Acta Informatica (2023) 60:179–208
https://doi.org/10.1007/s00236-022-00436-y

ORIGINAL ARTICLE

On partial information retrieval: the unconstrained 100 prisoner problem

Ivano Lodato[1] · Snehal M. Shekatkar[2] · Tian An Wong[3]

Received: 31 March 2022 / Accepted: 17 December 2022 / Published online: 30 December 2022

Abstract
We consider a generalization of the classical 100 prisoner problem and its variant, involving empty boxes, whereby winning probabilities for a team depend on the number of attempts, as well as on the number of winners. We call this the unconstrained 100 prisoner problem. After introducing the 3 main classes of strategies, we define a variety of 'hybrid' strategies and quantify their winning-efficiency. Whenever analytic results are not available, we make use of Monte Carlo simulations to estimate with high accuracy the winning probabilities. Based on the results obtained, we conjecture that *all* strategies, except for the strategy maximizing the winning probability of the classical (constrained) problem, converge to the random strategy under weak conditions on the number of players or empty boxes. We conclude by commenting on the possible applications of our results in understanding processes of information retrieval, such as "memory" in living organisms.

1 Introduction

The aim of this paper is to study, by analytical and computational means, mathematical models of information retrieval. Our starting point is a generalization of the problem originally proposed by Gál and Miltersen [5] in the context of data structures:

Ivano Lodato, Snehal M. Shekatkar and Tian An Wong have contributed equally to this work.

✉ Snehal M. Shekatkar
snehal@inferred.in

Ivano Lodato
ivano.lodato@allos.ai

Tian An Wong
tiananw@umich.edu

1 Allos Limited, 1 Hok Cheung Street, Kowloon, Hong Kong, China

2 Department of Scientific Computing, Modeling, and Simulation, Savitribai Phule Pune University, Ganeshkhind, Pune, Maharashtra 411007, India

3 Department of Mathematics and Statistics, University of Michigan-Dearborn, 4901 Evergreen Road, Dearborn, Michigan 48126, USA

> Consider a team of n prisoners $P_1, \ldots, P_n$, n keys labeled $1, \ldots, n$, distributed randomly in $N \geq n$ boxes so that each box contains *at most* one key. Each player P_i is allowed to open $a \leq N$ boxes to find the key i. The players cannot communicate after the game starts and the team wins if *at least* $w \leq n$ players find their key.

We dub this problem as the *unconstrained 100 prisoner problem* or the prisoners-search-game (PSG). The original motivation for the problem arose in computer science, concerning the trade-off between space (intended as storage space) and time (needed to perform a given task) for substring search algorithms. In colloquial terms, it asks what are the most efficient schemes to retrieve information encoded in data structures in which the information has been randomly stored. While it was clearly important in that context that the information be retrieved completely, i.e., the team wins if all players find their keys, this puzzle has an obvious generalization whereby the team wins if a number $w \leq n$ players find their keys in a attempts. The general analysis of strategies for partial information retrieval after random and constrained storage, treated in this paper constitutes a necessary step toward a precise description of complex memory processes in living organism as well-defined PSGs.

We consider a variety of strategies, each determines a family of probability distributions $P_S(a, w)$ as the maximum number of attempts a is varied for a fixed strategy S. This family can be thought of as a function of a and w which we will henceforth call a *P-function*: it gives the probability that following the strategy S, exactly w players win within a attempts.

We first analyze three main classes of strategies in the case $N = n$, i.e., when there are no empty boxes: the random strategy, the key strategy (also called the *pointer-following* strategy), and the box strategy in which players open the boxes in an arithmetic progression. We observe, both theoretically and experimentally, that for large N, the box strategy approximates the random strategy whose P-function is given by the binomial distribution in (3.1). Furthermore, while the key strategy remains *on average* the best strategy confirming the results of the constrained classical problem, its minimum-winner P-function is actually smaller, in certain parts of the a–w domain, than that of the random strategy!

We also consider a variety of hybrid strategies for $n = N$ and $n < N$, obtained combining these three main strategies (box, key, random) in various ways. As analytical methods become increasingly difficult in this setting, we turn to numerical experiments with Monte Carlo methods and quantify each strategy in absolute and relative terms (5.1) and (5.2), respectively. Using these tools, we compare the hybrid strategies we created with those of Avis–Devroye–Iwama (ADI) [2] and Goyal–Saks (GS) [6], both key-based algorithms. We experimentally verify that some of the former strategies are more *efficient* (more winners with less attempts) than the latter. Finally, based on our simulations, we formulate in Conjecture 5.1 the expectation that all properly bounded strategies, except the original key strategy, will approximate the random strategy whenever N grows large, or as n/N tends to zero.

1.1 Outline of the paper

The paper is organized as follows: in Sect. 2, we introduce the general setup of the problem, comment on the constraints that can be imposed on the prisoners' choices, and present some important definitions; in Sect. 3 we present and discuss the three most general classes of strategies logically allowed to solve the classical PSG and obtain, analytically whenever possible, the winning probabilities of the random, key, and box strategies. Generalized hybrid strategies, possible for $n = N$ and necessary when empty boxes are present, i.e., $n < N$, will be analyzed in Sect. 4 along with implementations of the ADI [2] and GS [6] algorithms. We then present a summary of all our results in Sect. 5, formulate Conjecture 5.1, and finally

discuss in Sect. 6. possible applications to mnemonic processes of information retrieval in living organism endowed with memory. We also include a set of appendices where we present some more details.

1.2 Code availability

All the codes used in this work are freely available as a part of a Python package `prisoners-search-game` [12].

2 Setup

2.1 Definitions

The problem we shall consider in this paper is a simple, yet broad generalization of the classical PSG and can be stated as follows:

> What is the optimal strategy for (at least) $w \leq n$ prisoners to find their key by opening $a \leq N$ boxes, knowing that the keys have been distributed uniformly randomly inside the boxes and each box can contain at most one key?

In this paper, we assume there are at least as many boxes as players, $n \leq N$. We will mostly consider the case $N = n$, leaving the variant $n < N$ for the second part of Sect. 4. Note that if $w < n$, a distinction must be made between the probabilities for an *exact* and a *minimum* number of winners in a attempts (P-functions), indicated by $P(a, w)$ and $P^{\min}(a, w)$, respectively. It is easy to see also that $P^{\min}(a, N) = P(a, N)$, since N is the upper bound on the number of winners.

There are in principle infinitely many strategies to approach the problem for N prisoners, each of which has $0 < a \leq N$ attempts, whereas the group wins if (at least) $0 < w \leq N$ prisoner find their key. Let $b_{i,j}$ be the box opened by prisoner i at attempt j and k_j the key found by the prisoner in box j. Any strategy must follow two algorithmic steps:

S1. Choose the first box to open: prisoner P_i decides an offset $D \in \mathbb{Z}$ from the box i, and opens first the box $(i + D) \bmod N$. There are two possible choices for the offset,

   (a) $D = 0$, each prisoner P_i opens box $b_{i,1} = i$ first,
   (b) $D = D_i$, each prisoner P_i opens box $b_{i,1} = (i + D_i) \bmod N$ and $D_i \neq 0$ for at least one i.

S2. Choose the remaining $j = 2, \ldots, a$ boxes, in search for the key. To do so we consider three possibilities, which can all be expressed in terms of a nonzero increment $I_{i,j}$, such that $b_{i,j+1} \equiv (b_{i,j} + I_{i,j}) \bmod N$ with:

   (a) **Key strategy**: $I_{i,j} = (k_j - b_{i,j})$, the $(j + 1)$th box prisoner i opens is decided by the number on the key found at the jth box opened, i.e., the prisoner opens the box suggested by the number of the key found in the previous box.
   (b) **Box strategy**: $I_{i,j} = d$: the $(j + 1)$th box each prisoner opens is decided by constant shift d of the box opened at the jth attempt.[1]

[1] This assumption can be extended, so that the $(j + 1)$th box opened may depend on more than just the previous opened box, but an some/all previously opened boxes. As we shall comment below, relaxing this hypothesis will not modify the P-functions.

(c) **Random strategy**: $I_{i,j} = d_{i,j}$: the $(j+1)$th box each prisoner opens is decided by a non-constant shift $d_{i,j}$, which depends on the prisoner and the attempt.[2]

Henceforth we will use the acronyms KS, BS and RS to indicate the above 3 strategies, respectively. We shall also use the superscript 0, as in KS^0 to indicate that the offset $D = 0$ for the key strategy. Note that the case KS^0 represents the optimal solution to the classical 100 prisoner problem.

Any strategy S, in principle, is a function of the players $P = \{P_1, \ldots, P_n\}$, boxes $B = \{b_1, \ldots, b_N\}$, and the keys $K = \{k_1, \ldots, k_N\}$ found inside the boxes, with $B, K \subset \{1, 2, \ldots, N\}$. So generally speaking, we have a map when $D = 0$,

$$S : P \times B \times K \to B.$$

The three possibilities for $S2$ discussed above, combined with a (specific) choice of D for $S1$, give rise to three natural classes of strategies, which we analyze in the next section. We will be interested mainly in strategies such that no prisoner revisits a previously opened box, regardless of choices of $S1$ and $S2$. This reasonable assumption will often constrain the offset D and/or increment I to take on only specific values. It is then helpful to give the following:

Definition 2.1 Let S be a strategy. We say:

1. S is *properly bounded* if all n prisoners will necessarily find their key for all $a \geq N$.
2. S is *bounded* if all prisoners will necessarily find their key for all $a \geq M$, for some integer $M > N$.
3. S is *unbounded* if not all prisoners necessarily find their key for any (finite) a.

We see that S is not properly bounded if and only if players can open the same box more than once. Hence properly bounded strategies force the increment I (or the offset D) to be such that no box is opened more than once. It follows by definition then that a properly bounded strategy S has higher probability of success for an *individual player* than any of its unbounded variants S',

$$P_S^i \geq P_{S'}^i,$$

where P^i specifies the probability of success of the player P_i. On the other hand, if we are interested in the probability of success for a group of *exactly* w players this inequality does not hold between the P-function $P_S(a, w)$ and $P_{S'}(a, w)$ in the whole a–w plane. The reason is that both these P-functions reach maxima (minima) not only when the probability for w players to win is maximized (minimized), but also when the probability for $N - w$ players to lose are maximized (minimized) (or equivalently, the probability for $N - w$ players to win are minimized (maximized)). However, by considering the P-functions for the group of *at least* w players to win, the remaining $N - w$ players are not required to lose. Consequently, we recover an inequality between the P-functions for a minimum number of winners:

$$P_S^{\min} \geq P_{S'}^{\min}.$$

In "Appendix A," we present an unbounded variant of the random strategy analyzed in Sect. 3.1 and show the correctedness of the above inequalities.

Aside from increasing the probability for individual players to win, there is another reason why the properly bounded property for strategies is important, though the above distinction

[2] There exist two middle cases, whereby the increment does (resp. does not) depend on the prisoner, but it does not (resp. does) depend on the attempt. However, these are automatically accounted for by the most general case $I = d_{i,j}$.

has been somehow hidden in previous approaches to this problem which fixed $w = N$. The reason is immediately apparent through the below lemma. Note that the symmetry assumption turns out to be true for both random and box strategies, as we will show later in Sect. 3, Eq. (3.2) and Appendix B, Lemma B6.

