Price of Locality in Permutation Mastermind: Are TikTok Influencers Chaotic Enough?

Bernardo Subercaseaux
Carnegie Mellon University, Pittburgh, PA, USA

Abstract

In the permutation Mastermind game, the goal is to uncover a secret permutation $\sigma^\star\colon [n] \to [n]$ by making a series of guesses $\pi_1, \ldots, \pi_T$ which must also be permutations of $[n]$, and receiving as feedback after guess π_t the number of positions i for which $\sigma^\star(i) = \pi_t(i)$. While the existing literature on permutation Mastermind suggests strategies in which π_t and π_{t+1} might be widely different permutations, a resurgence in popularity of this game as a TikTok trend shows that humans (or at least TikTok influencers) use strategies in which consecutive guesses are very similar. For example, it is common to see players attempt one transposition at a time and slowly see their score increase. Motivated by these observations, we study the theoretical impact of two forms of *locality* in permutation Mastermind strategies: ℓ_k-local strategies, in which any two consecutive guesses differ in at most k positions, and the even more restrictive class of w_k-local strategies, in which consecutive guesses differ in a window of length at most k. We show that, in broad terms, the optimal number of guesses for local strategies is quadratic, and thus much worse than the $O(n \lg n)$ guesses that suffice for non-local strategies. We also show NP-hardness of the satisfiability version for ℓ_3-local strategies, whereas in the ℓ_2-local variant the problem admits a randomized polynomial-time algorithm.

2012 ACM Subject Classification Mathematics of computing → Combinatorics

Keywords and phrases Permutation Mastermind, Locality, NP-hard

Digital Object Identifier 10.4230/LIPIcs.FUN.2026.39

Related Version *Full Version*: `https://arxiv.org/abs/2601.19161`

Funding This work is supported by the National Science Foundation under grant DMS-2434625.

1 Introduction

The classical game of Mastermind, and its many variations, have received significant attention from computer scientists and mathematicians since the 1970s [1,4,6,11,12,15,17,19]. However, it is only in the last couple of years that the *permutation black-peg Mastermind* variant has reached internet virality through a TikTok challenge often named "bottle match challenge" or "color match challenge". The goal of the challenge is to guess a secret permutation of colored bottles, based on guesses that also consist of permutations of an equivalent set of colored bottles. To each guess, a different player that we will call *codemaker*, responds with the number of positions for which the guess matches the secret permutation. In the TikTok trend, the secret permutation is hidden from the player under a table or a box, but visible to the viewer throughout the game (see Figure 1). Perhaps the success of the trend can be explained by paraphrasing filmmaker Alfred Hitchcock: suspense is not about a bomb exploding on scene, but rather about the audience watching people talk about trivial matters while a bomb ticks underneath their table.

After observing a variety of TikTok influencers play permutation Mastermind, a common aspect of their strategies came to my attention: *they usually only permute a couple of bottles between guesses*. In contrast, the best known upper bound for permutation Mastermind, which takes $O(n \lg n)$ guesses for n bottles, is based on guessing uniformly random permutations [17],

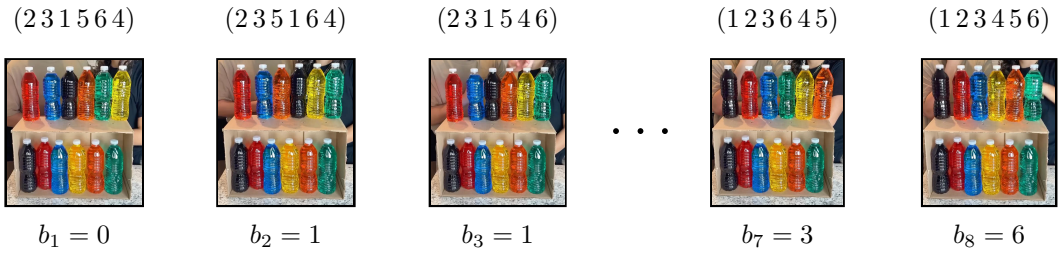

Figure 1 Illustration of a game of permutation Mastermind for $n = 6$ from TikTok [13]. Guesses are labeled by treating the secret permutation as the identity $\sigma^\star := (1\,2\,3\,4\,5\,6)$.

and thus almost all bottles are permuted between guesses on average.[1] My initial hypothesis was that these *"local"* strategies, which permute only a small number of bottles between guesses, are natural for humans since it might be easier to keep track of the information throughout the game. This is reminiscent, for instance, of the analysis of Mastermind with constant-sized memory, by Doerr and Winzen [5]. This paper aims to take the only reasonable course of action after TikTok has started to appropriate our well-studied game of Mastermind: to study a theoretical model of these local strategies and their implications both on the number of guesses required to win the game, and on its computational complexity.

1.1 Preliminaries

Before stating our main results, let us briefly formalize the problem at hand and present some of the best bounds known for the general case of permutation Mastermind.

Permutation Mastermind. Let $n \geqslant 2$ and S_n be the symmetric group, or in other words, the set of permutations $[n] \to [n]$ together with the composition operation. A secret code is a permutation $\sigma^\star \in S_n$ chosen by the *codemaker*. In round t, the *codebreaker* plays a guess $\pi_t \in S_n$ and receives the *black-peg score*[2]

$$b(\pi_t, \sigma^\star) \;:=\; \big|\{i \in [n] : \pi_t(i) = \sigma^\star(i)\}\big|.$$

An *adaptive strategy* is a function that maps the *transcript* $\big((\pi_1, b_1), \ldots, (\pi_{t-1}, b_{t-1})\big)$ to the next guess $\pi_t \in S_n$. A *static* (non-adaptive) strategy is a fixed list of guesses $(\pi_1, \ldots, \pi_T)$ chosen in advance. In the adaptive setting, a guess π_t wins the game for the codebreaker if $\pi_t = \sigma^\star$. In the static (i.e., non-adaptive) setting, after the codebreaker announces their fixed list of guesses $(\pi_1, \ldots, \pi_T)$, the codemaker responds with the scores $b(\pi_1, \sigma^\star), \ldots, b(\pi_T, \sigma^\star)$, after which the codebreaker gets to do one last guess to potentially win the game [9].

Locality notions and Cayley-Mastermind. We consider two different forms of locality, defined next based on the notation $D(\pi, \sigma) := \{i \in [n] : \pi(i) \neq \sigma(i)\}$ for the set of indices in which two permutations differ. For any $k \geqslant 2$, we will say a strategy is ℓ_k-local if for every two consecutive guesses π_t and π_{t+1} made by the strategy in any transcript, we have $|D(\pi_t, \pi_{t+1})| \leqslant k$. For a permutation $\pi \in S_n$, we call the set $D(\pi, \mathrm{id}_n) := \{i \in [n] : \pi(i) \neq i\}$ the *support* of π.

[1] Recall that a uniformly random permutation has a single fixed point in expectation.
[2] The name comes from the original physical version of the game, in which *black pegs* were used to indicate the number of correct positions guessed.

A strategy is w_k-local if for every two consecutive guesses π_t and π_{t+1} it holds that

$$\max(D(\pi_t, \pi_{t+1})) - \min(D(\pi_t, \pi_{t+1})) \leqslant k - 1.$$

Intuitively, this means there is some "window" $I := \{i, i+1, \ldots, i+k-1\}$ such that π_{t+1} differs from π_t only within that window (i.e., $D(\pi_t, \pi_{t+1}) \subseteq I$).

In the static case, we will assume that the last guess after receiving the scores is free of locality restrictions.

Our notions of locality are a particular case of playing Mastermind on a Cayley graph. Given a set of generators Γ of S_n, we can define the Γ-Permutation-Mastermind game in which the codemaker chooses a secret vertex of the Cayley graph $\mathsf{Cay}(S_n, \Gamma)$, then the codebreaker guesses an arbitrary starting vertex v_0, and for $t \geq 1$, the codebreaker must guess a vertex v_t adjacent in $\mathsf{Cay}(S_n, \Gamma)$ to v_{t-1}. Equivalently, a vertex v_t such that $v_{t-1}^{-1} v_t \in \Gamma$. Naturally, we can define in turn a Γ-strategy for the standard permutation Mastermind as one that always plays according to the rules of the Γ-Permutation-Mastermind game. For example, for any $k \geqslant 2$, we define $L_n^{(k)}$ as the set of all permutations in S_n whose support has size at most k. A strategy of permutation Mastermind is then said to be ℓ_k-local if its consecutive guesses differ in at most k positions, or equivalently, if it plays on the Cayley graph $\mathsf{Cay}(S_n, L_n^{(k)})$. We purposely present this more general version of the definition to motivate a natural direction of future research: studying permutation Mastermind under other sets of generators.

Query complexities. Let $A(n)$ be the minimum T such that there exists an adaptive strategy that determines $\sigma^\star$ within T guesses in the worst case. Let $S(n)$ be the analogous minimum for static strategies. For local versions, define: $A_\ell(n,k)$, $A_w(n,k)$, $S_\ell(n,k)$, $S_w(n,k)$ as the minimum worst-case number of guesses among adaptive/static strategies subject to the corresponding locality constraint (ℓ_k or w_k). The main previous results that contextualize our work are summarized in Table 1.

Table 1 Best-known asymptotic query complexities for permutation Mastermind.

Function	Lower bound	Upper bound
$A(n)$	$\Omega(n)$ [16, 20]	$O(n \lg n)$ [16, 20]
$S(n)$	$\Omega(n \lg n)$ [9]	$O(n \lg n)$ [17]

1.2 Our Contributions

We prove the following results.

▶ **Theorem 1.** *For the ℓ_k-local setting, we have* $\frac{n^2-3n}{2k} \leqslant A_\ell(n,k) \leqslant \frac{n^2 \lg n}{k}(1+o(1))$.

▶ **Theorem 2.** *In the ℓ_k-local static setting, we have*

$$\frac{n^2 - (1+o(1))n^{3/2}}{k} \leqslant S_\ell(n,k) \leqslant 28n \lg n \cdot \left\lceil \frac{n-1}{k-1} \right\rceil.$$

▶ **Theorem 3.** *For the w_k-local setting, we have $S_w(n,2) = \Theta(n^2)$.*

Our lower bounds are graph-theoretic in nature, and interestingly, they are based on giving the codebreaker even more advantage in order to simplify the analysis; we consider *generous* codemakers that reveal exactly which of the permuted positions of some guess is correct.

A similar setting was studied by Li and Zhu [18], who considered the more general case of Mastermind with "Wordle" feedback. Our upper bounds for Theorem 1 and Theorem 2 are based on simulating the best known upper bounds without locality restrictions. The upper bound of Theorem 3, in turn, uses a similar strategy to that of [18].

