JUGGLING POSET FOR INITAL AND TERMINAL STATE $\langle 1^k \rangle$

VIVIENE DO
ADVISED BY: LAURA COLMENAREJO

Abstract. We look at juggling from a combinatorial viewpoint. Juggling has been formalized in multiple ways. We consider two types of juggling: basic where at most one ball is thrown at a time, and multiplex where at most c balls are thrown at a time. For basic juggling, we look at Siteswaps, which are special sequences of numbers that represent a basic juggling pattern, and some known results for Siteswaps. Then we focus on an alternative representation of juggling sequences for multiplex juggling. Finally, we look at a generalized Poset for juggling sequences and consider the specific case where the initial and terminal state are $< 1^k >$.

Contents

1 Introduction to Juggling 1
1.1 Juggling States . . . 1
1.2 Siteswaps . . . 2

2 Siteswap Results 2
2.1 Minimal Juggling Patterns . . . 4
2.2 Eulerian Numbers . . . 6

3 Juggling States Experiment and Results 9
3.1 JS(a,b,n,m) . . . 10
3.2 The Poset . . . 11

4 Refrences 16

Date: July 22, 2021. Department of Mathematics and Statistics, University of Massachusetts, Amherst.

1. Introduction to Juggling

We have all probably seen someone juggle, whether it be the common form of toss juggling, or one of many other variations of juggling. Toss juggling is exactly as it sounds, the act of tossing and catching objects. When we say juggling we will be referring to toss juggling, more specifically throwing and catching balls.

If we were to go up to a juggler and ask them to juggle three balls, the pattern that we imagined may look quite different from what the juggler actually juggles. This is because there are many different ways one can juggle three balls, in fact as we'll soon see, there are infinite ways to juggle three balls. Though, physical limitations may rule out some of the hypothetical possibilities. Now let's say we wanted to describe the pattern we had in mind to the juggler without performing it for them. If it was simple enough, we could try and describe it and the juggler would probably understand what we were talking about, but what about not so simple cases. What if we wanted formal way to describe juggling patterns. Good news is that this has already been done. In the first part of this report we will go over methods and results already known for juggling.

In order to formalize juggling we will start with some basic assumptions. We consider that time is discrete, that the throws go forward in time, and that juggling is periodic (or cyclic). We also treat the balls as indistinguishable. In this study, we discuss two types of juggling: the *basic juggling*, in which the juggler can only catch or throw at most one ball at a time, and the *multiplex juggling*, in which the juggler can catch or throw at most c balls at the time. We refer to this constraint as the *hand capacity constraint.*

Our next step is to determine what information we should keep and how to denote the juggles. We can do so in two different ways: keeping track of the number of balls at each height or describing the pattern drawn by the balls. We describe each of them in the rest of this section.

Juggling States

One way to notate juggling patterns is through juggling states. Let us say we were juggling and time froze, and we then take this opportunity to record what information we have. In this case, we keep track of the heights of all the balls.

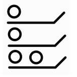
Definition 1.1 (Juggling State)**.** *We denote our juggling state by $s = \langle s_1, \dots s_n, \rangle$, where s_i represents the number of balls at height i, thus $\Sigma s_i = b$ is the total number balls. Note that as we can't have a negative number of balls at any given height $s_i \geq 0$.*

In basic juggling s_i is either 0 or 1, whereas in multiplex juggling, s_1 is less then or equal to c, the hand capacity constraint. Since a juggling state is a single snapshot in time, we can chain a sequence of juggling states to represent the entire juggling pattern.

Definition 1.2 (Sequence of Juggling States)**.** *We denote our sequence by $\boldsymbol{S} =< \boldsymbol{s}_1 \dots \boldsymbol{s}_n >$, where $\boldsymbol{s}_i$ represents the individual juggling state at time i. For $\boldsymbol{S}$ to be a valid juggling sequence, consecutive juggling states, $\boldsymbol{s}_{i-1} =< s_1, \dots, s_h >$ and $\boldsymbol{s}_i$ must satisfy:*

$$\boldsymbol{s}_i =< s_2 + b_1, s_3 + b_2, \dots, s_h + b_{h-1}, b_h, b_{h'} >,$$

where $b_j \geq 0$ and $\sum_{j=1}^{h'} b_j = s_1$. This requirement comes from gravity pulling down all the balls in the air down one from $\boldsymbol{s}_{i-1}$, and all the balls in the hand, or height 1, being thrown and distributed to various heights in the air.


(A) $s =< 2, 1, 1 >$

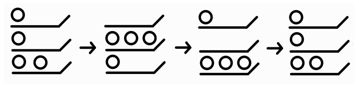

(B) $\mathbf{S} = (< 2, 1, 1 >, < 1, 3 >, < 3, 0, 1 >, < 2, 1, 1 >)$

FIGURE 1. Juggling State examples.

Siteswaps

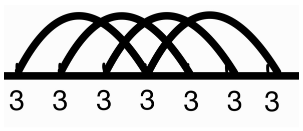
Rather then recording the state of all the balls at different times, we can record all our throws at once in something called *siteswaps*. This only works if we only catch one ball at a time, thus siteswaps can only be used for basic juggling.

Definition 1.3 (Siteswap)**.** *We denote our siteswap by $t = t_1 \dots t_n$, where $t_i \geq 0$ and $1+t_1, 2+t_2, \dots, n+t_n$ are all distinct mod n, in other words:*

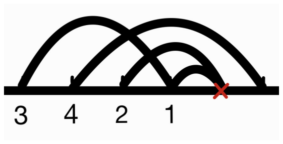
$$i + t_i \neq j + t_j (\operatorname{mod} n) \quad \forall i \neq j$$

In the siteswap t_i signifies a throw thrown on beat i that will land t_i beats in the future. Note that t is periodic, thus after the t_n throw we start over at t_1. We need one more thing for our siteswap $t_1 \dots t_n$, since in basic juggling we can't catch more then one ball at a time. So if we created a random string of numbers and considered it as a siteswap, it's quite possible that two or more balls will land at the same time, we call this a collision.

The condition that $t_i \geq 0$ comes from the fact that we can only throw forward in time, and the second restriction comes from avoiding *collisions*, that is, when two balls collapse in the same spot.



(A) Juggling diagram for the siteswap 3.



(B) Juggling diagram with collision.

FIGURE 2. Siteswap juggling diagrams.

2. Siteswap Results

The definitions and results presented in this section follow the YouTube videos created by Steve Butler in his channel (see [But15]).

Recall that a siteswap is valid if there are no collisions, which happens when one or more balls are caught at the same time. collisions only occur when $i + t_i = j + t_j (\operatorname{mod} n)$ for some i, j. This is because $i + t_i (\operatorname{mod} n)$ gives us the beat that the t_i^{th} throw will land. Thus no collisions happen if and only if:

$$i + t_i \neq j + t_j (\operatorname{mod} n) \quad \forall i \neq j$$

in other words $1 + t_1, 2 + t_2, \dots, n + t_n$ are all distinct mod n.

Remark 2.1. *If $t_1 \dots t_n$ a valid siteswap, we cannot guarantee that a reordering $t_{\sigma(1)} \dots t_{\sigma(n)}$ is valid for $\sigma \in S_n$. For example* 123 *is a siteswap, but* 321 *is not.*

The next result gives us a necessary condition for a sequence to be a siteswap.