Lemma 2.2 *If $P(a, w) = P(N - a, N - w)$, then the probability of success for* at least w *prisoners satisfies the identity*

$$P^{\min}(a, w) + P^{\min}(N - a, N - w + 1) = 1.$$

Proof Consider the left-hand side of the above identity, and write it down explicitly in terms of the exact probabilities, hence

$$\begin{aligned}\sum_{i=0}^{N-w} P(a, w+i) + \sum_{i=0}^{w-1} + P(N-a, N-i) \\ = \sum_{i=w}^{N} P(a, i) + \sum_{i=0}^{w-1} P(a, i) \\ = \sum_{i=0}^{N} P(a, i) = 1,\end{aligned}$$

where we have used the symmetry assumption to rewrite the second summation in the second line. The proof can also be followed in the opposite direction. □

In particular, under the properly bounded constraint, if the exact probability function possesses the diagonal symmetry above: the probabilities $P^{\min}$ of half the a–w plane determines the probability for the other half of the plane.

2.2 Monte Carlo sampling

Our aim in this paper is to explore the behavior of $P(a, w)$ over a range of a and w. We intend to study the problem for large N, and since analytic results are not available, we would like to compute the probabilities numerically by accounting for all possible permutations of the keys in the boxes. However, in this case it is difficult to obtain exact winning probability of a given strategy, since the total number of possible permutations of a set of N numbers is $N!$.

To approach this problem rigorously, we will make use of Monte Carlo simulations to describe a variety of different strategies. The basic idea of Monte Carlo methods is to sample the underlying space of configurations (too large to be considered fully) and then work with these sampled points, or simply "sample." Provided that the sample is drawn uniformly randomly, its properties must approximate the properties of the original space more and more as we increase the size of the sample (i.e., as we draw more and more permutations).

To apply the Monte Carlo technique to our problem, for a fixed value of N, n, and a, we draw a fixed sample of permutations of $\{1, 2, \ldots, N\}$ uniformly randomly from the set of all $N!$ permutations, and then simulate the game for each of those permutations with a given strategy. This allows us to estimate the P-function $P(a, w)$ to a great accuracy provided that an enough number of samples are drawn. At this point, we assume the conventional 95% confidence level on the probability P, which implies a *z-score* $z_{95} \sim 1.96$. Furthermore we take $\sigma^2 = p(1-p)$ to be the variance for the (possible) future samples of permutations, where

p is the percentage of permutations in a specific sample satisfying the condition "exactly w prisoners find their key within a attempts." We can then compute the margin of error $\mathcal{M}$ as:

$$\mathcal{M} = z_{95}\sqrt{\frac{p(1-p)}{s}} \tag{2.1}$$

with s sample size. It is easy to show that, under the above condition, by fixing $s \sim 10^4$, and because $0 \leq p \leq 1$, the margin of error will be always $\mathcal{M} \leq 1\%$. Hence, we will consider sample of size $s = 10^4$ to be sure of the accuracy, within small margin of errors of our Monte Carlo simulations. We remind that all the codes used in this paper are freely available as a part of a Python package `prisoners-search-game` [12].

3 Three main strategies

We now present the most general classes of properly bounded strategies discussed above. We start by considering the strategy based on random choices, for which the exact winning probabilities can be easily derived. This will be considered the benchmark we will compare all other results against. We then continue by considering the key and box strategies.

3.1 Random strategy RS

The random strategy is the benchmark of all properly bounded strategies. For all a attempts to be random, the offset D must be random for every prisoner. The algorithmic steps are:

1. Player B_i opens a random box from 1 to N, say j.
2. If Box j does not contain the key i, open another box randomly among the boxes not opened so far.
3. Repeat until the key i is found OR stop after a attempts.

The analytic formula for the probability of *exactly* w players finding their key after a attempts each is simply given by

$$P_{\mathrm{RS}}(a, w) = \binom{N}{w}\left(\frac{a}{N}\right)^w \left(\frac{N-a}{N}\right)^{N-w}, \tag{3.1}$$

where $\binom{N}{w}$ is the binomial coefficient which computes the number of winning w-tuples that can be formed from N elements, the second factor gives the probability for w prisoners to find their key after a attempts, and the last factor gives the probability of $N - w$ prisoners to not find their key in a attempts.

We further note that the above probability (3.1) is symmetric under exchange $(a, w) \to (N - a, N - w)$, the property of reflection with respect to the center point $(\frac{N}{2}, \frac{N}{2})$,

$$P_{\mathrm{RS}}(a, w) = P_{\mathrm{RS}}(N - a, N - w), \tag{3.2}$$

as can be shown using the symmetry of the binomial coefficient. Note that because of the above identity, the probability for half the a–w plane determines the probability for the other half of the plane (see Fig. 1).

Proposition 3.1 *The random strategy for a attempts and w winners has $P^{\min}$-function*

$$P^{\min}_{\mathrm{RS}}(a, w) = \sum_{w'=w}^{N} P(a, w') = \frac{1}{N^N}\sum_{w'=w}^{N}\binom{N}{w'} a^{w'}\,(N-a)^{N-w'}. \tag{3.3}$$

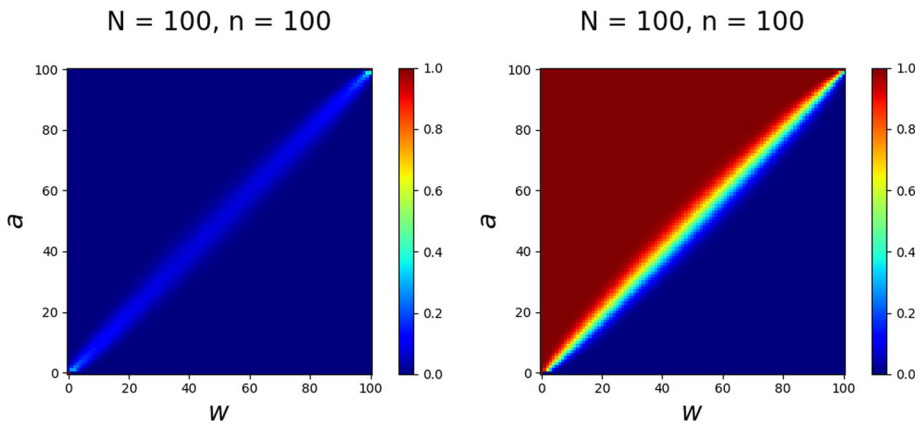

Fig. 1 Random strategy P-functions. Left: exact winners heatmap. Right: minimum winners heatmap

The proof of this follows immediately from (3.1). In Fig. 1 we show the heat-maps of the P-functions, i.e., the probability of winning, of the random strategy. The bifurcation that we observe can be seen as illustrating the cumulative distribution function for the binomial distribution,

$$P_{\mathrm{RS}}^{\min}(a, w+1) = 1 - P(X_{\mathrm{RS}} \leq w), \tag{3.4}$$

where the random variable X_{RS} is defined as

$$X_{\mathrm{RS}} = \sum_{i=1}^{n} \sum_{j=1}^{a} X_{ij}, \tag{3.5}$$

where each X_{ij} is either 0 or 1 depending on whether or not the P_i has found her key at the jth attempt. The exact winners P-function is peaked around the diagonal, with absolute maxima at $(a, w) = (0, 0), (100, 100)$. This is to be expected because of the symmetry shown in (3.2). More interesting is the minimum-winner P-function which is a very precise approximation for the Heaviside $\theta(x)$. Specifically, the graphs above show that

$$P_{\mathrm{RS}}^{\min}(a, w) \sim \theta(a - w). \tag{3.6}$$

The random strategy can be considered as the benchmark for all the other strategies since it is reasonable to expect that any clever strategy should at least be as good as the random strategy. We will see how this reasonable assumption is encoded in our study later, when we will define a numerical estimator for the efficiency of a strategy.

3.2 Key strategy KS^0

The second strategy we analyze corresponds to the solution which maximizes the winning probability for $w = N$ players in the classical (constrained) problem [3]. We shall see that, when extended to the whole a–w plane, its optimality persists but it is not evident as it was for the classical (constrained) problem. In fact, the key strategy maintains a high success probability only in specific regions, a feature which to our knowledge has never been observed before. The key strategy KS^0 coincides with the following algorithmic steps:

1. Player B_i opens Box i.

```
2. If Box i does not contain the key i, but key j, open Box j.
3. Repeat until the key i is found OR stop after a attempts.
```

In short, the strategy exploits the fact that the key numbers and box-numbers create a structure of permutation cycles, which each player simply follows. Hence, for this strategy to be optimal, the offset is $D_i = 0$ for all $i = 1, \ldots, N$. That is the only way a player is sure she will open only boxes-numbers which create, in combination with the numbers if the keys they contain, a cycle structure: a player, regardless from its identification number, will open only boxes, a in total, in a permutation cycle surely containing her own key. Furthermore, a player P^i, with initial offset $D_i = 0$, is sure to find her key in exactly l-attempts, if her key is in an l-cycle structure with the box-number.

The classical case then corresponds to $w = N$ and $a = N/2$, where the probability of success is given by

$$P_{\mathrm{KS}^0}(N/2, N) = 1 - \sum_{k=N/2+1}^{N} \frac{1}{k} = 1 - \ln 2 + o(1), \tag{3.7}$$

which follows from counting the number of cycles c of length $l_C \leq a = N/2$, which produce exactly l_c winners. Before proceeding, let us first recall the proof of this formula. Consider first the cycle decomposition of a given permutation of N numbers. It can be written as a partition:

$$N = \mathcal{N}_1 + \mathcal{N}_2 + \cdots + \mathcal{N}_N,$$

where $\mathcal{N}_k = k \cdot \alpha_k$ and α_k is the number of cycles of length k present in the partition. In the following, we will indicate a partition with p and its cycle structure with $(\alpha_1, \alpha_2, \ldots, \alpha_N)$ and unify the two notations by writing $p = (\alpha_1, \alpha_2, \ldots, \alpha_N)$. Clearly, for $k > \lfloor N/2 \rfloor$, the largest integer smaller than or equal to $N/2$, we have $\alpha_k = 0, 1$. Similarly, any permutation will probably have some $\mathcal{N}_k = 0$, because $\alpha_k = 0$.

Now, it is relatively easy to compute the number of permutations which contain at least (and obviously at most) a cycle of length $l > \lfloor N/2 \rfloor$ can be calculated as follows: first consider all possible ways to extract l elements from N. For each of these l-elements, there exist $(l-1)!$ inequivalent permutations which combine into an l-cycle (easy proof by induction). Finally, we can consider all the permutations of the remaining $(N-l)$ elements. The final formula hence reads:

$$\Omega_l^N = \binom{N}{l}(l-1)!(N-l)! = \frac{N!}{l}. \tag{3.8}$$

Dividing this formula by $N!$, we obtain the probability that a permutation contains an l-cycle, $l > N/2$, from which the formula (3.7) is derived.