Finally, we prove a hardness result, showing that processing the feedback received is not easy even if each guess was a single transposition from the previous one. It was first proved by de Bondt [3] that standard Mastermind, which includes the *white-peg* score, is NP-hard. Independently, Stuckman and Zhang [21] provided a different proof. Then, Goodrich [11] extended the result to only black-peg score feedback. We take these results further by proving hardness in the permutation variant, and even on the ℓ_3-local setting. In contrast, for the ℓ_2-local setting, we show the problem can be solved in randomized polynomial time, leveraging an algorithm for perfect matchings with parity constraints from Geelen and Kapadia [8].

▶ **Theorem 4.** *The* satisfiability *problem for permutation Mastermind in the ℓ_3-local setting is* NP*-hard. In other words, given a transcript $((\pi_1, b_1), \dots, (\pi_t, b_t))$, where each pair of consecutive guesses differs in at most 3 positions, it is* NP*-hard to decide whether there exists a secret $\sigma^\star$ such that $b(\pi_i, \sigma^\star) = b_i$ for every $1 \leqslant i \leqslant t$. In turn, for the ℓ_2-local setting there is a randomized polynomial-time algorithm.*

2 Adaptive Lower Bound on ℓ_k-local Strategies

This section is devoted to proving the lower bound of Theorem 1. The main object we will handle in the proof is a bipartite graph representing the potential secrets $\sigma^\star$ compatible with the information seen thus far. It will be convenient for this proof to think of $\sigma^\star$ as a string of length n over an alphabet $\Sigma := \{s_1, \dots, s_n\}$, so that *positions* of σ are just integers in $[n] := \{1, \dots, n\}$ but the *elements* of σ have a different type, Σ. This way, we think of $\sigma^\star$ as a function from $[n]$ to Σ. To any $\sigma\colon [n] \to \Sigma$, we associate a bipartite graph $B_\sigma := ([n] \sqcup \Sigma, E)$, where a pair $\{i, s_j\} \in E$ if and only if $\sigma(i) = s_j$. Note that these graphs B_σ are perfect matchings.

Recall that $b(\pi, \sigma) = |\{i \in [n] : \pi(i) = \sigma(i)\}|$ is the black-peg score of a guess. Let $b^{-1}(k, \sigma)$ be the set of all guesses $\pi\colon [n] \to \Sigma$ such that $b(\pi, \sigma) = k$.

Then, given a sequence of guesses $\Pi = (\pi_1, \dots, \pi_t)$, to which the codemaker has answered $B := (b_1, \dots, b_t)$, and some $\pi\colon [n] \to \Sigma$, we say π is *compatible* with (Π, B) if $b(\pi_i, \pi) = b_i$ for $1 \leqslant i \leqslant t$.

▶ **Definition 5** (Graph of possible secrets)**.** *Given a sequence of guesses $\Pi = (\pi_1, \dots, \pi_t)$, to which the codemaker has answered $B := (b_1, \dots, b_t)$, we define the* graph of possible secrets

$$P_t := \bigcup_{\sigma \text{ is compatible with } (\Pi, B)} B_\sigma.$$

As a degenerate case, P_0 is the complete bipartite graph between $[n]$ and Σ.

The following observation is immediate, but will be useful to translate our problem into a graph-theoretic one.

▶ **Observation 6.** *Given a sequence of guesses $\Pi = (\pi_1, \dots, \pi_t)$, to which the codemaker has answered $B := (b_1, \dots, b_t)$, the codebreaker can be certain of the secret permutation σ if and only if P_t has a unique perfect matching.*

We now prove that, for a graph P_t to have a unique perfect matching, it needs to be missing a significant number of edges. We will make use of the following folklore observation (illustrated in Figure 2):

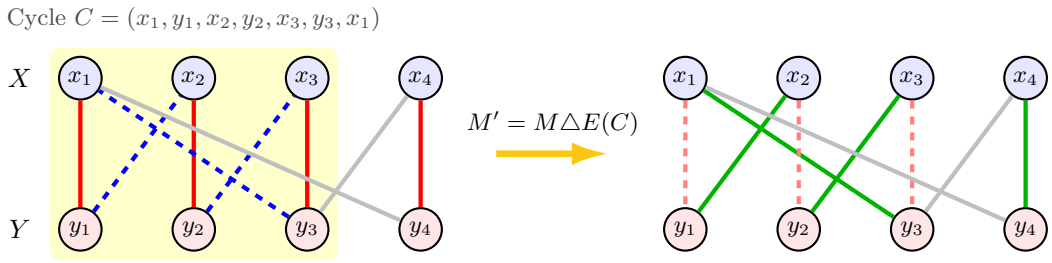

Figure 2 Illustration of the proof for Observation 7. Edges of the original matching M are colored red, and edges in $E(C) \setminus M$ are dashed blue. Gray edges are not part of the matchings, and the resulting matching M' is colored green.

▶ **Observation 7.** *If a finite bipartite graph B has minimum degree at least 2, it cannot have a unique perfect matching.*

Proof. If B has no perfect matchings, the statement holds trivially. Thus, let M be a perfect matching of B. For each vertex u, let $M(u)$ denote its neighbor in the matching M, and $D(u)$ denote one neighbor of u via an edge not in M, chosen arbitrarily, which must exist since u has degree at least 2. Now let u_0 be any vertex, and construct a walk $u_0, u_1, \ldots$ by

$$u_i := \begin{cases} M(u_{i-1}) & \text{if } i \text{ is odd} \\ D(u_{i-1}) & \text{otherwise.} \end{cases}$$

Since B is finite, some vertex appears twice in this walk. Let $u_j = u_k$ with $j < k$, chosen so that all intermediate vertices are distinct. Then the subwalk $u_j, u_{j+1}, \ldots, u_k$ forms a simple alternating cycle C, which necessarily has even length. Consider the matching $M' = M \triangle E(C)$. For vertices not in C, M' agrees with M. Each vertex of C is incident to exactly one edge of M and one edge of $E(C) \setminus M$, so it is matched by exactly one edge in M'. Thus M' is a perfect matching. Since C contains edges not in M, we have $M' \neq M$, contradicting the uniqueness of M. ◀

We now use Observation 7 to show a much stronger condition implied by having a unique perfect matching.

▶ **Lemma 8.** *Let $B = (X \sqcup Y, E)$ be a balanced bipartite graph with $n := |X| = |Y|$. Then, if B has a unique perfect matching, $|E| \leqslant \binom{n+1}{2}$.*

Proof. Let $f(n)$ be the maximum number of edges of a balanced bipartite graph with $2n$ vertices that has a unique perfect matching. We trivially have $f(1) = 1$. For $n \geqslant 2$, we claim that $f(n) \leqslant f(n-1) + n$. Before proving the claim, let us see that it is enough to conclude, since then, using induction we will have

$$f(n) \leqslant \binom{n}{2} + n = \frac{n(n-1)}{2} + n = \binom{n+1}{2}.$$

Now, to prove the claim, we consider an arbitrary balanced bipartite graph B_n on $2n$ vertices with a unique perfect matching and such that $|E(B_n)| = f(n)$. Let δ be the minimum degree in B_n, and note that $\delta = 1$: since B_n has a perfect matching we have $\delta \geqslant 1$, and by Observation 7 we have $\delta < 2$. Let $u \in V(B_n)$ be a vertex of degree 1, and v its only neighbor. It must be the case that u and v belong to opposite sides of the bipartition, and thus if we consider the graph resulting from B_n by removing vertices u and v, we get

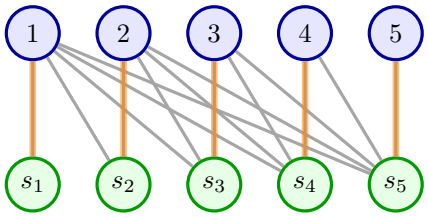

Figure 3 The half graph H_5, with $\binom{6}{2} = 15$ edges. Its only perfect matching is highlighted in orange.

a balanced bipartite graph B' on $2(n-1)$ vertices. Moreover, $\{u, v\}$ must belong to the (unique) perfect matching of B_n, and therefore any perfect matching of B' would extend (by adding $\{u, v\}$) to a perfect matching of B_n, and thus B' must also have a unique perfect matching. Therefore, $|E(B')| \leqslant f(n-1)$. On the other hand,

$$|E(B')| = |E(B_n)| - \deg_{B_n}(v) - (\deg_{B_n}(u) - 1) = |E(B_n)| - \deg_{B_n}(v) \geqslant |E(B_n)| - n.$$

We thus have

$$f(n) = |E(B_n)| \leqslant |E(B')| + n \leqslant f(n-1) + n,$$

concluding the claim. ◀

The tightness of Lemma 8 is witnessed by *half graphs* H_n (see Figure 3), which include the edge $\{i, s_j\}$ if and only if $i \leqslant j$. The uniqueness of a perfect matching in H_n can be easily seen by induction, noting that vertex n has degree one, and upon removing n together with its neighbor s_n, we are left exactly with H_{n-1}, for which the inductive hypothesis applies.

2.1 Generous and Supergenerous Codemakers

Perhaps paradoxically, I found that proving lower bounds on permutation Mastermind with locality becomes easier if we give more power to the codebreaker, but in a way that makes the game simpler.

Namely, while a normal codemaker will give $|B_\pi \cap B_{\sigma^\star}|$ as feedback (i.e., the number of correctly guessed positions), we will consider a *generous codemaker* that gives the set $B_\pi \cap B_\sigma$ as feedback. Note that, any strategy against a generous codemaker can be used against a standard codemaker without any loss, since from $B_\pi \cap B_\sigma$ the codebreaker can compute $|B_\pi \cap B_{\sigma^\star}|$.

In the original form of the game, each guess π_t can be described as well by $E_t := B_{\pi_t} \setminus B_{\pi_{t-1}}$, the set of newly guessed edges. Except for the first guess, which tests n edges, each guess π_t tests at most k edges in the ℓ_k setting. We will define a *supergenerous* codemaker as one that allows the codebreaker to guess individual edges one by one, giving generous feedback to each (i.e., revealing whether it is correct or not), but only accounting 1 guess per each k individual-edge guesses. Naturally, if we prove a lower bound of t individual-edge guesses against a supergenerous codemaker, then $(t-n)/k$ guesses are necessary against a standard codemaker.

