Overview of Ladder Lotteries

Patrick Di Salvo

March 2, 2020

1 Introduction

An Amidakuji/Ladder Lottery is a custom in Japan which allows for a pseudo-random assignment of children to prizes [6]. Usually done in Japanese schools, a teacher will draw N vertical lines, hereby known as lines, where N is the number of students in class. At the bottom of each line will be a unique prize. And at the top of each line will be the name of one of the students. The teacher will then draw 0 or more horizontal lines, hereby known as *bars*, connecting two adjacent lines. The more bars there are the more complicated (and fun) the Amidakuji is. No two endpoints of two bars can be touching. Each student then traces their line, and whenever they encounter an end point of a bar along their line, they must cross the bar and continue going down the adjacent line. The student continues tracing down the lines and crossing bars until they get to the end of the ladder lottery. The prize at the bottom of the ladder lottery is their prize [7]. See Figure 1 for an example of a ladder lottery.

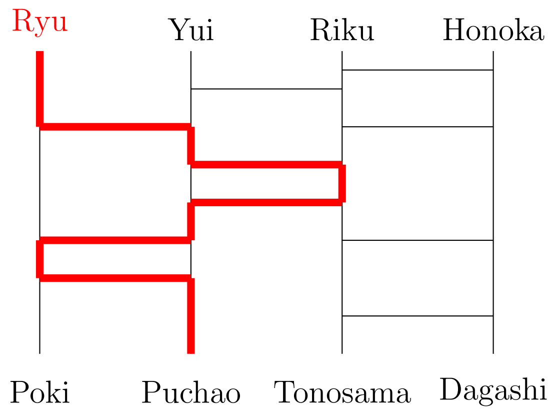


Figure 1: A ladder lottery where Ryu gets Puchao, Yui gets Dagashi, Riku gets Tonosama and Honoka gets Poki. You can see that Ryu's path is marked by red bars.

1.1 Optimal Ladder Lotteries

An interesting property about ladder lotteries is that they can be derived from a *permutation* which is a is a unique ordering of objects. For the purposes on this paper, the objects of a permutation will be integers ranging from $[1 \ldots N]$. Every permutation has a unique set of ladder lotteries associated with it. Moreover, each of the ladder lotteries in the set are a special case of a ladder lottery. These special cases of ladder lotteries are termed *optimal ladder lotteries* because they contain exactly one bar for each *inversion* in π. An *inversion* is a relation between two elements in π, A and B, such that if $A > B$ and A is to the left of B then A and B form an inversion. For example, given $\pi = (4, 3, 5, 1, 2)$, $Inv(\pi) = \{(4, 3), (4, 1), (4, 2), (3, 1), (3, 2), (5, 1), (5, 2)\}$. Thus, the set of optimal ladder lotteries associated with π, hereby known as $OptL\{\pi\}$, is the set containing all ladder lotteries with a number of bars equal to the number if inversions in π. See figure 1.1 for an example of an optimal ladder in $OptL\{\pi\}$ where $\pi = (4, 3, 2, 1)$. For each optimal ladder in $OptL\{\pi\}$, the N elements in π are listed at the top of a ladder and each element is given its own line. At the bottom of a ladder is the *sorted permutation*, hereby known as the *identity permutation* [7]. The identity permutation of size N is defined as follows - $I : (1, 2, 3, \ldots, N)$. Each ladder in $OptL\{\pi\}$ has the minimal number of horizontal bars to sort π into the identity permutation. Each bar in a ladder from $OptL\{\pi\}$ uninverts a single inversion in π exactly once. For the remainder of this paper, only optimal ladder lotteries will be discussed. Therefore when the term ladder lottery is used, assume optimal ladder lottery.