Next, we are interested in extending this formula to the cases $w < N$ winners and a unconstrained. The counting is not easy anymore, as cycles of length l not greater than $\lfloor N/2 \rfloor$ may appear with multiplicity $\alpha_l > 1$. There exists, however, a formula which counts the multiplicity of a certain partition of cycles of N, $p = (\alpha_1, \ldots, \alpha_N)$[3]:

$$\Omega^N_{(\alpha_1,\ldots,\alpha_N)} = \Omega^N_p = \frac{N!}{1^{\alpha_1} \ldots N^{\alpha_N}\, \alpha_1! \ldots \alpha_N!}. \tag{3.9}$$

[3] An extension of this formula, presented in (B3), allows for an explicit expression for $\Omega^N_{\alpha_l=k}$ in terms of more easily computed numbers $\Omega^{k_1 \cdot l}_{\alpha_l=k_1}$ and $\Omega^{N-\alpha_1 \cdot l}_{\alpha_l=k-k_1}$. For the scope of this section, this formula will not be necessary.

To give a simple example, assume N even, and we want to find how many permutations of N are such that their partition of cycles is

$$p = (0, \ldots, \alpha_{N/2} = 2, 0, \ldots, 0).$$

There are $\Omega_p^N = \frac{N!}{(N/2)^2 \cdot 2!}$ such permutations. For $N = 4$, the result is simply $\Omega^4_{(0,2,0,0)} = 3$, since there are 3 permutations of the numbers $(1, 2, 3, 4)$ which contain two 2-cycles: $(2, 1, 4, 3)$, $(3, 4, 1, 2)$ and $(4, 3, 2, 1)$.

We also note that, given a partition p of N as in the above, into cycles of length i, each cycle of length $i \leq a$ will produce $\mathcal{N}_i$ winners, while each cycle of length $j > a$ will produce j losers. Hence the total number of winners for a given partition p of N is given by:

$$w_p(a) = \sum_{\mathcal{N}_i \leq a} \mathcal{N}_i. \tag{3.10}$$

From the preceding discussion, we obtain the general formulas below.

Proposition 3.2 *The key strategy for a attempts and* exactly *w winners has P-function*

$$P(a, w) = \frac{1}{N!} \sum_p \Omega_p^N \cdot \delta\big(w, w_p(a)\big), \tag{3.11}$$

where δ is the Kronecker delta, equal to 1 *only when $w = w_p(a)$. Similarly, the P-function for the key strategy with a attempt and* at least *w winner reads:*

$$P(a, w) = \frac{1}{N!} \sum_p \Omega_p^N \cdot \theta\big(w - w_p(a)\big), \tag{3.12}$$

and 0 otherwise.

In the above equations, the sum is extended over all partitions p for which the number $w_p(a)$ in (3.10) is equal to or no less than w. In Fig. 2, we show the P-functions of the key strategy, which possess unique features: for small values of both a and w, both P-functions resemble the random P-functions (with a larger spread), while the minimum-winner P-function is symmetric under exchange $a \leftrightarrow w$ around the second diagonal.

3.3 Box strategy BS

Finally, we discuss the simplest strategy to implement, the box strategy. As we will observe heuristically and experimentally, the choice of nonzero offset(s) is immaterial to the box strategy P-functions, but for clarity of exposition, let us consider first the case $D_i = 0$. The algorithmic steps are:

1. Player P_i opens Box i.
2. If Box i does not contain the key i, open Box $(i + I) \bmod N$
3. Repeat until the key i is found OR stop after a attempts.

(C) The increment I is coprime to N.

The above condition (C) can be explained as follows: if $\gcd(I, N) = d > 1$, then each player will re-open the same box previously opened in attempt N/d, which could be considered a "period." This of course renders the strategy unbounded, significantly decreasing the probability of success. Hence we will not consider this unbounded variant, but instead always impose condition (C) on the box strategies.

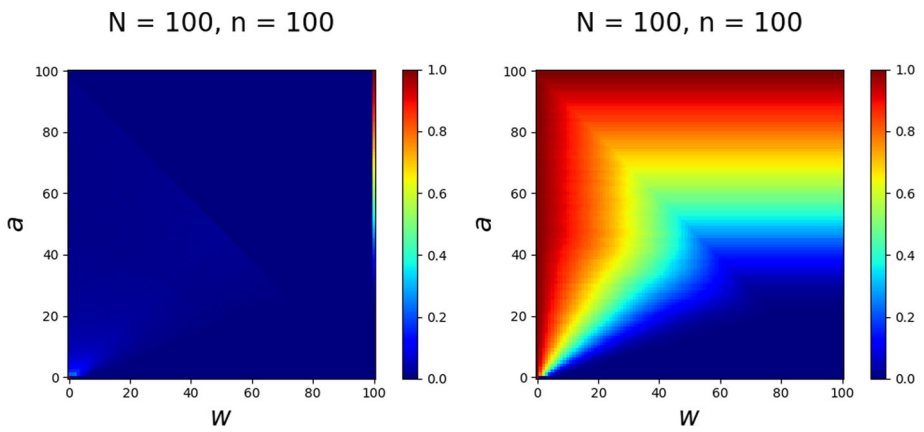

Fig. 2 Key strategy P-functions. Left: exact-winner heatmap. Right: minimum winner heatmap

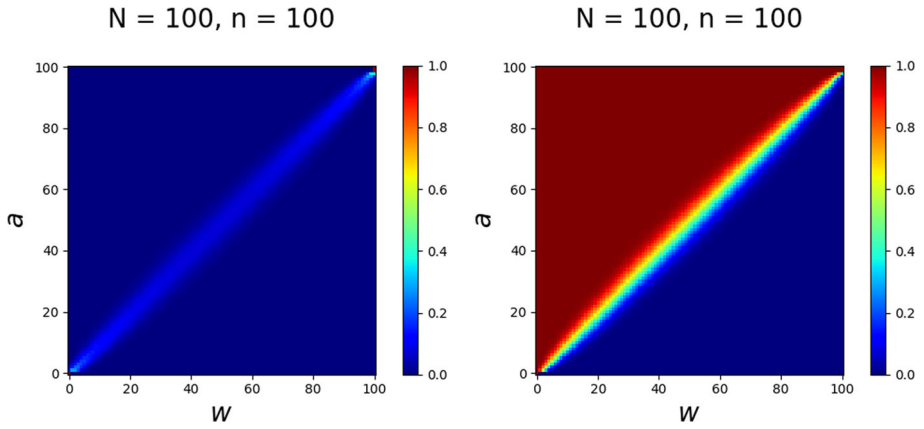

Fig. 3 Box strategy P-functions with increment 1, offset 0. Left: exact winners heatmap. Right: minimum winners heatmap

Remark 3.3 Although box strategies clearly aim to put an order in the selection of the boxes, they are by all means sub-cases of the random strategy. Unfortunately, neither strategy is equipped to "unravel" the random distribution of keys, i.e., to find an order in the disordered key positioning. The randomness of the keys' positioning is then the leading factor to be considered when calculating probabilities with box and random strategies. Whether the box selection is ordered or not, it is of little importance since the keys have been randomly distributed. Indeed, as we shall soon show by means of analytical and statistical approaches, the box strategy converges to the random strategy for N sufficiently large (see Fig. 3)

To compute the probability of winning with this strategy, we need to enumerate permutations with restricted positions. For clarity of exposition, let us first take the offset $D_i = 0$ for all i, and generic increment I coprime to N. This means that, for player i to win, her key i needs to be in a box between $i, i+I \ldots i+(a-1)I$. This problem is complementary, in a way succinctly expressed in the proof of Lemma B.2, to the famous "problème des rencontres" [9], "problème des ménages" [7, 8], and generalizations thereof, more generally known as the counting of restricted permutations, often expressed in the language of chessboards and

rooks positioning [10]. These problems have a long history and are notoriously hard to solve: to the best of our knowledge, a closed formula for the number of p-discordant permutations, i.e., permutations for which *no key* i is in its box i, or in any successive boxes until $i + p - 1$, exists only up to $p = 5$ [11].

Here we will present the results for the first two cases $a = 1, 2$, which are related to 1- and 2-discordant permutations, respectively.

Proposition 3.4 *For any* $1 \le w \le N$, *we have*

$$P_{\mathrm{BS}}(1, w) = \frac{1}{w!} \sum_{k=0}^{N-w} \frac{(-1)^k}{k!},$$

and

$$P_{\mathrm{BS}}(2, w) = \frac{c_w}{N!} = \frac{1}{N!} \sum_{k=w}^{N} (-1)^{k-w} \frac{2N}{2N-k} \binom{2N-k}{k} (N-k)! \binom{k}{w}.$$

For a proof of the above two formulas, we refer the reader to "Appendix B."

Corollary 3.5 $P_{\mathrm{BS}}(1, w)$ *converges to* $P_{\mathrm{RS}}(1, w)$ *for* N *large enough.*

Proof This follows from the relation $P_{\mathrm{BS}}(1, w) = D_{N,w}/N!$ in (B8) and (B7), and fact that

$$\lim_{N\to\infty} \frac{D_{N,w}}{N!} = \frac{e^{-1}}{w!},$$

so the limiting distribution is equal to the Poisson distribution with expected value 1, which is the same limit as the binomial distribution. □

Remark 3.6 As we have mentioned, analytic formulas exists for 3-,4- and 5-discordant permutations, namely for $a = 3, 4, 5$ attempts (we refer to [9, 11, 13] for details). In Fig. 3, we show the exact- and minimum-winner P-functions for the box strategy with increment 1 and zero offset. As we observe, the P-function appears to be identical (within small margin of errors) to the random-strategy P-function, as we expected and argued above.

Since the offset has been set to zero for simplicity, P_i can only find their key within boxes $(i, i+1, \ldots i+a-1) \bmod N$. Now consider the simple case $a = 1$. It is clear that as long as prisoners are assigned a box number in a one-to-one correspondence, it does not matter which box number that is, but the above arguments can be applied after a simple re-ordering of box (or player) numbers, e.g., P_i can be assigned box $i + D$ as her own fixed point. For the case $a = 2$, it does not matter whether P_i opens box $(i, i+1) \bmod N$ or $(i+D, i+D+I) \bmod N$.

Similarly, it does not depend on the fact that some prisoners may decide to change their offset and/or increment. This is the case, for instance, where the box $b_{i,j+1} = f(\{b_{i,j}\})$, where f is a bijective function on $[1, \ldots, N]$. This strategy would constitute an intermediate case between box strategy and random strategy, but since their P-functions are (almost) identical, a strategy that interpolates between the two will necessarily have (almost) identical P-function. It is natural then to ask whether all properly bounded strategy S, that is independent of the key-numbers K, will asymptotically approach the random strategy RS. Our many simulations have confirmed this to be true. We can then present the most general, naturally properly bounded strategy, not based on the key numbers, BS:

1. `Player` P_i `opens Box` $(i + D_i) \bmod N$.

2. If Box $(i + D_i)$ mod N does not contain the key i, open Box $(i + D_i + I_i)$ mod N, with I_i such that no box is opened more than once.
3. Repeat until the key i is found OR stop after a attempts.

(C) The increment $I_{i,j}$ should be such that no boxes are opened more than once.

We will formulate this 'convergence' to the random strategy in more general terms in Conjecture 5.1.