Theorem 2.2. *If $t_1 \dots t_n$ is a valid siteswap, then $(t_1 + \dots + t_n)/n \in \mathbb{N}$*

Proof. Assume $t_1 \dots t_n$ is a valid siteswap. By definition of a siteswap the values $1 + t_1, 2 + t_2, \dots, n + t_n$ are all distinct (mod n). That is, we have n distinct numbers, all from 0 to $n - 1$. Thus $1 + t_1 (\operatorname{mod} n), 2 + t_(\operatorname{mod} n), \dots, n + t_n (\operatorname{mod} n)$ is some rearrangement of $0, 1, \dots, n - 1$.

We now have that

$$\sum_{i=1}^{n} [i + t_i (\operatorname{mod} n)] = \sum_{i=1}^{n} [i (\operatorname{mod} n)] \to \sum_{i=1}^{n} (i + t_i) \equiv \sum_{i=1}^{n} i (\operatorname{mod} n).$$

Therefore, we get that

$$\sum_{i=1}^{n}(i+t_i) - \sum_{i=1}^{n} i \equiv 0(\operatorname{mod} n) \to \sum_{i=1}^{n} t_i \equiv 0(\operatorname{mod} n).$$

Thus there exists some $x \in \mathbb{N}$ such that $t_1 + \ldots + t_n = xn$. Note that we can say x is in $\mathbb{N}$ rather then $\mathbb{Z}$ since t_i and n are all nonnegative. Therefore $(t_1 + \ldots + t_n)/n = x \in \mathbb{N}$. □

Remark 2.3. *321 is a counterexample to the other direction, i.e. if $(t_1+\ldots+t_n)/n \in \mathbb{N}$, we cannot guarantee that $t_1 \ldots t_n$ is a valid siteswap. We have that $(3+2+1)/3 = 6/3 = 2 \in \mathbb{N}$, however 321 is not a valid sequence since $1+t_1 = 2+t_2 = 3+t_3 = 3$, thus giving us a collision.*

Theorem 2.4. *If $t_1 \ldots t_n$ is a valid siteswap, then we will need $b = (t_1 \ldots t_n)/n$ balls to juggle the sequence.*

We have two proofs, the first is a combinatorical proof where we have a counting argument in two different ways that then tells us how many balls are used. The second will give us an algorithm to create a new siteswap, which we then use to find the number of balls in the original sequence.

Proof. Let $t_1 \ldots t_n$ be a valid siteswap and b be the number of balls. Recall that t_i denotes a throw on beat i that lasts t_i beats.

We will count the cumulative *ball airtime* over one period, where ball airtime refers to the number of beats a ball is in the air for. Since a period is n beats and balls are thrown as soon as their caught, each ball will contribute n to our count. Thus, the cumulative ball airtime over one period is bn. Next, we compute this amount by looking at the throws.

To sum it up we look at each throw t_i in our period, this will toss a ball up in the air for t_i beats. If $i + t_i \leq n$ then we will catch the ball before the period ends, so the t_i throw contributes t_i to our sum. Now if $i + t_i > n$, so the throw goes into the next period, we consider the previous t_i throw(s) that will be in the air for our current period. So the t_i^{th} throw contributes exactly t_i to our sum regardless of its value. Thus we get that the total ball airtime is $t_1 + \ldots + t_n$.

Since we also know that the total ball airtime is bn, we get that $b = (t_1 \ldots t_n)/n$. □

Let's see the other proof.

Proof. First lets consider the case where all the throws are constant. i.e. $t_1 \ldots t_n = c \ldots c$. Now we want to find how many balls $c \ldots c$ uses.

Let's start juggling it, we toss the first ball so it will land c beats in the future, if $c > 1$ then on the second beat the first ball will not have landed yet so we toss a second ball. If $c > 2$ then on the third beat neither the first nor second ball will have landed yet so we toss a third ball, and so on. We will have tossed c balls by the time our first toss lands, and after the first ball lands, every beat we will catch and throw a ball. Thus c balls are needed to juggle $c \ldots c$. Also note that $(c + \ldots c)/n = (t_1 + \ldots + t_n)/n = c$.

Now, assume $t_1 \ldots t_n$ is a valid siteswap that is non constant. Then there exists some maximum throw t_j, where $t_j \geq t_i \; \forall i$, followed by a non maximum throw. If this weren't true, then a maximum throw would have to be followed by another maximum throw, thus the siteswap would have to be constant. note that t_1 is the throw after t_n. Thus we get that there exists j such that $t_j > t_{j+1(\text{mod} n)}$, the mod n is for the case where $j = n$ to account for $j+1$ being greater then n, and $t_j \geq t_i \; \forall i \leq n$. Note that the difference between t_j and t_{j+1} is at least 2; otherwise, there is a collision and the siteswap is not valid.

Our next goal is to construct a new a new siteswap from the current one, such that the new siteswap retains the same period and amount of balls used.

Let $s = s_1 \ldots s_n$ be a siteswap defined by setting $s_i = t_i$ for $i \neq j \neq j+1(\text{mod} n)$, $s_j = t_{j+1(\text{mod} n)} + 1$, and $s_{j+1(\text{mod} n)} = t_j - 1$, where j is the index corresponding to the maximum throw followed by a non maximum throw.

Note that the sequence of values $i + s_i$ is the same than the sequence of values of $i + t_i$, with the values $j + t_j$ and $j + 1 + t_{j+1}$ switched. Since the $i + t_i$'s are all distinct $\mod n$, we also have that the values $i + s_i$

are all distinct $\mod n$. Thus, $s_1 \dots s_n$ is a valid siteswap and $(t_1 + \dots + t_n)/n = (s_1 + \dots + s_n)/n$. This implies that this new siteswap $s_1 \dots s_n$ has the same period and uses the same amount of balls than $t_1 \dots t_n$.

If our result $s_1 \dots s_n$ is non constant, then we apply the same process to create a new siteswap. Eventually if we keep using this process to create new siteswaps we will eventually get some constant sequence. This is due to the fact that each time we apply our method we either reduce the maximum value or reduce the number of maximum throws if there was more then one, thus it has to stop eventually.

In the end we get some siteswap $b \dots b$ and we know that $(b + \dots b)/n = b = (t_1 + \dots + t_n)/n$. We also have that $b \dots b$ uses b balls, therefore $t_1 \dots t_n$ also requires $b = (t_1 + \dots + t_n)/n$ balls. You also get that all the sequences that you obtain in this process use b balls. □

Another question we can ask is the following: *How many possible siteswaps are there?* We start noticing that without restriction to our siteswaps, we easily have an infinite number of valid siteswaps. We will quickly go through an example of why this is true.

Lemma 2.5. *The sequence* $123 \dots n$ *is always a valid juggling sequence when* n *is odd.*

Proof. Let n be odd. Recall that for $123 \dots n$ to be valid, then $i + t_i (\mod n)$ or in this case $2i (\mod n)$ must be distinct. If we look at the first half of the terms (rounded down), $1, 2, \dots, (n-1)/2$, we see that $2i (\mod n)$ gives us $2, 4, \dots, n-1$, which are all distinct even numbers. Now looking at the remaining terms, $(n+1)/2, (n+3)/2, \dots, n-1, n$, applying $2i$ gives us $n+1, n+3, \dots, 2n-2, 2n$, then applying mod n we get $1, 3, \dots n-1, 0$, which are all distinct odd numbers with the addition of 0. Thus the values of $2i (\mod n)$ are distinct, proving the result. □

Since we have infinite odd numbers we have infinite juggling patterns. So without restriction there are infinite juggling sequences. Now, let us see what happens if we only look at those patterns of length n.

