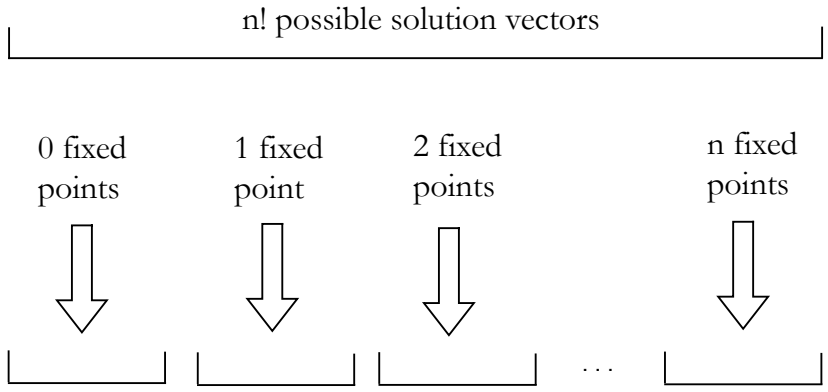
Mastermind with No Repeats: Lower Bound for Guessing Strategies

1. *Game.* In this version of Mastermind, there are n spots and n colors. All guesses and solutions are permutations.

2. *Trivial Lower Bound.* We have $n!$ possible hidden vectors, and each query has n possible responses.[1] Since we must distinguish between every pair of vectors, we must ask at least
$$\log_n(n!) \sim \left(n - \frac{n}{\ln(n)}\right)$$

3. *Buckets.* In order to beat the trivial lower bound, we use a more accurate picture of the distribution of the possible solutions after feedback from a given guess. For any guess, we can split the possible solutions up into groups, which we will call *buckets*; one bucket for each possible response given by the codemaker. For any guess we make, the number of possible solutions that give the response "k (black) hits" is the number of permutations with k fixed points with respect to our guess vector.



   The size of a bucket of possible solutions with k hits can be counted easily: we choose the k colors that are fixed points (wrt our guess vector), then we permute the remaining $n-k$ colors without any fixed points, giving $D(n-k)$, the derangements on an $(n-k)$-element vector:
$$\binom{n}{k} D(n-k)$$
   This tells us the initial size of each bucket, and as more guesses are made, the size of each bucket can only decrease, no matter which guess is considered.

4. *Notation.* Let $S_n(t)$ be the minimum number of possible solutions remaining after t guesses for any guessing strategy. We compute this lower bound by using a set of guesses which are guaranteed to eliminate at least as many solutions as any other guess we could make. The worst-case scenario for this set of guesses therefore

[1] The possible responses are $0, 1, \ldots, n-2, n$, since it is impossible to get $n-1$ black hits with no repeats and $n = k$.

provides a lower bound for the number of guesses required by any guessing strategy to guarantee finding the hidden vector.

5. **Recurrence Relation on Number of Possible Solutions after t Turns:**

   (a) We will inductively prove a lower bound, $S_n(t)$. By the discussion above, we know the maximum size of each bucket containing possible solutions. The best possible guess distributes the remaining solutions as evenly as possible so that the bucket containing the most remaining solutions is as small as possible. This best-possible distribution will fill some number of smaller buckets entirely and split the remaining solutions evenly into some number x larger buckets. That is, there are x incompletely filled buckets.

   (b) At the beginning of the t-th turn, we have at least $S_n(t-1)$ possible solutions remaining. As we showed before, the size of each bucket is bounded by its original size of $\binom{n}{k}D(n-k)$. Thus the number of solutions remaining $(t-1)$-th turn relates to the value of $S_n(t)$ as follows:

   Let $B_i(t)$ be the exact number of possible solutions remaining in the bucket holding possible solutions with i fixed points during the t-th turn (after submitting the guess vector, but before feedback). So

$$\begin{aligned} S_n(t-1) &= \sum_{i=0}^{n} B_i(t) \\ &= \sum_{i=0}^{x-1} B_i(t) + \sum_{i=x}^{n} B_i(t) \\ &\leq \sum_{i=0}^{x-1} S_n(t) + \sum_{i=x}^{n} \binom{n}{i} D(n-i) \quad \text{for optimal } x \\ &= x \cdot S_n(t) + \sum_{i=x}^{n} \binom{n}{i} D(n-i) \end{aligned}$$

   Once we know $S_n(t)$, picking x too large results in some smaller boxes overfilled. Picking x too small results in some larger boxes overfilled. So changing x to be something other than optimal always increases the RHS of this equation. Thus we have

$$S_n(t-1) \leq x \cdot S_n(t) + \frac{n!}{x!} \quad \forall x$$

   Where t is the number of turns taken, $S_n(t)$ is the number of possible solutions remaining after the t-th turn, and $D(k)$ is the derangements on a set of size k.

6. *Claim.*

$$\sum_{i=x}^{n} \binom{n}{i} D(n-i) \leq \frac{n!}{x!}$$

This inequality leads to our bound on $S_n(t)$ given below.[2]

Proof.

$$\begin{aligned}
\sum_{i=x}^{n} \binom{n}{i} D(n-i) &= \frac{n!}{x!} \sum_{i=x}^{n} \frac{x!}{i!(n-i)!} D(n-i) \\
&\qquad \text{we use the fact that } \frac{x!}{i!} \leq \frac{1}{(i-x)!} \\
&\leq \frac{n!}{x!} \sum_{i=x}^{n} \frac{1}{(i-x)!(n-i)!} D(n-i) \\
&\qquad \text{let } k = i - x \\
&= \frac{n!}{x!} \sum_{k=0}^{n-x} \frac{1}{k!(n-x-k)!} D(n-x-k) \\
&= \frac{n!}{x!(n-x)!} \sum_{k=0}^{n-x} \binom{n-x}{k} D(n-x-k) \\
&\qquad \text{note the summation counts all possible permutations} \\
&= \frac{n!}{x!(n-x)!} (n-x)! \\
&= \frac{n!}{x!}
\end{aligned}$$

□

(a) Plug the above result into the inequality to get

$$S_n(t-1) \leq x \cdot S_n(t) + \frac{n!}{x!}$$

So we have

$$S_n(t) \geq \frac{1}{x} \left(S_n(t-1) - \frac{n!}{x!} \right)$$

[2] A combinatorial argument can be made as follows: The LHS denotes the number of permutations of an n-element vector which have at least x fixed points. Suppose we choose x fixed points and simply permute the rest of the vector. This gives $\binom{n}{x}(n-x)! = \frac{n!}{x!}$ possible permutations. Clearly this includes all vectors with at least x fixed points (and overcounts by some margin), so the inequality holds.