2.2 Finishing the proof of Theorem 1's lower bound

▶ **Lemma 9.** *Let $k \geqslant 2$. Then, any deterministic adaptive strategy in the ℓ_k-local model requires $\frac{n^2-3n}{2k}$ guesses.*

Proof. We consider an adversarial *supergenerous* codemaker. The adversary will keep a graph G_t, with $G_0 := K_{n,n}$, and upon receiving a guess e_t for $t \geqslant 1$, it will answer *yes* if and only if e_t belongs to all perfect matchings of G_{t-1}. If it answers *no*, the adversary sets $G_t \leftarrow G_{t-1} \setminus e_t$, and otherwise $G_t \leftarrow G_{t-1}$. Let $\mathcal{S}_t$ be the set of all perfect matchings compatible with the answers given until time t (including to guess e_t), and $\mathcal{S}_0 = \mathsf{PM}(K_{n,n})$ the set of all perfect matchings of $K_{n,n}$. Then, by construction, we have that the adversary says *yes* to an edge e_t if and only if $e_t \in M$ for every $M \in \mathcal{S}_{t-1}$.

Let $\mathsf{PM}(G_t)$ be the set of all perfect matchings in G_t. We will prove by induction that $\mathcal{S}_t = \mathsf{PM}(G_t)$ for every $t \geqslant 0$. For $t = 0$ it holds trivially since both sets include all possible matchings. For $t \geqslant 1$, we consider two cases. First, if the adversary said *no* to guess e_t, we have

$$\mathsf{PM}(G_t) = \mathsf{PM}(G_{t-1}) \setminus \{M \in \mathsf{PM}(G_{t-1}) : e_t \in M\},$$

and similarly, $\mathcal{S}_t = \mathcal{S}_{t-1} \setminus \{M \in \mathcal{S}_{t-1} : e_t \in M\}$. But by the inductive hypothesis we then have

$$\mathcal{S}_t = \mathsf{PM}(G_{t-1}) \setminus \{M \in \mathsf{PM}(G_{t-1}) : e_t \in M\} = \mathsf{PM}(G_t).$$

On the other hand, if the adversary answered *yes*, we have by definition

$$\begin{aligned}
\mathcal{S}_t &= \mathcal{S}_{t-1} \setminus \{M : M \in \mathcal{S}_{t-1} \text{ and } e_t \notin M\} \\
&= \mathcal{S}_{t-1} \setminus \varnothing = \mathcal{S}_{t-1} && \text{(Since the answer to } e_t \text{ was } \textit{yes}) \\
&= \mathsf{PM}(G_{t-1}) && \text{(inductive hypothesis)} \\
&= \mathsf{PM}(G_t). && \text{(Since } G_t = G_{t-1} \text{ as the answer to } e_t \text{ was } \textit{yes})
\end{aligned}$$

We have thus proved that $\mathcal{S}_t = \mathsf{PM}(G_t)$. Now, observe that by definition the codebreaker can only win when $|\mathcal{S}_t| = 1$. But this implies $|\mathsf{PM}(G_t)| = 1$, and by Lemma 8, this implies $|E(G_t)| \leqslant \binom{n+1}{2}$. But $|E(G_t)| \geqslant |E(G_0)| - t$, from where we get

$$t \geqslant |E(G_0)| - |E(G_t)| = n^2 - |E(G_t)| \geqslant n^2 - \binom{n+1}{2} = \binom{n}{2}.$$

Therefore, the lower bound we get against a standard codemaker is at least

$$\frac{t-n}{k} \geqslant \frac{\binom{n}{2} - n}{k} = \frac{n^2 - 3n}{2k}.$$

◀

3 Lower Bound for the Static Variant

This section is devoted to the static version of permutation Mastermind with locality considerations. Recall that in this variant, the codebreaker must reveal a fixed sequence of guesses $\pi_1, \ldots, \pi_T$ upfront, subject to the locality constraints, after which they will receive the black-peg feedback $b_1, \ldots, b_T$, and then finally make their final guess that must equal the secret permutation $\sigma^\star$ in order to win. The final guess is not subject to the locality restrictions.

The main ingredient of our lower bound is a classic result on extremal graph theory (see [22] for a modern presentation).

▶ **Theorem 10** ([7, 14]). *The maximum number of edges an n-vertex graph can have without containing a $K_{2,2}$ is $(\frac{1}{2} + o(1))n^{3/2}$. Moreover, the maximum number of edges on a $K_{2,2}$-free bipartite graph with n vertices on each side is $(1 + o(1))n^{3/2}$.*

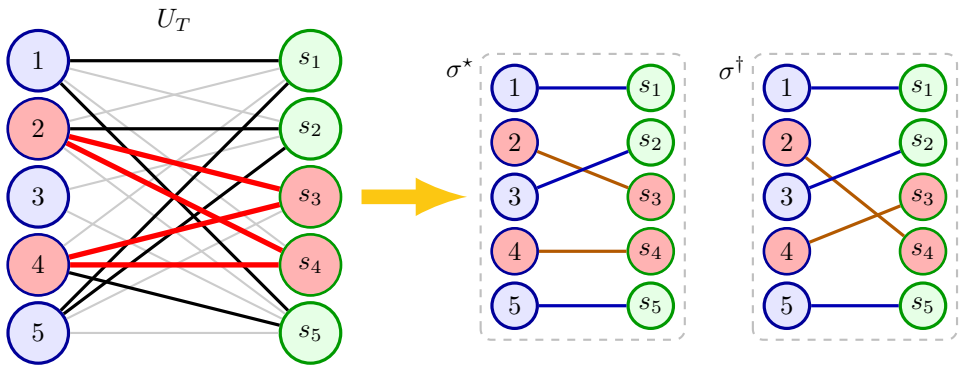

Figure 4 Illustration of the obstacle introduced by a $K_{2,2}$ for uniquely determining $\sigma^\star$. On the left, tested edges are colored light gray, while untested edges are colored black, except for those forming a $K_{2,2}$ which are highlighted.

For this proof we will consider a simpler graph than in Section 2. Let $U_t := ([n] \times \Sigma, E_t)$ be the bipartite graph where an edge $\{i, s_j\}$ is present if and only if no guess π_r with $r \leqslant t$ has $\pi_r(i) = s_j$. Intuitively, U_t corresponds to the *untested* matching edges up to time t. Now, we will prove the relevance of $K_{2,2}$-subgraphs. For this, let us say the static part of a strategy, $\pi_1, \ldots, \pi_T$, is *sufficient*, if for every secret $\sigma^\star$, the only permutation compatible with the transcript $((\pi_1, b(\pi_1, \sigma^\star)), \ldots, (\pi_T, b(\pi_T, \sigma^\star)))$ is exactly $\sigma^\star$. Naturally, the static part of any correct static strategy must be sufficient.

▶ **Lemma 11.** *If $\pi_1, \ldots, \pi_T$ is the static part of a correct static strategy, then U_T is $K_{2,2}$-free.*

Proof. Suppose for a contradiction that U_T contains a $K_{2,2}$ consisting of the four edges

$$\{i, s_x\}, \{i, s_y\}, \{j, s_x\}, \{j, s_y\}.$$

An illustration with a concrete example is depicted in Figure 4. Now, let $\sigma^\star$ be an arbitrary permutation such that $\sigma^\star(i) = s_x$ and $\sigma^\star(j) = s_y$. From $\sigma^\star$ we define now the permutation $\sigma^\dagger$ by

$$\sigma^\dagger(k) = \begin{cases} s_y & \text{if } k = i \\ s_x & \text{if } k = j \\ \sigma^\star(k) & \text{otherwise.} \end{cases}$$

In other words, $\sigma^\dagger$ is the matching obtained from $\sigma^\star$ via the alternating 4-cycle corresponding to the $K_{2,2}$. Now, note that for any guess π_t we have $b(\pi_t, \sigma^\star) = b(\pi_t, \sigma^\dagger)$. Indeed, let $k \in [n]$ and we prove by cases that $\sigma^\star(k) = \pi_t(k)$ if and only if $\sigma^\dagger(k) = \pi_t(k)$:

- If $k \notin \{i, j\}$, then $\sigma^\star(k) = \sigma^\dagger(k)$, and thus the claim holds.
- If $k \in \{i, j\}$, then by construction we have $\sigma^\star(k) \in \{s_x, s_y\}$ as well as $\sigma^\dagger(k) \in \{s_x, s_y\}$. But by definition of U_T, we have $\pi_t(k) \notin \{s_x, s_y\}$. Therefore neither $\sigma^\star(k) = \pi_t(k)$ nor $\sigma^\dagger(k) = \pi_t(k)$.

We thus have that $b(\pi_t, \sigma^\star) = b(\pi_t, \sigma^\dagger)$ for any guess π_t, which implies both $\sigma^\star$ and $\sigma^\dagger$ are compatible with transcript $((\pi_1, b(\pi_1, \sigma^\star)), \ldots, (\pi_T, b(\pi_T, \sigma^\star)))$, and thus $\pi_1, \ldots, \pi_T$ is not sufficient. ◀

Our lower bound is now a direct consequence of the previous lemma and the quantitative bounds on $K_{2,2}$-free graphs.

▶ **Theorem 12.** *For any $k \geqslant 2$, we have $S_\ell(n,k) \geqslant \frac{n^2-(1+o(1))n^{3/2}}{k}$.*

Proof. Consider a correct strategy whose static part is $\pi_1, \ldots, \pi_T$. Note that before the first guess we have $|E(U_0)| = n^2$. The first guess tests n edges, so $|E(U_1)| = n^2 - n$, and after this special first guess the ℓ_k-locality condition implies for $2 \leqslant t \leqslant T$ that

$$|E(U_t)| \geqslant |E(U_{t-1})| - k.$$

Therefore, $|E(U_T)| \geqslant |E(U_1)| - kT = n^2 - n - kT$. Now, since the static strategy is correct, we must have that U_T is $K_{2,2}$-free, and by Theorem 10 it must hold that $|E(U_T)| \leqslant (1+o(1))n^{3/2}$. Combining our inequalities, we have $(1+o(1))n^{3/2} \geqslant n^2 - n - kT$, from where $T \geqslant \frac{n^2-(1+o(1))n^{3/2}}{k}$, thus concluding the proof. ◀

4 Diameter-Based Upper Bounds

In this section we prove the upper bounds for Theorem 1 and Theorem 2 by a rather direct simulation adaptation of the $O(n \lg n)$ upper bounds of [17, 20]. In a nutshell, it suffices to show that we can pay $O(n/k)$ to simulate regular guesses in the ℓ_k-local model. This idea generalizes to the following lemma.