Lemma 2.6. *There are infinite siteswaps of length* n.

Proof. We get this from being able to construct new valid juggling sequences from existing ones in the following ways. Let $t_1 \dots t_n$ be a valid siteswap with b balls, we get the following siteswaps:

- $t_1 - 1, t_2 - 1, \dots, t_n - 1$ by subtracting 1 from all our throws we get a new siteswap with $b-1$ balls that's valid as long as our throws remain non negative. Instead of subtracting 1 from everything we could also add or subtract some integer c from everything as long as we end with no negative terms.
- $t_1 \dots, t_i + n, \dots t_n$ we can add the period to one of our throws to get a new siteswap with $b+1$ balls.
- $t_1 \dots, t_i - n, \dots t_n$ we can subtract the period from one of our throws to get a new siteswap with $b-1$ balls that's valid as long as the throw remains non negative. Instead of adding or subtracting n, we could also add or subtract a multiple of n as long as we have no negative terms.

□

Minimal Juggling Patterns

In order to finally get a finite set of siteswaps we need one more restriction.

Definition 2.7. *We say that a juggling pattern* $t_1 \dots t_n$ *is a* minimal (MJP) *if* $t_i \in 0, \dots, n-1$.

Our next goal is to count the number of MJP's of length n. To do so, we use non-attacking rook placements.

Definition 2.8. *Given a* $n \times n$ *board, a* k-rook placement *is an arrangement of* k *rooks on our board such that no two rooks are in the same row or column, i.e. they are non-attacking. We denote by* $r_k(B)$ *the number of* k*-rook placements in* B.

Note that we mostly deal with n-rook placements, where we have exactly one rook in every row and column.

Theorem 2.9.

(1) *There are $n!$ MJP's of period n.*
(2) *There is a bijection between MJP's of length n and $n-$rook placements on a $n \times n$ board.*
(3) *The number of balls needed for a given pattern is exactly the number of rooks below the diagonal.*

Proof. Let us start with part (2), and then show that there are $n!$ possible rook placements, which then proves part (1).

(2) To define a bijection from rook placements to minimal juggling patterns, consider a rook placement. There will be one rook per row and column, we then label the rows 1 through n starting at the top, and likewise the columns starting from the left side. We denote our rook placement by a string of numbers $r_1 \dots r_n$, where r_i represents a rook in the i^{th} row and the r_i^{th} column. Note that since there is exactly one rook per row and column $r_1 \dots r_n$ are all distinct. Now, the mapping ϕ from rook placements to MJP's is defined by

$$\phi(r_1, \dots, r_n) = t_1 \dots t_n, \quad \text{where } t_i = r_i - i (\operatorname{mod} n).$$

We claim that $t_1 \dots t_n$ is a valid minimal juggling pattern. Recall that for a juggling pattern to be valid there must be no collisions, in other words $i + t_i \neq j + t_j \quad \forall i \neq j$. Inherent to how we constructed t_i we get that $t_i < n$ and $t_i + i(\text{mod}n) = r_i$, and since r_i is unique $i + t_i(\text{mod}n)$ is unique. Thus $t_1 \dots t_n$ is a valid minimal juggling pattern.

Our mapping is well-defined since the rook placement $r_1, \dots, r_n$ maps to exactly one MJP. We now show that this mapping is bijective. First to show that ϕ is injective, let $R = r_1 \dots r_n$ and $S = s_1 \dots s_n$ be two distinct rook placements. Thus since $R \neq S$ there exists some i such that $r_i \neq s_i$. Let $\phi(R) = t_1 \dots t_n$ and $\phi(S) = u_1 \dots u_n$. Will now show that $\phi(R) \neq \phi(S)$, now since $r_i \neq s_i \to r_i - i \neq s_i - i$. The only way for $r_i - i(\text{mod}n) = s_i - i(\text{mod}n)$ to be true is if $r_i \equiv s_i(\text{mod}n)$, but this would mean that r_i and s_i are at least n apart, which cannot happen since they have to be between 1 to n. Thus $r_i - i(\text{mod}n) \neq s_i - i(\text{mod}n) \to t_i \neq u_i \to \phi(R) \neq \phi(S)$, which gives us that ϕ is injective.

All that is left is to show that ϕ is surjective. Consider a MJP $t_1 \dots t_n$, the preimage under ϕ would be $r_1, \dots, r_n \quad$ where $r_i = t_i + i(\text{mod}n)$ when $t_i + i(\text{mod}n) \neq 0$ otherwise $r_i = n$. We then have our rook placement $r_1 \dots r_n$ where the image under our mapping gives us $t_1 \dots t_n$, so ϕ is surjective.

Therefore ϕ is a bijection, which tells us that the $r_k(B)$ is the same as the number of MJP's of length n.

(1) In order to find the number of MJP's of length n we can just count the number of valid n-rook placements on a $n \times n$ board.

To count the rook placements we can just construct every possible configuration. We'll go through and place one rook per row. There are n possible options for the first rook, $n-1$ for the second, $n-2$ for the third, and so on. Thus there are $n!$ rook placements, which then tells us that there are $n!$ MJP's of length n. Alternatively we can think of our rook placements as permutations, since in our rook placement $r_1 \dots r_n$, each r_i is distinct. Thus the rook placement is a reordering of $1, \dots, n$, or in other words a permutation. Likewise given a permutation $\sigma = s_1 \dots s_n \in S_n$, we associate this with the rook placement with a rook in row i and column s_i. Thus we get another bijection from rook placements to the symmetric group, which has size $n!$ giving us that there are $n!$ rook placements, and $n!$ MJP's.

(3) Recall that for the MJP $t_1 \dots t_n$, there is a corresponding rook placement $r_1, \dots, r_n$ where r_i represents a rook in row i and column r_i. To get t_i from r_i, we use the formula $t_i = r_i - i(\operatorname{mod} n)$. Instead of using mod n, we could rewrite it as $t_i = r_i - i + (n \text{ if } i > r_i)$. Note that if $i > r_i$ then the rook is in a space where the row is greater then the column, in other words the rook is below the diagonal. So we have $t_i = r_i - i + (n \text{ if rook is below the diagonal})$.

From part (2), we have that the number of balls used in $t_1 \dots t_n$, is $b = (t_1 + \dots + t_n)/n$. So now we have $b = (\sum_{i=1}^{n} r_i - i + (n \text{ if rook is below diagonal}))/n = (\sum_{i=1}^{n} r_i - \sum_{i=1}^{n} i + n(\#$ rooks below diagonal))$/n$. Since $r_1, \dots, r_n$ is a reordering of $0, \dots n-1$, the two sums cancel and we are left with $b = \frac{n}{n}$(# rooks below diagonal). Therefore the number of balls need for the pattern is exactly the number of rooks below the diagonal.

□

Now, what happens if we add more restrictions to the MJP's? We use rook placements again and this would correspond to changing the shape of the board. For example, let us say we wanted all our throws to be odd, we can then cross out every square on our board with an even number, and we are left with something looking like a checkerboard.

Another example is the following. Consider an n-triangle board, a right triangle made of squares where the base and side are $n-1$. The number of non-attacking rook placements on an n-triangle board is exactly the Stirling numbers of second kind, which we denote by $S(n,k)$. The Stirling numbers of second kind also count the number of ways to partition the set $\{1 \dots n\} =: [n]$ into k non-empty sets.