Finally, even though analytic results are not generally known, approximations for the enumerators of discordant permutations have been studied. For instance, [10] shows that if $(a-1) < N^{1/3}$, the normalized probability $P^N_{\mathrm{BS}}(a, w)$ can be expanded in inverse powers of n (or $(n)_r = n \cdot (n-1) \cdots (n-r+1)$), with leading order term corresponding to a Poisson distribution. This suggests that, for N large enough, the P_{BS} will approximate P_{RS}, as it was hinted at earlier.

Let us give a heuristic argument to justify this. As we have seen, the random positioning of the key can be "compensated" by a smart choice of the strategy exploiting the cycle-decomposition of any random permutation of keys. On the other hand, without exploiting this cycle decomposition, it is hard to devise a way by which to find the keys in a specific order with any certainty. Hence, lacking a smart strategy to unravel the random distribution of keys, *any* other strategy will necessarily result in an approximate random probability of winning: even if the choice of boxes follows a certain order, as in the box strategy described above, the probability of winning will just reflect the randomness of the key distribution. If the choice of boxes is also random, then the probability will remain random, since a random shuffling of randomly distributed elements will still give a random distribution. This approximation will be more and more valid as the number of randomly shuffled elements increases. For small values of N, there may be finite differences, but they become negligible as N grows large. We shall describe this more concretely in (5.2).

4 Hybrid strategies

4.1 Escape routes

In the previous section, we analyzed the three general classes of properly bounded strategies to approach the PSG. Crucially, for the box strategy and the random strategy to be properly bounded, a constraint C was required. In the case of the key strategy, it is the condition $D = 0$ that makes the strategy properly bounded. To understand why, imagine player P_i first opens the box $j \neq i$. Now, P_i can either be inside her cycle or inside a different cycle. In the latter case, P_i may enter a cycle of length smaller than a and hence be forced to re-open certain boxes, never to find hers.

If for some reason the constraints on the increment (BS) or condition on the offset (KS) are not satisfied, the strategies become unbounded. Nevertheless, it is possible to recover proper boundedness by adding an extra algorithmic step, an escape route E, to all strategies:

E : If the next box to be opened has already been opened, choose another one to open, among the un-opened boxes.

This choice can again be made in one of three ways:

- choose the next box as indicated by the key number, $E = K$

- choose the next box randomly among the boxes not opened so far, $E = R$
- choose the next box sequentially, $E = B$

Note, however, that the first choice above does not necessarily, and on its own, imply boundedness. So we will neglect for the time being the key strategy as an escape route, though we will later present a (bounded) strategy that contains the KS as an escape. Another important feature, strengthened by the results of the previous section, is that the difference between a sequential or random choice of box to open is, within small statistical errors, inessential: as long as the key are distributed randomly, the P-function for the RS and BS are almost identical. This seems to suggests that the difference between random and sequential choice of the next box to open is small enough to be neglected. This fact seems to be confirmed by the simulations (we encourage the curious reader to check this statement making use of [12]), though as we shall see, our efficiency index will pick up on small differences between strategies with a box or a random strategy as an escape.

For the time being, we can, without loss of generality, consider the P-functions for the hybrid strategies KS and BS complemented by a random choice of the next box to be opened as an escape route. Note that, given our definition at the opening of Sect. 3, the random strategy will never require an escape route,[4] as the constraint assures proper boundedness. Hence, we can still consider its P-function as the benchmark of all hybrid strategies with $n = N$.

Further extensions of the game are possible. Of particular interest in practical applications is the case $n < N$, for which some boxes will not contain a key, they are empty. In this case, an escape route will be needed while using the key strategy if the last opened box is empty. We will also show an example of a bounded hybrid strategy [6], the Goyal–Saks algorithm, which uses $E = K$ escape route for the BS with $I = 1$ and a surplus. Finally, we will present simulations of a properly bounded strategy which uses a notion of fictitious keys as an escape route for the KS [2].

4.2 $n = N$ hybrid strategies

The first hybrid strategy we consider is the unbounded key strategy, $D \neq 0$, to which we add a random choice as an escape route whenever the player is about to open an already-opened box. The algorithmic steps are:

```
1. Player P_i opens any Box j ≠ i.
2. If Box j does not contain the key i, but key k, open Box k.
3. Repeat until the key i is found OR stop after a attempts.
(E) If the Player is forced to open a box she already opened, pick the next box randomly among the unopened boxes
```

We present in Fig. 5 results of the Monte Carlo simulation for this hybrid strategy.

As we see immediately, the P-function for the hybrid of a key strategy with a random (or sequential) escape route approximates with high precision the random strategy P-function in Fig. 1. The next hybrid strategy we consider is the unbounded box strategy, i.e., the increment I is not coprime to N, with $d = \gcd(I, N)$ complemented by a random-choice escape strategy. We call $\tau = N/d$ steps the *period*, the number of attempts after which all prisoners will be forced to open again an already-opened box. The algorithmic steps describing this BS hybrid are:

[4] In fact, the hybrid strategy random with a random escape route corresponds by definition to the bounded random strategy, whose P-function is shown in Fig. 1.

1. Player P_i opens any Box j.
2. If Box j does not contain the key i, open box $j + I \bmod N$, I not coprime to N
3. Repeat until the key i is found OR stop after a attempts.

(E) If the Player is forced to open a box she already opened, pick another box randomly among the unopened boxes

The P-function is plotted in Fig. 6. It again corresponds, with high accuracy, to the random strategy P-function.

4.3 $n < N$ hybrid strategies

The case of prisoner search games with empty boxes is one of the most explored variants in the literature. It clearly necessitates of an escape route to balance the presence of empty boxes and still enforce proper boundedness. We first consider a benchmark for these strategies, the properly bounded random strategy whose analytic P-function, shown in Fig. 7, reads:

$$P(a, w) = \binom{n}{w}\left(\frac{a}{N}\right)^w\left(\frac{N-a}{N}\right)^{n-w}. \tag{4.1}$$

In parallel to the previous subsection, we now define two types of hybrid strategies based on the unbounded key and box strategies.

First, we consider the hybrid strategy obtained from the unbounded box strategy complemented by a random escape route (Fig. 10). As one would expect simply by comparing the P-functions for random and box strategies, the exact- (and hence the minimum-)winners P-function for this hybrid will still be peaked only around the diagonal of equation: $a = \frac{N}{n}w$ (see Fig. 10 for the case $n = 50$).

The second case of hybrid strategy we consider is obtained from a key strategy with $D = 0$, KS^0, complemented by a random or sequential escape route. In this case, one can intuitively expect (the box-selection imposed by) this hybrid strategy to "collapse" to the random strategy if $n \ll N$: many boxes are empty, forcing prisoners to opt for a random selection of the next box to open. This argument implies that for $N - n$ large enough, the hybrid of an unbounded key strategy will reduce to a random strategy. In Fig. 11, we show that case $n = N - n = 50$ and note some very small differences, not connected to statistical errors, for small values of a and w.[5]

Interestingly, even for $N - n = 1$ (only one empty box), the P-functions for this hybrid strategy already possess the characteristic profile (along the diagonal) of the random strategy P-functions, see Figs. 8 and 9.

4.4 The Avis–Devroye–Iwama (ADI) strategy

So far we have presented strategies easily obtained as combinations of the three main strategies, random, key and box. There are of course examples of hybrid strategies whose escape does not belong to any of the above three categories. Examples of such strategies were presented in [2], where it was initially assumed $n < N$ and that all prisoners are surely aware of the value of n and N, and hence $N - n$.[6] We will consider here only the first example, named

[5] The same hybrid strategy, based on the key strategy with random or sequential escape route, but nonzero initial offset D gives instead efficiency slightly below 1.

[6] Note that, while it was always in principle possible for the prisoners to know their total number n and the boxes number N, all the strategies analyzed so far were constructed regardless of this information, i.e., all

PF-1, since the second example, PF-2(t), reduces to PF-1 for specific values of the parameter t and has been shown to produce lower winning probability in general. The strategy PF-1 can be described by the following algorithmic steps:

1. Player P_i opens Box i ($D = 0$) and sets an index $j_i = 0$, which counts the number of empty boxes that are opened during the search
2. If Box i contains key $j \neq i$, open box j; if box i is empty, increment j_i by 1, and go to open box $n + j_i$
3. Repeat step 2 until the key i is found OR stop after a attempts.

It is immediately clear why the information about the exact value of N and n is required, since otherwise the prisoner would not know which box to open after having opened an empty box. The index $j_i = 1, \ldots, N - n$ counts, in the same order they have been opened by each prisoner, the empty boxes. Since this opening order may vary from prisoner to prisoner, it is important to realize that the cycle structure of the keys, the union of the 'real' keys numbered from 1 to n, and the 'fictitious' keys which force the players to open the boxes from $n + 1$ to N (for which there is no actual key present), is not unique in this case, but it definitely varies from prisoner to prisoner (We refer the reader to the explicit example given in [2]). This of course does not happen when $N = n + 1$, only one empty box, since in that case all prisoners will agree on its box number and the P-functions will look exactly as Fig. 2. Hence, for this strategy alone, we will not show the case $n = 99$, but $n = 98$ instead. In Figs. 12 and 13, we show the P-function for the strategy PF-1, for $n = 98$ and $n = 95$, respectively. For n small enough, the P-function will approximate once again to the P-function for the random strategy, shown in Fig. 7.

4.5 The Goyal–Saks (GS) strategy

Finally, we simulate the P-function for the Goyal–Saks algorithm in [6, Theorem 1], in the general case $w < n$. This hybrid strategy is bounded, but not properly bounded. The strategy of Goyal and Saks is as follows. Let $d = \lfloor N/n \rfloor$ and let us denote by $[s, t]$ the set of integers $\{s, s + 1, \ldots, t - 1, t\}$ if $s \leq t$ and $\{s, \ldots, N\} \cup \{1, \ldots t\}$ if $s > t$. Define occupied$[s, t]$ to be the number of boxes that contain a key (hence are not empty) in $[s, t]$, and define

$$\text{surplus}[s, t] = \text{occupied}[s, t] - \frac{|s - t|}{d}.$$

In other words, the surplus function measures how much the number of non-empty boxes in the interval $\{s, t\}$ differ from the average number of non-empty boxes in the same interval. For each i, we let $m(i)$ be the smallest integer such that surplus$[i, m(i)]$ is non-negative. Finally, we partition the boxes into bins $B_1, \ldots, B_n$, where B_i contains boxes $[d(i-1)+1, d(i-1)+d]$ for $i \in [1, n - 1]$, and B_n contains boxes $[d(n - 1) + 1, N]$.[7] The algorithm is then the following.

1. Player P_i starts at the first box of the bin B_i.

Footnote 6 continued
prisoners would be given a unique list of boxes to open, even without knowledge of the exact number of empty boxes, $N - n$.

[7] Note that this is only one possible arrangement of boxes into bins when N/n is not an integer.

2. Check boxes sequentially, keeping track of the surplus until surplus is non-negative, hence $m(i)$.
3. If box $m(i)$ contains the key, done.
4. If not, then box $m(i)$ will contain key j, go to bin B_j. Re set surplus and repeat.

This means that the player will follow a BS until the surplus is negative, then use the key strategy as an escape when the surplus becomes non-negative, then continue with the sequential selection of boxes, and so on. It is clearly a hybrid strategy, but not a properly bounded one.