▶ **Lemma 13.** *Let Γ be a set of generators for S_n. Let $S_\Gamma(n)$ denote the minimum number of guesses for a correct static strategy for the Γ-Permutation-Mastermind game, and $A_\Gamma(n)$ denote the minimum number of guesses in the worst case of an adaptive strategy. Then,*

$$S_\Gamma(n) \leqslant S(n) \cdot \operatorname{diam}(\mathit{Cay}(S_n, \Gamma)) \quad \text{and} \quad A_\Gamma(n) \leqslant A(n) \cdot \operatorname{diam}(\mathit{Cay}(S_n, \Gamma))$$

Proof. We write the proof in terms of the static setting, but the adaptive one is identical. Let $\pi_1, \ldots, \pi_{S(n)}$ be a correct static strategy for the standard permutation Mastermind. Then, for each $1 \leqslant i \leqslant S(n)$, write $(\pi_{i-1})^{-1}\pi_i$ as a word $\omega_i := \gamma_1^{(i)} \ldots \gamma_{r_i}^{(i)}$ over Γ that minimizes r_i. Then, by definition of the diameter $r_i \leqslant \operatorname{diam}(\mathsf{Cay}(S_n, \Gamma))$. Now, construct the guess sequence

$$\pi_\Gamma := \pi_1 \gamma_1^{(2)} \ldots \gamma_{r_2}^{(2)} \gamma_1^{(3)} \ldots \gamma_{r_3}^{(3)} \ldots \gamma_1^{(S(n))} \ldots \gamma_{r_{S(n)}}^{(S(n))},$$

which is Γ-Permutation-Mastermind strategy, and has size at most

$$1 + \sum_{i=2}^{S(n)} r_i \leqslant 1 + (S(n)-1)\operatorname{diam}(\mathsf{Cay}(S_n, \Gamma)) \leqslant S(n) \cdot \operatorname{diam}(\mathsf{Cay}(S_n, \Gamma)).$$ ◀

We now prove a tight bound on the diameter of the Cayley graph corresponding to the ℓ_k-local setting.

▶ **Lemma 14.** *Fix $k \geqslant 2$. Recall that $L_n^{(k)}$ is the set of permutations with support on at most k elements. Then,*

$$\operatorname{diam}(\mathit{Cay}(S_n, L_n^{(k)})) \leqslant \left\lceil \frac{n-1}{k-1} \right\rceil.$$

Proof. Since $\mathsf{Cay}(S_n, L_n^{(k)})$ is a Cayley graph and thus vertex transitive, it suffices to upper bound the distance between the identity permutation id_n, and an arbitrary permutation π.

Let $M := D(\pi, \mathrm{id}_n) = \{i \in [n] : \pi(i) \neq i\}$ be the support of π, and let $m := |M|$. Letting $\mathsf{dist} := \mathsf{dist}_{\mathrm{Cay}(S_n, L_n^{(k)})}$, we will prove by induction on m that the stronger statement $\mathsf{dist}(\mathrm{id}, \pi) \leqslant \left\lceil \frac{m-1}{k-1} \right\rceil$ holds for any π.

If $m = 0$, then $\pi = \mathrm{id}_n$ and the bound holds. Otherwise $m \geqslant 2$, since $m = 1$ is not possible. If $m \leqslant k$, then $\pi \in L_n^{(k)}$, so $\mathsf{dist}(\mathrm{id}_n, \pi) = 1$, which agrees with the bound.

Assume now that $m > k$. We will construct $\gamma \in L_n^{(k)}$ such that $|D(\gamma^{-1}\pi, \mathrm{id})| \leqslant m-(k-1)$, and then invoke the inductive hypothesis.

Write π as a product of disjoint cycles $C_1, \ldots, C_r$, and for each nontrivial cycle C_i fix any cyclic ordering of its elements. By listing the elements of the nontrivial cycles in these cyclic orders (cycle by cycle), we obtain a sequence $v_1, \ldots, v_m$ that enumerates M with the property that for every $1 \leqslant j < m$, either $\pi(v_j) = v_{j+1}$, or else v_j is the last element of its cycle (in which case $\pi(v_j)$ is the first element of that same cycle and hence appears earlier in the list).

Let $T := \{v_1, \ldots, v_k\}$ be the first k elements of the previously obtained sequence, and observe that for every $x \in T \setminus \{v_k\}$ we have $\pi(x) \in T$: indeed, if $x = v_j$ with $j < k$, then either $\pi(x) = v_{j+1} \in T$, or x is the last element of its cycle and $\pi(x)$ is the first element of that cycle, which also lies in T.

Since π is injective and $\pi(T \setminus \{v_k\}) \subseteq T$, the set $\pi(T \setminus \{v_k\})$ has size $k-1$, so there is a unique element $y \in T \setminus \pi(T \setminus \{v_k\})$. Define $\gamma \in S_n$ by

$$\gamma(x) = \begin{cases} \pi(x) & \text{if } x \in T \setminus \{v_k\}, \\ y & \text{if } x = v_k, \\ x & \text{otherwise.} \end{cases}$$

By construction, γ only moves elements of T, so $\gamma \in L_n^{(k)}$. Moreover, for every $x \in T \setminus \{v_k\}$ we have $(\gamma^{-1}\pi)(x) = \gamma^{-1}(\pi(x)) = \gamma^{-1}(\gamma(x)) = x$, so $\gamma^{-1}\pi$ fixes all elements of $T \setminus \{v_k\}$. Since $T \subseteq M$, these are $k-1$ indices that were moved by π, and thus $|D(\gamma^{-1}\pi, \mathrm{id}_n)| \leqslant m-(k-1)$.

Now, since $\pi = \gamma \cdot (\gamma^{-1}\pi)$ and $\gamma \in L_n^{(k)}$, we have

$$\mathsf{dist}(\mathrm{id}_n, \pi) \leqslant 1 + \mathsf{dist}(\mathrm{id}_n, \gamma^{-1}\pi) \leqslant 1 + \left\lceil \frac{(m-(k-1))-1}{k-1} \right\rceil = \left\lceil \frac{m-1}{k-1} \right\rceil,$$

where the second inequality is the inductive hypothesis. This completes the induction.

Finally, since $m \leqslant n$, we conclude that for every $\pi \in S_n$, we have $\mathsf{dist}(\mathrm{id}_n, \pi) \leqslant \left\lceil \frac{n-1}{k-1} \right\rceil$, and hence $\mathsf{diam}(\mathrm{Cay}(S_n, L_n^{(k)})) \leqslant \left\lceil \frac{n-1}{k-1} \right\rceil$. ◀

▶ **Proposition 15** (Upper bounds). *In the static ℓ_k-local model,*

$$S_\ell(n,k) \leqslant 28n \lg n \cdot \left\lceil \frac{n-1}{k-1} \right\rceil,$$

and in the adaptive ℓ_k-local model,

$$A_\ell(n,k) \leqslant \frac{n^2 \lg n}{k}(1 + o(1)).$$

Proof. Larcher, Martinsson, and Steger proved that $S(n) \leqslant 28n \lg n$ [17], and thus combining Lemma 13 and Lemma 14 directly yields the first result.

On the other hand, El Ouali et al. [20] proved that $A(n) \leqslant (n-3)\lceil \lg n \rceil + \frac{5}{2}n - 1$. Again, combining Lemma 13 and Lemma 14 yields the result. ◀

5 Window Locality

In this section we consider the w_k-local model. Intuitively, one would expect this more restrictive model to require many more guesses than the ℓ_k-local model. However, perhaps surprisingly, showing a good separation result is one of the questions we will leave open.

Let us start by remarking that diameter-based upper bounds, as in Section 4, cannot help us here, since in the w_k-local model the diameter is $\Omega(n^2/k^2)$ as we show next.

▶ **Proposition 16.** *Let W_k be the set of permutations $\omega \in S_n$ for which there exists an index i such that $\omega(j) = j$ for $j \notin \{i, i+1, \dots, i+k-1\}$. Then,*

$$\mathit{diam}(\mathit{Cay}(S_n, W_k)) \geqslant \frac{\lceil n^2/2 \rceil}{k(k-1)} \geqslant \frac{1}{2}\left(\frac{n}{k}\right)^2.$$

Proof. Consider the permutation $\pi^\dagger \colon i \mapsto n+1-i$, and let us prove that $\mathsf{dist}(\pi^\dagger, \mathrm{id}_n) \geqslant \frac{\lceil n^2/2 \rceil}{k(k-1)}$. For this, define the function $\delta \colon S_n \to \mathbb{N}$ by $\delta(\pi) := \sum_{i=1}^{n} |\pi(i) - i|$. Note that $\delta(\mathrm{id}_n) = 0$, whereas

$$\delta(\pi^\dagger) = \sum_{i=1}^{n} |n+1-i-i| = \sum_{i=1}^{n} |n+1-2i| = \left\lceil \frac{n^2}{2} \right\rceil,$$

where the last equality can be seen by splitting the sum into two parts depending on whether the argument of the absolute value is positive or not.

To conclude, it suffices to show that, for any $\pi \in S_n$ and $\omega \in W_k$, we have

$$\delta(\omega\pi) - \delta(\pi) \leqslant k(k-1). \tag{1}$$

Indeed, if $\mathsf{dist}(\pi^\dagger, \mathrm{id}_n) < \frac{\lceil n^2/2 \rceil}{k(k-1)}$, we could write $\pi^\dagger$ as $\omega_1\omega_2\dots\omega_t\mathrm{id}_n$ with $t < \frac{\lceil n^2/2 \rceil}{k(k-1)}$ and $\omega_i \in W_k$ for all i. But then we would have

$$\delta(\pi^\dagger) = \sum_{i=1}^{t} \delta(\omega_i \dots \omega_1 \mathrm{id}_n) - \delta(\omega_{i+1} \dots \omega_1 \mathrm{id}_n) \leqslant t \cdot k(k-1) < \lceil n^2/2 \rceil,$$

which contradicts the fact that $\delta(\pi^\dagger) = \lceil n^2/2 \rceil$. We now prove (1). let I_ω be the interval $\{i, \dots, i+k-1\}$ from the definition of W_k in which ω is potentially different from id_n, and then observe that

$$\begin{aligned}
\delta(\omega\pi) &= \sum_{i=1}^{n} |\omega(\pi(i)) - i| \\
&= \sum_{i:\pi(i)\notin I_\omega} |\pi(i) - i| + \sum_{i:\pi(i)\in I_\omega} |\omega(\pi(i)) - i| \\
&= \sum_{i:\pi(i)\notin I_\omega} |\pi(i) - i| + \sum_{i:\pi(i)\in I_\omega} |(\omega(\pi(i)) - \pi(i)) + (\pi(i) - i)| \\
&\leqslant \sum_{i:\pi(i)\notin I_\omega} |\pi(i) - i| + \sum_{i:\pi(i)\in I_\omega} |\omega(\pi(i)) - \pi(i)| + \sum_{i:\pi(i)\in I_\omega} |\pi(i) - i| \\
&= \delta(\pi) + \sum_{i:\pi(i)\in I_\omega} |\omega(\pi(i)) - \pi(i)|.
\end{aligned}$$

But by definition of W_k, we have that $\omega(I_\omega) = I_\omega$, and thus $|\omega(j) - j| \leqslant k-1$ for $j \in I_\omega$. Therefore,

$$\delta(\omega\pi) \leqslant \delta(\pi) + \sum_{i:\pi(i)\in I_\omega} (k-1) = \delta(\pi) + k(k-1),$$

which proves (1) and thus concludes the proof. ◀

Algorithm 1 ConveyorBelt(n).

```
1: t ← 1
2: π_t ← id ▷ initial guess
3: for m = 1 to n do ▷ repeat a full left-to-right pass n times
4:     π' ← π_t
5:     for i = 1 to n − 1 do ▷ adjacent transposition (i  i + 1)
6:         swap π'(i) and π'(i + 1)
7:         π_{t+1} ← π'
8:         t ← t + 1
```

Naturally, the diameter of the Cayley graph yields a lower bound on $A_w(n,k)$, but in this case such a bound is weaker than the one we get from the trivial inequality $A_\ell(n,k) \leqslant A_w(n,k)$, combined with Theorem 1.