Eulerian Numbers

The Eulerian numbers $\left\langle \begin{matrix} n \\ k \end{matrix} \right\rangle$ can be defined as the number of ways to place n (non-attacking) rooks on a $n \times n$ board with k rooks below the diagonal. By Theorem 2.9, part (3), the Eulerian numbers also count the number of MJP's of length n with k balls.

The Eulerian numbers also count the number of permutations σ in S_n with k ascents ($\sigma_i < \sigma_{i+1}$, in two line notation), or k descents ($\sigma_i > \sigma i + 1$), or k drops ($i > \sigma_i$). Here we mean k of each type, not mixing them. For example $132 \in S_3$ has 1 ascent, 1 descent, and 1 drop.

Note that number of juggling patterns with k rooks below the diagonal naturally corresponds to permutations with k drops. Recall that we associated the permutation $\sigma = \sigma_1 \dots \sigma_n$ with a rook placement with rooks in row i and column σ_i, a drop then corresponds to when the row is greater then the column, which is exactly what it means to be below the diagonal.

What if we ask for the number of patterns with period n and b balls? We already know the minimal juggling patterns of period n and b balls, $\left\langle \begin{matrix} n \\ k \end{matrix} \right\rangle$. What about not only MJP's, but all juggling patterns, so we remove the restriction that $t_i < n$. We could use rook placements again if we allow having more than one rook in the cell. In this case, each subsequent rook would mean adding n to that throw.

Lemma 2.10. *The number of juggling patterns with period n with less then b balls is:*

$$\sum_{k=0}^{b-1} \left\langle \begin{matrix} n \\ k \end{matrix} \right\rangle \binom{n+b-1-k}{n}$$

Proof. Since every juggling pattern is associated with a unique minimal juggling pattern, we can start with an MJP and build the juggling patterns associated with it, that use less then b balls.

Recall that if $t_1 \dots t_n$ is a juggling pattern, then so is $t_1 \dots, t_i + n, \dots t_n$ and uses 1 more ball then the original pattern. Thus we can construct all the juggling patterns with less then b balls with the associated MJP by distributing extra rooks, or $+n$'s, among t_i. so for a minimal juggling pattern with k balls, we can distribute $b-1-k$, rooks/$+n$'s among our $n+1$ options, our n throws and the option to not add to any of them, to end with a pattern that has less then b balls.

We have that the number of ways to distribute α identical objects into β distict groups is

$$\binom{\beta + \alpha - 1}{\beta - 1}$$

Thus the ways we can distribute $b-1-k$, rooks/$+n$'s among $n+1$ groups is $\binom{n+b-1-k}{n}$.

So to count the number of juggling patterns of period n with less then b balls associated with a MJP of k balls, we have that there are $\left\langle {n \atop k} \right\rangle$ MJP's of length n with k balls, and from above, each MJP gives us $\binom{n+b-1-k}{n}$ more juggling patterns with less then b balls. So we have $\left\langle {n \atop k} \right\rangle \binom{n+b-1-k}{n}$. Thus to count the total number of juggling patterns with less then b balls, we can sum up our previous term to get $\sum_{k=0}^{b-1} \left\langle {n \atop k} \right\rangle \binom{n+b-1-k}{n}$ □

The following result presents some properties of the Eulerian numbers.

Theorem 2.11.

(1) $\left\langle {n \atop k} \right\rangle = \left\langle {n \atop n-k-1} \right\rangle$ [Symmetry].

(2) $\left\langle {n \atop k} \right\rangle = (k+1)\left\langle {n-1 \atop k} \right\rangle + (n-k)\left\langle {n-1 \atop k-1} \right\rangle$.

(3) $\sum_k \left\langle {n \atop k} \right\rangle \binom{x+k}{n} = x^n$ [Worpitzky's identity].

Proof. (1) We define a bijection from rook placements with k rooks below the diagonal to rook placements with $n-k-1$ rooks below the diagonal.

Consider some rook placement with k rooks below the diagonal, then there are still $n-k$ rooks on the rest of the board. Now if we think about the rooks not below the diagonal and not in the last column we have $n-k-1$ rooks, since there is only one rook per column. We can denote our rook placement by $r_1 \ldots r_n$, where r_i represents a rook in column i and row r_i. Since there are k rooks below the diagonal there are k instances where $i < r_i$. Likewise we have $n-k$ instances where $i \geq r_i$

We can then move the last column to the front so we get a new rook placement $s_1 \ldots s_n$, where $s_1 = r_n, s_i = r_{i-1}$. Now there are $k-n-1$ instances where $i > s_i$, this comes from the fact that we had $n-k$ instances where $i \geq r_i$ but now all our rooks have been shifted to the right, thus changing the $\geq$ to just $>$, and then shifting the last column to the front gives us the -1.

Now we can flip the board across the diagonal to get a new rook placement $u_1 \ldots u_n$ where $u_i = s_i$, but now u_i represents a rook in row i and column u_i. Since we had $n-k-1$ instances where $i > s_i$, we also have $n-k-1$ instances where $i > u_i$ which gives us that there are $n-k-1$ rooks below the diagonal.

For any rook placement with k rooks below the diagonal, we can repeat this process to get a unique rook placement with $n-k-1$ rooks below the diagonal. Likewise for any rook placement with $n-k-1$ rooks below the diagonal we can reverse this process to get a unique rook placement with k rooks below the diagonal. Therefore we get that:

$$\left\langle {n \atop k} \right\rangle = \left\langle {n \atop n-k-1} \right\rangle$$

(2) Given a rook placement, we can construct a new rook placement by modifying the position of two rooks. We can select two rooks, one in row i and column j and another in row $\tilde{i}$ and column $\tilde{j}$. Since there is only one rook per row and column we have that $i \neq \tilde{i}$ and $j \neq \tilde{j}$. We can now choose to switch the rows of our two rooks. So we will move the first rook from (i,j), (i,j) represents the position in row i and column j, to $(\tilde{i}, j)$ and we would move the second rook from $(\tilde{i}, \tilde{j})$ to $(i, \tilde{j})$. Note that by how we moved the rooks there are still only on rook per row and column.

We will now use this method of moving rooks to construct every rook placement on a nxn board with k rooks below the diagonal, in other words $\left\langle {n \atop k} \right\rangle$.

Consider a $(n-1)x(n-1)$ board with $n-1$ non-attacking rooks, we can then add a row and column and rook in the bottom corner. To now construct all our rook placements we can either swap the bottom right rook with another rook below the diagonal, on or above the diagonal, or make no swap.

If we make no swap, then in order to have k rooks below the diagonal, then we would need k rooks below the diagonal on the $(n-1)x(n-1)$ board as to which there are $\left\langle {n-1 \atop k} \right\rangle$ configurations.

If we make a swap with a rook below the diagonal, that rook remains below the diagonal, and thus the number of rooks remain unchanged, there are $\left\langle {n-1 \atop k} \right\rangle$ placements with k rooks below the diagonal and we can choose any of the k rooks to make the swap with, thus this gives us a total of $k\left\langle {n-1 \atop k} \right\rangle$ more configurations.

If we make a swap with a rook above/on the diagonal, that rook will go below the diagonal, then we would need $k-1$ rooks below the diagonal to start, so $\left\langle {n-1 \atop k-1} \right\rangle$, and we can make the swap with any of the rooks above/on the diagonal, to which there are $n-1-(k-1)=n-k$ rooks. Thus giving us a total of $(n-k)\left\langle {n-1 \atop k-1} \right\rangle$ more configurations.