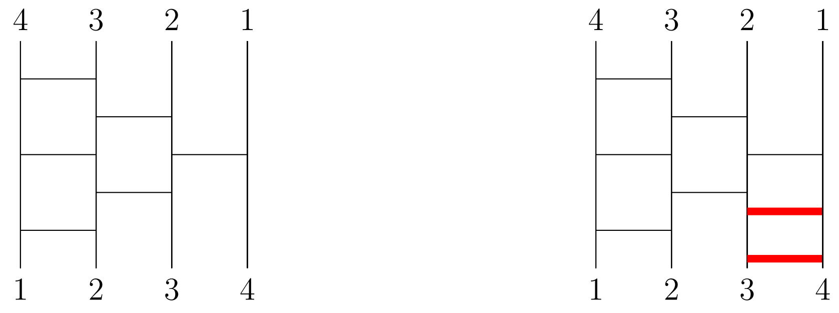
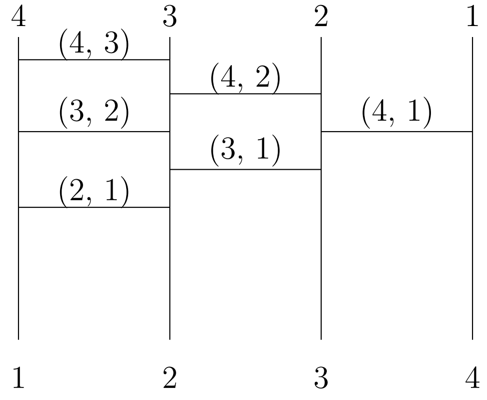


Figure 2: Two ladders for the permutation (4, 3, 2, 1). The left ladder is an optimal ladder and the right ladder is not. Therefore the left ladder belongs to $optL\{(4, 3, 2, 1)\}$. The bold bars in the right ladder are redundant, thus the right ladder is not optimal

1.2 Ladders and Adjacent Transpositions

A ladder lottery is a way of sorting a permutation, yet it can also be thought of as a decomposition of a permutation into *adjacent transpositions.* [5] An *adjacent transposition* is simply a swap of two adjacent elements in a permutation. For example, given the permutation (1, 3, 4, 2), an adjacent transposition could be done on the following pairs of elements: (1, 3), (3, 4) or (4, 2). Each would result in a unique permutation. Simply put, given any arbitrary starting permutation, π, keep swapping adjacent inversions until the identity permutation is reached. An optimal ladder lottery from $\pi's$ optimal ladder set is a minimal sequence of adjacent transpositions such that π is sorted into the identity permutation; each ladder in the set represents a sequence of adjacent transpositions for sorting π into the identity permutation. For example, given the permutation (4, 3, 2, 1) there exists eight ladders in this permutation's optimal ladder set. Two of these ladders are found in 1.2:



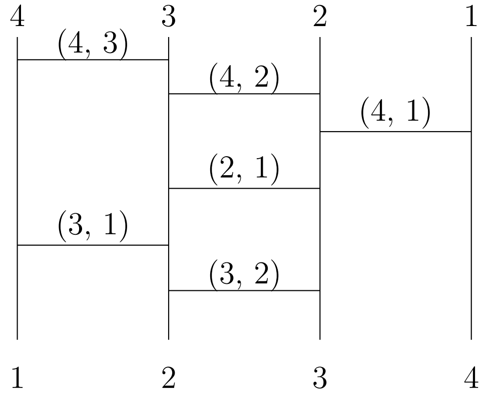


Figure 3: The left ladder is one of eight unique ladders from (4,3,2,1)'s optimal ladder set. The right ladder is another one of eight unique ladders form (4,3,2,1)'s optimal ladder set

From looking at the above ladders, going from top left to bottom right, the left ladder represents the sequence of adjacent transpositions (4,3), (4,2), (4,1),(3,2),(3,1),(2,1) whereas the right ladder represent the sequence of adjacent transpositions (4, 3),(4, 2),(4, 1),(2, 1),(3, 1),(3, 2). Notice how the length of the sequences are the same,because both lengths are equal to the minimal number of swaps to sort (4, 3, 2, 1) it is simply the order in which the adjacent transpositions occur in the sequence that makes the sequences different from each other.

Theorem 1: Let π be an arbitrary permutation with N unique elements, then there is a sequence of adjacent transpositions of length $0 \leq N(N-1))/2$ such that π is sorted into the identity permutation.