Theorem 1. *For all n, and for any constant C_n, we have*

$$\frac{S_n(t)}{n!} \geq \frac{C_n! - (H_{C_n+t} - H_{C_n})}{(C_n+t)!}$$

$$\textit{where} \quad 0 \leq t \leq n - C_n$$

Here we use H_n to denote the n-th harmonic number.[3]

Proof. While this equation holds for all x, to achieve a desirable bound, we will let $x = t + C_n$ for any positive constant $C_n = C_n(n)$. We have

Base Case. When $t = 0$, we have $S_n(0) = n!$ and

$$\frac{S_n(0)}{n!} \geq \frac{C_n! - (H_{C_n} - H_{C_n})}{C_n!} = 1$$

Inductive Step. Assume the induction hypothesis holds for $t - 1$.

$$\frac{S_n(t)}{n!} \geq \frac{1}{x}\left(\frac{S_n(t-1)}{n!} - \frac{1}{x!}\right)$$

Let $x = C_n + t$.

$$\frac{S_n(t)}{n!} \geq \frac{1}{C_n+t}\left(\frac{S_n(t-1)}{n!} - \frac{1}{(C_n+t)!}\right)$$

Inductively,

$$\begin{aligned}\frac{S_n(t)}{n!} &\geq \frac{1}{C_n+t}\left(\frac{C_n! - (H_{C_n+t} - H_{C_n})}{(C_n+t-1)!} - \frac{1}{(C_n+t)!}\right) \\ &\geq \left(\frac{C_n! - (H_{C_n+t-1} - H_{C_n}) - \frac{1}{C_n+t}}{(C_n+t)!}\right) \\ &\geq \left(\frac{C_n! - (H_{C_n+t} - H_{C_n})}{(C_n+t)!}\right)\end{aligned}$$

Then

$$\begin{aligned}S_n(n - C_n) &\geq n!\left(\frac{C_n! - (H_n - H_{C_n})}{n!}\right) \\ &\geq C_n! - (H_n - H_{C_n})\end{aligned}$$

Thus the inequality holds for all n.

□

[3] $H_n = \sum_{i=1}^{n} \frac{1}{i}$

We choose C_n by the following reasoning: we want to achieve a lower bound of $n - C_n$ turns. Thus, we want $S_n(n - C_n) > 1$, which follows if

$$C_n! - (H_n - H_{C_n}) > 1$$

For $C_n = \log\log n$, this is ≥ 1 for $n \geq e^{198}$.
For $C_n = \log n$, this is true for $n \geq 10$.

In conclusion, when n is sufficiently large, the minimum number of remaining possible solutions after $n - \log\log n$ guesses is at least

$$\begin{aligned} S_n(n - \log\log n) &\geq (\log\log n)! - (H_n - H_{\log\log n}) \\ &> 1 \end{aligned}$$

Thus there is no strategy that can reduce the set of possible solutions to one, and guarantee finding the codemaker's hidden vector, in $n - \log\log n$ turns or fewer.

Lower Bound on Guessing Effectiveness for Permutation Game

Theorem 2. *In Mastermind with no repeats with n spots and k colors, for any set of possible remaining solutions, there exists a guess vector for which any response will eliminate at least $1/nk$ of the remaining solutions.*

1. *Notation.*

   i. *Basics.* Let n be the number of spots in the hidden vector. Let k be the number of colors available. Since we allow no repeats in this variant of Mastermind, we have $k \geq n$. We will use single-count responses, which are the number of spots in which our guess vector and the hidden vector have the same color in that spot.

   ii. *Remaining Solution Set.* As the game proceeds, we maintain a set S that contains all possible solution vectors that haven't been ruled out by a previous guess and response.

   iii. *Bucket.* As defined in the previous section, a guess splits the remaining possible solutions into "buckets," where a bucket B_i contains all the remaining possible solutions with i fixed points with respect to the guess vector.

   iv. *Sub-Bucket.* Within a bucket, we will divide the vectors based on a chosen component, say the first element of each vector. Then there will be k sub-buckets D_i, where D_1 contains all vectors from the bucket with a 1 in the first spot, D_2 contains all those with a 2 in the first spot, and so on.

2. *Proof.* We give a proof by induction on n.

   Base Case. When $n = 1$, we have a single-element vector. So there are at most k possible solutions remaining at any step, thus $|S| \leq k$. If we guess one of the remaining solutions $v \in S$, this will eliminate 1 solution (v itself) if the response is 0 and $|S| - 1$ solutions if the response is 1. So we're guaranteed to eliminate at least $1/|S| \geq 1/(nk)$ solutions.

   Inductive Step. Assume that for $n-1$ spots and any number of colors $\widehat{k}$ (where $n \geq 2$), we are guaranteed that there is a guess that eliminates a fraction at least $1/(n\widehat{k})$ of the solutions.

   Consider a guess vector $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$. We will analyze the number of solutions eliminated from S by guessing σ in the worst-case. We divide S into n buckets $B_1, B_2, \ldots, B_n$, where B_i is as defined above (i fixed points with respect to σ). Consider the response j that eliminates as few possible solutions as possible from S, so $|B_j| = \max\{|B_1|, |B_2|, \ldots, |B_n|\}$.

   Suppose that

   $$|B_j| \leq \frac{nk-1}{nk} \cdot |S|$$

Then the response j would eliminate at least $|S|/(nk)$ solutions from S, and since j eliminates the fewest possible solutions of any response, all responses to σ eliminate a fraction at least $1/(nk)$ of the possible solutions in S.