Therefore this is all the ways we can have k rooks below the diagonal on a $n\times n$ board, so we get

$$\left\langle {n \atop k} \right\rangle = \left\langle {n-1 \atop k} \right\rangle + k\left\langle {n-1 \atop k} \right\rangle + (n-k)\left\langle {n-1 \atop k-1} \right\rangle = (k+1)\left\langle {n-1 \atop k} \right\rangle + (n-k)\left\langle {n-1 \atop k-1} \right\rangle.$$

(3) Lastly we will show Worpitzky's identity through induction

$$\sum_k \left\langle {n \atop k} \right\rangle \binom{x+k}{n} = x^n$$

To begin we will start with showing the identity is true for the base case, $n=0$.

$$\sum_{k=0}^{0} \left\langle {0 \atop k} \right\rangle \binom{x+k}{0} = \left\langle {0 \atop k} \right\rangle \binom{x+k}{0} = 1(1) = 1 = x^0$$

Now assume the identity is true for $n \geq 0$, we will show that it is true for $n+1$.

$$\sum_{k=0}^{n+1} \left\langle {n+1 \atop k} \right\rangle \binom{x+k}{n+1} = \sum_{k=0}^{n+1}\left((k+1)\binom{x+k}{n+1}\left\langle {n \atop k} \right\rangle\right) + \sum_{k=0}^{n+1}\left((n-k)\binom{x+k}{n+1}\left\langle {n \atop k-1} \right\rangle\right)$$

If we look at the last term of our first sum, we compute $\left\langle {n \atop n+1} \right\rangle = 0$, so we can remove it. Now Looking at the first term of the second term, we compute $\left\langle {n \atop -1} \right\rangle = 0$, so we can remove it, and reindex the terms to get,

$$\sum_{k=0}^{n}\left((k+1)\binom{x+k}{n+1}\left\langle {n \atop k} \right\rangle\right) + \sum_{k=0}^{n}\left((n-k)\binom{x+k+1}{n+1}\left\langle {n \atop k} \right\rangle\right) =$$
$$\sum_{k=0}^{n}\left((k+1)\binom{x+k}{n+1} + (n-k)\binom{x+k+1}{n+1}\right)\left\langle {n \atop k} \right\rangle$$

To finish the proof, we will use the following identity,

$$(k+1)\binom{x+k}{n+1} + (n-k)\binom{x+k+1}{n+1} = x\binom{x+k}{n}$$

To verify this is true, one can just expand the terms and simplify. Now plugging this back in to our previous equation we get,

$$\sum_{k=0}^{n+1} \left\langle {n+1 \atop k} \right\rangle \binom{x+k}{n+1} = \sum_{k=0}^{n} x \binom{x+k}{n} \left\langle {n \atop k} \right\rangle = x \sum_{k=0}^{n} \binom{x+k}{n} \left\langle {n \atop k} \right\rangle = x(x^n) = x^{n+1}$$

Thus proving the inductive step.

□

Now we are ready to present our next result.

Theorem 2.12. *The number of juggling patterns of period n with less then b balls is b^n.*

Proof. By Lemma 2.10, we have that the number of juggling patterns of period n with less then b balls is

$$\sum_{k=0}^{b-1} \left\langle {n \atop k} \right\rangle \binom{n+b-1-k}{n}.$$

We want to simplify this expression. By the symmetry of the Eulerian numbers, Theorem 2.11, we substitute $\left\langle {n \atop k} \right\rangle$ with $\left\langle {n \atop n-k-1} \right\rangle$. To make things simpler we now substitute $n-k-1$ with t, to end up with $\sum_t \left\langle {n \atop t} \right\rangle \binom{t+b}{n}$. Then using Worpitzky's identity we get:

$$\sum_t \left\langle {n \atop t} \right\rangle \binom{t+b}{n} = b^n.$$

□

Corollary 2.13. *The number of juggling patterns of period n with exactly b balls is $(b+1)^n - b^n$.*

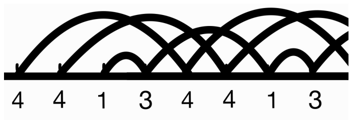
Proof. This is a direct result from Theorem 2.12, the number of juggling patterns of period n with less then b balls is b^n. Thus, there are $(b+1)^n$ juggling patterns with b balls or less, and b^n patterns with less then b balls. To get the ones with exactly b balls, we subtract b^n from $(b+1)^n$. □

3. Juggling States Experiment and Results

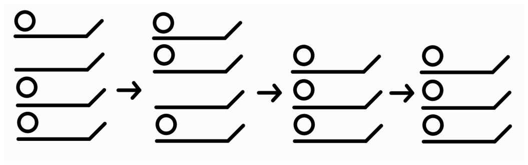
In this section we look into a poset defined for juggling states in general, and some properties for juggling states.

Recall that juggling states are an alternative way to notating a juggling pattern. The difference being, rather then recording just the throws, we record the position of each ball throughout the pattern. A question we can ask is, can we convert a siteswap into juggling states? We present how to do it with an example.

Example 3.1. *The juggling state notation of 4413 is $<1,1,0,1>, <1,0,1,1>, <1,1,1>, <1,1,1>$. The first state $<1,1,0,1>$ corresponds to the position of the balls after the first "4" throw, thus we have a ball at height 4, $<1,1,0,\underline{1}>$. For the next state all the ball above height 1 fall one position, and since we have another "4" throw, the first ball goes to height 4, $<1,0,1,\underline{1}>$. Again all the balls fall one position and since we have a "1" throw, the first ball goes to height 1, $<\underline{1},1,1>$. Likewise the balls all fall one position and since we have a "3" throw, the first ball goes to height 3, $<1,1,\underline{1}>$. Then similarly to how siteswaps are periodic, the juggling states would repeat from the beginning. Note that unlike siteswaps, juggling states do not have to be periodic. If the initial and terminal states are the same, then we could juggle it indefinitely, otherwise we just have a finite pattern.*



(A) Juggling diagram for the siteswap 4413.



(B) 4413 converted to juggling states.

FIGURE 3. Siteswap to juggling state conversion.

To convert from a siteswap to juggling states, after finding the initial juggling state it's quite simple. Once we have the initial juggling state, to find the rest of the states we just follow a simple algorithm. Let our siteswap be $t_1 \dots t_n$, and let $s_{i-1} =< s_1, s_2, \dots, s_j >$ be the previous juggling state, then to get the next juggling state, we shift everything down one position and add a ball in the t_i position, to get $s_i =< s_2, s_3, \dots, \underline{1}, \dots, s_{j-1} >$, note that the underlined 1 is at height t_i.

Now we just need to get the initial juggling state from $t_1 \dots t_n$. To do so we can start by putting a one at height t_1. If $t_1 > 1$ then we will need a ball in our hand/at height one for the next throw t_2, unless t_2 is 0. Then to check if we need a ball at height 2, we check if either the first or second throw will be caught by then, if so we leave it empty, if not we add a ball at height 2 for our third throw. We then continue this process until all balls are accounted for. This is somewhat poorly worded, so we'll show an example.

Consider the siteswap 4413 again. We know that this uses a total of $(4+4+1+3)/4 = 3$ balls. Thus our initial state has three balls, and since this is basic juggling, i.e. one ball at a time, we get three instances where we have a ball/or a 1, at a particular height. Our first throw is a "4", so we can fill the fourth position, $<?, ?, ?, \underline{1}>$, if we think about the next juggling state, this first throw will still be in the air, thus we'll need a ball in hand for the second "4" throw, thus we have $< \underline{1}, ?, ?, 1 >$. Now for the next throw, the "1", our first two balls are still in the air, so we'll need a ball at height 2, $< 1, \underline{1}, ?, 1 >$. Now since all three balls are accounted for, we fill in the rest of the heights with 0's, to get $< 1, 1, 0, 1 >$ as our initial juggling state.