Proof. The rationale for the above theorem is as follows. The minimum number of inversions π can have is 0. Therefore π would already be sorted and there would be no adjacent transpositions. If π is in descending order, then the number of inversions in π is $N(N-1)/2$. Let element N be the maximum element in π, assuming π is descending. In order to put N in its correct position, it must undergo $N-1$ adjacent transpositions. For example, given permutation (5, 4, 3, 2, 1), element 5 must undergo four adjacent transpositions in order to be put in its correct spot, thus resulting in the permutation (4, 3, 2, 1, 5). The result is the first $N - 1$ elements are descending, and the last $(N - (N - 1))$ elements are sorted. Applying the same process to the left $N - 1$ elements, element 4 would be swapped 3 times, resulting in (3, 2, 1, 4, 5) leaving to the first $N - 2$ elements in descending order and the last $(N - (N - 2))$ elements sorted. In general one can say, that given the descending permutation, applying adjacent transpositions starting with the leftmost element and working to the right, the first $N - i$ elements will be descending and the right $(N - (N - i))$ elements will be sorted where i is the index in π. In order for the entire descending π to be sorted, each element, p, must undergo $p - 1$ adjacent transpositions. Thus, given a descending permutation of size N, the number of adjacent transpositions is the following summation.

$$\sum_{x=1}^{n-1} x = N(N-1)/2 \tag{1}$$

This summation represents the maximum length of a sequence of adjacent transpositions used to sort an arbitrary permutation π of size N. $\square$

Corollary 1.1: Since an adjacent transposition sequence can be represented as a ladder lottery, where each adjacent transposition is a bar in a ladder lottery, then one can conclude that given any π of size N, where $N > 0$, and given any ladder, L_k, in π's optimal ladder set, $0 \leq$ the number of bars in $L_k \leq N(N - 1)/2$.

1.3 Ladder Lotteries and their Relation to other Combinatorial Objects

Ladder Lotteries are closely related to a number of other combinatorial objects. Ladder Lotteries share a commonality with primitive sorting networks, pseudo-line arrangements and wiring diagrams. [8] [**?**] [7] [3] When an optimal ladder set is derived from a reverse permutation, $\pi = (N, N - 1, N - 2, ...1)$, the number of ladders in the set shares the same integer sequence as the number of primitive sorting networks, wiring diagrams and pseudo-line arrangement for the same permutation. There is currently no known closed form solution for this integer sequence. In their paper Counting *Primitive Sorting Networks by πDDs*, the authors have computed the value for this integer sequence when $N = 13$. Lastly, in 2009 Armstrong computed the value for $N = 15$ [2] which is currently the greatest value of N this sequence has been computed for. [1]

2 Problems Related to Ladder Lotteries

The focus of this paper will be to cover three open problems related to Ladder Lotteries. The three problems are the following

1. When enumerating all permutations of size N, is there a corresponding Gray Code for enumerating all root ladders for each permutation of size N?
2. Given permutation of size N, is there a polynomial solution to determine a ladder with a minimal height.
3. The third problem has already been mentioned, but will be more generalized, which is given a permutation of size N, is there a closed form solution for determining the number of ladders in its optimal ladder set.

2.1 Solved Problem: Ladder Lottery Realization

In their paper *Ladder Lottery Realization*, Yamanka, Horiyama, Uno and Wasa pose a problem they termed the *Ladder Lottery Realization Problem.* A6 The question this problem poses is the following, given a target permutation, π of size N, and a multi-set of bars, is it possible to construct a ladder that uses every bar in the multi-set such that the ladder can sort the ascending permutation of size N into the target permutation? The authors prove that the problem is NP-complete. An example of the problem will now be provided.

Let $N = 6$
Let the target permutation, π be (4, 1, 6, 3, 5, 2).
Let the ascending permutation be (1, 2, 3, 4, 5, 6)
Let the multi-set, S = $\{(1,3)^2,(1,4),(2,3),(2,4)^3,(2,5)^3,(2,6),(3,4),(3,6),(5,6)^3\}$.
The exponent represents the number of times the bar must occur in the ladder. Assume exponent 1 if no exponent is given.
Problem: Is there a ladder such that every bar in the multi-set occurs exactly the number of times as its exponent in the multi-set and the ladder sorts the identity permutation into the target permutation?
The answer is yes, the ladder in figure 2.1 is such a ladder.[4]