Then the main result of [6] is that their strategy, GS, for $w = n$ and $a = N/k$ (k also depends on n) has success probability at least

$$2^{-9\sqrt{kn}\log_2 n - k}.$$

It is known that this lower bound can be improved to $2^{-3\sqrt{kn}\log_2 n - 2k\log_2 e/3}$, but it is ineffective in the following sense. The classical case corresponds to $N = n = 100$ and $k = 2$, and substituting into the latter expression we have

$$2^{-3\sqrt{200}\log_2 100 - 4\log_2 e/3} \sim 2 \times 10^{-85},$$

which is much smaller than the exact probability 0.311 calculated from the classical solution. Indeed, it remains an open problem to determine whether the probability $P(N/2, n)$ with varying n tends to zero. To answer this question, we simulated the GS strategy for different values of $N/2 \leq n < N$ confirming that indeed $P(N/2, n)$ is vanishingly small. Given our limited samples of all permutations, it would be clearly impossible to verify with precision the bound (to obtain a probability of 10^{-85} we would have to consider at least 10^{85} permutations and verify that in at most one of those can n players win with $N/2$ attempts).

In Fig. 14, we show the case $N = 100$ and $n = 99$: it is easy to see that the strategy is not properly bounded, but only bounded, since when $a = N$, the P-function $P(N, w)$ never reaches 1 (see exact winners histogram in Fig. 14).

5 Convergence to the random strategy

So far, we have analyzed and presented the P-functions for a variety of old and new strategies for the resolution of the generalized 100 prisoner problem, where not only the number of attempts a, but also the minimum number of winners w can be varied. Although the numerical plots illustrate clearly the differences/similarities between certain strategies, in the section below we aim to quantify the absolute *efficiency* of a strategy, as well as the *error* between two strategies, by means of two estimators of our own making.

5.1 Estimators of a strategy

The first estimator we introduce here, called the efficiency η of a strategy, measures the performance of a given strategy in terms of producing the exact number of winners within the right number of attempts. In fact, it is logical to assume that a strategy is more efficient than another if it produces more (equal) winners with the same (less) number of maximum attempts. In other words, a more efficient strategy is one for which $P(a, w)$ tends to take high values for low values of a and high values of w. If we multiply each value $P(a, w)$ by the

function $(w/a)^\beta$ with $\beta > 0$, it is easy to see that for an efficient strategy, the product takes high values in the correct region (low a, high w) of the a–w plane, whereas for a less efficient strategy this will not be the case. Hence, if we sum this product over all the combinations of a and w, it should tell us which strategy is efficient on average. Thus, we have

$$\eta = \frac{1}{C} \sum_{a,w} P(a, w) \cdot \left(\frac{w}{a}\right)^\beta, \tag{5.1}$$

where C is a normalization constant which is fixed below to correspond to the value of η for the random strategy (normalization constant), (3.1) for $n = N$ or (4.1) for $n < N$. It should also be clear that small values of β may not be able to differentiate strategies well enough since $(w/a)^\beta$ would take similar values all over the plane. We find that $\beta = 2$ leads to a good resolution of the strategies, and henceforth we will fix this value for β.

In Table 1, we indicate zero offset $D = 0$ by adding a 0 index to the acronym for the strategy, e.g., KS^0. Also, since the box strategy requires the specification of a (constant) increment I, we will indicate it as $\mathrm{BS}I$ in the tables.

We have quantified how 'favorable' (more winners with less attempts) each of the strategies are, using the efficiency index η (5.1) which is always normalized by a RS, (3.1) for $n = N$ or (4.1) for $n < N$. From Table 1, it is clear that the box strategy with an arbitrary increment is only as efficient as the random strategy. The same is true for the key strategy when players do not start by opening their own box. It is also evident from the table that as the number of empty boxes increases, all strategies P-functions converge to the random strategy P-functions (4.1).

The second index ϵ_{ij} directly compares how much two P-functions differ from each other by taking the sum of the absolute differences between their values for each combination of a and w, and then averaging these differences over all possible values of a and w:

$$\epsilon_{ij} = \frac{1}{(n+1)N} \sum_{a,w} |P_i(a, w) - P_j(a, w)|, \tag{5.2}$$

where the normalization factor is easily obtained since w takes $n + 1$ values $0, 1, 2, \ldots, n$ and a takes N values $1, 2, \ldots, N$. We note that the error ϵ_{ij} is related to the usual variational distance, given by

$$\delta(P_i, P_j) = \frac{1}{2} \sum_{\omega \in \Omega} |P_i(\omega) - P_j(\omega)|,$$

where Ω denotes the sample space, and P_i, P_j are probability measures, by the formula

$$\epsilon_{ij} = \frac{1}{(n+1)N} \sum_{a,w} |P_i(a, w) - P_j(a, w)| = \frac{2}{(n+1)N} \sum_{k=1}^{a} \delta(P_i(k, \cdot), P_j(k, \cdot)).$$

In other words, the error ϵ_{ij} can be viewed as an average distance between two families of probability measures determined by two strategies. With this in view, we observe that other measurements of error can be defined through different choices of statistical distance δ. In Fig. 15 in Appendix, we show ϵ_{ij} heatmap for different combinations of strategies presented in this paper. It is easy to see from these how strategies with similar (if not identical) efficiency also have small absolute global errors $\lessapprox 1\%$.

Table 1 Efficiency of various strategies described in the paper for three different values of n

Strategy	Escape	η
$n = 100$		
KS^0	–	1.35
KS	RS	1.00
KS	BS	1.00
BS1	–	1.00
BS5	RS	1.00
BS5	BS	1.00
Goyal–Saks	–	1.35
ADI	–	1.35
RS	–	1
$n = 99$		
KS^0	RS	1.21
KS^0	BS	1.26
KS	RS	1.00
KS	BS	1.00
BS1	–	1.00
BS5	RS	1.00
BS5	BS	1.00
Goyal–Saks	–	1.12
ADI *	–	1.30
RS	–	1
$n = 50$		
KS^0	RS	1.01
KS^0	BS	1.04
KS	RS	1.00
KS	BS	0.99
BS1	–	0.99
BS5	RS	1.00
BS5	BS	1.00
Goyal–Saks	–	0.99
ADI	–	1.00
RS	–	1

Note that, since the values are obtained from P-functions simulations using Monte Carlo sampling, they are not exact. Above, we show only the statistically meaningful decimals for the estimated efficiency. (*Here the η value is written for $n = 98$ since ADI for $n = 99$ is identical to $n = 100$)

5.2 Convergence to random

Finally, to conclude our analysis we present a conjecture that is suggested by our simulations and results (see [12] for experiments). Consider the (discrete) space of all properly bounded strategies $\mathcal{S}$ which determines the space $\mathcal{P}$ of all corresponding P-functions. The error $\epsilon_{ij} = \epsilon_{ij}^{n,N}$ between two strategies S_i and S_j defined in 5.2 depends on n and N. It is straightforward

to check that ϵ_{ij} defines a metric on $\mathcal{P}$ for every value of n and N. Hence we present the following.

Conjecture 5.1 *For any properly bounded strategy S, excluding* KS^0*, its P-function converges to the P-function of the random strategy as N grows large or as n/N grows small, in particular*

$$\lim_{N\to\infty} \epsilon_{S,\mathrm{RS}} = 0\,, \qquad \text{for } n = N, \tag{5.3}$$

$$\lim_{n/N\to 0} \epsilon_{S,\mathrm{RS}} = 0\,, \qquad \text{for } n < N. \tag{5.4}$$

To phrase (5.3) and (5.4) differently, we also recall that a sequence of random variables X_k is said to converge in distribution to X if their associated cumulative distribution functions $F_k(x)$ converge to $F(x)$ for all x. In our case, where X_k is associated with a strategy S_k, similar to (3.5), we have for any fixed a, n, and N,

$$F_k(w) = P(X_k \le w) = 1 - P^{\min}_{S_k}(a, w+1). \tag{5.5}$$

Then we may refine the conjecture to the statement that for any properly bounded strategy S different from KS^0, the associated random variable S converges to X_{RS} as N grows large. Furthermore, even the KS^0, for which the chances of winning the classic (constrained) problem were many orders of magnitude larger than the chances of winning using the random strategy, when considering the unconstrained problem treated is this paper, is only slightly ($\sim 25\%$) more efficient than the random strategy in the a–w plane.

Remark 5.2 Finally, we present informally an alternative formulation of the above conjecture, which does not rely on limits for N or n. A nontrivial change in the algorithmic steps of a strategy S (e.g., the value of S on $B \times K$ for each prisoner $P_i \in P$) will move the point S in $\mathcal{S}$. The closer two points S and S' in $\mathcal{S}$, the smaller the differences (absolute value of the error or estimated efficiency) between the two strategies. Then we expect that there is a sequence of modifications of a strategy S_k such that S_k converges to the RS in distribution. Put differently, we can say that the random strategy acts as an *attractor* in the space of all (properly bounded) strategies one can generate to solve the PSG.

6 Future directions: rigged search games, storage strategies and memory

In this paper, we discussed a multitude of inequivalent strategies to approach the unconstrained 100 prisoner problem, or PSG. Of course, any optimal retrieval process must clearly depend on the storage processes preceding it. Here, we analyzed the case of storage strategy in which n keys are randomly distributed in N boxes of capacity 0 or 1 (each box can contain at most one key) and show that all classes of strategies (except the key strategy with fixed initial conditions, $D = 0$ and $n = N$) will achieve performances comparable, or converging, to the random strategy performances. However, as it turns out the key strategy itself is not much more efficient than the random strategy, as it is easy to verify by computing the efficiency of a perfect God-strategy (see footnote 3), $\eta \sim 5.6$. As expected, these results indicate that classical retrieval strategies for the unconstrained problem cannot improve by much the random strategy efficiency, since the storing process was fixed to be random.

On the other hand, the quantum (constrained) variant of the locker puzzle [1] has been shown to be trivial, as it allows for maximum winners with 100% chances in all fair games for $a \ge \frac{\pi}{4}\sqrt{N}$. Based on recent discoveries [14], it would seem that neither the classical nor the

quantum variant of this search game is fit to accurately describe memory retrieval processes in living organisms, but instead they constitute the two extreme mathematical abstractions of realistic memory retrieval processes. Natural organisms endowed with memory certainly do not store information in random fashion nor do they retrieve them randomly, and are most likely not constrained by quantum mechanical rules, at least at the collective level. Rather, the storage follows specific patterns, and consequently the same happens during retrieval processes. Furthermore, in various circumstances, memory retrieval processes are facilitated by external (or internal) cues (see for instance [4]), which allow for the right neuronal connection to be turned on, producing memory recall.

The goal of this paper was to pave the way for the mathematically rigorous but biology-driven analysis of memory in living organism, animate or inanimate. Of course, to describe accurately biological memory processes one would need to consider further extensions or modifications of the PSG. For instance, crucial to resolve is the equivalent problem posed on storing strategies: given a fixed retrieval strategy, what is the storing strategy which optimizes the probability of (partial) information retrieval? Alternatively, is there a storing strategy which optimizes the probability of (partial) information retrieval, regardless of the retrieval strategy?