JS(a,b,n,m)

Let us denote by JS(a,b,n,m), the set of juggling states such that the initial state is **a**, the terminal state is **b**, its length is n and its hand capacity is m. We omit the parameter m if there is no hand capacity constraint.

Example 3.2. *JS($< 1 >, < 1 >, 3$) refers to the set of all sequences of juggling states with initial and terminal state $< 1 >$ and length 3. These elements would be* $(< 1 >, < 1 >, < 1 >, < 1 >), (< 1 >, < 1 >, < 0, 1 >, < 1 >), (< 1 >, < 0, 1 >, < 1 >, < 1 >)$, *and* $(< 1 >, < 0, 0, 1 >, < 0, 1 >, < 1 >)$.

In Appendix **??**, we include the code for generating the set JS(a, b, n) in Sage [The21]. Let us see how it works here. I created a program in sage that generates the set JS(a,b,n). It works by constructing every element simultaneously. The juggling patterns are determined by the initial state, and the throws that take place between each state. The program works by starting with the initial state and then creates every possible juggling state that can follow the initial state, note that the height of the balls in this state are limited by where the balls need to be in the terminal state. This process continues until every possible juggling state with the correct initial state and length are made. Last it removes the sequences with the wrong terminal state. I originally went about this a different way. Instead of generating all possible elements at once, I made a function that would generate a random element of JS(a,b,n). I would then run this function repeatedly to hopefully generate every element. At a certain size this method took too long and was inconsistent in actually generating every element. I then created the current method based on ideas explored in my first attempt.

The Poset

We then look at a poset on JS(a,b,n,m) described in [BHH+20]. It was original described for another object Tesler matrices, in specific cases they are bijective to sequences of juggling states. The poset was then generalized to more general juggling sequences. See [O'N17] for more on the Tesler poset.

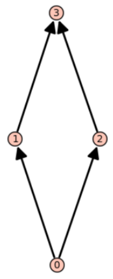
Let us see how the poset is defined for juggling state sequences. In order to define the poset we need to specify the set of elements together with a cover relation. Poset stands for partially ordered set, so for a poset we have the set and all the pairs of relations. Not every element has to be or will be comparable.

Definition 3.3. *We denote the poset PJS(a,b,n,m) on the set of juggling sequences JS(a,b,n,m). Given two juggling sequences from the set S and S', the cover relation is defined as $S \lessdot S'$ if S' can be obtained from S by taking two throws in S, one at time i and the second when the first throw lands, and combining them into one throw at time i that lands when the second throw would of landed.*

Example 3.4. *We will go through PJS($<1>, <1>, 3$). Using the sage program we can construct the set.*

```
0 [[1], [1], [1], [1]]
1 [[1], [1], [0, 1], [1]]
2 [[1], [0, 1], [1], [1]]
3 [[1], [0, 0, 1], [0, 1], [1]]
```

We then have the following relations $0 \lessdot 1, 0 \lessdot 2, 1 \lessdot 3$, and $2 \lessdot 3$. First let's look at $0 \lessdot 1$. 0 consists of just throws to the hand every interval, and 1 consists of throws to the hand except at time 3 which is a throw to height 1. 1 can be obtained by 0 by combining the throw at time 3 to height 1, and the immediate next throw, the one at time 4. These two throws get combined into one larger throw at time 3 to height 2. Similarly for $0 \lessdot 2$, 2 can be obtained by 0 by combing the throw at time 2 with the throw at time 3, to one throw at time 2 to height 2. For $1 \lessdot 3$, 3 can be obtained from 1 by combing the throw at time 2 with throw at time 3, into one larger throw at time 2 to height 3. Lastly $2 \lessdot 3$, 3 can be obtained from 2 by combing the throw at time 2 with the throw at time 4, into one larger throw at time 2 to height 3. The resulting poset looks like this,



Furthermore this poset is isomorphic to the boolean lattice of order 2. In fact the poset PJS($<1>, <1>, n$) is isomorphic to the boolean lattice of order n. See [BHH+20] for more on this specific case.

In general sense, given two juggling sequences, one covers the other if two throws can be combined into one larger throw, with the sequence with less throws covering the other.

I programmed a function in SAGE [The21], see Appendix **??**, that can generate this relation between two elements of JS(a,b,n). It works by checking the number of differences between the throws of two elements. It works out that if there are exactly three differences than there is a relation, with the element with more throws covering the other. I then used functions in sage to generate the poset for various examples of JS(a,b,n).

To elaborate on why this function works, we need to clarify some ideas. Instead of thinking about our juggling pattern as a sequence of juggling states we could think about it as the initial state along with a sequence of throws. For example, consider the juggling pattern $(<1,1>, <1,0,1>, <1,1>)$. Recall that we can only throw a ball when it's in our hand (or at height 1), and if a ball isn't in our hand then in the next state it will have dropped one height. Now if we look at our first two states $<1,1>$ and $<1,0,1>$, to go from the first state to the second state we throw the ball in our hand to height three, $<\underline{1},1>$ to $<1,0,\underline{1}>$. We will denote this throw by $<0,0,1>$, if we were looking at the states $<2,1>$ and $<1,0,2>$

instead, then the throw would be $< 0, 0, 2 >$ and we would consider this as two throws. Likewise to go from the second state to the third state, we throw the ball in our hand to height 1, $< \underline{1}, 0, 1 >$ to $< \underline{1}, 1 >$, we denote this throw by $< 1 >$. Thus the juggling pattern $(< 1, 1 >, < 1, 0, 1 >, < 1, 1 >)$ can be characterized by the initial state $< 1, 1 >$ along with the two throws $(< 0, 0, 1 >, < 1 >)$.

More formally to describe the throws, recall that consecutive juggling states, $\mathbf{s}_{i-1} =< s_1, \dots, s_h >$ and $\mathbf{s}_i$ must satisfy:

$$\mathbf{s}_i =< s_2 + b_1, s_3 + b_2, \dots, s_h + b_{h-1}, b_h, b_{h'} >,$$

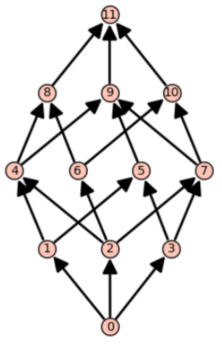
where $b_j \geq 0$ and $\sum_{j=1}^{h'} b_j = s_1$. The throws between $\mathbf{s}_{i-1}$ and $\mathbf{s}_i$ would be $< b_1, b_2, \dots, b_{h'} >$ and s_1 would be the number of throws.

We can think of our poset in another way. Lets say $S < S'$, if we compare the throws of S and S' we will see that there are exactly three differences, in S there is a throw at time i to height j that is missing in S', likewise in S' there is a throw at time i to height $j + k$ that is missing in S, and lastly in S there is a throw at time $i + j$ to height k that is missing in S'.