Otherwise we have

$$|B_j| > \frac{nk-1}{nk} \cdot |S|$$

Case 1. If $j > 0$, the total number of hits among all vectors in B_j is $j \cdot |B_j|$. These hits are divided up among n spots, therefore by the pigeonhole principle, at least one position must have at least

$$\frac{j}{n} \cdot |B_j|$$

Vectors that match the guess at that spot. WLOG, we assume this is position one. We now divide the bucket B_j into sub-buckets D_i, where permutation $s \in B_j$ is put into sub-bucket D_{s_1}. From above, we have

$$|D_{\sigma_1}| \geq \frac{j}{n} \cdot |B_j|$$

We have two sub-cases:

i. $|D_{\sigma_1}| \leq \dfrac{n-1}{n} \cdot |B_j|$:
Then we apply the pigeonhole principle on the remaining solutions in B_j to find the second-largest sub-bucket: By our assumption, there are at least $|B_j|/n$ solutions remaining in B_j, and there are $k-1$ remaining sub-buckets D_i. By the pigeonhole principle, there exists a D_α such that

$$|D_\alpha| \geq \frac{1}{n} \cdot \frac{1}{k-1} \cdot |B_j|$$

If α is an element of σ, then WLOG let the color be σ_2. Then in this case we consider the guess vector $\overline{\sigma} = \{\sigma_2, \sigma_1, \sigma_3, \ldots, \sigma_n\}$. Otherwise, consider the guess vector $\overline{\sigma} = \{\alpha, \sigma_2, \sigma_3, \ldots, \sigma_n\}$. Consider the buckets C_i formed from the guess vector $\overline{\sigma}$. In either case, the vectors in D_{σ_1} will have at least one fewer hit with $\overline{\sigma}$ than with σ, and the guess vectors in D_α will have at least one more hit. Therefore, any response to $\overline{\sigma}$ which corresponds to some vectors in D_{σ_1} corresponds to none of the vectors in D_α and vice versa. So we know that any response to $\overline{\sigma}$ will entirely eliminate at least one of those two sets. From above, we know

$$\begin{aligned}
|D_{\sigma_1}| &\geq \frac{j}{n} \cdot |B_j| \\
&\geq \frac{j}{n} \cdot \frac{nk-1}{nk} \cdot |S| \\
&\geq \frac{1}{n} \cdot \frac{n}{nk} \cdot |S| \\
&= \frac{1}{nk} \cdot |S|
\end{aligned}$$

And

$$\begin{aligned}|D_\alpha| &\geq \frac{1}{n} \cdot \frac{1}{k-1} \cdot |B_j| \\ &\geq \frac{1}{nk-n} \cdot \frac{nk-1}{nk} \cdot |S| \\ &\geq \frac{1}{nk} \cdot |S|\end{aligned}$$

Since $\overline{\sigma}$ will always eliminate at least one of these two sets, we also have that $\overline{\sigma}$ will eliminate at least $|S|/(nk)$ vectors from S with any response.

ii. $|D_{\sigma_1}| > \dfrac{n-1}{n} \cdot |B_j|$:
Consider the set of vectors in D_{σ_1}. By definition, all of them have σ_1 as their first entry. Their remaining $n-1$ entries are made up of the remaining $k-1$ colors, and no two of them coincide in all spots $2, 3, \ldots, n$ (otherwise they would be the same vector). Therefore, if we consider this new set of $n-1$ vectors formed by taking spots $2, 3, \ldots, n$ of each vector in D_{σ_1}, they are a set of possible solutions for this Mastermind game with $n-1$ spots and $k-1$ colors. By the induction hypothesis, there is a guess of size $n-1$ that is guaranteed to eliminate $1/((n-1)(k-1))$ of these vectors. If we add σ_1 to the beginning of this guess, it becomes a guess of size n. For any response r, any of the vectors in D_{σ_1} will be eliminated if and only if their vector of spots $2, 3, \ldots, n$ was eliminated by the response $r-1$ in the smaller game of Mastermind. Therefore, this guess still guarantees that we eliminate

$$\begin{aligned}\frac{1}{(n-1)(k-1)}|D_{\sigma_1}| &\geq \frac{1}{(n-1)(k-1)} \cdot \frac{n-1}{n} \cdot |B_j| \\ &\geq \frac{1}{(k-1)} \cdot \frac{1}{n} \cdot \frac{nk-1}{nk} \cdot |S| \\ &\geq \frac{1}{nk-n} \cdot \frac{nk-1}{nk} \cdot |S| \\ &\geq \frac{1}{nk} \cdot |S|\end{aligned}$$

Case 2. If $j = 0$, we consider the first component of all vectors in the bucket B_0. Since $j = 0$ and $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, no vector in B_0 can contain σ_1 in the first position, so there are $k-1$ possible colors in the first position. By the pigeonhole principle, there must be one color which appears in the first spot of at least

$$\frac{1}{k-1} \cdot |B_0|$$

vectors in B_0. If this color appears in σ, WLOG let the color be σ_2. Then in this case we consider the guess vector $\overline{\sigma} = \{\sigma_2, \sigma_1, \sigma_3, \ldots, \sigma_n\}$. Otherwise, call this new color τ and consider the guess vector $\overline{\sigma} = \{\tau, \sigma_2, \sigma_3, \ldots, \sigma_n\}$. In either case, $\overline{\sigma}$ matches these $|B_0|/(k-1)$ vectors in either one or two positions.