While using the diameter bound between guesses of a normal permutation Mastermind strategy would only yield an $O\left(\frac{n^3 \lg n}{k^2}\right)$ upper bound, we show that $O(n^2)$ is always a valid upper bound for $S_w(n,k)$, and it holds already for $k=2$. Since the diameter lower bound immediately shows $S_w(n,2) = \Omega(n^2)$, we focus on proving the upper bound of Theorem 3.

▶ **Proposition 17.** *We have* $S_w(n,2) = O(n^2)$.

Proof. First, let us assume the codemaker is *generous*, in the sense of Section 2, and thus reveals after every guess which of the $k=2$ changed positions is correct. We will then show how to get rid of this assumption with only a constant overhead.

We will now use the following "conveyor belt" strategy (see Algorithm 1): take the sequence of $n-1$ adjacent transpositions $(1\,2), (2\,3), \ldots, (n-1\;n)$ and repeat it n times. For example for $n=4$, this strategy guesses:

1. $(1\,2\,3\,4)$
2. $(2\,1\,3\,4)$
3. $(2\,3\,1\,4)$
4. $(2\,3\,4\,1)$
5. $(3\,2\,4\,1)$
6. $(3\,4\,2\,1)$
7. $(3\,4\,1\,2)$
8. $(4\,3\,1\,2)$
9. $(4\,1\,3\,2)$
10. $(4\,1\,2\,3)$

The nice property of this sequence of adjacent transpositions is that each element goes through every position, and thus, since the codemaker is generous, after this sequence of guesses, the codebreaker will know exactly what the secret permutation is. More formally, for each value $1 \leqslant m \leqslant n$ (line 3 of Algorithm 1), the first element of π' right after line 4 is m, and it will be at position $2 \leqslant j \leqslant n$ after exactly $j-1$ guesses. Thus, the exact position of m must be revealed by the generous codemaker on that pass.

After having identified the position of each element, the codebreaker knows the target permutation, and can reach the secret permutation through a shortest path. Using the well-known identity $\mathsf{diam}(S_n, W_2) = \binom{n}{2}$, the codebreaker will reach the secret permutation in $O(n^2)$ guesses.

Now, it only remains to show how to simulate the generous codemaker with only a constant factor overhead. For this, consider for example that for $n=3$ the last guess was $(1\,2\,3)$ and we now want to guess $(2\,1\,3)$ but receiving generous feedback. We will do the sequence of guesses: $(1\,2\,3) \to (1\,3\,2) \to (3\,1\,2) \to (3\,2\,1) \to (2\,3\,1) \to (2\,1\,3)$. Intuitively, the scores received throughout this sequence will uniquely determine the "generous" feedback for the guess $(2\,1\,3)$. More in general, we will replace each transposition $(i\,i+1)$ by 6 guesses that

traverse the 6 possible permutations of $\{i, i+1, i+2\}$ (or $\{i-1, i, i+1\}$ in case $i = n-1$). The fact that going through the 6 permutations allows us to determine if any of the 3 cycled elements should go into one of the 3 positions of the cycle can be checked via finite computation. Indeed, the following Python3 code is enough:

```python
import itertools

possible_masks = list(set(itertools.permutations([0,0,1,2,3], 3)))
guesses = list(itertools.permutations([1,2,3]))

results = {}
for pm in possible_masks:
    results[pm] = []
    for g in guesses:
        results[pm].append(sum(1 for m, g in zip(pm, g) if m == g))

for pm1, pm2 in itertools.combinations(possible_masks, 2):
    assert results[pm1] != results[pm2]
```

To conclude the proof, we observe that the strategy is indeed static. ◀

6 The Complexity of Locality

This section is dedicated to proving Theorem 4. We will start by proving NP-hardness of the permutation variant without considering the locality restriction, and then show how a modification of the proof yields hardness even in the ℓ_3-local setting. We will conclude the section by proving that in the ℓ_2-local setting there is a randomized polynomial algorithm.

Our reduction is from Monotone-1-in-3-SAT, where the input is a 3-CNF formula with only positive literals, and the goal is to decide if some assignment of its variables assigns exactly 1 literal per clause to true. A more formal description is presented below. Interestingly, the first proof of NP-completeness of Mastermind was a reduction from 1-in-3-SAT (non-monotone) [3], but our construction is different.

PROBLEM :	Monotone-1-in-3-SAT
INPUT :	A 3-CNF formula $\varphi := \bigwedge_{i=1}^{m} C_i$, over variables $x_1, \ldots, x_n$, with each clause $C_i := (x_1^{(i)} \vee x_2^{(i)} \vee x_3^{(i)})$, and each $x_j^{(i)} \in \{x_1, \ldots, x_n\}$.
OUTPUT :	YES, if there is an assignment $\tau\colon \{x_1, \ldots, x_n\} \to \{0,1\}$ such that for every clause C_i we have $\tau\left(x_1^{(i)}\right) + \tau\left(x_2^{(i)}\right) + \tau\left(x_3^{(i)}\right) = 1$. NO otherwise.

▶ **Theorem 18** ([10], see [2]). *Monotone-1-in-3-SAT is* NP*-complete.*

Now we present the Permutation-Mastermind-SAT problem.

PROBLEM :	Permutation-Mastermind-SAT
INPUT :	A sequence of permutations $\pi_1, \ldots, \pi_T \in S_n$, and a sequence of black-peg scores $(b_1, \ldots, b_T) \in \{0, \ldots, n\}$.
OUTPUT :	YES, if there is some permutation $\sigma^\star \in S_n$ such that $b(\pi_t, \sigma^\star) = b_t$ for each $1 \leqslant t \leqslant T$. NO otherwise.

▶ **Theorem 19.** *Permutation-Mastermind-SAT is* NP*-complete.*

Proof. Membership is trivial, since it suffices to guess $\sigma^\star \in S_n$. For hardness, we reduce from Monotone-1-in-3-SAT. Let $\varphi := \bigwedge_{i=1}^{m} C_i$, over variables $x_1, \ldots, x_n$, be an input instance of Monotone-1-in-3-SAT. Then, we set $N = 3n$, and we will construct an instance of Permutation-Mastermind-SAT over S_N.

The first guess will be simply the identity $\pi_1 := \mathrm{id}_N$, and the score is $b_1 := 0$. This enforces that the secret permutation $\sigma^\star$ holds $\sigma^\star(i) \neq i$ for each $1 \leqslant i \leqslant N$. In order to define the next guesses we will need some notation. For each $1 \leqslant i \leqslant n$, let us define the *block* $B_i := (3i-2, 3i-1, 3i)$. Intuitively, the first part of our construction will enforce that in the secret permutation $\sigma^\star = (s_1\, s_2 \ldots s_N)$, we have $\{s_{3i-2}, s_{3i-1}, s_{3i}\} = \{3i-2, 3i-1, 3i\}$, meaning that the secret permutation is only permuting within blocks of 3 elements.

Using notation $\oplus$ for concatenation, we can define the identity permutation $\mathrm{id}_N \in S_N$ as $B_1 \oplus B_2 \cdots \oplus B_n$. Now, for each $1 \leqslant i < j \leqslant n$ and $\sigma \in S_3$, we define the permutation

$$\gamma_{i,j,\sigma} := B_1 \oplus \cdots \oplus B_{i-1} \oplus \sigma(B_j) \oplus \cdots \oplus B_{j-1} \oplus \sigma(B_i) \oplus \cdots \oplus B_n.$$

For example, if $n = 4$, and $\sigma = (1\,3\,2)$, then

$$\gamma_{1,3,\sigma} = \sigma(B_3) \oplus B_2 \oplus \sigma(B_1) \oplus B_4 = (7\,9\,8\,4\,5\,6\,1\,3\,2\,10\,11\,12).$$

Let $\sigma_1, \ldots, \sigma_6$ be an arbitrary enumeration of S_3. We now introduce $6\binom{n}{2}$ new guesses, corresponding to the permutations γ_{i,j,σ_k} for $1 \leqslant i < j \leqslant n$ and $1 \leqslant k \leqslant 6$. The score for each of them will be 0.

▷ Claim 20. Any permutation $\sigma^\star$ compatible with the scores for the $1 + 6\binom{n}{2}$ guesses made thus far satisfies

$$\forall 1 \leqslant i \leqslant n, \quad \{\sigma^\star(3i-2), \sigma^\star(3i-1), \sigma^\star(3i)\} = \{3i-2, 3i-1, 3i\}.$$

Proof. Assume the condition fails for some $1 \leqslant i \leqslant n$. Then, some element of $\{\sigma^\star(3i-2), \sigma^\star(3i-1), \sigma^\star(3i)\}$ belongs to a block B_j for $j \neq i$. Let us assume $i < j$.[3] Then, for some k it must be the case that guess γ_{i,j,σ_k} gets score at least 1, which contradicts the compatibility assumption with the received scores which are all 0. ◁

We therefore have that the secret permutation $\sigma^\star$ must be of the form

$$\sigma^{(1)}(B_1) \oplus \sigma^{(2)}(B_2) \oplus \cdots \oplus \sigma^{(n)}(B_n),$$

where each $\sigma^{(i)}$ is a permutation with no fixed points. In particular, the only two possibilities in S_3 are $\alpha := (2\,3\,1)$ and $\beta := (3\,1\,2)$. Intuitively, we will have $\sigma^{(i)} = \alpha$ if variable x_i should be assigned to true in φ, and β if it should be assigned to false.