Finally, we have that if there are exactly three differences between the throws of S and S' then there is a cover relation between them, with the pattern with less throws covering the other, without loss of generality lets say S' has less throws. The first instance of a difference between throws we will call time i and we get at least two differences, a throw in S gets replaced by a higher throw in S'. If the throws at time i have more then two differences then we must be switching more then one throw, every time we switch two throws we get two differences, and thus no cover relation occurs. The second instance of a difference we will call time j and S has a throw that S' is missing. the only way from the rest of the throws to be the same is if the higher throw at i in S' lands at the same time as the throw at j in S, which exactly describes the cover relation, thus $S < S'$. In addition we get that if $S < S'$ then S has exactly one more throw then S'.

Example 3.5. *Let's again look at PJS($< 1 >, < 1 >, < 3 >$), specifically the cover relation $1 \lessdot 3$ from Example 3.2. 1 was the juggling sequence $(< 1 >, < 1 >, < 0, 1 >, < 1 >)$ and 3 was the juggling sequence $(< 1 >, < 0, 0, 1 >, < 0, 1 >, < 1 >)$. The throws for 1 are $(< 1 >, < 0, 1 >, < 0 >)$ and the throws for 3 are $(< 0, 0, 1 >, < 0 >, < 0 >)$. To get the number of differences we first "subtract" each corresponding element from the throws from each other. So we get a new list $(< 1 >, < 0, 1 >, < 0 >) - (< 0, 0, 1 >, < 0 >, < 0 >) = (< 1, 0, -1 >, < 0, 1 >, < 0 >)$. Finally we total the absolute value of each number in the list, $< 1, 0, -1 >$ gives us 2, $< 0, 1 >$ gives us 1, and $< 0 >$ gives nothing, thus in total we have 3 differences.*

In Figure 4 we see some examples of Hasse diagrams for the poset. Note that in the diagrams the numbers shown are the index of the juggling pattern in JS(a,b,n).


(A) PJS($< 4 >, < 1, 2, 1 >, 2$)

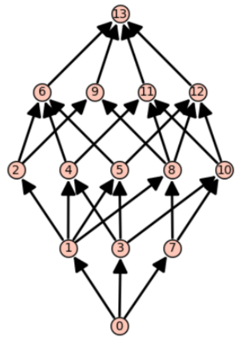

(B) PJS($< 1, 0, 1 >, < 0, 2 >, 4$)

FIGURE 4. Poset examples

We will now focus on a particular case of PJS(a,b,n) where $a = b =< 1^k >$ and $n \geq k$, see examples in Figure 5. Here $< 1^k >$ refers to the sequence of k 1s, for example if $k = 3$ then $a = b =< 1, 1, 1 >$. The

$n \geq k$ comes from the balls falling one between states. If $n < k$ then there will be ball(s) that never touch are hand, thus if they were removed it wouldn't change the poset. These posets have unique minimal element, but their maximal element is not unique. In fact, we have the following result.

Theorem 3.6. *Consider the poset PJS(a,b,n), where $a = b =< 1^k >$ and $n \geq k$. There exists a bijection between the maximal elements of PJS(a,b,n) and the symmetric group S_k.*

Before proving this result, we describe the maximal elements of PJS($< 1^k >, < 1^k, n >$) in the following result.

Proposition 3.7. *The maximal elements of PJS($< 1^k >, < 1^k >, n$) are exactly the elements with k throws.*

Proof. We claim that there are no elements with less then k throws. To show this, let's construct a pattern with the minimum number of throws, consider the initial state $< 1, 1, \ldots, 1 >$ to get to the next state a throw must occur, note that a throw to height 1 is still considered a throw since we currently have a ball in our hand, if we throw it to a height less then n then it will be in our hand again before the sequence is over, and we must then throw it again. Thus to minimize the number of throws we use, we throw it to a height greater or equal to n so the ball won't land before the terminal state, we have this first ball contributing 1 throw to our total. Now if we look at the second state the ball at height 2 from the initial state, $< 1, \underline{1}, \ldots, 1 >$, will be in our hand and so we must throw it. Again to minimize the number of throws we throw it to a height greater or equal to $n-1$ so it won't land before the terminal state. We repeat this for the third ball that will be in our hand in the third state, the fourth ball in our hand in the fourth state, and so on, note that since n is greater then k, each ball will have been in our hand at least once by the time the sequence is over. So with minimal throws we will still have had to have thrown k balls, where each ball is thrown exactly once. Thus there are no elements with less then k throws.

We can then say that all elements with exactly k throws are maximal. Let's assume the opposite, let S be an element with k throws and S' be an element such that $S < S'$. Then from how we defined our poset, S' has one less throw then S, so S' has $k-1$ throws. However there are no elements with $k-1$ throws thus S' can't exist. Thus an element with k throws isn't covered by another element, which makes it a maximal element.

All that's left is to show that elements with more then k throws are not maximal. To do so, let's consider and element with more then k throws and call it S. At least one of these throws must land before the terminal state, for if it didn't then all the balls will only have been thrown once, and we'd only get k throws. Looking at the throw that lands before the terminal state, lets call the time it was thrown i and the height j. Since the ball lands before the terminal state then during time $i + j$ we throw it again to some height. We now consider a new sequence S' where replace the throw in at i and the throw at $i + j$ in S with a throw at i to their combined height, this new throw in S' will land at the same time as the $i + j$ throw in S, thus keeping everything else constant S' will have the same terminal state as S. Therefore S' is also in our set and $S < S'$, thus S is not maximal. So we have just shown that elements with more then k throws are not maximal.

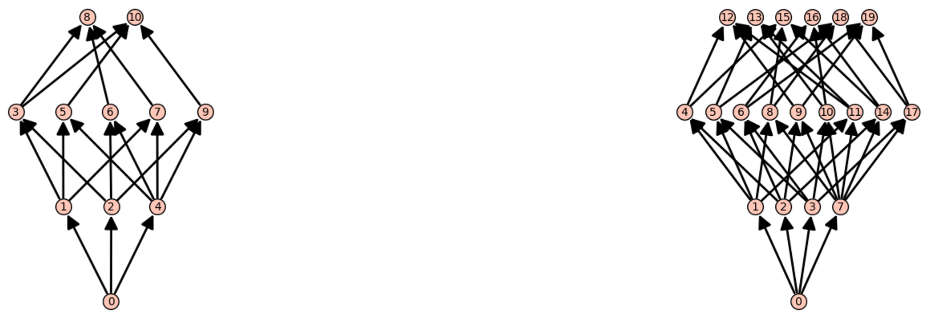
The maximal elements of PJS(a,b,n) are the elements with exactly k throws. □

Proof of Theorem 3.6. From Proposition 3.7 we have that the maximal elements of the poset are the elements with exactly k throws. We will now show that there are $k!$ of these elements.

For an element to have k throws each ball is thrown only once. So once a ball is thrown, it will fall until it reaches it's position in the terminal state $< 1, 1, \ldots, 1 >$, of which there are k possible positions. So the element with k throws are determined by where each ball from the initial state maps to in the terminal state. There are then k possible options to throw our first ball too. $k-1$ options for the second, since the first ball already takes up a spot. $k-2$ for the third, and so on. Thus there are $k!$ elements that have k throws, which gives us that there are $k!$ maximal elements.

This also gives us the mapping from the maximal elements to the symmetric group S_k. We look at the balls in the initial state $< 1, 1, \ldots, 1 >$ and see where they map too in the terminal state $< 1, 1, \ldots >$, for example if the ball at height 1 in the initial state goes to height 3 in the terminal state, then we say 1 maps to 3 and to get the rest of the permutation we look at the rest of the balls. A more concrete example, consider PJS($< 1,1,1 >, < 1,1,1 >, 3$), the maximal element ($< 1,1,1 >, < 1,1,0,0,1 >, < 1,0,1,1 >, < 1,1,1 >$) would map to the permutation in one line notation 321. Since each maximal element has exactly one corresponding permutation we have that this mapping is well defined and injective.

