AUSTRALASIAN JOURNAL OF COMBINATORICS
Volume **91(2)** (2025), Pages 301–325

Combinatorial aspects of the card game War

Tanya Khovanova

Mathematics Department
Massachusetts Institute of Technology
Cambridge, MA 02139, U.S.A.
tanya@math.mit.edu

Atharva Pathak

Princeton University
Princeton, NJ 08544, U.S.A.
atharva.pathak@princeton.edu

Abstract

This paper studies a single-suit version of the card game War on a finite deck of cards. There are varying methods of how players put the cards that they win back into their hands, but we primarily consider randomly putting the cards back and deterministically always putting the winning card before the losing card. The concept of a passthrough is defined, which refers to a player playing through all cards in their hand from a particular point in the game. We mostly consider games in which the second player wins during their first passthrough. We introduce several combinatorial objects related to the game: game graphs, win-loss sequences, win-loss binary trees, and game posets. We show how these objects relate to each other. We enumerate states depending on the number of rounds and the number of passthroughs.

1 Introduction

Card games are interesting to children and mathematicians alike. Games like poker and blackjack are fascinating applications of game theory, probability, and statistics. Bayer and Diaconis's research into card shuffling [3] has had practical applications in casinos, summarized in Diaconis's discussion with Haran [8]. There is also ongoing research in games where one player must sequentially guess cards in a deck given varying shuffling methods and varying information feedback regarding the guesses [2, 4, 5, 7].

This paper focuses on the card game War, for which some previous research exists. Haqq-Misra [10] finds a linear regression on the chance of a player winning a game on

ISSN: 2202-3518

a standard 52-card deck given two initial statistics: the difference in the sum of card values for the two players and the number of rounds a player wins during the first 26 rounds. Lakshtanov and Roshchina [6] show that, with random putback of cards, the expected duration of the game is finite. Alexeev and Tsimerman [1] investigate a game similar to War, where instead players randomly draw cards from their hands, and the player with the higher card in a round wins the lower card but discards the higher card. Ben-Naim and Krapivsky [9] consider a similar stochastic model of War where cards are drawn randomly from the players' hands, but the winner of a round keeps both cards. They find the expected length of a game on N cards is $\mathcal{O}(N^2)$ or $\mathcal{O}(N^2 \log N)$ depending on the initial advantage of the eventual winner. Spivey [13] investigates and classifies some types of cycles in games of War given deterministic putback.

We note that previous analysis in War generally applies to variations where the players randomly draw cards from their hands [1, 9] or are statistical analyses of large decks [10]. We mostly focus our analysis on standard rules of War in situations where one player has many more cards than the opponent. In contrast to Spivey [13], we focus on situations where one player wins, rather than games that cycle.

Here we define the rules of War in the formulation we investigate. The game is played between two players, whom we refer to as Alice and Bob. A deck of n cards, labeled 1 through n, is divided between them, with their hands face down. In each round, the players reveal the top cards from their hands, with the player whose card was higher collecting both cards to the bottom of their hand. The player who loses all their cards first loses the game.

Section 2 introduces the notation for how we represent games and some additional definitions. A single-use game is a game where Bob wins while going through his initial hand. Depending on how the players collect cards to the bottom of their hands, our results do or do not depend on whether the game is single-use. We define a passthrough as a player going through the cards in their hand from a certain point in the game. In a single-use game, the game ends on Bob's first passthrough. The number of passthroughs for Alice is an important parameter in our calculations. We describe two ways that players may put cards they win back into their hands: WL-putback, where the winning card is put before the losing card, and random putback, where the order is chosen at random.

Section 3 introduces the game graph. The game graph has cards as vertices, and cards are connected if they play against each other. We prove that the game graph of a single-use game is a forest. We introduce the notion of blocks, which are irreducible components of the game.

In Section 4 we introduce win-loss sequences that describe the results of every round and prove that the number of win-loss sequences corresponding to a single-use game is an entry in the Catalan triangle. We also prove a recursion for the number of win-loss sequences corresponding to games where Alice undergoes at most k passthroughs. We describe a bijection between win-loss sequences corresponding to k passthroughs and binary trees of height k. We show how to build the game

graph given the win-loss sequence.

In Section 5 we find the probability that a randomly chosen initial state leads to a game with (1) exactly R rounds or (2) at most k passthroughs for Alice. The results hold for random putback even if the game is not single-use as long as neither player loses at any point during the game. We also show that these probabilities are the same for WL-putback when single-use is assumed.

In Section 6 we count the number of initial states of single-use games that necessarily follow a given win-loss sequence using WL-putback and random putback. To do so, we construct posets that encode the relationships between cards that must be satisfied and count the number of initial states that satisfy the relationships.

2 Preliminaries and Definitions

We consider two paradigms for how players put the cards they win in a round back to the bottom of their hand, which we call the *putback*. Players may put the cards back *randomly*, with Alice's card and Bob's card in either order, or deterministically. There are a few plausible ways of deterministically putting cards back, but we only consider *WL-putback*, where the winning card is put back first and the losing card is put back second.

We represent a *state* of the game by a string of the form $a_1 \dots a_i | a_{i+1} \dots a_n$, where $a_1 \dots a_i$ represents Alice's hand from top to bottom, $a_{i+1} \dots a_n$ represents Bob's hand from top to bottom, and the vertical bar character | denotes the separation between the two players' hands. The values of these cards range from 1 to n, as stated earlier. An example of this notation is given in Figure 1.

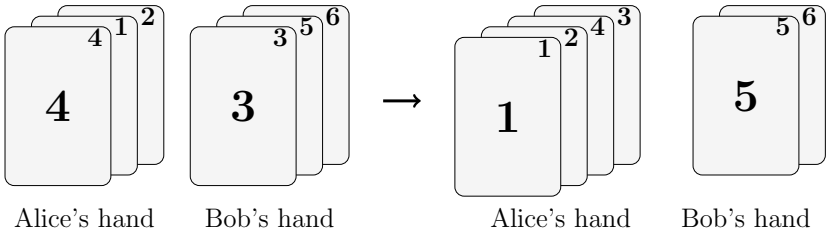


Figure 1: The first state is 412|356. After the round where Alice's card 4 beats Bob's card 3, and after Alice uses WL-putback, the state becomes 1243|56.

Example 1. Consider the game with initial state 2|13 under random putback. Alice wins the first round because her card 2 is greater than Bob's 1. She randomly puts the cards back into her hand, so the new state is either 12|3 or 21|3. In both these cases, Bob wins the second round because he has the highest card of 3, and then the possible states are $2|13, 2|31, 1|23, 1|32$. The first of these four cases is the initial state that we analyzed previously. In the latter three of the four cases, Bob wins the next round and, therefore, the game.

The player with the highest card can never lose in a single-suit game of War.

However, a loop is possible, as the game can alternate indefinitely between the states 12|3 and 2|13.

It is then immediate that there are $(n+1)!$ states of the game: each permutation of the bar and the numbers 1 through n represents a unique state of the game. This number includes states where the bar is at the beginning, corresponding to Bob winning, and states where the bar is at the end, corresponding to Alice winning.

We define an *m-card state* as a state where Alice has m cards. We refer to 1-card states as *unicard states*. Note that for any n, there are $n!$ m-card states for each $0 \leq m \leq n$.

The concept of a *passthrough* is vital to our discussion throughout the paper. Given a state where a player has m cards, we say that their hand undergoes a passthrough after m rounds, and that these m rounds occur during the passthrough. The concept of a passthrough encodes when a player's hand has been "used up". We normally refer to passthroughs from the initial game state.

A *single-use game* is a game in which Bob wins during his first passthrough. A game that ends within k passthroughs for Alice is a *k-passthrough game*. A game that ends in exactly R rounds is an *R-round game*. Note the difference between ending *within* k passthroughs and ending in *exactly* R rounds.