To answer these questions in general terms, it will be necessary to consider boxes with capacity $l > 1$, i.e., each box can contain at most l keys: this simple change will destroy the key-box cycle structure created by randomly distributing keys in boxes all with exact capacity 1, and create branching as well as overlapping cycles structures. Equivalently intriguing is the generalization whereby each prisoner P_i looks for her key i, but also for other distinct keys $j, k, \ldots$ which also open her cell door.

One would certainly expect these extended PSGs to possess novel, less-smooth P-functions, obviously related to the chosen storing strategy, which may be directly confronted to efficiency or performance of biological mnemonic processes.

Acknowledgements IL acknowledges two interviewers from Alibaba who accidentally made him aware of the problem. SMS acknowledges funding from the DST-INSPIRE Faculty Fellowship (DST/INSPIRE/04/2018/002664) by DST India. TAW was partially supported by NSF grant DMS-2212924.

Author Contributions IL, SMS and TAW have contributed equally.

Declarations

Conflict of interest The authors declare no competing interests.

Appendix A: Unbounded (pure) random strategy

In this appendix, we present the analytic P-function formula for the unbounded random strategy. This variant of the strategy (for $n = N$) presented in 3.1 allows every prisoner to choose and open a box that was already opened. This could happen, for instance, if the prisoner suffers from short-term memory loss or if the amount of boxes she opened is so large that she would not remember them all. At each box selection a prisoner has 1 chance of getting the right box with the right key, and $N - 1$ chances of getting the wrong box. At the next selection, she again has 1 chance of finding the right box, and again $N - 1$ of failing. Instead of summing all chances that the prisoner gets her key in the first attempt, or the second, or multiple attempts, we can compute directly the chances that a prisoner does not obtain her key in a attempts,

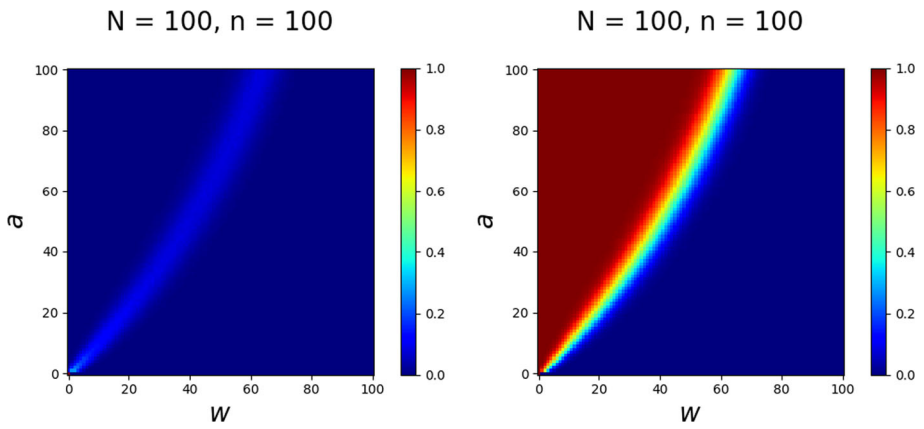

Fig. 4 The P-functions for the pure random strategy, the only example of unbounded strategy presented in this paper

$$\left(\frac{N-1}{N}\right)^a. \tag{A1}$$

Consequently, the probability that a prisoner does get her key within a attempts is easily obtained by subtracting the above number from the unity. If we consider N players and impose that exactly w must be winners, we obtain the P-function for the pure random strategy:

$$P_{\text{pureR}}(a, w) = \binom{N}{w}\left[1 - \left(\frac{N-1}{N}\right)^a\right]^w \left[\left(\frac{N-1}{N}\right)^a\right]^{N-w}. \tag{A2}$$

In Fig. 4, we present the P-function for $N = 100$. The minimum-winner P-function for the pure random strategy is everywhere smaller than (or equal to) the properly bounded random-strategy P-function, as expected from the discussion in Sect. 2.1, but this inequality does not hold for the exact-winner P-function in the whole a–w plane.
This is also confirmed by the estimation of the efficiency for this unbounded strategy, $\eta = 0.66$, the lowest efficiency value encountered throughout the analysis.

Appendix B: Proofs of estimates

In this appendix, we prove the formulas presented in the main text for the P-functions of the Key and Box strategies.

Lemma B.1 *The probability for all N players to find their key within a attempts, where $1 \le a \le N$ is*

$$P_{\text{KS}^0}(a, N) = 1 - \sum_{k=a+1}^{N} \frac{\Omega_k^N}{N!}, \tag{B1}$$

where Ω_k^N is defined by (B5) *and* (B3).

Proof Call $r = \lfloor N/l \rfloor$, so $r \geq 1$ always and in (3.8) $r = 1$. By definition, for $r \geq 1$:

$$\Omega_l^N = \sum_{i=1}^{r} \Omega_{\alpha_l=i}^N, \qquad \Omega_{\alpha_l=0}^N = N! - \Omega_l^N, \tag{B2}$$

where $\Omega_{\alpha_l=i}$ is the number of permutations which contain exactly i many l-cycles. Now, a known formula counts the number of permutations which have a specific cycle decomposition or partition of cycles $p = (\alpha_1, \ldots, \alpha_N)$, Eq. (3.9). This means that if all partitions of N containing a certain number of l-cycles are known, a repeated use of the (B2) gives us Ω_l^N. However, the number of partitions grows rapidly with N, so the brute-force approach is not feasible. To solve this problem, we proved an identity which counts the number of permutations containing a precise number k of l-cycle, i.e., $\alpha_l = k$, in terms of the number of permutations containing exactly $k - k_1 \geq 0$ and k_1 l-cycles, respectively:

$$\Omega_{\alpha_l=k}^N = \frac{\binom{N}{k_1 \cdot l}}{\binom{k}{k_1}} \Omega_{\alpha_l=k_1}^{k_1 \cdot l} \, \Omega_{\alpha_l=k-k_1}^{N-k_1 \cdot l} \tag{B3}$$

valid for all values of $N, k > 0$ and for all k_1 such that $k \geq k_1 \geq 1$ (in fact we obtain a trivial identity for $k_1 = 0$). In the above, the binomial coefficients count all the ways $k_1 \cdot l$ elements can be chosen from N elements and the number of ways k_1 l-tuples can be chosen among k. Note also that (3.8) is a special case of (B3). Of course, some values of the factors Ω can be easily computed by using (3.9) or simple definitions, such as:

$$\Omega_{\alpha_l=k_1}^{k_1 \cdot l} = \frac{k_1 \cdot l!}{l^{k_1} k_1!}, \qquad \Omega_{\alpha_l=1}^{l} = (l-1)!, \qquad \Omega_{\alpha_l=0}^{N-l} = (N-l)! \quad \text{for } l > N - l, \tag{B4}$$

where the last formula extends the result (B3) to the case $k = 0$, but only whenever it is not possible to create an l-cycle among $N - l$ elements, simply because there are not enough elements available. In all other circumstances, $\Omega_{\alpha_l}^N = 0$ can only be computed indirectly from the second of (B2). It is a simple bookkeeping exercise to show that the above formula (3.8) can be easily obtained from (B3) by making use of the last two equations in (B4).

We can now derive the generalization of (3.8) to the cases $l < N/2$, i.e., $r \geq 2$: from both (B2), using (B3) and fixing $k_1 = 1$ (to account for the permutations of N containing only 1 l-cycle), we get

$$\begin{aligned}
\Omega_l^N &= \binom{N}{l} \Omega_{\alpha_l=1}^{l} \sum_{i=1}^{r} \frac{\Omega_{\alpha_l=i-1}^{N-l}}{\binom{i}{1}} \\
&= \frac{N!}{l \cdot (N-l)!} \left(\Omega_{\alpha_l=0}^{N-l} + \sum_{i=2}^{r} \frac{\Omega_{\alpha_l=i-1}^{N-l}}{i} \right) \\
&= \frac{N!}{l \cdot (N-l)!} \left[(N-l)! - \sum_{i=1}^{r-1} \Omega_{\alpha_l=i}^{N-l} + \sum_{i=2}^{r} \frac{\Omega_{\alpha_l=i-1}^{N-l}}{i} \right] \\
&= \frac{N!}{l \cdot (N-l)!} \left[(N-l)! - \sum_{i=1}^{r-1} \frac{i}{i+1} \Omega_{\alpha_l=i}^{N-l} \right].
\end{aligned} \tag{B5}$$

Clearly, if $r = 1$ the above formula reduces to (3.8), whereas if $r > 1$, namely $l < N$, the recursive use of (B3) is convenient to obtain $\Omega_{\alpha_l=i}^{N-l}$. □

For the Box strategy, we first show that the P-function satisfies the same reflection-symmetry property of the random strategy.

Lemma B.2 *For any $1 \leq a \leq N$, $0 \leq w \leq N$, and I coprime to N, the box-strategy exact-winner P-function satisfies:*

$$P_{\mathrm{BS}}(a, w) = P_{\mathrm{BS}}(N - a, N - w). \tag{B6}$$

Proof If I is coprime to N, the box strategy is properly bounded. Hence, any prisoner P_i, given N attempts would open all N boxes, in the order

$$i + D, i + D + I, i + D + 2I, \ldots, i + D + (N - 1)I \pmod{N},$$

with D arbitrary offset. Now, for every permutation such that w players P_j are winners in a attempts, i.e., find their key in one of the boxes

$$j + D, j + D + I, \ldots, j + D + (a - 1)I \pmod{N},$$

there are $N - w$ players P_l which did not find their key in a attempts, but would have certainly found their key in one of the unopened boxes

$$l + D + aI, l + D + (a + 1)I, \ldots, l + D + (N - 1)I \pmod{N}.$$

Hence, the players P_l would have won, i.e., would have found their key, if they used the box strategy with offset $D' = D + aI$, same increment I and also had at their disposal $N - a$ attempts. We then arrive at the equality $P^{D}_{\mathrm{BS}}(a, w) = P^{D'}_{\mathrm{BS}}(N - a, N - w)$. Finally, because the P-functions of the box strategy are independent of D, as we shall argue at the end of the section, the identity (B6) follows. □

It should be quite clear why the same property will not hold for the key strategy. In that case, in fact, no player can know, before playing, which box(es) she will open for any given attempt $a > 1$. If w players find their key in a attempts, the remaining $N - w$ players could not now a priori the box to start from, hence would not have certainly found their cycle, and hence their key.