Therefore since the maximal elements of PJS(a,b,n) and S_k have the same size, and the described mapping from the maximal elements of PJS(a,b,n) to S_k is injective, this mapping is a bijection. □



(A) $k = 2, n = 3$ (B) $k = 3, n = 3$

FIGURE 5. Examples of PJS($< 1^k >, < 1^k >, n$)

Example 3.8. *A more thorough example of the bijection described in Theorem 3.6. Consider PJS*($< 1,1,1 >, < 1,1,1 >, 3$)*, the maximal elements are,*

```
12 [[1, 1, 1], [1, 1, 1], [1, 1, 1], [1, 1, 1]]
13 [[1, 1, 1], [1, 1, 1], [1, 1, 0, 1], [1, 1, 1]]
15 [[1, 1, 1], [1, 1, 0, 1], [1, 1, 1], [1, 1, 1]]
16 [[1, 1, 1], [1, 1, 0, 1], [1, 0, 1, 1], [1, 1, 1]]
18 [[1, 1, 1], [1, 1, 0, 0, 1], [1, 1, 0, 1], [1, 1, 1]]
19 [[1, 1, 1], [1, 1, 0, 0, 1], [1, 0, 1, 1], [1, 1, 1]]
```

By our theorem this set is bijective to the symmetric group S_3. To read the bijection we trace the balls from the initial to terminal state. In 12*, the position of the balls in the initial and terminal state do not chance, thus* 12 *corresponds to the identity permutation* 123*. In* 13*, the second and third ball switch, thus corresponding to the permutation* 132*. In* 15*, the first and second ball switch, thus corresponding to the permutation* 213*. In* 16*, the first ball goes to the second, the second goes to the third, and the third goes to the first, thus corresponding to the permutation* 231*. In* 18*, the first ball goes to the third, the second goes to the first, and the third goes to the second, thus corresponding to the permutation* 312*. Finally in* 19*, the first ball and third ball switch, thus corresponding to the permutation* 321*.*

Next, we look at the subposets induced by the minimal element and each of the maximal elements. In Figure 6, we present the subposets induced by the maximal elements for $n = k = 3$. In this case, the subsets are all isomorphic to each other and we have the following result.

Proposition 3.9. *For $n = k \leq 5$, the subposets induced by the maximal elements in PJS($< 1^k >, < 1^k >, n$) are all isomorphic to each other.*

Proof. Shown computationally using sage program in Appendix **??**. □

A way to construct these subposets of our juggling patterns is by choosing a maximal elements and then constructing the subposet by adding the elements the maximal elements covers, and then the elements each of those elements covers and so on.

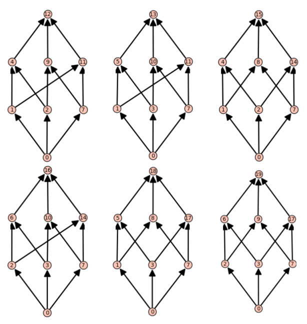


FIGURE 6. Subposets of PJS(< 1, 1, 1 >, < 1, 1, 1 >, 3)

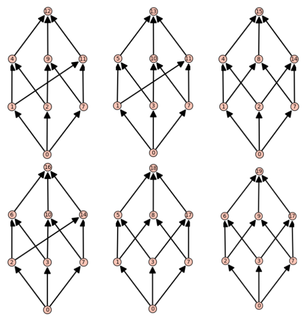
In the case where we consider JS(< 1, 1, 1 >, < 1, 1, 1 >, 3) we actually have that these subposets are isomorphic to the boolean lattice of order 3. For example we consider the maximal element (< 1, 1, 1 >, < 1, 1, 1 >, < 1, 1, 1 >, < 1, 1, 1 >) corresponding to the permutation 123. The analogue maximal element in the boolean case would be $[x, y, z]$, which covers the elements $[x, y], [x, z], [y, z]$. To get the elements that our juggling pattern covers, we look at the throws that make up the sequence, that would be (< 0, 0, 1 >, < 0, 0, 1 >, < 0, 0, 1 >).

Recall in that determining the cover relation we were "combining" throws, going in the opposite direction we can decompose the throws. In other words we take a throw at time i to height k, and split it into two throws, one at time i to height j and another at time $i + j$ to height $k - j$. If we look at our three throws, there are two was to decompose the first throw, we could have the throws (< 1 >, < 0, 1, 1 >, < 0, 0, 1 >) or (< 0, 1 >, < 0, 0, 1 >, < 1, 0, 1 >). There is one way to decompose the second throw, that would be (< 0, 0, 1 >, < 1 >, < 0, 1, 1 >), and there are no ways to decompose the third throw. Thus we get the three corresponding juggling patterns (< 1, 1, 1 >, < 2, 1 >, < 1, 1, 1 >, < 1, 1, 1 >), (< 1, 1, 1 >, < 1, 2 >, < 2, 0, 1 >, < 1, 1, 1 >), and (< 1, 1, 1 >, < 1, 1, 1 >, < 2, 1 >, < 1, 1, 1 >) as the elements analogues to $[x, y], [x, z], [y, z]$ from the boolean lattice. These three juggling patterns decompose in a similar fashion to $[x, y]$ decomposing in to $[x]$ and $[y]$. Each of the juggling patterns on this level can decompose into two different ways giving us the three elements (< 1, 1, 1 >, < 2, 1 >, < 2, 0, 1 >, < 1, 1, 1 >), (< 1, 1, 1 >, < 2, 1 >, < 2, 1 >, < 1, 1, 1 >), and (< 1, 1, 1 >, < 1, 2 >, < 3 >, < 1, 1, 1 >) as the elements analogue to $[x], [y], [z]$. Finally each of these three juggling patterns only decompose into (< 1, 1, 1 >, < 2, 1 >, < 3 >, < 1, 1, 1 >) the element analogues to $[\,]$.

We can follow similar steps for the remaining maximal elements.

The general result is still a conjecture.

Conjecture 3.10. *For $k \geq 6$, the subposets induced by the maximal elements in PJS($< 1^k >, < 1^k >, n$) are all isomorphic to each other.*


(A) Subposets of PJS($< 1, 1, 1 >, < 1, 1, 1 >, 3$)

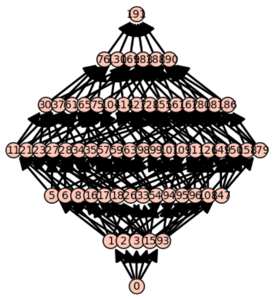

(B) Subposet of PJS($< 1, 1, 1, 1 >, < 1, 1, 1, 1 >, 4$)

FIGURE 7. Subposet examples

4. Refrences

[BHH+20] Carolina Benedetti, Christopher R. H. Hanusa, Pamela E. Harris, Alejandro H. Morales, and Anthony Simpson. Kostant's partition function and magic multiplex juggling sequences. *Ann. Comb.*, 24(3):439–473, 2020.

[But15] Steve Butler. Maths of juggling: Juggling diagrams and siteswaps Link to the YouTube channel, 2015.

[O'N17] Jason O'Neill. On the poset and asymptotics of tesler matrices, 2017.

[The21] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.2)*, 2021. `https://www.sagemath.org`.