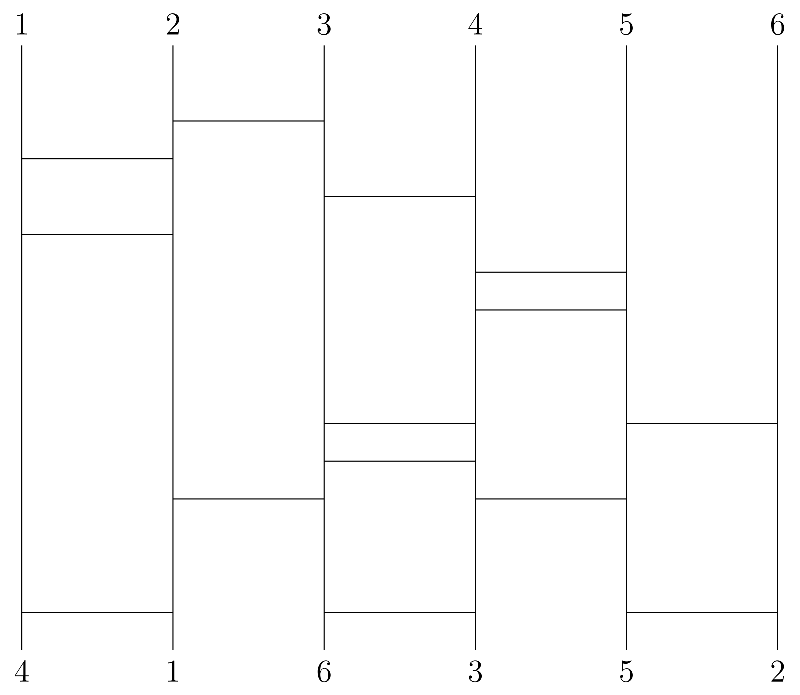


Figure 4: A ladder that is a solution to the above instance of the Ladder Lottery Realization Problem

2.2 Open Problems:

Some other open problems related to ladder lotteries are the following

1. Token Swapping Problem: Given a permutation and a set of allowed transpositions, where each transposition in the set can be done an infinite number of times, what is the minimum number of transpositions required to sort the permutation? [4]

2.3 Areas of Application

Outside of theoretical interest, ladder lotteries can be applied to the following areas. Firstly, there is potential to apply ladder lotteries to FPGAs (Field Programmable Gate Array). For example, assume an FPGA designer is looking to implements a sorting network. In order to save space, the designer could design a ladder lottery as sorting network such that the ladder had the minimal height; by minimal height, what is meant that the maximum number of heights can be placed at the same level in the ladder.

A second area of application is the following. Consider a permutation of N computers in a network. Each computer has a unique label between 1 and N. Next consider that a signal from each computer must travel through wires and cross wires such that its signal reaches the end of a wire and is in the position corresponding to its computer's label. In other words, once the signals travel through the wiring, they are sorted. If one thinks of the wires and cross-wires as a ladder lottery, then it becomes clear that there is a reliable way to sort the signals from the computer network.

3 Algorithms, Formulas and Properties

3.1 Introduction

The following chapter pertains to a variety of algortithms and formulas used for the preliminary research of this thesis. The algorithms and formulas are contingent upon certain properties of ladder lotteries which will be explained prior to the algorithmic analysis. The primary algorithm used for this research is a ramified version of Yamanaka's enumeration algorithm in his paper Efficient Enumeration of Optimal Ladder Lotteries and its Application. The algorithm in Yamanaka's paper was updated for this research due to certain inconsistencies in the original algorithm. The ramifications are original to this thesis. These ramifications will be explained throughout the chapter.

3.2 Explanation of Properties

This section pertains explaining properties of ladder lotteries that are required for the algorithms and formulas to work.

3.2.1 Decreasing subsequence of length three

Given a permutation, π, of length N where $N \geq 3$ a *decreasing subsequence of length 3* is a relation between three elements in the π, p, q, and r such that if $p > q > r$ and p is to the left of q which is to the left of r then p, q and r form a decreasing subsequence of length 3. The importance of decreasing subsequences of length three, hereby known as $DSS3$ is that $DSS3s$ directly translate to enumerating ladders in $optL\{\pi\}$. For an example of a permutation with $DSS3s$ see 3.2.1.

$$\pi = (6, 3, 2, 1, 5, 4)$$

Figure 5: $\pi's$ inversion tuple $((6,3),(6,2),(6,1),(6,5),(6,4),(5,4),(3,2),(3,1),(2,1))$ and $\pi's$ $DSS3s$ tuple is $((6,3,2),(6,3,1),(6,2,1),(6,5,4),(3,2,1))$. Note that the number of inversions is not the same as the number of DSS3s

