Spring 2016

The Diameter of a Rouquier Block

Andrew Mayer
asm70@zips.uakron.edu

The Diameter of a Rouquier Block

Andrew Mayer

Submitted to The University of Akron Honors College

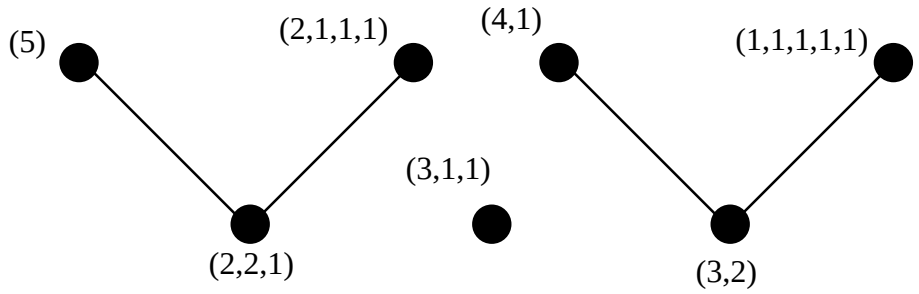
Sponsor: Dr. James P. Cossey

First Reader: Dr. Jeffrey Riedl

Second Reader: Dr. Stefan Forcey

Introduction

Recall that a partition of an integer n is a way of writing n as a sum of positive integers. We write the sum from largest to smallest; for example, $8 = 4 + 2 + 1 + 1$. We write this as $\lambda=(4,2,1,1)$ and say $\lambda \rightarrow 8$. We will fix two positive integers n and p; note that p is typically prime when working in the context of representation theory, but for our strictly combinatorial purposes p does not have to be prime. Once we fix n, we will have some number of partitions of n. We will represent each of these partitions with a point of a graph. We say that two partitions λ and σ are connected by an edge in the graph if they satisfy a special condition which will be discussed later. For example, suppose we chose n = 5, p = 3. Our graph will look like this [2]:



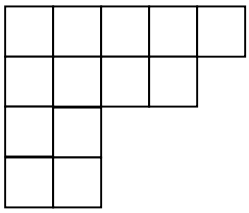
Notice that we have several distinct connected components of our graph. We will define a block for the integers n and p to be any connected component of this graph. We are interested in Rouquier blocks, specifically; these will be defined later. We are going to find an upper bound for the furthest apart two vertices can be in a Rouquier block; this is called the diameter of that block. We see in the above example that we have two blocks of diameter two and one block of diameter zero. To find this bound, we need to find the weight of the block, which is an invariant which we will define later. With this, we can state our main result.

Theorem (1) – If B is a Rouquier block for the integer p with weight w, then the diameter of B is at most $2\log_2|w| + p + 2$.

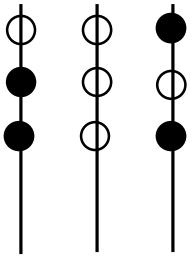
Section 1: Definitions

We now need to define several terms in order to progress. First, we will introduce the idea of a Young diagram (see [3] for more information). To build a Young diagram, we take our partition λ and begin building rows of boxes; we put a number of boxes in the first row equal to the value of the first integer of λ, then a second row equal to the second integer, etc. What we have ends up looking like an upside down staircase, like the following:

Let $\lambda=(5,4,2,2)$. Then our Young diagram will look like this:

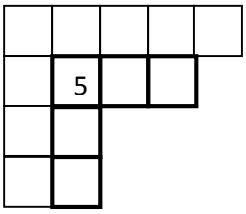


Next, we will discuss the abacus of a partition [2]. To build the abacus of a partition, we will draw p vertical lines (recall that p is fixed). Next, we will start putting "beads" on the abacus, either open beads or closed beads, going left to right, top to bottom. We start by putting in a number of open beads equal to the smallest number in our partition, then a number of closed beads equal to how many times that particular number appears in our partition. We will repeat this process for each integer that appears in our partition smallest to largest; however, we will always put in a number of closed beads equal to the difference between the current integer and the previous integer.