Consider the buckets C_i formed by the guess $\overline{\sigma}$. We know

$$\begin{aligned}|C_1| + |C_2| &\geq \frac{1}{k-1} \cdot |B_0| \\ &\geq \frac{1}{k-1} \cdot \frac{nk-1}{nk} \cdot |S| \\ &\geq \frac{1}{k-1} \cdot \frac{nk-n}{nk} \cdot |S| \\ &= \frac{n}{nk} \cdot |S| \\ &\geq \frac{2}{nk} \cdot |S|\end{aligned}$$

Therefore we have either

$$|C_1| \geq \frac{1}{nk} \cdot |S| \qquad \text{or} \qquad |C_2| \geq \frac{1}{nk} \cdot |S|$$

Let this larger bucket be C_α, then suppose

$$|C_\alpha| \leq \frac{nk-1}{nk}|S|$$

Then getting the response α will eliminate at least $|S|/(nk)$ possible solutions from S, and getting any other response will eliminate at least $|C_\alpha| \geq |S|/(nk)$ possible solutions. Therefore the guess $\overline{\sigma}$ is guaranteed to eliminate at least $|S|/(nk)$ solutions from S.
Otherwise we have

$$|C_\alpha| > \frac{nk-1}{nk}|S|$$

Then, this satisfies the conditions for Case 1 above, where we have a guess whose largest bucket is B_j with $j > 0$.

□

3. The minimax algorithm is a guessing strategy described by Knuth [DK76]. At each turn, the algorithm assigns each of the possible guesses a score equal to the minimum number of solutions that they would eliminate over all possible responses from the codemaker. It then picks the guess with the maximum score.

   Corollary. The minimax algorithm on Mastermind with n spots, k colors, and no repeats takes at most

   $$\frac{\log\left(\frac{k!}{(k-n)!}\right)}{\log\left(\frac{nk}{nk-1}\right)}$$

   turns to find the hidden vector.

Proof. Each step in the minimax algorithm cuts the set of possible solutions S down by a factor of $(nk-1)/(nk)$ at each step. To complete the game, we need to cut S down by

$$\frac{1}{\left(\frac{k!}{(k-n)!}\right)}$$

Thus the required number of steps is

$$\log_{\left(\frac{nk-1}{nk}\right)}\left(\frac{1}{\frac{k!}{(k-n)!}}\right)$$

Which is equivalent to the claim. □

This gives us the following asymptotic bounds:

i. The minimax algorithm for Mastermind with n spots, k colors, and no repeats takes at most $O(n^2 k \log k)$ turns to find the hidden vector.

ii. In the same game where $n = k$, the minimax algorithm takes at most $O(n^3 \log n)$ turns to find the hidden vector.

Both of these asymptotic bounds are actually strict upper bounds, and each of them differs from the exact bound by a constant factor which approaches 1 as n approaches infinity.

Extension of Previous Section to Repeated Colors

In order to replicate the bound of nk with repeated colors, we need to make the following assumption:[4] There exists a guess vector v such that all of the solutions in set S correspond to a single response r after making the guess v. Note that this is always satisfied if the solutions are any set of solutions remaining after any set of guesses in an actual game of Mastermind. We simply let the guess vector v be any guess vector that has already been guessed. Clearly, all solutions remaining correspond to whichever response was given to that guess vector, since guesses corresponding to any other response were eliminated. We will also prove the theorem for the special case of the very first guess separately.

Theorem 3. *In a game of Mastermind with n spots and k colors, for any set of remaining solutions that can actually be achieved during a game of Mastermind, there is always a guess for which any response will eliminate at least $1/nk$ of the remaining solutions.*

Proof. We will follow the same logic as taken in the previous proof, but will describe only the parts that differ from the previous proof.

1. Because of the problem statement, we will separately prove the base case where S is the set of all k^n possible solutions before any guesses have been made. In this case, we guess the vector $\{1, 1, \ldots, 1\}$. The bucket B_j has size

$$|B_j| = \binom{n}{j}(k-1)^{n-j}$$

So the greatest fraction of possible solutions remaining is

$$\max_j \left(\frac{\binom{n}{j}(k-1)^{n-j}}{k^n} \right)$$

When $k \geq n$, we have

$$\begin{aligned} \frac{\binom{n}{j}(k-1)^j}{k^n} &\leq \frac{n^{n-j}(k-1)^j}{k^n} \\ &\leq \frac{k^{n-j}(k-1)^j}{k^n} \\ &= \frac{(k-1)^j}{k^j} \\ &\leq \frac{k-1}{k} \end{aligned}$$

Therefore, we always eliminate a fraction at least $1/k \geq 1/nk$ of the solutions.

[4]The proof in the previous section can be extended easily to general n and k: Substitute to show that there is always a guess that eliminates a factor of $1/n^2k$ possible solutions at each step. For a tighter bound with repeated colors, however, we need to make more restrictive assumptions.

When $k < n$, it is difficult to find the size of one bucket compared to the entire set, so instead we will compare one bucket to the sum of itself and the next bucket. This ratio will always be at least the ratio of one bucket compared to the entire set.

$$\begin{aligned}\frac{|B_{j+1}|}{|S|} &\leq \frac{|B_{j+1}|}{|B_j| + |B_{j+1}|} \\ &\leq \frac{1}{\frac{|B_j|}{|B_{j+1}|} + 1} \\ &\leq \frac{1}{\frac{\binom{n}{j}(k-1)^{n-j}}{\binom{n}{j+1}(k-1)^{n-j-1}} + 1} \\ &\leq \frac{1}{\frac{j+1}{n-j}(k-1) + 1}\end{aligned}$$

We plug in $k \geq 2$ (the entire problem is trivial when $k = 1$, as there is only 1 solution).

$$\begin{aligned}\cdots \quad &\leq \frac{1}{\frac{j+1}{n-j} + 1} \\ &\leq \frac{n-j}{n+1} \\ &\leq \frac{n}{n+1} \\ &= 1 - \frac{1}{n+1} \\ &\leq 1 - \frac{1}{nk}\end{aligned}$$

Additionally, for bucket 0, we have:

$$\begin{aligned}\frac{|B_0|}{|S|} &= \frac{(k-1)^n}{k^n} \\ &\leq \frac{k-1}{k} \\ &\leq 1 - \frac{1}{nk}\end{aligned}$$

So we have that the first guess of any game will always eliminate at least $1/nk$ of the remaining solutions.