3.2.2 Allowable Inversions

Given a permutation π of length N, an *allowable inversion* in π is an inversion in π that can be uninverted by performing an adjacent transposition on the elements of the allowable inversion. For an example of a permutation with an allowable inversion see 3.2.2

$$\pi = (4, 5, 2, 3, 1)$$

Figure 6: The allowable inversions in π are $((5, 2), (3, 1))$ and the unallowable inversions in π are $((4, 2)(4, 3), (4, 1), (5, 3)(5, 1), (3, 1), (2, 1))$

Theorem 2: Let π be an arbitrary permutation of size N. If π has zero $DSS3s$ then $|OptL\{\pi\}| = 1$ and if $|OptL\{\pi\}| = 1$ then π has zero $DSS3s$.

Proof. Think of a ladder lottery as a unique sequence of uninverting allowable inversions in parallel. By uninverting the allowable inversions of π, the result is a derivative permutation. Let $C(\pi)$ be the child of some other π. One can say that by performing parallel adjacent transpositions of allowable inversions on pi one gets $C(pi)$. Assuming the original permutation is the root permutation,π_{root}, every derivative permutation is a descendant of the original permutation. The final descendant of the original permutation is the sorted permutation because the sorted permutation has zero inversions. Refer to the sorted permutation as π_{sort}. Thus, $\pi_{sort} = C(C(C(C \ldots C(\pi_{root}))))$.

Corollary 2.1: If π has an inversion, then π has an allowable inversion.

Proof. Let π be a permutation of size $N >= 3$. Let π have one or more inversions.

Case 1: If the elements of inv_i are adjacent to each other then the proof is complete.

Let the greater element be in position π_t and the lesser element be in position π_{t+r} where $r >= 2$. There must be some elements between πt and π_{t+r} in π.

Case 2: If all the elements between π_t and π_{t+r} are greater than π_t then the element at $\pi_{(t+r)-1}$ must form an allowable inversion with the element at π_{t+r}.

Case 3: If all the elements between π_t and π_{t+r} are less than π_r then the element at π_{t+1} forms an allowable inversion with π_t.

Case 4: Next consider that some of the elements between π_t and π_{t+r} are $>$ than π_t and the others are $< \pi_{t+r}$. Let r be any element $>$ than π_t and be between π_t, π_{t+r}. Let q be any element $<$ than π_{t+r} and be between π_t and π_{t+r}. If every q is to the left of every r then there is some q at position π_{t+1} thus forming an allowable inversion with π_t and there is some r at position $\pi_{(t+r)-1}$ forming an allowable inversion with pi_{t+r}. On the other hand, if some q is to the right of some r then r and q will form an allowable inversion.

Case 5: Lastly consider that all elements between π_t and π_{t+r} are $< pi_t$ and $> pi_{t+r}$ then the element at pi_{t+1} will form an allowable inversion with π_t and the element at $\pi_{(t+r)-1}$ forms an allowable inversion with pi_{t+r}.

□

Two or more allowable inversions can be uninverted in parallel if uninverting the elements of one allowable inversion does not reposition the elements of another allowable in π. For example, let $\pi_{root} = (1, 3, 6, 2, 4, 5)$. The inversions in π_{root} are $((6, 2), (6, 4), (6, 5), (3, 2))$. At stage one of performing adjacent transpositions on π_{root}, the only allowable inversion is $(6, 2)$. Once this allowable inversion is uninverted, the resulting $C(\pi_{root}) = (1, 3, 2, 6, 4, 5)$ The only allowable inversions in $C(pi_{root})$ are $((6, 4), (3, 2))$. The question is whether these allowable inversions can be uninverted in parallel. To help answer this question, let the elements of an allowable inversion be termed the set $elem\{AllowedInv\}$. Next consider two allowable inversions A and B. There are two rules for determining whether A and B can be uninverted in parallel.The first rule is that A and B can be uninverted in parrallel if uninverting A does not remove a $DSS3$ formed by $elem\{B\}$ in π_i and conversely uninverting B does remove a $DSS3$ formed by $elem\{A\}$ in π_i. That is to say, if A and B can be uninverted in parallel then $elem\{A\} \cap elem\{B\} = \emptyset$ and $elem\{A\}$ does not form a $DSS3$ with the elements in $elem\{B\}$ nor does $elem\{B\}$ form a $DSS3$ with the elements in $elem\{A\}$. The second rule is that uninverting A does not remove a $DSS3$ in pi_i formed by $elem\{B\}$ and uninverting B does not remove a $DSS3$ in pi_i formed by $elem\{A\}$. In the above example, the allowable inversions in $C(\pi_{root})$ can be uninverted in parallel because uninverting $(6, 4)$ does not reposition the elements of $(3, 2)$ and conversely uninverting $(3, 2)$ does not reposition the elements of $(6, 4)$. Nor do the elements in $(6, 4)$ form a $DSS3$ with $(3, 2)$ in $C(\pi_{root})$. Nor do the elements in $(3, 2)$ form a $DSS3$ with $(6, 4)$ in $C(\pi_{root})$. Refer to figure 3.2.2 to see a sequence of uninverting allowable inversions for a permutation with only one ladder in its set and a sequence of uninverting allowable inversions for a permutation with two ladders in its set.