Concretely, for each clause $C := (x_a \vee x_b \vee x_c)$ with $a < b < c$, we create a permutation

$$\pi_C := B_1 \oplus \cdots \oplus B_{a-1} \oplus \alpha(B_a) \oplus B_{a+1} \oplus \cdots \oplus \alpha(B_b) \oplus \cdots \oplus \alpha(B_c) \oplus \cdots \oplus B_n.$$

The score given to each of these guesses will be 3.

We are now ready to prove the correctness of the reduction. Let $\pi_1, \ldots, \pi_T$ with $T = 1 + 6\binom{n}{2} + m$, and $b_1, \ldots, b_T$ be the sequence of guesses and scores of our construction.

▷ Claim 21. If φ is a YES-instance for Monotone-1-in-3-SAT, then $(\pi_1, \ldots, \pi_T), (b_1, \ldots, b_T)$ is a YES-instance for Permutation-Mastermind-SAT.

Proof. Let τ be the satisfying assignment for φ which witnesses its membership in Monotone-1-in-3-SAT. Then, let $\sigma^\star$ be the permutation $\sigma^{(1)}(B_1) \oplus \sigma^{(2)}(B_2) \oplus \cdots \oplus \sigma^{(n)}(B_n)$ where

$$\sigma^{(i)} = \begin{cases} \alpha & \text{if } \tau(x_i) = 1 \\ \beta & \text{otherwise.} \end{cases}$$

[3] Otherwise we implicitly do $i := \min(i,j), j = \max(i,j)$.

It suffices to check that all scores on $\sigma^\star$ are as constructed. First, since both α and β have no fixed points, we indeed have $b(\mathrm{id}_N, \sigma^\star) = 0$. Let us write $\sigma[i] := (\sigma(3i-2), \sigma(3i-1), \sigma(3i))$ for the i-th block of a permutation σ.

Then, we claim that $b(\gamma_{i,j,\sigma}, \sigma^\star) = 0$ for every $1 \leqslant i < j \leqslant n$ and $\sigma \in S_3$. Indeed, for $k \in \{1, \ldots, n\} \setminus \{i, j\}$, we have $\gamma_{i,j,\sigma}[k] = B_k$, whereas $\sigma^\star[k]$ is either $\alpha(B_k)$ or $\beta(B_k)$, and in either case $b(\gamma_{i,j,\sigma}[k], \sigma^\star[k]) = 0$. For $k \in \{i, j\}$, we also have that $\sigma^\star[k]$ is either $\alpha(B_k)$ or $\beta(B_k)$, whereas $\gamma_{i,j,\sigma}[k] = \sigma(B_{\{i,j\}\setminus\{k\}})$, and thus $b(\gamma_{i,j,\sigma}[k], \sigma^\star[k]) = 0$.

Finally, for each clause C, we claim that $b(\pi_C, \sigma^\star) = 3$. Indeed, by assumption, there is exactly one variable $x_i \in C$ such that $\tau(x_i) = 1$, so $\sigma^{(i)} = \alpha$, and thus $\pi_C[i] = \alpha(B_i)$ and by construction $\sigma^\star[i] = \alpha(B_i)$, so $b(\pi_C[i], \sigma^\star[i]) = 3$. If the other variables of C are x_j and x_k, then for any $\ell \notin \{i, j, k\}$ we have $\pi_C[\ell] = B_\ell$, whereas $\sigma^\star[\ell]$ is either $\alpha(B_\ell)$ or $\beta(B_\ell)$, and in either case $b(\pi_C[\ell], \sigma^\star[\ell]) = 0$. For $\ell \in \{j, k\}$ we have $\tau(x_\ell) = 0$ by assumption, and thus $\sigma^\star[\ell] = \beta(B_\ell)$, whereas $\pi_C[\ell] = \alpha(B_\ell)$, and again $b(\pi_C[\ell], \sigma^\star[\ell]) = 0$. ◁

To complete our proof, we prove the other direction of the reduction.

▷ Claim 22. If $(\pi_1, \ldots, \pi_T), (b_1, \ldots, b_T)$ is a YES-instance for Permutation-Mastermind-SAT, then φ is a YES-instance for Monotone-1-in-3-SAT.

Proof. Assume there is some $\sigma^\star$ compatible with the transcript $(\pi_1, \ldots, \pi_T), (b_1, \ldots, b_T)$. Then, based on Claim 20, we have for each $1 \leqslant i \leqslant n$ that $\sigma^\star[i] = \sigma^{(i)}(B_i)$ for some $\sigma^{(i)} \in S_3$. Because $\pi_1 = \mathrm{id}_N$ and $b_1 = 0$, all the $\sigma^{(i)}$ must have no fixed points. But on S_3 the only two permutations without fixed points are exactly $\alpha := (2\,3\,1)$ and $\beta := (3\,1\,2)$. Therefore, $\sigma^{(i)} \in \{\alpha, \beta\}$ for each $1 \leqslant i \leqslant n$. Now, define the assignment $\tau \colon \{x_1, \ldots, x_n\} \to \{0, 1\}$ by

$$\tau(x_i) := \begin{cases} 1 & \text{if } \sigma^{(i)} = \alpha \\ 0 & \text{otherwise.} \end{cases}$$

Now, let $C := (x_a \vee x_b \vee x_c)$ be an arbitrary clause in φ. Then, since guess π_C got score 3, we have that

$$b(\alpha(B_a), \sigma^\star[a]) + b(\alpha(B_b), \sigma^\star[b]) + b(\alpha(B_c), \sigma^\star[c]) = 3. \tag{2}$$

However, for each $g \in \{a, b, c\}$, we have $\sigma^\star[g] = \alpha(B_g)$, in which case $b(\alpha(B_g), \sigma^\star[g]) = 3$ or $\sigma^\star[g] = \beta(B_g)$, in which case $b(\alpha(B_g), \sigma^\star[g]) = 0$. Thus, the only way to satisfy Equation (2) is that for exactly one $g \in \{a, b, c\}$, $\sigma^{(g)} = \alpha$, and thus τ satisfies exactly one variable of C. ◁

Since the reduction clearly takes polynomial time, this completes the proof. ◀

We will now show that the construction above can be adapted so that each pair of consecutive guesses differs in a fixed number of positions. The main difficulty is that, while the sequence of guesses $\pi_1, \ldots, \pi_T$ from the proof of Theorem 19 could be "interpolated" by a series of transpositions between each π_t and π_{t+1}, the reduction needs to construct scores for these intermediate guesses as well, and it must do so in such a way that does not require knowing what a satisfying assignment (if there is any) looks like.

PROBLEM :	k-Local-PM-SAT
INPUT :	A sequence of permutations $\pi_1, \ldots, \pi_T \in S_n$, where each $\pi_{t+1} := \pi_t \circ \omega_{t+1}$, for some permutation ω_{t+1} of support size at most k, and a sequence of black-peg scores $(b_1, \ldots, b_T) \in \{0, \ldots, n\}$.
OUTPUT :	YES, if there is some permutation $\sigma^\star \in S_n$ such that $b(\pi_t, \sigma^\star) = b_t$ for each $1 \leqslant t \leqslant T$. NO otherwise.

▶ **Theorem 23.** *3-Local-PM-SAT is* NP*-complete.*

Proof. Membership is again trivial since it suffices to guess $\sigma^\star \in S_n$. For hardness, the proof is very similar to that of Theorem 19, also via a reduction from Monotone-1-in-3-SAT, but we will need some non-trivial modifications. Once again, if the input instance φ has n variables, we will construct an instance of 3-Local-PM-SAT over S_N for $N := 3n$. The first guess will again be the identity id_N with a score of 0, implying the secret permutation must not have any fixed points. Then, similarly to the proof of Theorem 19, we will construct guesses and scores enforcing that

$$\forall 1 \leqslant i \leqslant n, \quad \{\sigma^\star(3i-2), \sigma^\star(3i-1), \sigma^\star(3i)\} = \{3i-2, 3i-1, 3i\}. \tag{3}$$

This part can be done in fact under the stronger ℓ_2-local restriction. It suffices to take every pair of indices $1 \leqslant i < j \leqslant 3n$ such that $\lceil i/3 \rceil \neq \lceil j/3 \rceil$ (i.e., i and j belong to different blocks of size 3), and guess the permutation $\delta_{i,j}$ defined by

$$\delta_{i,j}(k) = \begin{cases} k & \text{if } k \notin \{i, j\} \\ i & \text{if } k = j \\ j & \text{if } k = i. \end{cases}$$

The associated score of each guess $\delta_{i,j}$ will be 0, and after each such guess, we will guess again the identity permutation id_N with a score of 0, before making the next guess $\delta_{i',j'}$. This stage makes $O(n^2)$ guesses, and indeed guarantees Equation (3) since any position in $B_i := \{3i-2, 3i-1, 3i\}$ is guessed in every position outside of B_i and receives a score of 0. Now, we turn our attention to the *clause guesses*. Let $C = (x_i \vee x_j \vee x_k)$ be a clause in φ, with $i < j < k$. Our goal is to make the guess

$$\pi_C := B_1 \oplus \cdots \oplus B_{i-1} \oplus \alpha(B_i) \oplus B_{i+1} \oplus \cdots \oplus \alpha(B_j) \oplus \cdots \oplus \alpha(B_k) \oplus \cdots \oplus B_n.$$

However, we will have to "walk" to such a guess by permutations of support size at most 3. Let us show an example for $n = 4$, with the clause $C = (x_1 \vee x_3 \vee x_4)$. For ease of notation, we will identify the identity permutation with $b_1 b_2 \dots b_{12}$, and highlight in blue the permuted elements of each guess.

$b_1b_2b_3\ b_4b_5b_6\ b_7b_8b_9\ b_{10}b_{11}b_{12}$
$b_7b_2b_3\ b_4b_5b_6\ b_{10}b_8b_9\ b_1b_{11}b_{12}$
$b_7b_8b_3\ b_4b_5b_6\ b_{10}b_{11}b_9\ b_1b_2b_{12}$
$b_7b_8b_9\ b_4b_5b_6\ b_{10}b_{11}b_{12}\ b_1b_2b_3$
$b_8b_9b_7\ b_4b_5b_6\ b_{10}b_{11}b_{12}\ b_1b_2b_3$
$b_8b_9b_7\ b_4b_5b_6\ b_{11}b_{12}b_{10}\ b_1b_2b_3$
$b_8b_9b_7\ b_4b_5b_6\ b_{11}b_{12}b_{10}\ b_2b_3b_1$
$b_2b_9b_7\ b_4b_5b_6\ b_8b_{12}b_{10}\ b_{11}b_3b_1$
$b_2b_3b_7\ b_4b_5b_6\ b_8b_9b_{10}\ b_{11}b_{12}b_1$
$b_2b_3b_1\ b_4b_5b_6\ b_8b_9b_7\ b_{11}b_{12}b_{10}$

Naturally, this sequence of permutations can be inverted to obtain back the identity. More formally, let us introduce notation $\alpha(i, j, k)$ for the permutation that applies $\alpha \in S_3$ to positions i, j, k, and similarly, we define $\beta(i, j, k)$. Then, the sequence of guesses corresponding to a clause $C = (x_i \vee x_j \vee x_k)$ is:

Table 2 Correctness of the sequence of clause guesses.