2. The argument for the base case where $n = 1$ proceeds exactly as in the previous section.

3. The induction hypothesis remains almost the same: Assume that for $n - 1$ spots and any number of colors $\widehat{k}$ (where $n \geq 2$), we are guaranteed that there is a guess that eliminates a fraction at least $1/(n\widehat{k})$ of the solutions, given that there is a guess such that all solutions fall into the same bucket.

4. *Main Proof.*
Let this bucket be j. From our assumptions $B_j = S$. Then we have two subcases as before:

Case 1. If $j > 0$, we make the following adjustments to the previous proof.

i. For the first subcase, we are now allowed repeats, so we can always let

$$\overline{\sigma} = \{\alpha, \sigma_2, \sigma_3, \ldots, \sigma_n\}$$

The proof of the rest of this subcase follows.

ii. For the second subcase, we proceed as the previous section did by considering D_{σ_1}, but now we are no longer guaranteed that the spots 2 through n of these vectors have no instance of σ_1 since colors can be repeated. Whereas before, inducted on $n-1$ spots with $k-1$ possible colors, we now have to induct on $n-1$ spots still with k possible colors. This is fine, though, as by the induction hypothesis we can therefore eliminate

$$\begin{aligned}\frac{1}{(n-1)k} \cdot |D_{\sigma_1}| &\geq \frac{1}{(n-1)k} \cdot \frac{n-1}{n} \cdot |B_j| \\ &= \frac{1}{nk} \cdot |S|\end{aligned}$$

It is important to check that the induction hypothesis holds- namely, that there exists a guess of size $n-1$ such that all of the $(n-1)$-vectors we're considering produce the same response.

However, by our assumptions, σ is an n-vector such that all of the n vectors fall into a single bucket j. Since every vector we're considering for induction is in the same sub-bucket D_{σ_1}, if we consider only spots 2 through n, all of these $(n-1)$-vectors will have exactly one fewer hit than they did as n-vectors, (as all of them had a hit in the first spot), so all of them fall into the same bucket (bucket $j-1$) for the guess $\{\sigma_2, \sigma_3, \ldots, \sigma_n\}$, so the condition is satisfied.

Case 2. If $j = 0$, we can no longer find a new $\overline{\sigma}$ and switch to Case 1, so instead we induct similarly to Case 1. We split the bucket up into sub-buckets as before. By pigeonhole principle, there must exist an α such that $|D_\alpha| \geq |B_j|/(k-1) \geq |S|/(nk)$ (since $j = 0$ we have no hits, so $|D_{\sigma_1}| = 0$). We then split into two subcases based on the size of D_α compared to the size of the entire bucket as we did in Case 1 and the proof proceeds in exactly the same manner.

□

Optimality of Mastermind for $(n, k) = (4, 6)$

Up to isomorphism, the first guess for this game will be one of the following:

$$\{(0,0,0,0), (0,0,0,1), (0,0,1,1), (0,0,1,2), (0,1,2,3)\}$$

Their respective worst-case responses and remaining solutions after that response are

$$\{(0,0) \to 625, (1,0) \to 317, (0,0) \to 256, (0,1) \to 276, (0,2) \to 312\}$$

For each of these guesses, there exists a response that leaves at least 256 possible remaining solutions, as shown above. After this point, for every possible guess made, there will be at most fourteen possible responses given. Thus, at every step, the number of possible solutions remaining is reduced by at most a factor of fourteen. We use this information-theoretic lower bound to show that after the third guess, there will still be at least two solutions remaining:

$$\left\lceil \frac{\left\lceil \frac{256}{14} \right\rceil}{14} \right\rceil = 2$$

Thus no strategy could guess the hidden vector on the fourth turn, so it is not possible for a deterministic guessing strategy to guarantee finding the hidden vector in fewer than five turns.

Lower Bound for Non-Adaptive Strategies Using Entropy

Introduction to Entropy

For a formal introduction to some of the properties of entropy, the authors found the paper "Three tutorial lectures on entropy and counting" by David Galvin to be a helpful resource. Here we will describe a few of the relevant properties of entropy and provide informal justification.

Entropy is a metric for characterizing a random variable, which can be roughly described as the expected amount of information yielded by that random variable when it takes on one of the values in its domain. The entropy of a random variable X with domain D is defined as

$$H(X) = \sum_{x \in D} -\Pr[X = x] \cdot \log_2 (\Pr[X = x])$$

Entropy has the following properties:

Property 1 If X, Y uniquely determine each others' outcomes, then $H(X) = H(Y)$.

Proof. For every outcome $x_i \in D$, let y_i be the outcome such that $\{X = x_i\} \Leftrightarrow \{Y = y_i\}$. Therefore, $\Pr[X = x_i] = \Pr[Y = y_i]$ for every pair of corresponding outcomes x_i and y_i. Then we have

$$\begin{aligned} H(X) &= \sum_i -\Pr[X = x_i] \cdot \log_2(\Pr[X = x_i]) \\ &= \sum_i -\Pr[Y = y_i] \cdot \log_2(\Pr[Y = y_i]) \\ &= H(Y) \end{aligned}$$

□

Property 2 (Subadditivity) A vector $X = (X_1, X_2, \ldots, X_n)$ of random variables (which is itself a random variable) has

$$H(X) \leq \sum_{i=1}^{n} H(X_i)$$

Justification. To find the total information gained from X, we sum the information gained from each component. However, the components are not necessarily independent, therefore we might get some repeated information. The amount of *unique* information we get from X should therefore be at most the sum of the information gained from the individual components. For a formal proof, see pp. 8-9 in [DG14].

Proof

Theorem 4. *Every non-adaptive strategy for Mastermind with no repeats must submit at least $O(n \log k)$ queries to uniquely identify all possible hidden vectors.*

Proof. Our proof follows along similar lines as [Doerr et al]. Let q_i be the i-th guess of our deterministic strategy. Let s be the smallest index such that all $k!/(k-n)!$ codes are uniquely determined by the responses to $q_1, q_2, \ldots, q_s$.