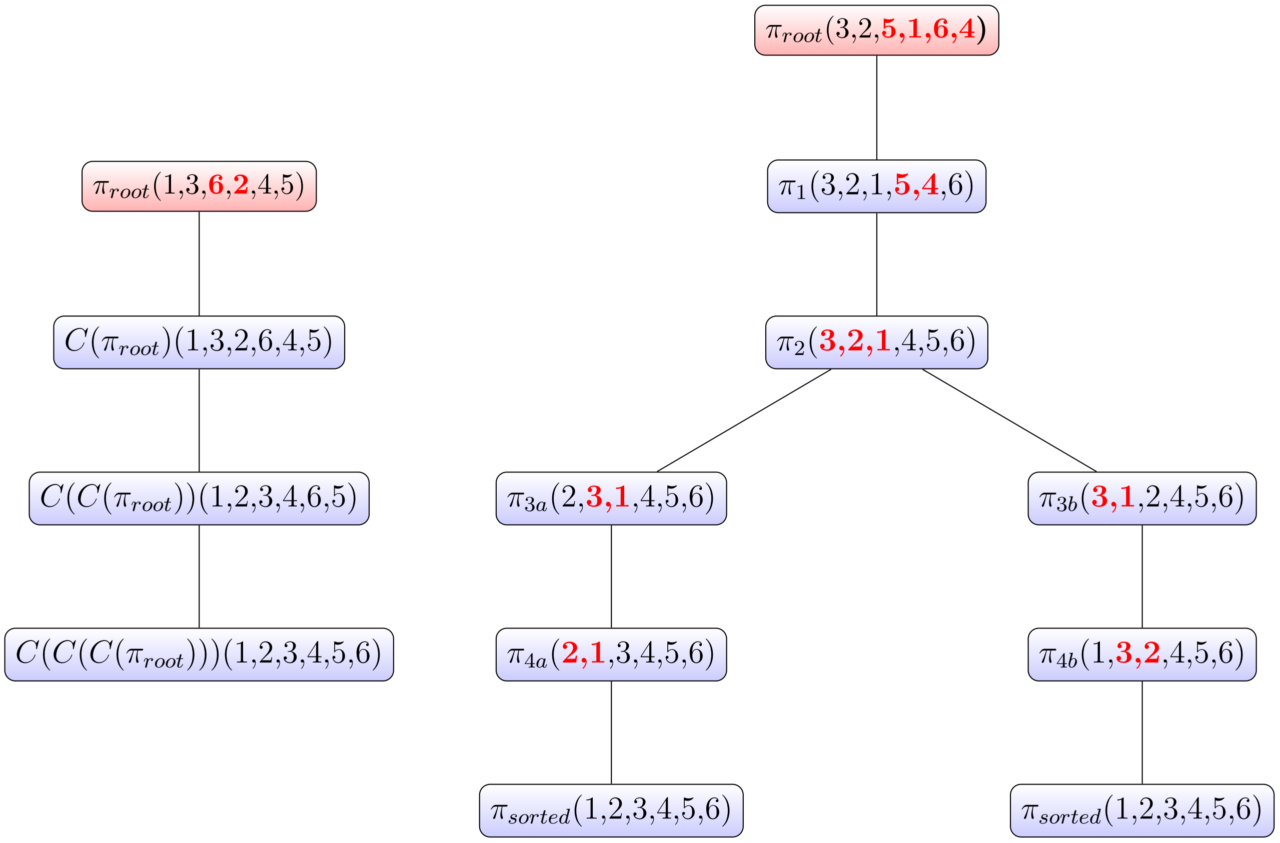

π_{root}(1,3,6,2,4,5)
$C(\pi_{root})$(1,3,2,6,4,5)
$C(C(\pi_{root}))$(1,2,3,4,6,5)
$C(C(C(\pi_{root})))$(1,2,3,4,5,6)
π_{root}(3,2,5,1,6,4)
π_1(3,2,1,5,4,6)
π_2(3,2,1,4,5,6)
π_{3a}(2,3,1,4,5,6)
π_{3b}(3,1,2,4,5,6)
π_{4a}(2,1,3,4,5,6)
π_{4b}(1,3,2,4,5,6)
π_{sorted}(1,2,3,4,5,6)
π_{sorted}(1,2,3,4,5,6)

Figure 7: The above figure illustrates sequences of uninverting allowable inversions. Note that for $\pi_{root} = (3, 2, 5, 1, 6, 4)$ the tree has one branch until the $elem\{3, 2, 1\}$ are adjacent to each other, then it splits off into two branches. These separate branches correspond to two the only two ladders in $OptL\{3, 2, 5, 1, 5, 4\}$. Whereas in $\pi_{root} = (1, 3, 6, 2, 4, 5)$ there is only one branch. This is because $\pi_{root} = (1, 3, 6, 2, 4, 5)$ has no $DSS3s$.

In fact, given the starting permutation, $\pi_{root} = (1, 3, 6, 2, 4, 5)$, for pi_{root} and all of its descendants, if there is more than one allowable inversion in pi_i, then the allowable inversions will be uninverted in parallel. This is because the π_{root} has no $DSS3s$. The term *false descendant* refers to a possible descendant of π that has two or more allowable inversions such that $elemA$

When a permutation has a $DSS3$ then the permutation, or one of its decedents, have two or more allowable inversions such that the lesser element of allowable inversion $A =$ the greater element of allowable inversion B or the lesser element of allowable inversion $B =$ the greater element of allowable inversion A. In other words, $elem\{A\} \cap elem\{B\} \neq \emptyset$ for π_{root} or some descendent of π_{root}. When this is the case a choice is presented as to which allowable inversion to uninvert first seeing as uninverting one allowable inversion will reposition the shared element between the two allowable inversions. Each of these choinces correspond to another ladder in $OptL\{\pi\}$.