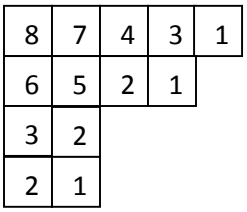


This is what the abacus would look like for $\lambda = (5,4,2,2)$ if we chose p=3

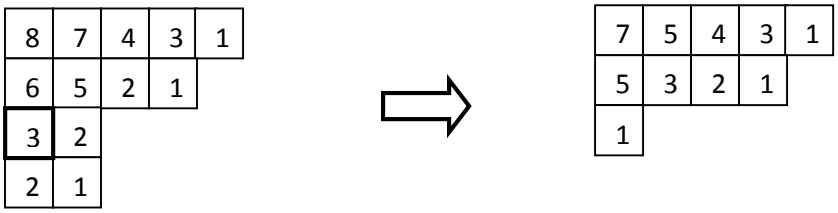
Now we can discuss a process called hook removal on a Young diagram. First we set a value for p; we will use this to remove p-hooks. In our Young diagram, we will label each box with the total number of boxes to the right and below each box, like so:



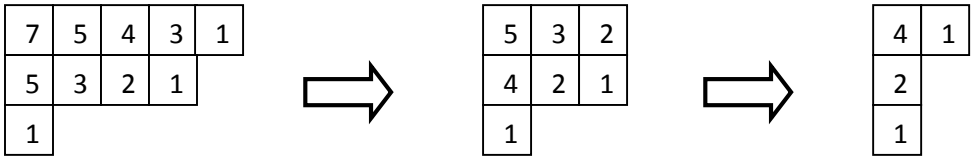
Completely done, it will look like this:



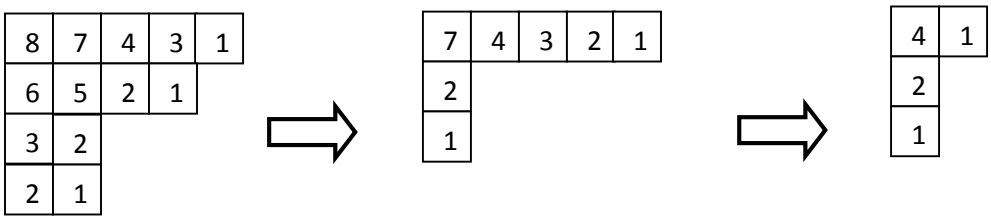
To remove a p-hook from this diagram, we will pick a box whose number is divisible by p and remove its outer layer. In the previous example, let us choose p to be 3. We can remove a 3-hook from the bottom left like so:



There are two important things to note. One, we do not necessarily remove the box that we choose; and two, values in some of the boxes changed while others stayed the same. We must reevaluate every box's value after removing a p-hook before we try to remove another. As we relabel, we see that there are more 3-hooks to remove:



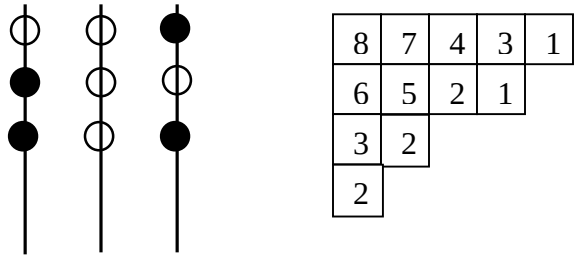
Here we do not have any more values that are divisible by 3. This is what we call the core of our Young diagram for p=3. Note that it does not matter what order we remove the p-hooks. For example, if we had started with the 6, our process would have looked like this.