Consider a code Z sampled uniformly at random from the set of $k!/(k-n)!$ codes. Then consider the random variables $Y_i = \text{eq}(Z, q_i)$ be the response (# black hits) to guess q_i. Then the vector $Y = (Y_1, Y_2, \ldots, Y_s)$ always uniquely determines, and is uniquely determined by, Z. So by Property 1, $H(Z) = H(Y)$. Since Z is a random variable with $k!/(k-n)!$ outcomes of equal probability, we know $H(Z) = \log_2(k!/(k-n)!)$.

By Property 2, we know

$$H(Y) \leq \sum_{i=1}^{s} H(Y_i).$$

So now we bound $H(Y_i)$. By its definition,

$$H(Y_i) = -\sum_{x=0}^{n} \Pr[Y_i = x] \cdot \log_2(\Pr[Y_i = x]). \tag{1}$$

What is $\Pr[Y_i = x]$? This is the probability that X is a solution vector with x fixed points with respect to the query q_i. Using our previous terminology, this is the probability that X is in bucket x. So we get[5]

$$\begin{aligned}
\Pr[Y_i = x] &= \frac{|B_x|}{\left(\frac{k!}{(k-n)!}\right)} \\
&\leq \frac{\binom{n}{x}\frac{(k-x)!}{(k-n)!}}{\left(\frac{k!}{(k-n)!}\right)} \\
&= \frac{1}{x!} \cdot \frac{\left(\frac{n!}{(n-x)!}\right)}{\left(\frac{k!}{(k-x)!}\right)} \\
&= \frac{1}{x!} \cdot \frac{n(n-1)\cdots(n-x+1)}{k(k-1)\cdots(k-x+1)} \\
&\leq \frac{1}{x!}
\end{aligned}$$

[5]In going from the first to the second line, we make the same combinatorial argument as before: Choosing x fixed points, choosing $n-x$ colors from the remaining $k-x$ possibilities and permuting those $n-x$ positions overcounts all vectors with no repeated colors and at least x fixed points and thus overcounts the number of vectors with no repeats and exactly x fixed points (which is $|B_x|$).

We want to plug this upper bound into equation 1. For $\alpha < 1/e$, $f(\alpha) = -\alpha \log_2 \alpha$ is an increasing function. So we can plug in the upper bound of $\Pr[X = x] \leq 1/x!$ when $x \geq 3$ and still have an upper bound. For the first three values, we will use the upper bound $f(\alpha) \leq 1/(e \log 2)$.

$$\begin{aligned}
H(Y_i) &\leq \frac{3}{e \log 2} + \sum_{x=3}^{n} -\frac{1}{x!} \cdot \log_2(\frac{1}{x!}) \\
&= \frac{3}{e \log 2} + \sum_{x=3}^{n} \frac{\log_2(x!)}{x!} \\
&\leq \frac{3}{e \log 2} + \sum_{x=3}^{n} \frac{x \log_2 x}{x!} \\
&\leq \frac{3}{e \log 2} + \sum_{x=3}^{n} \frac{x(x-1)}{x!} \\
&= \frac{3}{e \log 2} + \sum_{x=3}^{n} \frac{1}{(x-2)!} \\
&\leq \frac{3}{e \log 2} + e - 1 \\
&< 4
\end{aligned}$$

Combining this with the summation above gives $H(Y) \leq 4s$, so we get

$$\begin{aligned}
H(X) &\leq H(Y) \\
\log_2 \left(\frac{k!}{(k-n)!} \right) &\leq 4s \\
s &\geq \frac{1}{4} \log_2 \left(\frac{k!}{(k-n)!} \right)
\end{aligned}$$

This gives us a lower bound of $O(n \log k)$ turns for any non-adaptive strategy for Mastermind with no repeats. □

Lower Bound on Non-Adaptive Strategies with Dual-Color Response: Entropy Argument

We provide an entropy argument to give a lower bound for guessing strategies for a game of mastermind on n pegs and $k \geq n^2$ colors (without repeats).

We let x be the total number of hits (black hits plus white hits). Let D_x be the set of buckets of hidden vectors corresponding to responses (b, w), where $b + w = x$. The number of hidden vectors in D_x is equal to the number of ways to choose the x hit colors out of n colors in our guess, times the number of ways to choose the remaining $n-x$ colors out of the other $k - n$ colors, times the number of ways to permute all those elements. That is,

$$\sum_{B \in D_x} |B| = \binom{n}{x}\binom{k-n}{n-x} n!$$

Let Y_i be the response to the ith guess. So $Y_i = (b, w)$, a 2-vector giving the number of black hits and white hits. We have

$$\begin{aligned} H(Y_i) &= \sum_{x=0}^{n} \sum_{B \in D_x} -\Pr[Y_i = B] \cdot \log_2 \Pr[Y_i = B] \\ &\leq \sum_{x=0}^{n} -|D_x| \cdot \frac{\Pr[Y_i \in D_x]}{|D_x|} \cdot \log_2 \left(\frac{\Pr[Y_i \in D_x]}{|D_x|} \right). \end{aligned}$$

This is because $f(\alpha) = -\alpha \cdot \log_2 \alpha$ is concave down on $[0, 1]$, and thus replacing each term of $f(\alpha)$ in the second summation with f evaluated on the average of all α is guaranteed to increase the total sum. Also, note that $|D_x| = x + 1$, as there are $x + 1$ ways to have $b + w = x$ for non-negative b and w.

$$\begin{aligned} \Pr[Y_i \in D_x] &= \frac{\sum_{B \in D_x} |B|}{\sum_B |B|} \\ &= \frac{\binom{n}{x}\binom{k-n}{n-x} n!}{\frac{k!}{(k-n)!}} \\ &= \frac{(n!)^2((k-n)!)^2}{x!(k-x)!((n-x)!)^2 k!} \\ &\leq \frac{(n!)^2((k-n)!)^2}{x!(k-n)!((n-n)!)^2 k!} \\ &= \frac{(n!)^2(k-n)!}{x!k!} \\ &= \frac{1}{x!} \cdot \frac{n^2(n-1)^2 \cdots 2^2 \cdot 1^2}{k(k-1)\cdots(k-n+2)(k-n+1)} \\ &\leq \frac{1}{x!}, \text{ as } k \geq n^2 \text{ and } (n-i)^2 \leq n^2 - i \text{ for } 0 \leq i < n. \end{aligned}$$