Theorem 3: Uninverting one of the allowable inversions formed by an adjacent $DSS3$ will remove the other allowable inversion formed by the same $DSS3$ without uninverting it.

Proof. Consider an arbitrary $DSS3$ composed of p, q, and r in which $p > q > r$. There are two allowable inversions, $((p, q), (q, r))$, thus two cases for uninverting an allowable inversion.

Case 6: Uninverting the allowable inversion (p, q) puts q to the left of p and r. The resulting triad is q, p, r. Since $q > r$ and q is not adjacent to r then the allowable inversion (q, r) has been removed without being uninverted.

Case 7: Uninverting the allowable inversion (q, r) puts q tp the right of r and to the left of p. The resulting triad is p, r, q. Since $p > q$ and p is not adjacent to q then the allowable inversion (p, q) has been removed without being uninverted.

□

Since p, q and r are arbitrary, the proof holds for any p, q or r insofar as they form an adjacent $DSS3$. The significance of this proof is that it demonstrates two important properties of ladder lotteries. First it demonstrates under what conditions $OptL\{\pi\}$ has only one ladder in its set. Secondly, it demonstrates that if $OptL\{\pi\}$ has more than one ladder in its set, π_{root} must have at least one $DSS3$. Furthermore, assuming there is a $DSS3$ in pi_{root}, it is known when the next ladder in $OptL\{\pi\}$ will emerge; that point being when π_{root} or some decendant of π_{root} has a $DSS3$ that forms two allwable inversions that are adjacent to each other and share a common element. It is at this point another ladder emerges because two allowable inversions exist, but they cannot be uninverted in parallel. Hence, two distinct uninverting sequences exist for sorting π_{root} into π_{sorted}, each of which is represented by a different ladder lottery in $OptL\{\pi\}$. □