Proof of Proposition 3.4 We first consider $a = 1$. This case corresponds to the "problème des rencontres": we want to evaluate the number of permutations of N such that no players will find their key in its box. If k elements are fixed in a position, there are exactly $(N - k)!$ permutations of the remaining elements in the remaining positions. Furthermore, there are $\binom{N}{D}$ ways to select those elements. Hence, from the inclusion–exclusion principle,[8] we get:

$$D_{N,0} = \sum_{k=0}^{N} (-1)^k \binom{N}{k} (N - k)! = N! \sum_{k=0}^{N} \frac{(-1)^k}{k!}, \tag{B7}$$

where the index 0 indicates that all elements possess the property a, or no element possesses the property $\bar{a}$. This formula is well known and counts the number of "derangements" $D_{N,0}$, permutations with no trivial cycles. It can easily be generalized: if we are interested only in $N - w$ elements satisfying the property a, or equivalently w elements satisfying $\bar{a}$ (and hence permutations with w trivial cycles), we then obtain:

$$D_{N,w} = \binom{N}{w} D_{N-w,0} = \binom{N}{w} (N - w)! \sum_{k=0}^{N-w} \frac{(-1)^k}{k!} = \binom{N}{w} !(N - w), \tag{B8}$$

where the symbol $!p$ indicates the derangements and the sum has been truncated to reflect the fact that now we want at most $(N - w)$-tuples of elements to satisfy the condition a at

[8] This principle was in fact firstly used to find a solution to the "problème des rencontres" [10].

the same time. We have also multiplied by a binomial coefficient, which simply counts the w-tuples of elements falling in their positions, also called the "hits," within N total elements. Now it is straightforward to obtain the explicit formula for the probability for w players to find their key at the first attempt

$$P_{\mathrm{BS}}(1, w) = \frac{D_{N,w}}{N!} = \binom{N}{w} \frac{(N-w)!}{N!} \sum_{k=0}^{N-w} \frac{(-1)^k}{k!}$$
$$= \frac{1}{w!} \sum_{k=0}^{N-w} \frac{(-1)^k}{k!} \tag{B9}$$

as desired. Note that, from (B6) we get:

$$P_{\mathrm{BS}}^{N}(N-1, N-w) = \frac{1}{(N-w)!} \sum_{k=0}^{w} \frac{(-1)^k}{k!}. \tag{B10}$$

It is also instructive to derive the above formula from the generating polynomial of derangements [10],

$$\sum_{k=0}^{N} \frac{N!}{k!} (x-1)^k. \tag{B11}$$

The coefficient of the term x^w enumerates the number of permutations such that w players have hits, i.e., they find their key in their-number box in one attempt. By expanding the polynomial, using the fact that the diagonals of the Pascal's triangle are given by binomials,[9] we get:

$$P_{\mathrm{BS}}^{N}(1, w) = \frac{1}{N!} \sum_{k=w}^{N} (-1)^{k-w} \frac{N!}{k!} \binom{k}{w}, \tag{B12}$$

which is equal to (B9) after a few simplifications and a shift in the range of k.

Next, we consider the case $a = 2$. This case is complementary to the "problème des ménages" [7, 8] formulated as: What is the number of permutations such that no element i is found in position $(i, i+1)$ mod N (meaning that the last element N cannot be in position N or 1)?

We can derive the formula directly from the known generating polynomial [10]:

$$\sum_{k=0}^{N} \frac{2N}{2N-k} \binom{2N-k}{k} (N-k)!(x-1)^k. \tag{B13}$$

As before, the coefficient $C_{N,w}$ of the term x^w enumerates the number of permutations such that w players have hits, i.e., find their key in their-number box or in the following one (clearly no overlap is allowed, each box contains one and only one key, so no two players can find their key in the same box). The probability then reads

$$P_{\mathrm{BS}}^{N}(2, w) = \frac{C_{N,w}}{N!} = \frac{1}{N!} \sum_{k=w}^{N} (-1)^{k-w} \frac{2N}{2N-k} \binom{2N-k}{k} (N-k)! \binom{k}{w} \tag{B14}$$

as desired. It is easy to see the similarities with (B12). □

[9] To be more precise, consider the ith elements on the left (right) edge of the Pascal's triangle. The elements jth lying on the diagonal starting from the element, then reaching the lower right (left) successive elements is simply given by $\binom{i}{j-1}$.

Appendix C: Hybrid strategies P-functions plots

In this appendix, we present the plots for the P-function of the hybrid strategies discussed in Sect. 4 (Figs. 5, 6, 7, 8, 9, 10, 11, 12, 13, 14).

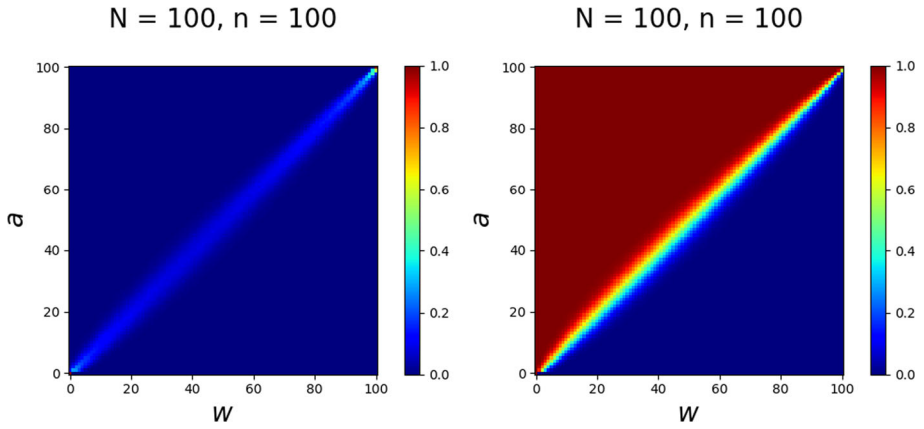

Fig. 5 The key strategy P-function with $D \neq 0$, made properly bounded by the use of the random strategy as an escape, $E = R$. Note that the graph does not change if the value of the offset is different, or not constant for different prisoners, and also if the escape strategy is not the random but the box strategy $E = B$

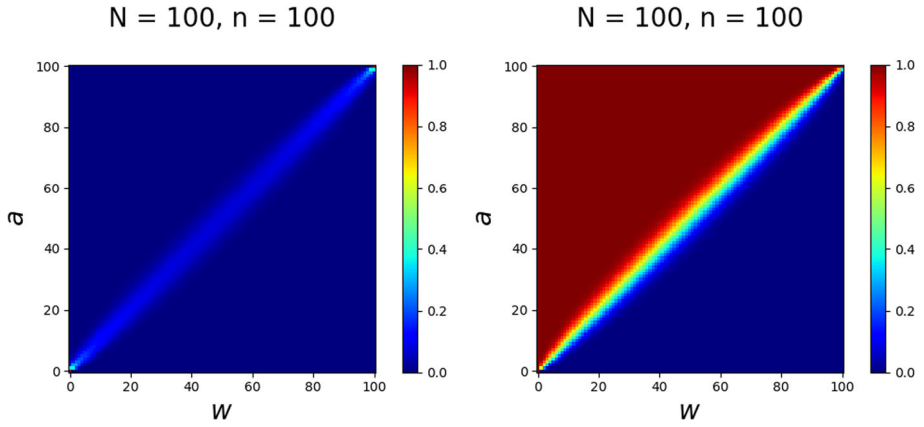

Fig. 6 The box strategy P-function for $I = 5$ (not coprime with $N = 100$) and $D = 0$, made properly bounded by the use of the random strategy as an escape. Note that the graph does not change for a different choice of the offsets or if the escape strategy chosen is the box strategy with a different increment

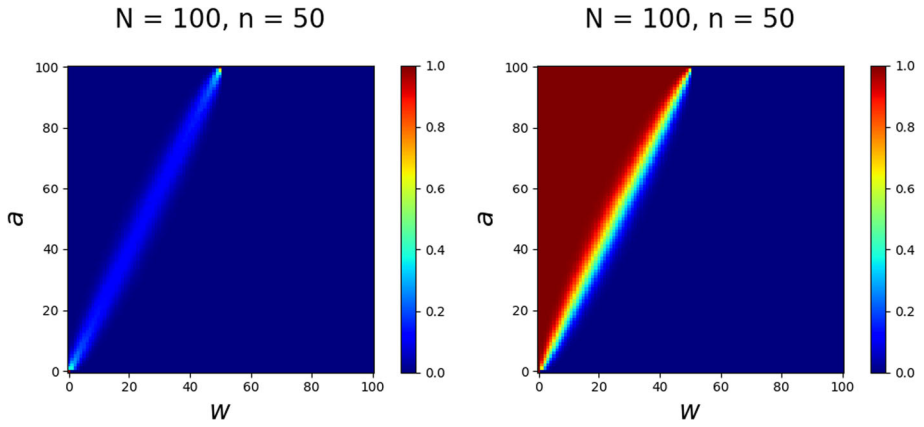

Fig. 7 The properly bounded random-strategy P-functions for $n < N$

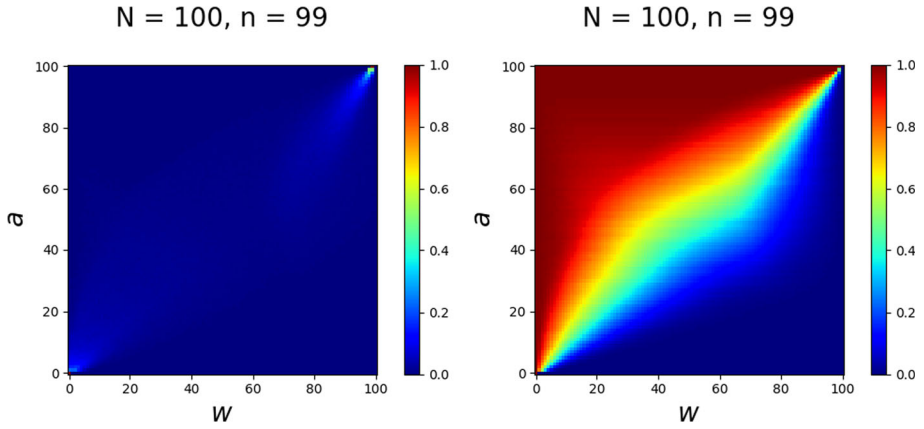

Fig. 8 P-functions of the key strategy with $D = 0$ and escape $E = R$

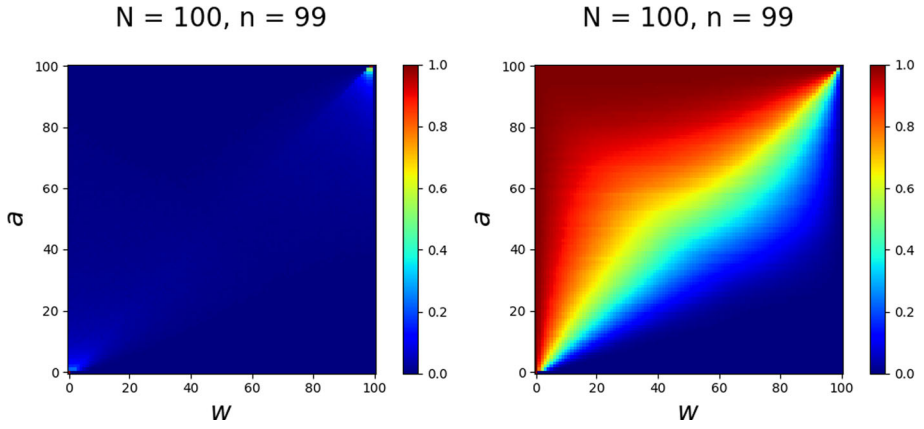

Fig. 9 P-functions of the key strategy with $D = 0$ and escape $E = B$

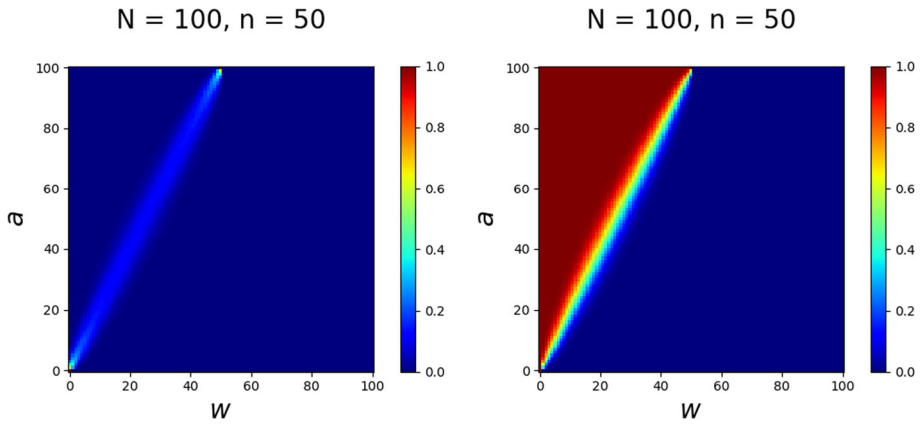

Fig. 10 The unbounded box-strategy P-function with $D = 2$ and $I = 5$ (not coprime to $N = 100$), made properly bounded by the use of the random strategy as an escape, $E = R$. Note that the graph does not change for a different choice of the offsets or escape route