We want to find an upper bound for $H(Y_i)$. For $\alpha < 1/e$, we know $f(\alpha) = -\alpha \log_2 \alpha$ is an increasing function. So as $\Pr[Y_i \in D_x]/|D_x| \leq 1/(x! \cdot (x+1)) = 1/(x+1)!$, for all

$x \geq 2$ we have $f(1/(x+1)!)$ is an upper bound for $f(\Pr[Y_i \in D_x]/|D_x|)$. For the first two values, we will use the upper bound $f(\alpha) \leq 1/(e \log 2)$.

$$\begin{aligned} H(Y_i) &\leq \sum_{x=0}^{1}(x+1)\frac{1}{e\log 2} \;+\; \sum_{x=2}^{n}\frac{-(x+1)}{(x+1)!}\cdot\log_2\left(\frac{1}{(x+1)!}\right) \\ &\leq \frac{3}{e\log 2} + \frac{\log_2 6}{2} + \frac{\log_2 24}{6} + \sum_{x=4}^{n}\frac{\log_2((x+1)!)}{x!} \end{aligned}$$

For $x \geq 4$, we have $\log_2((x+1)!) \leq (x+1)\log(x+1)/\log 2 \leq x(x-1)$. Thus,

$$\begin{aligned} H(Y_i) &\leq \frac{3}{e\log 2} + \frac{\log_2 6}{2} + \frac{\log_2 24}{6} + \sum_{x=4}^{n}\frac{1}{(x-2)!} \\ &\leq \frac{3}{e\log 2} + \frac{\log_2 6}{2} + \frac{\log_2 24}{6} + e - 2 \\ &= \frac{3}{e\log 2} + \frac{2\log 3}{3\log 2} + e - 1 \\ &< 5 \end{aligned}$$

Combining this with the summation above gives $H(Y) < 5s$, so we get

$$H(X) \leq H(Y)$$

$$\log_2\left(\frac{k!}{(k-n)!}\right) \leq 5s$$

$$s \geq \frac{1}{5}\log_2\left(\frac{k!}{(k-n)!}\right)$$

This gives us a lower bound of $O(n \log k)$ turns for any non-adaptive strategy for Mastermind without repeats using both black and white hits.

Upper Bound on Non-Adaptive Strategies for Mastermind

For a game of mastermind with n spots and k colors (with or without repeats), every hidden vector and query can be represented by a matrix $A \in \mathbb{R}^{n \times k}$, defined by $A_{ij} = 1$ if the i-th peg is the j-th color, and 0 otherwise. In that way, for a given hidden vector matrix X and query matrix Q, the codemaker's response to the query is given by $X \cdot Q$. Let S be the space which the set of possible queries spans. Then, for any $Q_1, \ldots, Q_k$ which span S, any matrix $X \in S$, including all hidden permutations, can be uniquely determined by the numbers $X \cdot Q_i$ for $i \in [1, k]$.

As the set of all possible queries spans S, there must exist a subset of queries, K, with $|K| = \dim(S)$ and K spanning S. Thus, the queries of K uniquely determine any hidden permutation. $S \subset \mathbb{R}^{n \times k}$, so $\dim(S) \leq \dim(\mathbb{R}^{n \times k}) = n \cdot k$.

Thus, there exists a set of at most $n \cdot k$ queries which uniquely determines every possible hidden permutation.

Existence of a Deterministic $O(k \log k)$ Non-Adaptive Strategy

This proof was inspired by [GK00].

Theorem 5. *When $k \geq n$, there always exists a set of $O(k \log k)$ queries, the responses to which will distinguish between all possible n-vectors from an alphabet of size k.*

Proof. To prove the theorem, we select a set of $4k \log k$ random guesses and show that the with probability < 1 there exists a pair of vectors that are indistinguishable by these guesses. This would imply the existance of a set of guesses that distinguishes between every pair of possible solutions. As in [GK00], we only look at distinguishing between "critical pairs", which are two vectors in which we only analyze the spots that differ. For example, the following vectors produce the following critical pair:

$$\begin{matrix}(1, 2, 3, 4, 5, 3)\\(1, 3, 6, 8, 5, 8)\end{matrix} \quad \Longrightarrow \quad \begin{matrix}(\_, 2, 3, 4, \_, 3)\\(\_, 3, 6, 8, \_, 8)\end{matrix}$$

The theorem follows directly if we can show that there's a non-zero probability that $O(k \log k)$ queries distinguish between every possible critical pair. We use the following lemma:

Lemma 1. *Given a query q and a critical pair v_1, v_2 of size x, the probability that v_1 and v_2 receive the same response is at most $(1 - 1/k)^x$. That is,*

$$\sum_{i=0}^{\lfloor x/2 \rfloor} \Pr[\mathrm{Eq}(q, v_1) = i \cap \mathrm{Eq}(q, v_2) = i] \leq \left(1 - \frac{1}{k}\right)^x$$

Proof. We prove two separate cases:

Case 1. When $x \leq k - 3$, we have

$$\begin{aligned}
&\Pr[\mathrm{Eq}(q, v_1) = i \cap \mathrm{Eq}(q, v_2) = i] \\
&\qquad = \binom{x}{i}\binom{x-i}{i}\left(\frac{1}{k}\right)^{2i}\left(\frac{k-2}{k}\right)^{x-2i} \\
&\qquad = \frac{x!(k-2)^{x-2i}}{(i!)^2(k^x)(x-2i)!} \\
&\qquad \leq \frac{x^{2i}}{(k-2)^{2i}(i!)}\left(\frac{k-2}{k}\right)^x.
\end{aligned}$$