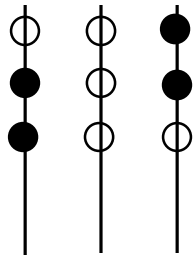
We can see that the core that we get this way turns out to be the same as the core we had by taking a different path. It is a basic combinatorial fact that for a partition λ, the core is independent of the choice of order that the hooks are removed [2].

On an abacus, we will define hook removal to be the process of moving beads up the abacus. For each column of beads on the abacus, push all the closed beads up to replace the open bead spots until there is no open bead between the first closed bead and the last on that column. We will do this for each column on the abacus. We do this in order to remove p-hooks. This process removes p-hooks because every time we move a bead up, we are essentially omitting the beads that were in between the moved bead and the space to which we moved it. By the design of an abacus, the number of beads we skip is divisible by p; thus, removing a p-hook. The abacus with which we are left contains no p-hooks; we also say that this abacus is the core of our partition.

Let us look at an example of an abacus and its corresponding Young diagram as we remove hooks from the abacus of the partition λ=(5,4,2,2) with p=3.

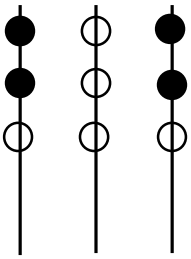


If we move the bottom right bead up one space, we will get this:



6	5	2	1
5	4	1	
3	2		
2	1		

We can see that we have lost the 3-hook that was on the top right corner of the Young diagram. If we push all the closed beads on the left runner to the top, we will get this:



4	1
2	
1	

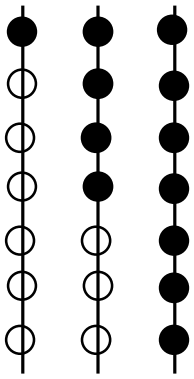
Notice that, whenever we move a closed bead up, we replace it with an open bead. We see now that the partition with which we are left has no 3-hooks. By moving all the closed beads up all the way on the runners, we have eliminated all of the 3-hooks. So, instead of removing all the p-hooks from a Young diagram, we can simply discover its abacus, push all the closed beads up, and reform a Young diagram from the new abacus, one without any p-hooks.

We can now define a block of n with respect to p as all partitions of n that have the same core with respect to p.

We will now define the quotient for p. The p-quotient (we will refer to this as the quotient if the p is clear) of a partition λ is a sequence of partitions $(\lambda(0), \lambda(1),\ldots, \lambda(p-1))$ formed by considering the partition of each runner of the abacus. The quotient for our previous $\lambda=(5,4,2,2)$ would be $((1,1), \emptyset, (1))$. We can also define the weight w of our partition λ as the sum of the

sizes of the partitions in the quotient λ(0), λ(1),…, λ(p-1). In our previous example, the weight is three.

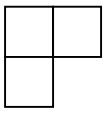
A p-block B of weight w is said to be Rouquier if the number of closed beads on each runner of the core's abacus increases by at least w-1 from left to right.



In this abacus, we have a Rouquier block with weight at most 4.

Now we must observe the process called linking, focusing on Rouquier blocks. To do this, we must use Littlewood-Richardson coefficients [3]. This process is referred to as "multiplying two partitions." To start, we will select two Young diagrams, α and β. Next, we will number every box in the first row of β with a 1, the second row with a 2, etc.

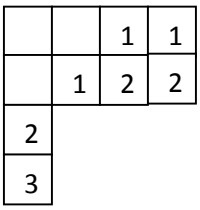
α



β

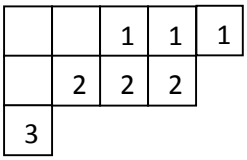
1	1	1
2	2	2
3		

Finally, we put each box from β onto the right side of α in any place we choose, so long as they follow a set of rules. First, the numbers in each row in the resulting partition must be nondecreasing. Second, the numbers in each column must be strictly increasing. Third, the boxes read right to left, top to bottom form a ballot sequence: A ballot sequence is a finite series of natural numbers such that, at any point in the sequence, for each value of k, the number of terms equal to k-1 is greater than or equal to the number of terms equal to k up to that point.