For simplicity, we sometimes implicitly refer to the game played from some initial state as simply that state. For example, when we say a state is R-round, we mean the game initialized from that state is R-round.

We will need some standard definitions in combinatorics. To recall, a *full binary tree* is a tree in which every vertex other than the leaves has two children.

The entry $C(n,k)$ of the Catalan triangle is defined as the number of strings of n U's and k D's such that each initial substring has at least as many U's as D's [11]. It is known that

$$C(n,k) = \binom{n+k}{k} - \binom{n+k}{k-1} = \frac{n+1-k}{n+1}\binom{n+k}{k}.$$

We use C_r to denote the rth Catalan number $\frac{1}{r+1}\binom{2r}{r}$. The Catalan numbers correspond to the diagonal of the Catalan triangle: $C_r = C(r,r)$.

3 Game Graphs and Blocks

We construct a *game graph* as follows. We begin with vertices for each of the n cards. As we watch the game progress, we add edges between cards that play each other. Game graphs can have multiple edges between two cards if they are played against each other multiple times.

Proposition 3.1. *The game graph of a single-use game is a forest with m nontrivial trees and x isolated vertices, where m is the number of Alice's initial cards and x is the number of Bob's cards that never get played.*

Proof. We initialize the game graph with all cards as nodes, and then we add edges corresponding to rounds. We color nodes for each of Alice's m cards as amber and Bob's as blue. For every round, we add an edge between the two cards that are played and make both cards amber. Because the game is single-use, Bob only ever plays cards from his initial hand, so Bob only ever plays blue cards. And since Alice's initial cards are amber and any cards she wins from Bob are recolored to amber, Alice only ever plays amber cards.

Therefore, amber connected components grow only by the addition of blue vertices which are subsequently recolored amber, so the amber connected components stay disconnected from each other and grow as trees. Moreover, each of Alice's initial cards plays at least once, so they are all in nontrivial trees. Cards that never get played for Bob remain as isolated blue vertices. □

We note that the diameter of the game graph when Alice has gone through k passthroughs and Bob is still in his first passthrough is at most $2k - 1$. This is immediate by induction. After the 1st passthrough, the maximal distance between two connected cards is 1. Each card in Alice's hand at the end of the kth passthrough plays one card in the $(k + 1)$st passthrough. So if some pair of cards with maximal distance after the kth passthrough are in fact both still in Alice's hand after the kth passthrough, then the cards they play against in the $(k + 1)$st passthrough have distance 2 more than the the maximal distance after k passthroughs, so the maximal distance after the $(k + 1)$st passthrough increases by 2.

For a single-use game, the game graph decomposes into connected components. Treating cards outside a particular component as invisible, the component behaves as its own independent subgame of War. We call these subgames *blocks*. The components that are not isolated vertices contain exactly one card from Alice's initial deck. This is why the unicard states play an important role: they are the building blocks of single-use games.

There are two ways we can turn our game graph into a digraph. We can give direction to edges from the card that won the round to the card that lost the round, or we can give direction to edges from the card Alice played in the round to the card Bob played in the round.

The first method turns the game graph into a poset, where greater cards point towards lesser cards. We call such posets *winner-to-loser game digraphs*. These posets will be explored in greater depth in Section 6.

We call the digraph resulting from the second method an *Alice-to-Bob game digraph*. In an Alice-to-Bob game digraph for a single-use game, the m directed trees have unique roots. These unique roots are Alice's m initial cards, because the trees were built starting from her initial cards and adding edges pointing to cards Bob played.

Consider what the game graph would have been if we started tracking rounds from some later point in the game. If we built the game graph starting from round i instead of round 1, the effect would be to delete the edges corresponding to rounds 1 to $i - 1$. In the Alice-to-Bob digraph starting from round i, connected components

are trees whose roots are the cards Alice had at that point in the game. We refer to subgames involving cards and rounds in each of these trees as *subblocks* of the original game. We say a subblock is *induced* by the card Alice plays in the first round of the subblock, which is the root of the tree. A card induces a subblock each time it plays, so we always specify the round from which the subblock we want to discuss is begun.

We note that a block, and all the rounds and cards in it, only becomes well-defined for a deterministic putback game after an initial state is selected and only becomes well-defined for a random putback game after the game is played in its entirety. Nevertheless, we will utilize the notion of the block that a card will go on to induce to make claims regarding the probabilities that certain events take place.

Note that although a unicard game consists of only one block, it immediately separates into two subblocks after a first round win for Alice. For example, from the state $a|bcd\ldots$, if Alice wins the first round and uses WL-putback, the state becomes $ab|cd\ldots$, and from this point on a and b induce subblocks.

4 Win-loss Sequences and Binary Trees

4.1 Win-loss sequences

A *win-loss sequence* is a string $x_1x_2\ldots$ of Ws and Ls describing the progress of the game from the point of view of Alice. A W represents a round Alice wins, and an L represents a round Alice loses. We use R to denote the total number of rounds, w_i and ℓ_i to denote the numbers of Ws and Ls within the first i rounds, and we say $w = w_R$ and $\ell = \ell_R$. As a reminder, we only consider games where Alice loses.

We can "stylize" a win-loss sequence with forward slashes to separate passthroughs of Alice's hand. For example, if Alice started with a single card, the win-loss sequence WLL would be stylized as W/LL because the first round constitutes Alice's first passthrough.

The number of letters before the first slash is the number of Alice's cards at the start. This is the same as the number of rounds in the first passthrough. After that, the number of letters between slashes is twice the number of W's in the previous passthrough. As we only consider finite games, the rounds in the final passthrough are all L's. Since each W nets $+1$ cards for Alice and each L nets -1 cards for Alice, the number of cards she has after i rounds is $m + w_i - \ell_i$. She must have a positive number of cards until the end of the game when she has none, so for $1 \leq i < R$ we have $m + w_i - \ell_i > 0$ and $m + w_R - \ell_R = m + w - \ell = 0$.

Note that $w_i + \ell_i = i$, so $w + \ell = R$, and combining this with $m + w - \ell = 0$ finds that m and R have the same parity. We also trivially have $R \geq m$.

We now enumerate the number of win-loss sequences that are exactly R rounds.

Theorem 4.1. *The number of win-loss sequences corresponding to an m-card R-round game is the entry $C(\frac{m+R}{2} - 1, \frac{R-m}{2})$ of the Catalan triangle.*

Proof. Every win-loss sequence ends with an L because Alice must go from having one card to having no cards. Ignore this final L. Then the condition on the remaining win-loss sequence is that no terminal segment of the string can have more Ws than Ls, because that would mean Alice had zero cards at some point before the end of the game. This is equivalent to the condition on strings that Catalan triangle numbers count, with the roles of terminal and initial segments reversed. Since we have ignored the final L, the remaining number of L's and W's is $\ell - 1$ and w respectively, and thus there are $C(\ell - 1, w) = C(\frac{m+R}{2} - 1, \frac{R-m}{2})$ win-loss sequences corresponding to an m-card R-round game. □

Corollary 4.2. *There are $C_{\frac{R-1}{2}}$ win-loss sequences corresponding to unicard games ending in R rounds.*

Proof. Win-loss sequences corresponding to unicard games have $\ell - w = m = 1$, so substitution gives $C(\ell - 1, w) = C(w, w) = C_w$. Unicard games have $w = \frac{R-1}{2}$, so $C_w = C_{\frac{R-1}{2}}$. □

Here, we enumerate the number of win-loss sequences that end within k passthroughs of Alice's hand.

Theorem 4.3. *The number of win-loss sequences corresponding to an m-card k-passthrough game is A_k^m, where A_k is recursively defined by $A_1 = 1$ and $A_{k+1} = 1+A_k^2$.*

Proof. For enumerating win-loss sequences, Bob's cards do not matter, so we can assume he has enough so that the game is single-use for any k-passthrough game. We show inductively that the number of k-passthrough win-loss sequences is A_k for unicard blocks. Then, combining the m blocks' win-loss sequences gives A_k^m combined win-loss sequences. Combining the sequences occurs by concatenating each of the sequences' first passthroughs, then each of their second passthroughs, and so on.