Now we have

$$\begin{aligned}
\sum_{i=0}^{\lfloor x/2 \rfloor} \Pr[\mathrm{Eq}(q, v_1) = i \cap \mathrm{Eq}(q, v_2) = i] \left(\frac{k}{k-1}\right)^x & \\
&\leq \sum_{i=0}^{\lfloor x/2 \rfloor} \frac{x^{2i}}{(k-2)^{2i}(i!)} \left(\frac{k-2}{k}\right)^x \left(\frac{k}{k-1}\right)^x \\
&= \sum_{i=0}^{\lfloor x/2 \rfloor} \frac{x^{2i}}{(k-2)^{2i}(i!)} \left(1 - \frac{1}{k-1}\right)^x \\
&< \left(1 - \frac{1}{k-1}\right)^x \sum_{i=0}^{\infty} \left(\frac{x^2}{(k-2)^2}\right)^i \left(\frac{1}{i!}\right) \\
&< e^{-\frac{x}{k-1}} \cdot e^{\frac{x^2}{(k-2)^2}} \\
&< 1 \quad \text{for } x \leq (k-3).
\end{aligned}$$

Multiplying both sides of this inequality by $(1 - 1/k)^x$ results in the claim.

Case 2. When $k - 2 \leq x \leq k$, we manipulate the sum slightly differently: First we multiply both sides of the inequality by $(k-2)^x/k^x$ to give,

$$\sum_{i=0}^{\lfloor x/2 \rfloor} \binom{x}{i} \binom{x-i}{i} (k-2)^{-2i} \leq \left(\frac{k-1}{k-2}\right)^x$$

We will bound the LHS from above by a constant:

$$\sum_{i=0}^{\lfloor x/2 \rfloor} \binom{x}{i} \binom{x-i}{i} (k-2)^{-2i} = \sum_{i=0}^{\lfloor x/2 \rfloor} \frac{\frac{x(x-1)\cdots(x-2i+1)}{(k-2)^{2i}}}{i!^2}$$

When $i \geq 3$,

$$\begin{aligned}
x(x-1)\cdots(x-2i+1) &\leq k(k-1)\cdots(k-2i+1) \\
&\leq (k-2)^{2i}
\end{aligned}$$

Thus,

$$\begin{aligned}
\sum_{i=3}^{\lfloor x/2 \rfloor} \binom{x}{i} \binom{x-i}{i} (k-2)^{-2i} &\leq \sum_{i=3}^{\lfloor x/2 \rfloor} \frac{1}{(i!)^2} \\
&\leq \sum_{i=3}^{\infty} \frac{1}{(i!)^2} \\
&< .0296.
\end{aligned}$$

We separately bound the contribution of the terms for $i = 0, 1, 2$:

$$\begin{aligned}\sum_{i=0}^{2}\binom{x}{i}\binom{x-i}{i}(k-2)^{-2i} &= 1 + \frac{x(x-1)}{(k-2)^2} + \frac{x(x-1)(x-2)(x-3)}{4(k-2)^4} \\ &\leq 1 + \frac{k(k-1)}{(k-2)^2} + \frac{k(k-1)(k-2)(k-3)}{4(k-2)^4} \\ &< 2.5536 \text{ for } k \geq 14.\end{aligned}$$

Combining the sub-cases when $0 \leq i < 3$ and $i \geq 3$ gives

$$\begin{aligned}\sum_{i=0}^{\lfloor x/2 \rfloor}\binom{x}{i}\binom{x-i}{i}(k-2)^{-2i} &< 2.5832 \\ &< \left(1 + \frac{1}{k-2}\right)^{k-2} \text{ for } k \geq 14 \\ &= \left(\frac{k-1}{k-2}\right)^{x}.\end{aligned}$$

Cases 1 and 2 together show that for all critical pairs v_1, v_2 with size x, the probability that v_1 and v_2 receive the same response with respect to a given query is at most $(1-1/k)^x$. □

Using Lemma 1, we get

$$\Pr[v_1, v_2 \text{ not distinguished by } s \text{ random queries}] \leq \left(1 - \frac{1}{k}\right)^{xs}$$

Now we want to calculate

$$\begin{aligned}&\Pr[\exists \text{ pair } v_1, v_2 \text{ not distinguished}] \\ &\qquad = \Pr\left[\bigcup_{v_1, v_2 \text{ critical pair}} \mathrm{Eq}(q, v_1) = \mathrm{Eq}(q, v_2)\right] \\ &\qquad \leq \sum_{v_1, v_2 \text{ critical pair}} \Pr[\mathrm{Eq}(q, v_1) = i \cap \mathrm{Eq}(q, v_2) = i]\end{aligned}$$

By Boole's inequality.
But for a fixed length x, this probability is the same for all critical pairs (v_1, v_2) of length x. This allows us to rewrite the above sum as

$$\sum_{x=1}^{n}(\#\text{ of critical pairs of size } x) \cdot \Pr[\text{single pair of lenteh } x \text{ not distinguished}].$$

So we now use $s = 4k \log k$ queries to bound

$$\begin{aligned}
&\sum_{x=1}^{n} (\#\text{ of critical pairs of size } x) \cdot \Pr[\text{single pair of lenteh } x \text{ not distinguished}] \\
&\quad = \sum_{x=1}^{n} \left[\binom{n}{x} k^x (k-1)^x \cdot \left(\sum_{i=0}^{\lfloor x/2 \rfloor} \Pr[\mathrm{Eq}(q, v_1) = i \cap \mathrm{Eq}(q, v_2) = i] \right)^s \right] \\
&\quad \leq \sum_{x=1}^{n} k^{3x} \left(1 - \frac{1}{k}\right)^{(4k \log k)x} \\
&\quad < \sum_{x=1}^{n} k^{3x} \left(\frac{1}{e}\right)^{(4 \log k)x} \\
&\quad = \sum_{x=1}^{n} k^{3x} \left(\frac{1}{k}\right)^{4x} \\
&\quad \leq \sum_{x=1}^{n} \frac{1}{k} \\
&\quad \leq 1.
\end{aligned}$$

So we have $\Pr[\exists \text{ pair } v_1, v_2 \text{ not distinguished}] < 1$. Thus, for $k \geq n$ there always exists a set of $O(k \log k)$ queries that distinguishes between all possible n-vectors from an alphabet of size k. □