This is one legal way in which to multiply these two partitions; the ballot sequence for this partition is 1,1,2,2,1,2,3.

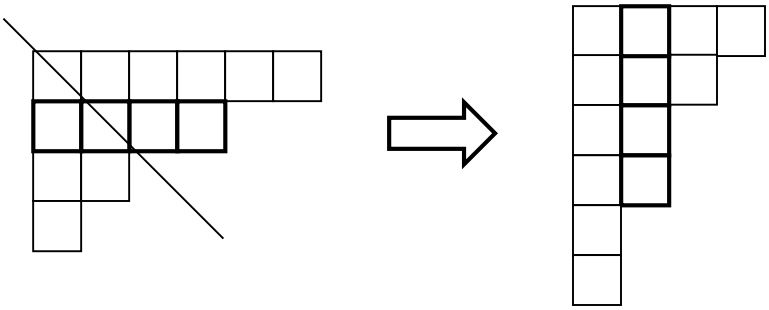
Note that the new partition must also have nonincreasing row lengths, just like any other partition.



This is another legal way in which we can multiply two partitions. This sequence is 1,1,1,2,2,2,3.

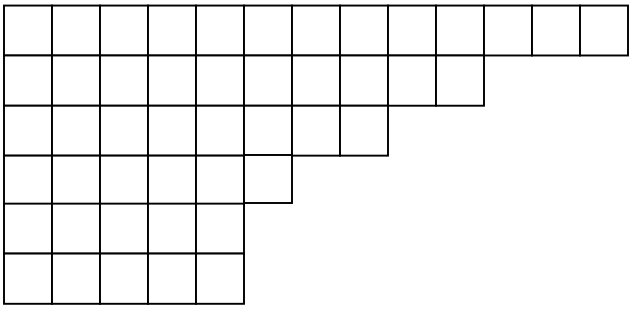
We define a ballot sequence to be simple if it is of the form 1,1,…,1,2,2,…,2,3,3,…

Now that we have defined the Littlewood-Richardson process, we can now begin describing adjacency in the graph of a Rouquier block (for more details, see [1] and [4]). First, we will write the quotient of λ as ($\lambda(0)$,…, $\lambda(p-1)$) in their Young diagram forms. Then, for each $\lambda(i)$, we are going to split $\lambda(i)$ into a left piece and a right piece, $\lambda_L(i)$ and $\lambda_R(i)$. Here, $\lambda_L(i)$ and $\lambda_R(i)$ are partitions such that λ can be obtained by applying the Littlewood-Richardson process to $\lambda_L(i)$ and $\lambda_R(i)$. We can make these two partitions in any way we choose, adhering to the Littlewood-Richardson process. One additional requirement is that λ must be partitioned such that $\lambda_L(0)$ is empty and $\lambda_R(p-1)$ is empty. Once we have split all the partitions in the quotient into two pieces, we then conjugate $\lambda_R(i)$ by drawing an imaginary line from the top left corner diagonally down and flipping $\lambda_R(i)$ across that axis. Row 1 becomes column 1, row 2 becomes column 2, etc., etc., like so:

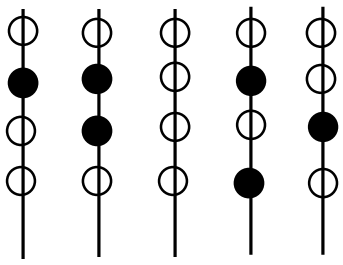


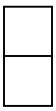
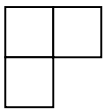
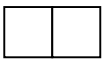
Once we have conjugated, we move each of the conjugated $\lambda_R(i)^*$ over one slot; when we do this, our new partition set will be of the form $(\emptyset, \mu(1), \mu(2),\ldots,\mu(p-1))$, where $\mu(i)$ can be any partition formed by the Littlewood-Richardson multiplication of $\lambda_R(i-1)^* \times \lambda_L(i)^*$, as described above. Finally, we are going to do the process in reverse. So, each new partition $\mu(i)$ will be split into a left and right piece, as above. The right pieces $\mu_R(i)$ will now be conjugated and moved one slot to the left and combined with $\mu_L(i-1)$ via the Littlewood-Richardson process again. Once all this is done, we will have a new quotient whose total weight is equivalent to the starting weight.