$(\sigma^{(1)}, \sigma^{(2)}, \sigma^{(3)})$	(α,α,α)	(α,α,β)	(α,β,α)	(α,β,β)	(β,α,α)	(β,α,β)	(β,β,α)	(β,β,β)
$b_1b_2b_3\ b_4b_5b_6\ b_7b_8b_9$	0	0	0	0	0	0	0	0
$b_4b_2b_3\ b_7b_5b_6\ b_1b_8b_9$	0	0	0	0	0	0	0	0
$b_4b_5b_3\ b_7b_8b_6\ b_1b_2b_9$	0	0	0	0	0	0	0	0
$b_4b_5b_6\ b_7b_8b_9\ b_1b_2b_3$	0	0	0	0	0	0	0	0
$b_5b_6b_4\ b_7b_8b_9\ b_1b_2b_3$	0	0	0	0	0	0	0	0
$b_5b_6b_4\ b_8b_9b_7\ b_1b_2b_3$	0	0	0	0	0	0	0	0
$b_5b_6b_4\ b_8b_9b_7\ b_2b_3b_1$	0	0	0	0	0	0	0	0
$b_2b_6b_4\ b_5b_9b_7\ b_8b_3b_1$	3	2	2	1	2	1	1	0
$b_2b_3b_4\ b_5b_6b_7\ b_8b_9b_1$	6	4	4	2	4	2	2	0
$b_2b_3b_1\ b_5b_6b_4\ b_8b_9b_7$	9	6	6	3	6	3	3	0

1. $\alpha(3i-2, 3j-2, 3k-2)$,
2. $\alpha(3i-1, 3j-1, 3k-1)$,
3. $\alpha(3i, 3j, 3k)$,
4. $\alpha(3i-2, 3i-1, 3i)$,
5. $\alpha(3j-2, 3j-1, 3j)$,
6. $\alpha(3k-2, 3k-1, 3k)$,
7. $\beta(3i-2, 3j-2, 3k-2)$,
8. $\beta(3i-1, 3j-1, 3k-1)$,
9. $\beta(3i, 3j, 3k)$.

Their corresponding scores are: $(0,0,0,0,0,0,1,2,3)$. Consecutive "clause gadgets" can be chained by appending the inverse walk back to id_N between clauses, with the corresponding scores. We now make the main claim for the correctness of the reduction.

▷ Claim 24. A permutation $\sigma^\star$ is compatible with the scores constructed thus far if and only if we have

$$\sigma^\star = \sigma^{(1)}(B_1) \oplus \sigma^{(2)}(B_2) \oplus \cdots \oplus \sigma^{(n)}(B_n)$$

where each $\sigma^{(i)}$ is either $\alpha := (2\,3\,1)$ or $\beta := (3\,1\,2)$, and for each clause $(x_i \vee x_j \vee x_k) \in \varphi$, exactly one of $\sigma^{(i)}, \sigma^{(j)}, \sigma^{(k)}$ equals α.

Proof. The first part is direct from Equation (3) and the fact that the identity guess id_N has a score of 0, since α and β are the only permutations in S_3 without fixed points.

The second part can be checked mechanically. It suffices to consider clause $(x_1 \vee x_2 \vee x_3)$ without loss of generality, and for each of the 8 possibilities for $(\sigma^{(1)}, \sigma^{(2)}, \sigma^{(3)})$, observe that the only ones that give the scores $(0,0,0,0,0,0,1,2,3)$ are those with exactly one α. This is shown in Table 2, and concludes the proof of the claim. ◁

Having proven Claim 24, we will proceed to prove the correctness of the reduction.

▷ Claim 25. If φ is a YES-instance for Monotone-1-in-3-SAT, then the sequence of guesses and scores $(\pi_1, \ldots, \pi_T), (b_1, \ldots, b_T)$ constructed above is a YES-instance for 3-Local-PM-SAT.

Proof. Let τ be the satisfying assignment for φ which witnesses its membership in Monotone-1-in-3-SAT. Then, let $\sigma^\star$ be the permutation

$$\sigma^{(1)}(B_1) \oplus \sigma^{(2)}(B_2) \oplus \cdots \oplus \sigma^{(n)}(B_n)$$

where

$$\sigma^{(i)} = \begin{cases} \alpha & \text{if } \tau(x_i) = 1 \\ \beta & \text{otherwise.} \end{cases}$$

Then, by Claim 24 we have that $\sigma^\star$ is compatible with $(\pi_1, \ldots, \pi_T), (b_1, \ldots, b_T)$, thus proving it is a YES-instance for 3-Local-PM-SAT. ◁

▷ Claim 26. If $(\pi_1, \ldots, \pi_T), (b_1, \ldots, b_T)$ is a YES-instance for 3-Local-PM-SAT, then φ is a YES-instance for Monotone-1-in-3-SAT.

Proof. Directly by Claim 24 it suffices to take the assignment τ that assigns each x_i to 1 if $\sigma^{(i)} = \alpha$, and to 0 otherwise. ◁

Since the reduction can clearly be constructed in polynomial time, we conclude the proof. ◀

Finally, we show that 2-Local-PM-SAT can be solved in randomized polynomial time, by formulating it as a search over perfect matchings with parity constraints. In particular, we consider the following problem.

PROBLEM :	t-Dimensional Parity Perfect Matching
INPUT :	A $G = (V, E)$, a weight function $w\colon E \to \mathbb{Z}^{\geqslant 0}$, a function $\gamma\colon E \to \mathbb{F}_2^t$, a constant $\mathbf{c} \in \mathbb{F}_2^t$, and a number $r \in \mathbb{Z}$.
OUTPUT :	A perfect matching M of G such that $w(M) := \sum_{e \in M} w(e) = r$, and such that $\sum_{e \in M} \gamma(e) = \mathbf{c}$.

▶ **Theorem 27** ([8, Thm. 5.6]). *There is a randomized algorithm that solves the t-Dimensional Parity Perfect Matching problem, over instances on n vertices, in time $O(n^6 \lg^2 n \cdot \max_{e \in E(G)} w(e))$, with probability of success $1 - o(1)$.*

Strictly speaking, the algorithm of Geelen and Kapadia [8] is intended for minimizing $w(M)$, but this is only done in step 3 of their evaluation algorithm, where the minimum exponent of the variable z is considered; by considering the term including z^r instead, from the same polynomial, we obtain the form of Theorem 27.

▶ **Theorem 28.** 2-Local-PM-SAT *is in* RP*, and in particular, can be solved in time $O(n^7 \lg^2 n)$ with success probability $1 - o(1)$.*

Proof. Let $(\pi_1, \ldots, \pi_T)$ and $(b_1, \ldots, b_T)$ be an instance of 2-Local-PM-SAT over S_n. Then, as in Section 2, we identify a permutation $\sigma \in S_n$ with the perfect matching $B_\sigma := \big\{\{i, \sigma(i)\} : i \in [n]\big\}$ in the complete bipartite graph $K_{n,n}$, and note that $b(\pi_t, \sigma) = |B_{\pi_t} \cap B_\sigma|$.

We can assume without loss of generality that the transcript contains no consecutive guesses being equal, i.e., $\pi_{t+1} \neq \pi_t$, since otherwise we can remove the consecutive duplicates from the transcript without altering satisfiability. Because of the ℓ_2-locality restriction, for each $t \in [T-1]$ there is a (unique) transposition $\omega_{t+1} = (a_t\, b_t)$ such that $\pi_{t+1} = \pi_t \circ \omega_{t+1}$. Let $u_t := \pi_t(a_t)$, and $v_t := \pi_t(b_t)$, and define the four edges

$$e_t^1 := \{a_t, u_t\}, \quad e_t^2 := \{b_t, v_t\}, \quad f_t^1 := \{a_t, v_t\}, \quad f_t^2 := \{b_t, u_t\}.$$

Then $B_{\pi_t} \triangle B_{\pi_{t+1}} = \{e_t^1, e_t^2, f_t^1, f_t^2\}$, and only these two positions can change their contribution to the black-peg score when passing from π_t to π_{t+1}. Let $\Delta_t := b_{t+1} - b_t$. Letting $B_{\sigma^\star}$ be the matching associated to a secret permutation $\sigma^\star$ compatible with the scores (if any exists), shows:

$$\begin{aligned}
\Delta_t = 2 &\iff \{f_t^1, f_t^2\} \subseteq B_{\sigma^\star},\\
\Delta_t = -2 &\iff \{e_t^1, e_t^2\} \subseteq B_{\sigma^\star},\\
\Delta_t = 1 &\iff \left|B_{\sigma^\star} \cap \{f_t^1, f_t^2\}\right| = 1 \text{ and } B_{\sigma^\star} \cap \{e_t^1, e_t^2\} = \emptyset,\\
\Delta_t = -1 &\iff \left|B_{\sigma^\star} \cap \{e_t^1, e_t^2\}\right| = 1 \text{ and } B_{\sigma^\star} \cap \{f_t^1, f_t^2\} = \emptyset,\\
\Delta_t = 0 &\iff B_{\sigma^\star} \cap \{e_t^1, e_t^2, f_t^1, f_t^2\} = \emptyset.
\end{aligned}$$

Consequently, we will construct an instance of perfect matching with the following constraint sets: a set $\mathcal{F}$ of edges that must necessarily be present in the secret perfect matching $B_{\sigma^\star}$, a set $\mathcal{N}$ of edges that must necessarily *not* be present in the secret perfect matching $B_{\sigma^\star}$, and a set $\mathcal{C}$ of pairs of edges such that for every pair $C \in \mathcal{C}$, exactly one of the edges in C must be present in the secret perfect matching $B_{\sigma^\star}$. These sets of constraints are constructed by Algorithm 2.

Algorithm 2 Construct Constraint Sets $\mathcal{C}, \mathcal{F}, \mathcal{N}$.