There is only $A_1 = 1$ unicard block 1-passthrough win-loss sequence, namely L itself. Then, there are two cases for a $(k+1)$-passthrough win-loss sequence: either an immediate L, or a W and a continuation to a subsequent passthrough. In the second case, the win-loss sequences for the subblocks induced by the two cards yielded back to Alice from the W must end within k passthroughs, so there are A_k^2 possibilities here. Thus there are $A_{k+1} = 1 + A_k^2$ k-passthrough win-loss sequences for unicard blocks, as desired. □

4.2 Win-loss binary trees

We describe a bijection between unicard games and full binary trees. Recall that "full" means each node has either zero children, in which case it is a leaf, or two children, in which case it is a non-leaf.

We note that the bijection to binary trees in the following theorem works for any win-loss sequence, not just those corresponding to single-use games. However, we do restrict ourselves to games that Alice loses because such games end with a

passthrough of Ls. Also, the concept of win-loss binary trees is easily extended to m-card games and forests of m full binary trees. If the game is single-use, then each of the trees corresponds to a block, but even if the game is not single-use, the concept of win-loss binary trees still applies.

Theorem 4.4. *Win-loss sequences corresponding to k-passthrough unicard games in which Alice loses are in bijection with full binary trees of height k.*

Proof. We describe a construction of the bijection by demonstrating the conversion from win-loss sequences to binary trees and vice versa.

We begin with a root node for the tree at the top associated with the first letter of the win-loss sequence. Each round corresponds to a node in the tree, which we label with W and L for win or loss. Every L yields no cards back to Alice's hand, so we make L nodes terminal leaf nodes. Every W yields two cards back to Alice, and these two cards are thereby associated with this particular W. We give every W two children nodes, which we label W or L depending on whether the corresponding card goes on to win or lose its next round. Importantly, the first card put back corresponds to the left child node, and the second card put back corresponds to the right child node. Once Alice undergoes a passthrough, the cards available for the next passthrough are those won during the prior passthrough. Since we wrote in the left-to-right direction and because the order of cards in Alice's stack is preserved after being put back, each passthrough in the win-loss sequence read left-to-right is exactly a level of the tree read left-to-right.

To convert back from a binary tree to a win-loss sequence, we orient the tree with the root at the top and then write W on all non-leaves and L on all leaves. Then we read off the sequence left-to-right, top-to-bottom. Since the number of leaves is always less than or equal to the number of non-leaves until the very end of the tree, the associated win-loss sequence always has at least as many wins as losses until the end, which means that Alice will have a positive number of cards until the end. Therefore, the win-loss sequence is valid, and the bijection is established both ways. □

Since each round corresponds to a node in a win-loss binary tree, we interchangeably refer to rounds and nodes in the win-loss binary tree.

Example 2. The tree corresponding to the sequence $W/WW/LWWL/LLLL$ played from initial state $a|bcdefghijkl$ with WL-putback is in Figure 2. Because the game is single-use, the cards Bob wins are regarded as discarded. In parentheses, we say which card from Alice plays against which card from Bob. Passthrough is abbreviated as PT.

Now, we note that for single-use games, each subtree encodes the win-loss sequence for a subblock. For example, take the subblock induced by card a in the round a/c in the second passthrough. The subblock plays as $a|cefij$, which follows the win-loss sequence $W/LW/LL$, as shown by the subtree starting from the round a against c in the second level of the tree.

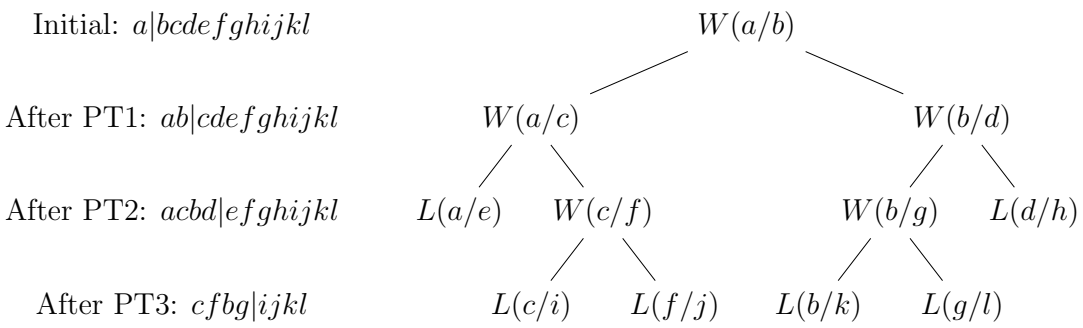


Figure 2: The tree for win-loss sequence $W/WW/LWWL/LLLL$. In the first passthrough, Alice's card a beats Bob's card b, so we create a W for this round, and Alice's hand becomes ab. In the second passthrough, Alice's a beats Bob's c and Alice's b beats Bob's d, so we create a $W(a/c)$ and a $W(b/d)$ as children of $W(a/b)$. Alice's hand becomes $acbd$, of which a and d lose their next round and c and b win their next round, so we create $L(a/e)$ and $W(c/f)$ as children of $W(a/c)$ and $W(b/g)$ and $L(d/h)$ as children of $W(b/d)$. The two wins $W(c/f)$ and $W(b/g)$ yield $cfbg$ for Alice's hand in the final passthrough, where all four cards lose.

4.3 Building the game graph from the win-loss tree

We can create the game graph from a win-loss binary tree labeled with the two cards played in each round: we create a vertex for each card and an edge for each round. We want to show a formal algorithm to convert the win-loss binary tree into the game graph, but we need some definitions first.

Recall an ancestor of a node x is a node on the path between the root and x, inclusive. We say that the *right parent* of a node x is the parent of the closest ancestor of x which happens to be a right child. Note, however, that a node whose ancestors are all left children has no right parent.

Consider a game with WL-putback. At each node of a win-loss binary tree, write the two cards that played in the corresponding round. We have the following lemma that describes the card Bob plays against through the win-loss binary tree.

Lemma 4.5. *Consider a single-use m-card game played with WL-putback. Consider a round Q where card y plays for Bob against card x for Alice. If Q in a win-loss binary tree has right parent node P, then x is the card Bob played in round P. If Q does not have a right parent, then x is the card from Alice's initial hand which induced the block that Q is in.*

Proof. After Alice wins a card x from Bob in round P, it is put back second and hence plays in the right child round the next time it plays. Each time x wins a subsequent round, it gets put back first due to WL-putback and hence plays in the left child round for the next time it plays. Eventually, x plays y in round Q. Working backwards to get from node Q to node P in the win-loss binary tree, we have to find the nearest ancestor of Q which is a right child, corresponding to x being

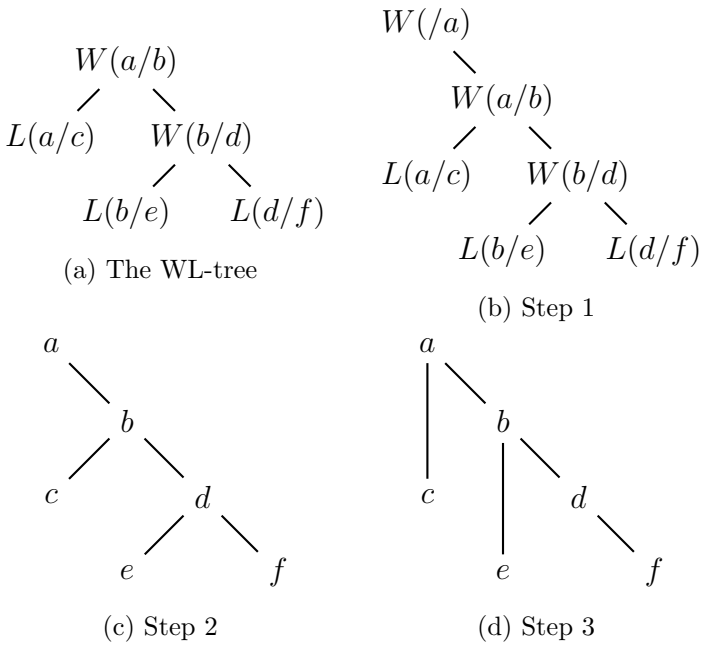
put back second after being won from Bob, and then this nearest ancestor's parent, corresponding to the round P where Bob played x.