Algorithm 1: Algorithm for Right Rotating a Degenerative Triadic Subsequence

```
Input: $l \leftarrow$ 2D matrix representing a ladder. $val \leftarrow$ integer value
  representing the bottom bar of the degenerative triadic subsequence:

$upperNeighbor \leftarrow getUpperNeighbor(l, curVal)$
$rightNeighbor \leftarrow getRightNeighbor(l, curVal)$
$bottomRightNeighbor \leftarrow getBottomRightNeighbor(l, curVal)$
$l \leftarrow shiftChildren(l, rightNeighbor, 2)$

$swap(upperNeighbor, bottomRightNeighbor)$

$swap(curVal, rightNeighbor)$
```

Algorithm 2: Algorithm for getting the upper bar of a degeneartive triadic subsequence

```
Input: $l \leftarrow$ 2D matrix representing a ladder, $val \leftarrow$ integer value
  representing the bottom bar of the degenerative triadic subsequence:

$row \leftarrow val's$ row in $l$
$col \leftarrow val's$ col in $l$

for $i \leftarrow row - 1 \ldots 0$ do
    if $l[i][col] > 0$ then
        return l[i][col]
    end
end
```

Algorithm 3: Algorithm for shifting the children of a bottom bar of a degenerative triadic bar formation downwards

Input: l ←2d matrix representing a ladder. val ←integer representing the current bar, either the initial bar of the degenerative triadic bar pattern, or one of its descendants. $offset$ ← integer representing the offset to shift the bars downwards in the ladder

```
$leftChild \leftarrow getLeftChild(l, val)$
$rightChild \leftarrow getRightChild(l, val)$

if (leftChild ≤ 0) and (rightChild ≤ 0) then
    $col \leftarrow val's$ column index in ladder
    $row \leftarrow lval's$ row index in ladder
    $l[row + offset][col] \leftarrow val$
    $l[row][col] \leftarrow 0$
    return
end
else if (leftChild > 0) and (rightChild ≤ 0) then
    shiftChildren(l, leftChild, offset)
end
else if (rightChild > 0) and (leftChild ≤ 0) then
    shiftChildren(l, rightChild, offset)
end
else
    shiftChildrendDown(l, leftChild, offset)
    shiftChildrenDown(l, rightChild, offset)
end
$col \leftarrow val's$ column index in ladder
$row \leftarrow lval's$ row index in ladder
$l[row + offset][col] \leftarrow val$
$l[row][col] \leftarrow 0$
return
```

Algorithm 4: Algorithm for Left Rotating a degenerative triadic bar pattern. The inverse of the Algorithm for Right Rotating a degenerative triadic bar pattern

```
Input: $l \leftarrow$ 2D matrix representing a ladder. $val \leftarrow$ integer value
  representing the bottom bar of the degenerative triadic subsequence:

$row \leftarrow getRow(l, val)$
  $col \leftarrow getCol(l, col)$
  $swap(l[row][col], l[row+1][col-1])$
  $swap(l[row+2][col], l[row-1][col-1])$
  $l \leftarrow shiftChildren(l, rightChild, -2)$
```

References

[1] https://oeis.org/search?q=2%2C+8%2C+62language=engli. Accessed: 2020-01-21.

[2] D. Armstrong. The sorting order on a coxeter group. *Journal of Combinatorial Theory*, 116:1285–1305, 2009.

[3] S. N. K. Yamanaka. Listing all plane graphs. *Graph Algorithms Appl*, 13:5–18, 2009.

[4] T. U. K. W. Katsuhisa Yamanaka, Takashi Horiyama. Ladder-lottery realization. *CCCG*, aug 2018.

[5] M. Kiyomi. Reverse search; enumeration algorithms. *Encyclopedia of Algorithms*, pages 1840–1842, 2016.

[6] K. Yamanaka and S. Nakano. Enumeration, counting, and random generation of ladder lotteries. *IEICE Transactions Information and Systems*, 100-D(3):444– 451, 2017.

[7] M. U. N. Yamanaka, Nakano. Efficient enumeration of all ladder lotteries and its application. *Theoretical Computer Science*, 411:1714–1722, 2010.

[8] S. Z.Li. Efficient generation of plane triangulations without repetitions. *The 28th International Colloquium of Automata, Languages and Programming*, LNCS(2076):433–443, 2001.