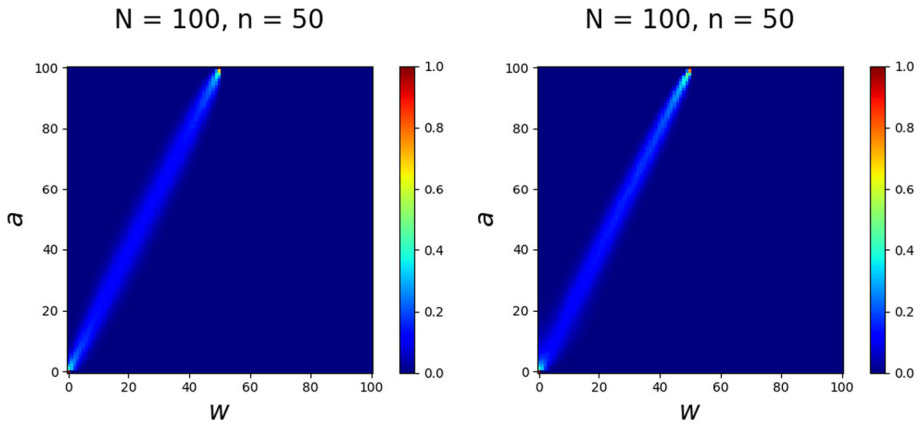

Fig. 11 The exact winners P-function for the hybrid strategy, KS with $D = 0$, made properly bounded by either the random (top) or sequential (bottom) escape route. Though almost identical, note that (non-statistical) differences for $a, w < 20$, showing the two strategies are in fact different

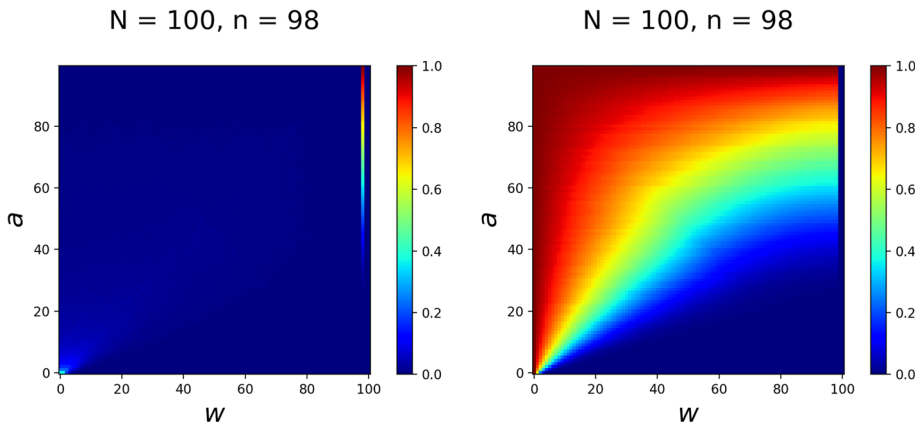

Fig. 12 The ADI strategy, for $n = 98$

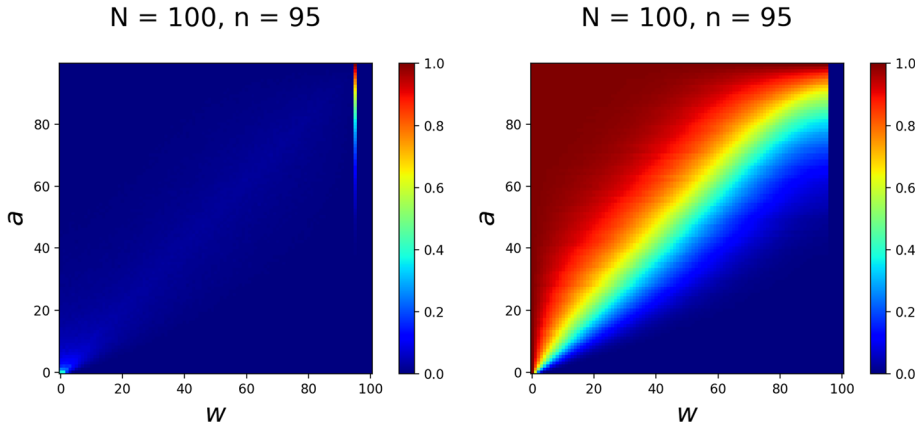

Fig. 13 The ADI strategy, for $n = 95$

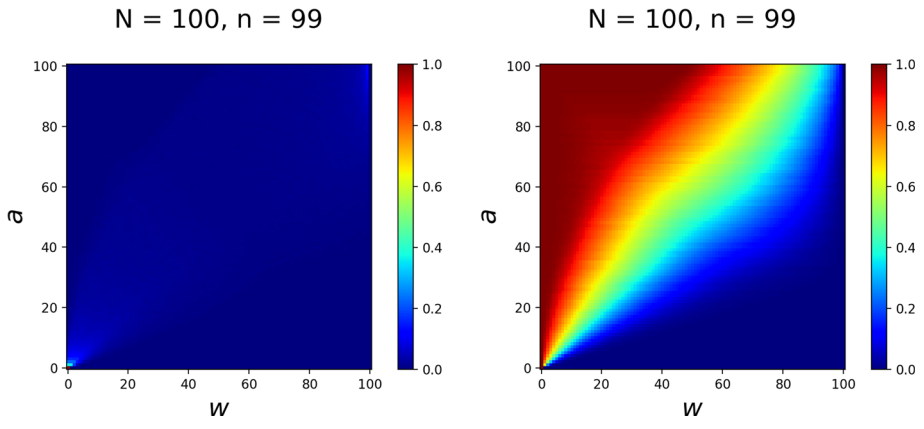

Fig. 14 The Goyal–Saks strategy, the only example of a (not properly) bounded strategy presented in this paper

Appendix D: Error ε_{ij}

See Fig. 15.

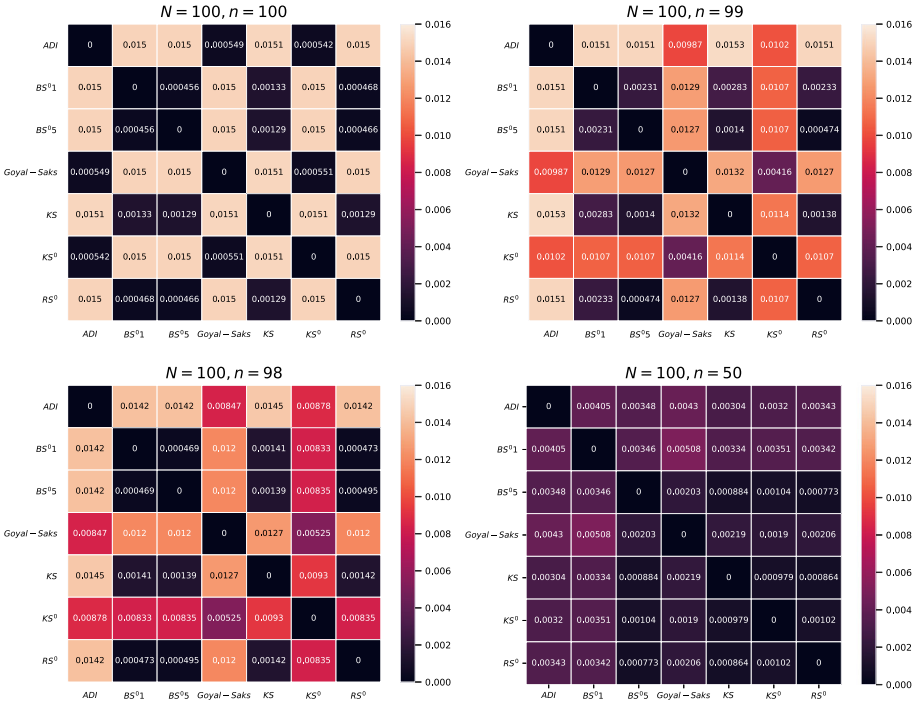

Fig. 15 Errors heatmap for various values of n. For each n, and for each pair of strategies, the absolute error is calculated using (5.2)

References

1. Avis, D., Broadbent, A.: The quantum locker puzzle. In: 2009 Third International Conference on Quantum, Nano and Micro Technologies, IEEE, pp. 63–66 (2009)
2. Avis, D., Devroye, L., Iwama, K.: The locker problem with empty lockers. Int. J. Comput. Inf. Eng. **3**(12), 2747–2750 (2009)
3. Curtin, E., Warshauer, M.: The locker puzzle. Math. Intell. **28**(1), 28–31 (2006)
4. Frankland, P.W., Josselyn, S.A., Köhler, S.: The neurobiological foundation of memory retrieval. Nat. Neurosci. **22**, 1576–1585 (2019). https://doi.org/10.1038/s41593-019-0493-1
5. Gál, A., Miltersen, P.B.: The cell probe complexity of succinct data structures. Theoret. Comput. Sci. **379**(3), 405–417 (2007)
6. Goyal, N., Saks, M.: A parallel search game. Random Struct. Algorithms **27**(2), 227–234 (2005)
7. Kaplansky, I.: Solution of the problème des ménages. Bull. Am. Math. Soc. **49**, 784–785 (1943)
8. Kaplansky, I., Riordan, J.: The problème des ménages. Scripta Math. **12**, 113–124 (1946)
9. Riordan, J.: Discordant permutations. Scripta Math. **20**, 14–23 (1954)
10. Riordan, J.: An Introduction to Combinatorial Analysis. Wiley Publications in Mathematical Statistics, Wiley, New York (1958)
11. Santmyer, J.M.: Five discordant permutations. Graphs Comb. **9**(3), 279–292 (1993)

12. Shekatkar, S.M., Lodato, I.: The `prisoners-search-game` package. https://figshare.com/articles/prisoners-search-game/13482591
13. Whitehead, E.G., Jr.: Four-discordant permutations. J. Aust. Math. Soc. **28**(3), 369–377 (1979)
14. Zheng, J., et al.: Neurons detect cognitive boundaries to structure episodic memories in humans. Nat. Neurosci. **25**, 358–368 (2022). https://doi.org/10.1038/s41593-022-01020-w