A special case occurs if Alice never won card x from Bob; i.e., when Alice began the game with x. In this case, x is card a from Alice's initial hand, and as long as x continues winning, it gets put back first and next plays in the left child round. In this case, each round x plays in has no right parent. Note that x induces the block in which round Q occurs. □

Now we show the steps to obtain the game graph for a single-use WL-putback game from the forest of win-loss binary trees labeled with the two cards played in each round.

1. Add a node for each of Alice's initial cards above and to the left of the root of the tree they induced, so that the true root node is effectively a right child of this new node.
2. Change the labels to the card played by Bob.
3. Replace all edges with edges connecting each node to its right parent.

We show the steps in Figure 3 for the unicard game starting from initial state $a|bcdef$ and following the win-loss sequence $W/LW/LL$.



(a) The WL-tree (b) Step 1 (c) Step 2 (d) Step 3

Figure 3: Getting the game graph from a win-loss binary tree with WL-putback

Theorem 4.6. *The algorithm described above converts a forest of labeled win-loss binary trees into the game graph.*

Proof. All second cards in the labeling of a win-loss tree, i.e., cards Bob plays in the game, are distinct due to the game being single-use. This is why step two gives us nodes with all possible cards in the game graph. Edges created in Step 3 describe pairs of cards that play against each other due to Lemma 4.5. □

We now motivate Step 1 of the algorithm using a unicard game as an example. Why does this trick of putting a above and to the left of b work? Let us say Bob somehow started with all the cards in the state $|abcd\ldots$. Suppose Alice has a "null" card that beats a and wins it over from Bob. Due to WL-putback, she puts a second, which means it will next play in the round corresponding to the right child of this null versus a round. From here on, the game operates normally from the state $a|bcd\ldots$. So we can treat the true root of the win-loss binary tree as the right child of a null round.

5 Counting R-round and k-passthrough Games

For a randomly initialized m-card game played with

(a) random putback, or
(b) WL-putback,

we can ask the probability the game ends up being

1. R-round, or
2. k-passthrough.

We will show that the probabilities are the same whether Alice and Bob each play with WL-putback or random putback for each of the questions of R-round or k-passthrough. If both players use random putback, the necessary condition is only that Bob never loses at any point during the game, whereas either player using WL-putback breaks a certain symmetry which requires the necessary condition that the game is single-use. The game being single-use means that whichever putback Bob chooses does not matter because the cards he puts back would only get played again in a second passthrough.

For this section, we need the following key lemma. We consider m-card states, where we assume that the total number of cards is n and $0 < m < n$. Recall that the number of m-card states is $n!$, and likewise for $(m-1)$-card and $(m+1)$-card states.

Lemma 5.1. *Consider the uniform probability distribution over the m-card states. After one round is played from each of these states, half have Alice winning the first round and half have Bob winning the first round. Taking the half of cases where Alice wins, the probability distribution of the resulting states after Alice uses random putback is the uniform distribution over all $(m+1)$-card states. Similarly, taking the half of cases where Bob wins, the probability distribution of the resulting states after Bob uses random putback is the uniform distribution over all $(m-1)$-card states.*

Proof. Each $(m+1)$-card state is preceded by exactly one m-card state, which is the state obtained by reversing the process of a round that Alice won. This is done by removing the bottom two cards from Alice and putting the larger at the top of her hand and the smaller at the top of Bob's hand. Therefore, each $(m+1)$-card state is attainable from the $n!/2$ m-card states with a uniform probability distribution over the states. Likewise, we get the result for $(m-1)$-card states. □

5.1 Counting R-round single-use states

Lemma 5.1 allows us to prove the following theorem.

Theorem 5.2. *The probability that a random putback game with randomly chosen initial state follows any particular win-loss sequence so long as neither player loses during the sequence, and even if the win-loss sequence does not end at the end of the game, is $\frac{1}{2^R}$ where the win-loss sequence consists of R rounds.*

Proof. By Lemma 5.1, we have that the probability distribution of the state after any number of rounds is the uniform distribution over states with the appropriate number of cards. Due to uniformity, the probability the card at the top of Alice's hand is greater than the card at the top of Bob's hand is $\frac{1}{2}$ at every point in the game. Thus, the chance Alice wins each round is $\frac{1}{2}$ independent of what happened prior, so long as both players have a nonzero number of cards. So the probability of following a win-loss sequence consisting of R rounds is $\frac{1}{2^R}$. □

Now, Theorem 5.2 allows us to prove the following corollary, where the condition on the number of cards n is just so that Bob does not lose during any R-round game. By our definition of an R-round game, Alice loses at the Rth round. Assuming she starts with m cards, then at most the first $\frac{R-m}{2}$ rounds can be Ws if Alice is to lose after the Rth round. So Bob's initial hand size $n-m$ must be at least $\frac{R-m}{2}+1$ if he is to survive until the Rth round, i.e., $n \geq \frac{R+m}{2}+1$.

Corollary 5.3. *The probability a random putback game with randomly chosen m-card initial state is R-round, where $n \geq \frac{R+m}{2}+1$, is*

$$\frac{C\left(\frac{R+m}{2}-1, \frac{R-m}{2}\right)}{2^R}.$$

Proof. The condition that $n \geq \frac{R+m}{2}+1$ guarantees that Bob survives until the Rth round for every R-round win-loss sequence that results in Alice losing. By Theorem 4.1, there are $C\left(\frac{R+m}{2}-1, \frac{R-m}{2}\right)$ such R-round win-loss sequences, and by Theorem 5.2, each has a $\frac{1}{2^R}$ probability of occurring, so the desired result holds. □

If Alice uses WL-putback, we do not have the same randomness after each round: states where she puts the lesser card before the greater card do not occur. For example, after Alice wins the first round of 3|124, WL-putback makes the next state necessarily 31|24 instead of 31|24 or 13|24 with equal probability. In particular, in a

non-single-use game without random putback, complications arise in terms of actual card values when computing the probabilities that a randomly initialized game is R-round or k-passthrough. However, under the assumption that the game is single-use, we still have the same probability of a game with a randomly chosen m-card initial state being R-round.

First, we prove two lemmas. Note that for a unicard game, the system of equations on the number of wins and losses is $\ell + w = R$ and $\ell - w = 1$. In particular, for the following lemma, we use $R = 2w + 1$.

Lemma 5.4. *The probability that a randomly chosen single-use unicard state $a_1|a_2a_3 \dots a_n$ played with WL-putback is R-round, where $n - 1 \geq R$ so that the game is single-use, is*

$$\frac{C_{\frac{R-1}{2}}}{2^R}.$$

Proof. We use strong induction on w.

The base case $w = 0$ gives $\frac{C_w}{2^{2w+1}} = \frac{1}{2}$, which is indeed the probability that the game ends in exactly $2 \cdot 0 + 1 = 1$ round: this happens when a_2 beats a_1 in the first round itself.

For the inductive hypothesis, assume that the probability a randomly chosen single-use game lasts $2x + 1$ rounds is $\frac{C_x}{2^{2x+1}}$ for every x from 0 to r. In order for the game to last exactly $2(w + 1) + 1 = (2w + 3)$ rounds, Alice must first win the first round. So we must have $a_1 > a_2$, and then the state becomes $a_1a_2|a_3 \dots a_n$ after putting the cards back in the WL order. The $\frac{1}{2}$ chance that $a_1 > a_2$ is used again later.