$\mathcal{C} \leftarrow \emptyset, \mathcal{F} \leftarrow \emptyset, \mathcal{N} \leftarrow \emptyset$
for $t = 1$ **to** $T - 1$ **do**
  if $\Delta_t = 2$ **then**
    $\mathcal{F} \leftarrow \mathcal{F} \cup \{f_t^1, f_t^2\}$
    $\mathcal{N} \leftarrow \mathcal{N} \cup \{e_t^1, e_t^2\}$
  else if $\Delta_t = -2$ **then**
    $\mathcal{F} \leftarrow \mathcal{F} \cup \{e_t^1, e_t^2\}$
    $\mathcal{N} \leftarrow \mathcal{N} \cup \{f_t^1, f_t^2\}$
  else if $\Delta_t = 1$ **then**
    $\mathcal{N} \leftarrow \mathcal{N} \cup \{e_t^1, e_t^2\}$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\{f_t^1, f_t^2\}\}$
  else if $\Delta_t = -1$ **then**
    $\mathcal{N} \leftarrow \mathcal{N} \cup \{f_t^1, f_t^2\}$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\{e_t^1, e_t^2\}\}$
  else if $\Delta_t = 0$ **then**
    $\mathcal{N} \leftarrow \mathcal{N} \cup \{f_t^1, f_t^2, e_t^1, e_t^2\}$

By the previous analysis, we have the following observation.

▶ **Observation 29.** *There exists a secret permutation $\sigma^\star$ compatible with the transcript $(\pi_1, \ldots, \pi_T), (b_1, \ldots, b_T)$ if and only if $K_{n,n}$ has a perfect matching $M^\star$ such that:*

1. $|M^\star \cap B_{\pi_1}| = b_1$,
2. $|M^\star \cap \mathcal{N}| = 0$,
3. *for every* $e \in \mathcal{F}$, $|M^\star \cap \{e\}| = 1$,
4. *for every pair of edges* $C \in \mathcal{C}$, $|M^\star \cap C| = 1$.

We will now show how to decide the existence of such a matching $M^\star$ by reducing to the t-Dimensional Parity Perfect Matching problem. First, we define the edge weight function $w\colon E(K_{n,n}) \to \mathbb{Z}^{\geqslant 0}$ as follows:

$$w(e) := \begin{cases} n & \text{if } e \in \mathcal{N},\\ 1 & \text{if } e \notin \mathcal{N} \text{ and } e \notin B_{\pi_1},\\ 0 & \text{otherwise.} \end{cases}$$

Now, note that, for any perfect matching M of $K_{n,n}$, we have

$$w(M) := \sum_{e \in M} w(e) = n \cdot |M \cap \mathcal{N}| + |(M \setminus \mathcal{N}) \setminus B_{\pi_1}|,$$

from where (i) if $|M \cap \mathcal{N}| > 0$, then $w(M) \geqslant n$, and (ii) if $|M \cap \mathcal{N}| = 0$, then $w(M) = n - |M \cap B_{\pi_1}|$.

Therefore any perfect matching M of $K_{n,n}$ satisfies conditions 1 and 2 of Observation 29 if and only if $w(M) = n - b_1$.

Now, let $t := |\mathcal{C}| + |\mathcal{F}|$. Let $C_1, \ldots, C_m$ be an arbitrary enumeration of $\mathcal{C}$, and $e_1, \ldots, e_f$ an arbitrary enumeration of $\mathcal{F}$. For each $1 \leqslant i \leqslant t$, let $\mathbf{e}_i \in \mathbb{F}_2^t$ denote the i-th standard basis vector. We now define a label function $\gamma \colon E(K_{n,n}) \to \mathbb{F}_2^t$ by

$$\gamma(e) = \left(\sum_{i \text{ s.t. } e \in C_i} \mathbf{e}_i \right) + \left(\sum_{i=1}^{f} \mathbf{e}_{i+m} \cdot \mathbb{1}_{[e=e_i]} \right). \tag{4}$$

We are now ready for the main claim that will prove the theorem. Let us write $\mathbf{1}$ for the vector of all ones in $\mathbb{F}_2^t$.

▷ Claim 30. There exists a secret $\sigma^\star$ compatible with the transcript $(\pi_1, \ldots, \pi_T), (b_1, \ldots, b_T)$ if and only if $K_{n,n}$ has a perfect matching M such that $\gamma(M) := \sum_{e \in M} \gamma(e) = \mathbf{1}$ and $w(M) = n - b_1$.

Proof. We have shown above that a matching M of $K_{n,n}$ satisfies conditions 1 and 2 of Observation 29 if and only if $w(M) = n - b_1$.

Now we show that M satisfies conditions 3 and 4 if and only if $\gamma(M) = \mathbf{1}$. Let us use notation $\mathbf{v}[j]$ for the j-th coordinate of a vector $\mathbf{v}$, and we will show that $\gamma(M)[j] = 1$ for $j \in \{1, \ldots, m+f\}$. We consider two cases:

- **(Case 1: $j \in \{m+1, \ldots, m+f\}$).** In this case, $j = i+m$ for a unique $i \in \{1, \ldots, f\}$, so only the second sum of (4) is non-zero, so we have

  $$\gamma(M)[j] = \sum_{e \in M} \mathbb{1}_{[e=e_i]} = |M \cap \{e_i\}| \bmod 2 = |M \cap \{e_i\}|,$$

  from where $\gamma(M)[j] = 1$ if and only if $|M \cap \{e_i\}| = 1$ (i.e., condition 3. holds for e_i).
- **(Case 2: $j \in \{1, \ldots, m\}$).** In this case, only the first sum of (4) is non-zero, so we have

  $$\gamma(M)[j] = \sum_{e \in M} \sum_{i \text{ s.t. } e \in C_i} \mathbb{1}_{i=j} = \sum_{e \in M} \mathbb{1}_{e \in C_j} = |M \cap C_j| \bmod 2,$$

  from where $\gamma(M)[j] = 1$ if and only if $|M \cap C_j|$ is odd. But since $|C_j| = 2$, the only possibility is $|M \cap C_j| = 1$. Thus, $\gamma(M)[j] = 1$ if and only if condition 4 holds for C_j.

This concludes the proof. ◁

Applying Theorem 27 to the instance $(G, w, \gamma, \mathbf{1})$, and then using Claim 30, we conclude the proof. The final runtime results from the fact that $\max_e w(e) = n$. ◀

References

1 Peyman Afshani, Manindra Agrawal, Benjamin Doerr, Carola Doerr, Kasper Green Larsen, and Kurt Mehlhorn. The query complexity of a permutation-based variant of mastermind. *Discrete Applied Mathematics*, 260:28–50, 2019. doi:10.1016/J.DAM.2019.01.007.

2 Andreas Darmann and Janosch Döcker. On simplified NP-complete variants of Monotone 3 -Sat. *Discrete Applied Mathematics*, 292:45–58, 2021. doi:10.1016/j.dam.2020.12.010.

3 MC de Bondt. NP-completeness of master mind and minesweeper, 2004.

4 Benjamin Doerr, Carola Doerr, Reto Spöhel, and Henning Thomas. Playing mastermind with many colors. *Journal of the ACM (JACM)*, 63(5):1–23, 2016. doi:10.1145/2987372.

5 Benjamin Doerr and Carola Winzen. Playing mastermind with constant-size memory, 2011. arXiv:1110.3619.

6 Mourad El Ouali and Volkmar Sauerland. The exact query complexity of yes-no permutation mastermind. *Games*, 11(2):19, 2020. doi:10.3390/G11020019.

7 Pál Erdős, Alfréd Rényi, and Vera T. Sós. On a problem of graph theory. *Stud. Sci. Math. Hung.*, 1:215–235, 1966.

8 Jim Geelen and Rohan Kapadia. Computing girth and cogirth in perturbed graphic matroids. *Combinatorica*, 38(1):167–191, January 2017. doi:10.1007/s00493-016-3445-3.

9 Christian Glazik, Gerold Jäger, Jan Schiemann, and Anand Srivastav. Bounds for the static permutation mastermind game. *Discrete Mathematics*, 344(112253), 2021. doi:10.1016/j.disc.2020.112253.

10 E Mark Gold. Complexity of automaton identification from given data. *Information and Control*, 37(3):302–320, 1978. doi:10.1016/S0019-9958(78)90562-4.

11 Michael T Goodrich. On the algorithmic complexity of the mastermind game with black-peg results. *Information Processing Letters*, 109(13):675–678, 2009. doi:10.1016/J.IPL.2009.02.021.

12 Gerold Jäger and Marcin Peczarski. The number of pessimistic guesses in generalized mastermind. *Information Processing Letters*, 109(12):635–641, 2009. doi:10.1016/J.IPL.2009.02.016.

13 jasminandjulio. TikTok - Make Your Day — jasminandjulio. https://www.tiktok.com/@jasminandjulio/video/7333730349534727470. [Accessed 12-01-2026].

14 Thomas KővÃąri, Vera SÃşs, and PÃąl TurÃąn. On a problem of K. Zarankiewicz. *Colloquium Mathematicae*, 3(1):50–57, 1954. URL: http://eudml.org/doc/210011.

15 Donald Ervin Knuth. The computer as master mind. *Journal of Recreational Mathematics*, 9:1–6, 1977.

16 Ker-I Ko and Shia-Chung Teng. On the number of queries necessary to identify a permutation. *Journal of Algorithms*, 7(4):449–462, 1986. doi:10.1016/0196-6774(86)90013-1.

17 Maxime Larcher, Anders Martinsson, and Angelika Steger. Solving static permutation mastermind using $O(n \log n)$ queries. *The Electronic Journal of Combinatorics*, 29(1), January 2022. doi:10.37236/10280.

18 Renyuan Li and Shenglong Zhu. Playing mastermind with wordle-like feedback. *The American Mathematical Monthly*, 131(5):390–399, February 2024. doi:10.1080/00029890.2024.2308489.

19 Anders Martinsson and Pascal Su. Mastermind with a linear number of queries. *Combinatorics, Probability and Computing*, 33(2):143–156, 2024. doi:10.1017/S0963548323000366.

20 Mourad El Ouali, Christian Glazik, Volkmar Sauerland, and Anand Srivastav. On the query complexity of black-peg ab-mastermind, 2016. arXiv:1611.05907.

21 Jeff Stuckman and Guo-Qiang Zhang. Mastermind is np-complete, 2005. arXiv:cs/0512049.

22 Yufei Zhao. *Graph Theory and Additive Combinatorics: Exploring Structure and Randomness*. Cambridge University Press, Cambridge, 2023. doi:10.1017/9781009310956.