi(i)} = 1$ for all i, which is exactly a perfect matching. □

4 Review

CHECK I checked:

- the minimax recursion matches the rules (one cell per move, forced values 0 or 1)
- the recursion implements optimal play (existential choice for the 0 player, universal choice for the 1 player)
- the implication $\det(A) \neq 0 \Rightarrow$ existence of an all-ones permutation term
- the starting-player issue by testing both starting players for $n \leq 4$.

A potential concern was whether my friend could place two 1's in the same pair. However, the turn-based nature of the game ensures that the pairing response always occurs immediately, preventing this situation.

MISTAKE My first instinct was to try tofill a whole row with zeros. However, I quickly realised that this is not optimal, as my friend can always place a 1 in that row on their first opportunity, and then the plan collapses. That forced me to switch from local goals (a single row) to global obstructions.

REFLECT The most useful move was switching away from raw determinant calculation and instead asking what must be true even for $\det(A) \neq 0$ to be possible? It also clarified why the game is not about computing determinants but about preventing a global combinatorial structure.

EXTEND Possible extensions:

- **Starting player.** I assumed I move first throughout. It would be interesting to characterise the outcome when the 0-player moves second for general n (my minimax suggests the 0-player can still win at least for $n = 3$).
- **Alternative viewpoints.** Proposition 3.1 gives a clean necessary condition for $\det(A) \neq 0$ via perfect matchings, but the actual winning strategy uses a stronger linear-algebraic obstruction. It would be nice to relate these perspectives more systematically.
- **Variants.** If the win condition is changed (e.g. determinant modulo p), cancellations can occur even when perfect matchings exist, so the above obstruction no longer directly applies.

Code

The code for this assignment can be found on my GitHub page:
`https://github.com/cathal0317/Problem_Solving_RUBRIC`