Temporarily assume that Alice had done random putback for this round before proceeding with WL-putback for the remainder of the game. Then the state is described as $b_1b_2|a_3 \dots a_n$, where b_1 and b_2 are a_1 and a_2 in some order. One round has taken place, so for the remainder of the game to last exactly $2w+2$ total rounds, the subblocks induced by the next rounds of b_1 and b_2 must end in $2x+1$ and $2y+1$ rounds exactly, where x and y are numbers such that $(2x + 1) + (2y + 1) = 2w + 2$ rounds.

The state $a_1a_2|a_3 \dots a_n$ does not follow the uniform distribution over all 2-card states, because we know $a_1 > a_2$, but the state $b_1b_2|a_3 \dots a_n$ does follow the uniform distribution over all 2-card states, because we can have either $b_1 > b_2$ or $b_1 < b_2$. Since the game is single-use, Theorem 3.1 implies that the subblocks induced by b_1 and b_2 are totally independent. The uniformity of the state $b_1b_2|a_3 \dots a_n$, together with the inductive hypothesis, guarantees that the probabilities for the subblocks induced by b_1 and b_2 to end in $2x + 1$ and $2y + 1$ rounds are independently $\frac{C_x}{2^{2x+1}}$ and $\frac{C_y}{2^{2y+1}}$, so the combined probability that the subblocks end in $2x + 1$ and $2y + 1$ rounds is

$$\frac{C_x}{2^{2x+1}} \frac{C_y}{2^{2y+1}} = \frac{C_x C_y}{2^{2w+2}}.$$

Let A_x denote the probability that a_1's subblock ends in $2x + 1$ rounds and a_2's subblock ends in $2w+2-(2x+1) = 2(w-x)+1$ rounds from the state $a_1a_2|a_3 \dots a_n$.

Then $\frac{A_x+A_{w-x}}{2}$ is the average of two cases of

1. a_1's subblock ending in $2x+1$ rounds and a_2's subblock ending in $2(w-x)+1$ rounds, and
2. a_1's subblock ending in $2(w-x)+1$ rounds and a_2's subblock ending in $2x+1$ rounds.

Importantly, this averaging of cases is exactly what we have done with using b_1 and b_2 as a_1 and a_2 in either order. Therefore, we have

$$\frac{A_x + A_{w-x}}{2} = \frac{C_x C_{w-x}}{2^{2r+2}}.$$

The total probability of the game ending in $2w+3$ rounds is then

$$\frac{1}{2}\sum_{x=0}^{x=w} A_x,$$

where the first $\frac{1}{2}$ comes from the probability that a_1 beat a_2 initially. We have

$$\begin{aligned}\frac{1}{2}\sum_{x=0}^{x=w} A_x &= \frac{1}{2}\left(\frac{A_0 + A_w}{2} + \frac{A_1 + A_{w-1}}{2} + \cdots + \frac{A_w + A_0}{2}\right)\\ &= \frac{1}{2}\sum_{x=0}^{x=w} \frac{C_x C_{w-x}}{2^{2w+2}}.\end{aligned}$$

Finishing with the Catalan recursive formula, we get

$$\frac{1}{2}\sum_{x=0}^{x=w} \frac{C_x C_{w-x}}{2^{2w+2}} = \frac{1}{2^{2w+3}} C_{w+1} = \frac{1}{2^{R+1}} C_{w+1},$$

and the induction is complete. □

To generalize to m-card initial states, we need the following lemma about the properties of the Catalan triangle.

Lemma 5.5. *We have*

$$C(m+k-1,k) = \sum_{\substack{x_1+x_2+\cdots+x_m=k\\ x_1,\ldots,x_m\geq 0}} \prod_{i=1}^{i=m} C_{x_i}.$$

Proof. By definition $C(p,q)$ is the number of paths from $(0,0)$ to (p,q) consisting of up (U) and right (R) moves that never go above the line $x-y=0$.

We describe a bijection between

- Type $\mathcal{A}$: paths from $(0,0)$ to $(m+k,k)$ consisting of $m+k$ copies of R and k copies of U that never go above the line $x-y=1$ except before the first move to the right.
- Type $\mathcal{B}$: sequences of m subpaths that go from $(0,0)$ to (x_i+1,x_i) consisting of x_i copies of U and x_i+1 copies of R that never go above the line $x-y=1$ except before the first move to the right, such that $x_1+\cdots+x_m=k$ and $x_1,\ldots,x_m\geq 0$.

Let A and B denote the numbers of Type $\mathcal{A}$ paths and Type $\mathcal{B}$ sequences.

Type $\mathcal{A}$ paths are equivalent to paths from $(0,0)$ to $(m+k-1,k)$ but shifted one unit to the right and with an extra R attached to the beginning. Therefore, we have

$$A=C(m+k-1,k).$$

The regular Catalan numbers C_k count paths from $(0,0)$ to (k,k) that never go above $x-y=0$, and Type $\mathcal{B}$ are exactly sequences of such paths, again with an extra R attached at the beginning of each path. Therefore, B is the sum over all possible values of $x_1,\ldots,x_m$ of the product of the amounts of possibilities for the first through mth path. Then

$$B=\sum_{\substack{x_1+x_2+\cdots+x_m=k\\ x_1,\ldots,x_m\geq 0}}\prod_{i=1}^{i=m}C_{x_i}.$$

Once we establish the desired bijection, we will have shown $A=B$, as desired.

First, we describe how to convert from a Type $\mathcal{A}$ path to a Type $\mathcal{B}$ sequence. The path must touch the lines $x-y=i$ a last time for every i from 1 to $m-1$, because the path goes from an initial point $(0,0)$ for which $x-y=0-0=0$ to a final point $(m+k,k)$ for which $x-y=m+k-k=m$. Say that these points are $(x_1,y_1),\ldots,(x_{m-1},y_{m-1})$. Splitting the path at each of these points, we get a sequence of m subpaths. Since (x_i,y_i) is on the line $x-y=i$, the point can be expressed as (y_i+i,y_i). So the subpath from (y_i+i,y_i) to $(y_{i+1}+i+1,y_{i+1})$ has $(y_{i+1}-y_i)$ copies of U and $(y_{i+1}-y_i+1)$ copies of R. Moreover, the subpath does not go above the line $x-y=i+1$ except before the first R of the subpath, so each of the subpaths along this sequence of m subpaths fits the description of Type B.

Now, we describe how to convert from a Type $\mathcal{B}$ sequence to a Type $\mathcal{A}$ path. We simply concatenate the m subpaths into a full path. Each subpath contributes x_i copies of U and x_i+1 copies of R, so in total there are $x_1+\cdots+x_m=k$ copies of U and $(x_1+1)+\cdots+(x_m+1)=k+m$ copies of R. This is exactly how many U and R moves are necessary for a Type $\mathcal{A}$ path from $(0,0)$ to $(m+k,k)$. Further, this concatenated path cannot go above the line $x-y=1$ except at the starting point because each of the m subpaths increases the value of $x-y$ by 1 from an initial value of $0-0=0$, and the first subpath only has $x-y$ value of 0 before the very first R.

Thus the bijection is established both ways, and $A=B$ as desired. □

We can now combine Lemma 5.4 and Lemma 5.5 into Theorem 5.6.

Theorem 5.6. *The probability that an m-card game played with WL-putback is R-round, assuming $n \geq m + R$ so that the game is single-use, is*

$$\frac{1}{2^R} C\left(\frac{R+m}{2} - 1, \frac{R-m}{2}\right).$$

Proof. In this proof, we use w_i, ℓ_i, and R_i for the numbers of wins, losses, and rounds in the ith block, instead of the numbers of such quantities within the first i rounds. We have $R_i = \ell_i + w_i$, and since blocks are unicard, we have $\ell_i = w_i + 1$. Adding the wins and losses of all blocks, we have $w = w_1 + \cdots + w_m$, $\ell = \ell_1 + \cdots + \ell_m$, and $R = R_1 + \cdots + R_m$.

By Lemma 5.4, there is a $\frac{C_{w_i}}{2^{R_i}}$ probability that a particular block consists of exactly w_i wins. Blocks are independent, so the probability of the m blocks having $w_1, w_2, \ldots, w_m$ wins is

$$\prod_{i=1}^{i=m} \frac{C_{w_i}}{2^{R_i}} = \frac{1}{2^{R_1+\cdots+R_m}} \prod_{i=1}^{i=m} C_{w_i} = \frac{1}{2^R} \prod_{i=1}^{i=m} C_{w_i}.$$

This must be added over all possible assignments of numbers of wins to the blocks, so we have a total probability of

$$\sum_{w_1+\cdots+w_m=w} \left(\frac{1}{2^R} \prod_{i=1}^{i=m} C_{w_i}\right) = \frac{1}{2^R} \sum_{w_1+\cdots+w_m=w} \prod_{i=1}^{i=m} C_{w_i}.$$

Applying Lemma 5.5, this simplifies as

$$\frac{1}{2^R} \sum_{w_1+\cdots+w_m=w} \prod_{i=1}^{i=m} C_{w_i} = \frac{1}{2^R} C(m + w - 1, w).$$

We have $\ell - w = m$ and $\ell + w = R$, so we have $w = \frac{R-m}{2}$. Thus,

$$\frac{1}{2^R} C(m + w - 1, w) = \frac{1}{2^R} C\left(\frac{R+m}{2} - 1, \frac{R-m}{2}\right),$$

as claimed. □

5.2 Counting k-passthrough single-use games

Previously, our main parameter was the number of rounds. Now, we look at the number of passthroughs.

As in Section 5.1, the formula in the main theorem here, Theorem 5.7, holds for random putback so long as Bob never loses at any point in any k-passthrough win-loss sequence, whereas for WL-putback the game must be single-use for the formula to hold.

Here we get the conditions for the number of cards n necessary for the random putback and for the WL-putback cases, assuming that Alice begins with m cards. A k-passthrough win-loss sequence can contain at most $m \cdot 2^{i-1}$ W's in the ith passthrough, and at most the first $k-1$ passthroughs can consist of W's, so for the random putback result, Bob must start with at least $1+(m+m\cdot 2+\cdots+m\cdot 2^{k-2}) = 1+m\cdot(2^{k-1}-1) = 2^{k-1}m-m+1$ cards, so $n \geq 2^{k-1}m+1$. A k-passthrough game can last at most $m+m\cdot 2+\cdots+m\cdot 2^{k-1} = m(2^k-1)$ rounds, so for the game to be single-use, Bob must have at least $m(2^k-1)$ cards, so $n \geq 2^k m$.

Theorem 5.7. *The probability that a game initialized with m-card initial state is k-passthrough single-use is P_k^m, where P_k is recursively defined by $P_1 = \frac{1}{2}$ and $P_{k+1} = \frac{1}{2}+\frac{1}{2}P_k^2$, both for when*

(a) the game is played with random putback as long as $n \geq 2^{k-1}m+1$ so that Bob never loses, and

(b) when the game is played with WL-putback as long as $n \geq 2^k m$ so that the game is single-use.

Proof. (a) We show that for a random putback game with the necessary condition, the probability that an m-card initial state leads to a k-passthrough game is P_k^m. This immediately yields the desired recursion on P_k: the probability that a unicard initial state is k-passthrough is the sum of (1) the $\frac{1}{2}$ probability that Alice immediately loses and (2) the $\frac{1}{2}P_k^2$ probability that Alice wins the first round and the resulting state, which is uniform over all 2-card states, is subsequently k-passthrough.

Let $S_{m,k}$ denote the set of win-loss sequences corresponding to k-passthrough games with m-card initial state, and let $P_{m,k}$ be the probability that an m-card initial state ends up being k-passthrough, i.e., the probability that a sequence from $S_{m,k}$ is followed. Each sequence in $S_{m,k}$ corresponds exactly to an element of $S_{1,k}\times\cdots\times S_{1,k}$, where the association is made by turning the sequence from $S_{m,k}$ into a list of m binary trees of height at most k and turning each tree into a sequence in $S_{1,k}$. Also, by Theorem 5.2, every sequence $s \in S_{m,k}$ has probability $\frac{1}{2^{\text{length}(S)}}$ of occurring, so we have

$$\begin{aligned} P_{m,k} &= \sum_{s\in S_{m,k}} \frac{1}{2^{\text{length}(s)}} \\ &= \sum_{s_1,\ldots,s_m\in S_{1,k}} \frac{1}{2^{\text{length}(s_1)+\cdots+\text{length}(s_m)}} \\ &= \prod_{i=1}^{m} \sum_{s_i\in S_{1,k}} \frac{1}{2^{s_i}} = P_k^m, \end{aligned}$$

as desired.

(b) Again, we first prove that if the probability is P_k that a unicard initial state ends up being k-passthrough, then the probability an m-card initial state ends up being k-passthrough is P_k^m. By single-useness, the m blocks induced by each of Alice's original cards stay independent, and by the uniformity of the original state, each block has a probability P_k of ending up being k-passthrough, so the total probability is just P_k^m.

Proving the recursive formula takes an extra step. We begin with the inductive hypothesis that the probability a WL-putback game is k-passthrough is P_k. Consider a unicard initial state $a_1|a_2a_3\ldots$. If $a_1 < a_2$ then the state is $(k+1)$-passthrough. Otherwise, Alice wins the first round, and suppose that for this one round she uses random putback instead of WL-putback, so that the state is $b_1b_2|a_3\ldots$ where b_1 and b_2 are a_1 and a_2 in some order. From here, the probability that the rest of the game is k-passthrough is P_k^2. By the symmetry obtained from the uniformity of the initial state, this probability applies for both possible ways Alice could have put back her cards, and in particular the WL-putback case. Therefore, in the $a_1 > a_2$ case, we have a probability P_k^2 that the initial game is $k+1$ passthrough. Combining the $a_2 > a_1$ and $a_1 > a_2$ cases as before, we obtain the same recursion, as claimed.

□

The first few values in the sequence P_k are $P_2 = \frac{5}{8}$ and $P_3 = \frac{89}{128}$.

Example 3. We show the possible permutations for a 2-passthrough unicard game played with WL-putback. Such an initial state has form $a|bcd$. Alice may lose in the first passthrough, in which case the only constraint is $a < b$. Otherwise, if Alice wins the first round, then Bob must win from the state $ab|cd$. In this case, the constraints are $a > b$, $c > a$, and $d > b$. Out of the 24 initial unicard states on 4 cards, 12 satisfy $a < b$, and additionally 2|143, 2|134, and 3|142 satisfy $a > b$, $c > a$, and $d > b$. Thus, there is a $\frac{15}{24} = \frac{5}{8}$ chance that a game played with WL-putback initialized with random unicard state $a|bcd$ results in Bob winning within two passthroughs of Alice's hand.

Example 4. Now, we compute the probability that a game randomly initialized from a state of the form $a|bc$ and played with random putback is 2-passthrough. The three initial states where $a < b$ result in Alice losing immediately on the first round with probability 1. The two initial states where Alice starts with her card a being 3 mean she never loses, so the game is never 2-passthrough. The last initial state to consider is 2|13, which leads to 12|3 or 21|3 with equal probability. The 12|3 case only ends up being 2-passthrough in the $\frac{1}{2}$ chance that Bob puts his cards back as in 2|31. Finally, from 21|3, Alice necessarily loses the next two rounds, which guarantees that the game is 2-passthrough. So the combined probability is $\frac{3}{6} + \frac{1}{6} \cdot (\frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2}) = \frac{5}{8}$, as claimed.

Corollary 5.8. *The probability a unicard state is k-passthrough tends to 1 as k tends to infinity.*

Proof. Multiplying both sides of the recurrence relation $P_{k+1} = \frac{1}{2} + \frac{1}{2}P_k^2$ by 2 and subtracting $2P_k$ from both sides yields $2P_{k+1} - 2P_k = 1 + P_k^2 - 2P_k$, or $2(P_{k+1} - P_k) = (1 - P_k)^2$. The only steady-state of this recurrence is $P = 1$, and the sequence starting at $P_1 = \frac{1}{2}$ is monotone increasing and bounded by 1. Therefore, the sequence converges to the steady-state $P_\infty = 1$. □

This is a nontrivial result. It was plausible that the chance Alice does not lose within Bob's first passthrough would approach some positive value if high-value initial cards for Alice "survived" long enough, but this is not the case.

6 Counting games that necessarily follow a win-loss sequence

In this section, we count the number of initial states which are guaranteed to follow a certain win-loss sequence when the game is played with WL-putback and when the game is played with random putback. We do so by constructing a poset on card values which encodes the greater-than/less-than relationships between them.

Any tree inherits a natural poset structure, where the root is the largest element. We call this poset a *tree poset*. A *linear extension* of a finite poset of n elements is an assignment of the integers 1 through n such that poset relations are satisfied. The number of linear extensions is the number of initial states that satisfy the constraints of a poset on the cards, which is our aim in this section. An element of a poset is said to *cover* a lesser element if there is no element between them.

For both the WL-putback and random putback cases, the poset contains a tree structure, so we need the following lemma.

Lemma 6.1 (Ruskey [12])**.** *The number of linear extensions of a tree poset with n elements is*

$$\frac{n!}{\prod_{i=1}^{n} h(v_i)},$$

where $h(v)$ denotes the number of elements less than or equal to v.

Note that this lemma is similar to the hook-length formula. A quick understanding of this formula arises from the fact that for each v, the probability it is the greatest in the subtree where v is the root is $\frac{1}{h(v)}$. Multiplying these probabilities for every vertex, we get the expression in the lemma.

Also, we only consider single-use games for both WL-putback and random putback in this section. Otherwise, cards could play against each other in more than one round. This would result in situations where both cards must be greater than the other, interfering with our goal of constructing a poset and counting the initial states which satisfy its relations.

6.1 Poset for WL-putback

We can take any game graph and make it a directed graph where each edge points from the card that won to the card that lost a round. In fact, since this is a directed

acyclic graph, it constitutes a poset on the cards. Posets can be graphically represented by Hasse diagrams, where greater elements are drawn above lesser elements. The conversion from a win-loss binary tree or forest to the game graph as described in Section 4 naturally yields the poset by placing leaves above their right parents and non-leaves below their right parent.

Example 5. Consider the game with WL-putback and the win-loss sequence $W/LW/LL$ with initial state $a|bcdef$. The game progresses as

$$\begin{aligned} & a|bcdef \\ \Longrightarrow\ & ab|cdef \\ \Longrightarrow\ & b|defca \\ \Longrightarrow\ & bd|efca \\ \Longrightarrow\ & d|fcaeb \\ \Longrightarrow\ & |caebfd. \end{aligned}$$

The directed game graph, i.e., the poset, is shown in Figure 4.

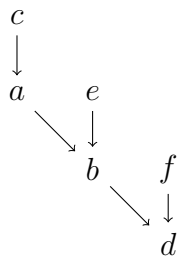


Figure 4: Game poset for WL-putback on $W/LW/LL$

Example 6. Considering the win-loss sequence $W/WW/LWWW/LLLLLL$ with the initial state $a|bcdefghijklmn$, we build the winner-to-loser game digraph. The result is shown in Figure 5.

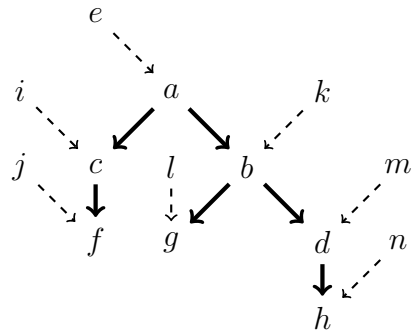


Figure 5: Game poset for WL-putback on $W/WW/LWWW/LLLLLL$

In this Hasse diagram, we have drawn thin dashed edges to connect a card from Bob that beats a card from Alice. Taken alone, the solid edges and the cards they

connect form a tree poset structure rooted at a. Nodes of this tree poset are the cards that have been in Alice's hand at any point in the game. In the Hasse diagram, each such node has one node above it corresponding to the card from Bob's hand that beat it. These cards from Bob's hand sit outside the tree.

For non-unicard games, the game digraph consists of m connected components, and for each connected component, the non-leaves form a tree poset.

We use the variable k with $n = 2k$, so that there are k rounds of L and $k-1$ rounds of W. Note that each card that ever appears in Alice's hand loses exactly once.

We now finish with a theorem to count linear extensions of the poset, which enumerates initial states that satisfy the win-loss sequence.

Theorem 6.2. *The number of $2k$-round single-use WL-putback unicard states $a_1|a_2 \dots a_{2k}$ that satisfy a particular win-loss sequence is*

$$\frac{(2k)!}{2^k \prod_{i=1}^{k} h(v_i)},$$

where v_i's are vertices in the tree component of the associated poset and $h(v_i)$ is the number of vertices less than or equal to v_i in the tree component of the poset.

Proof. There are $\frac{(2k)!}{k! \cdot 2^k}$ partitionings of $2k$ cards into k pairs, where the lesser element is in the tree component and the greater element sticks outside the tree. As described earlier, the number of linear extensions of just the tree component of the poset is

$$\frac{k!}{\prod_{i=1}^{k} h(v_i)}.$$

Multiplying these terms, we get

$$\frac{(2k)!}{k! \cdot 2^k} \cdot \frac{k!}{\prod_{i=1}^{k} h(v_i)} = \frac{(2k)!}{2^k \prod_{i=1}^{k} h(v_i)},$$

as claimed. □

We illustrate the theorem with an example.

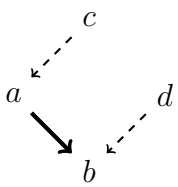


Figure 6: Game poset for WL-putback on W/LL

Example 7. Consider the game with initial state $a|bcd$ with win-loss sequence W/LL, played with WL-putback. We have four cards, so $k = 2$. The game poset is

shown in Figure 6. Within the tree component of the poset, consisting of a and b, we have $h(a) = 2$ and $h(b) = 1$. Plugging these values into formula in Theorem 6.2, we have

$$\frac{4!}{2^2 \cdot 2 \cdot 1} = \frac{24}{8} = 3.$$

These three initial states are 2|134, 2|143, and 3|142.

We can now also enumerate m-card single-use games with a given win-loss sequence. Following the win-loss sequence breaks the game up into m unicard blocks. We apply Theorem 6.2 to each of the m blocks and multiply the possibilities together.

6.2 Poset for random putback

We want to count unicard single-use states of the form $a_1|a_2 \dots a_n$ that *necessarily* follow a certain win-loss sequence under random putback. We first describe how to construct the poset associated with the win-loss sequence and then show how to count its linear extensions. We restrict ourselves to consider single-use games, so that when dealing with an m-card rather than unicard state, the claimed method of constructing the poset applies to each unicard block, after which we can simply multiply the possibilities for each block.

It is important to note the word "necessarily" in a state *necessarily* following a win-loss sequence. A given initial state can follow multiple possible win-loss sequences given different possibilities in random putback. For example, consider the initial state 3|14256. Alice wins the first round, and the state becomes either 31|4256 or 13|4256. In the first case, Alice loses the next two rounds, so the win-loss sequence is W/LL. In the second case, Alice loses and then wins, so that she has the cards 3 and 2 in some order. Both of these cards are less than both of 5 and 6, so Alice loses the next two rounds. This case follows the win-loss sequence $W/LW/LL$. Therefore, this initial state 3|14256 does not necessarily follow any win-loss sequence.

Example 8. Suppose an initial state $a|bcd$ is to necessarily follow win-loss sequence W/LL. Then, necessarily $a > b$ to get to the state $ab|cd$ or $ba|cd$, after which both a and b are less than c and d to necessarily have two Ls. The only initial states that satisfy these constraints are 2|143 and 2|134. The relationships are summarized in the poset in Figure 7.

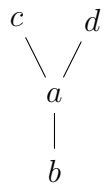


Figure 7: Poset for random putback on W/LL

Now, we describe the general method of obtaining the poset from a unicard state where Alice begins with card a.

1. Initialize the poset with a node for card a.
2. At each node of a win-loss binary tree, write down which of Bob's cards plays against Alice in the round.
3. Prune all leaves of the win-loss binary tree and put each corresponding card from the leaves as a node above a in the Hasse diagram.
4. Take the remainder of the tree, consisting of Bob's cards involved in W rounds, and place the entire structure under a in the Hasse diagram.

The algorithm is illustrated in Figure 8 using the example of the win-loss sequence $W/WW/LWWW/LLLLLL$ on the initial state $a|bcd\ldots mn$. Card a initializes the poset. Then, cards e, i, j, k, l, m, and n, which participated in L rounds, are placed above a. The remainder of the binary tree, consisting of b, c, d, f, g, and h, is placed under a.

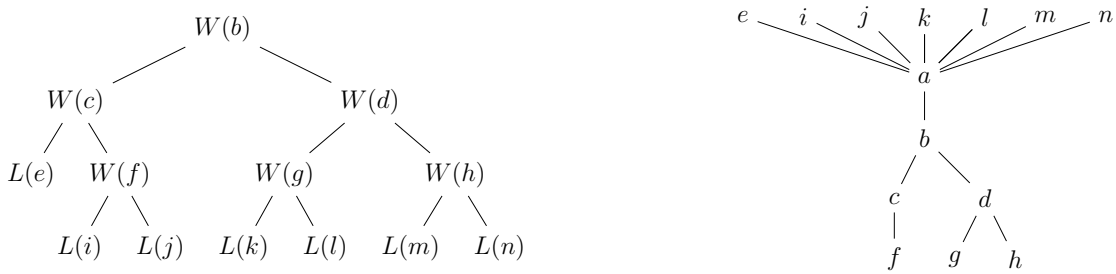


Figure 8: Win-loss binary tree and poset with random putback for necessarily following $W/WW/LWWW/LLLLLL$

Now, we explain why the algorithm for writing the Hasse diagram works. Take a W round where Alice's card x beats Bob's card y. Each such y then appears only in the subblock induced by that round. This subblock is exactly the subtree rooted at that round in the win-loss binary tree. And y can appear anywhere in the subblock until the end of the subblock due to random putback: if put back first, y begins a subblock corresponding to the left child subtree of the x/y round, and if put back second, y begins a subblock corresponding to the right child subtree of the x/y round. Therefore, every one of Bob's cards involved in a subsequent W in the subblock induced by x/y must be less than y for that subsequent W to necessarily occur. This continues recursively through all W's: each one of Bob's cards involved in a W must be less than each of Bob's cards involved in a W higher up in the tree. Moreover, card a must be greater than every card played by Bob in a W round, because a can be anywhere in Alice's hand at every point in the game. This is exactly what we do when taking the structure of W's in the binary tree and placing it under a in the Hasse diagram.

Again, card a can be anywhere in Alice's stack at every point in the game. Therefore, for L's to occur necessarily, cards Bob plays in L rounds must be greater

than a. This is what we do when pruning L's and putting the card Bob plays in those rounds above a in the Hasse diagram.

We now enumerate states that are guaranteed to follow a particular win-loss sequence under random putback by counting linear extensions of the poset.

Theorem 6.3. *The number of single-use random putback games with initial unicard state $a_1|a_2 \dots a_{2k}$ that necessarily satisfy a particular win-loss sequence is*

$$\frac{(k!)^2}{\prod_{i=1}^{k} h(v_i)},$$

where v_i's are vertices in the bottom tree component of the associated poset and $h(v_i)$ is the number of vertices less than or equal to v_i.

Proof. The Hasse diagram consists of two components: a tree below and including a, that happens to be a not-necessarily-full binary tree, and a tree above and including a, which has height 2. The top layer of the poset has as many elements as L's in the win-loss sequence, namely k, where $n = 2k$. These must be the values $k+1$ through n, because they are greater than every other element of the poset, so there are $k!$ assignments of values to cards in the top layer. For the tree consisting of the remaining k cards from card a and down, we use Lemma 6.1 for a factor of

$$\frac{k!}{\prod_{i=1}^{k} h(v_i)}$$

again. Combining these two factors, we get

$$k! \cdot \frac{k!}{\prod_{i=1}^{k} h(v_i)} = \frac{(k!)^2}{\prod_{i=1}^{k} h(v_i)},$$

as claimed. □

7 Conclusion

We defined the notions we needed, like passthroughs, WL-putback, and random putback. We introduced the basic objects of the game graph and the win-loss binary tree. For random putback we showed the basic lemma that the state of the game is always uniformly drawn over all states with the appropriate number of cards for each player.

From these, we computed the answers to the questions of: the chance that a game ends in R rounds, the chance that a game ends in k passthroughs, and the chance that a game follows a specific win-loss sequence. The uniformity property of random putback allowed us to use for the first two questions the weaker necessary condition that Bob must never lose. Otherwise, the necessary condition throughout is that the game is single-use to avoid the complicated interactions between cards as they come back up through Bob's hand.

Even though the game, as investigated in this paper, has no strategy, it still has interesting combinatorial structures.

Acknowledgements

We would like to thank the MIT PRIMES-USA program for the opportunity to conduct this research. We are grateful to the anonymous reviewers for their useful and thoughtful comments.

References

[1] B. Alexeev and J. Tsimerman, Note on a War-like Card Game, *Amer. Math. Monthly.* 119 (2012), 793–795.

[2] M. Ciucu, No-feedback card guessing for dovetail shuffles, *Ann. Appl. Probab.* 8 (1998), 1251–1269.

[3] D. Bayer and P. Diaconis, Trailing the dovetail shuffle to its lair, *Ann. Appl. Probab.* 2 (1992), 294–313.

[4] P. Diaconis, R. Graham, X. He and S. Spiro, Card guessing with partial feedback, *Combin. Probab. Computing* 31 (2022), 1–20.

[5] P. Diaconis, R. Graham and S. Spiro, Guessing about guessing: Practical strategies for card guessing with feedback, *arXiv preprint* arXiv:2012.04019 (2020).

[6] E. Lakshtanov and V. Roshchina, On finiteness in the card game of war, *Amer. Math. Monthly* 119 (2012), 318–323.

[7] P. Liu, On card guessing game with one time riffle shuffle and complete feedback, *Discrete Appl. Math.* 288 (2021), 270–278.

[8] P. Diaconis, Shuffling extra footage (3/3), filmed by B. Haran, *YouTube* (2015).

[9] E. Ben-Naim and P. Krapivsky, Parity and ruin in a stochastic game, *Eur. Phys. J. B* 25 (2002), 239–243.

[10] J. Haqq-Misra, Predictability in the game of war, *The Science Creative Quarterly* (2006).

[11] S. Reuveni, Catalan's trapezoids, *Probab. Engrg. Inform. Sci.* 28 (2014), 353–361.

[12] F. Ruskey, Generating linear extensions of posets by transpositions, *J. Combin. Theory Ser. B* 54 (1992), 77–101.

[13] M. Z. Spivey, Cycles in war, *Integers* 10 (2010), 747–764.

(Received 23 June 2023; revised 29 Nov 2024)