Now we will do an example to illustrate this process. Combinatorially, we can do this linking process for any block. However, the algebraic applications only apply to Rouquier blocks. Rouquier blocks are often very large, so for ease of understanding our examples will not necessarily be Rouquier blocks; the combinatorial process remains the same. We will pick p=5 for our example. Let $\lambda = (13, 10, 8, 6, 5, 5)$. We know the Young diagram will look like this:

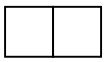
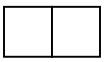


Its abacus looks like this:

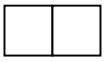
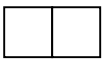


The quotient of λ is ( ,  , ø,  , ).

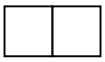
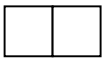
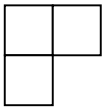
Now that we have the quotient, we can start our linking process. First, we will split each part of the quotient into a left and right piece:

(ø x  ,  x  , ø x ø,  x  ,  x ø)

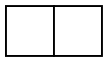
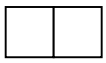
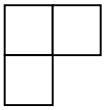
We conjugate each right piece and shift them once over, giving us the following:

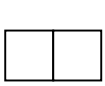
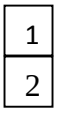
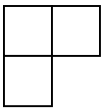
(ø,  x  , ø x  ,  x ø,  x )

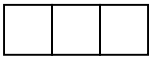
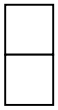
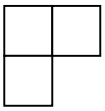
Combining with Littlewood-Richardson, we could make this quotient set:

(ø,  ,  ,  , )

Now we reverse the process:

(ø x ø,  x ø, ø x  , ø x  ,  x ø)

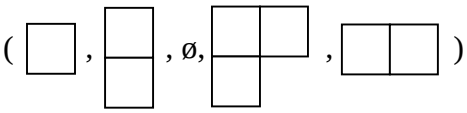
(ø x ø,  x  , ø x  , ø x ø,  x ø)

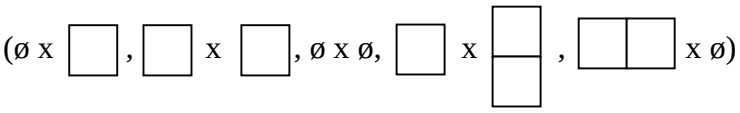
(ø,  ,  , ø, )

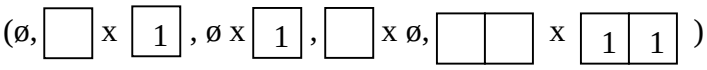
We will call our new quotient set the partition β. We can see that λ and β look very different. Other than the fact that they both have a weight of eight, it seems like they are not similar. However, because we were able to get from λ to β using the linking process, we call λ and β connected. Now we can define how two partitions get connected with an edge. If we can take

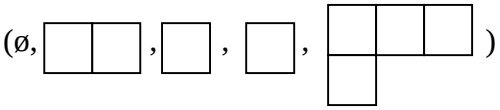
one partition λ, go through this process of splitting, linking, splitting, and linking again and get another partition β, then we say λ and β are connected; we connect their points with an edge.

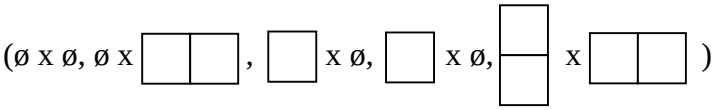
Note that this is just one way that we could link λ. We could have also done something like this:

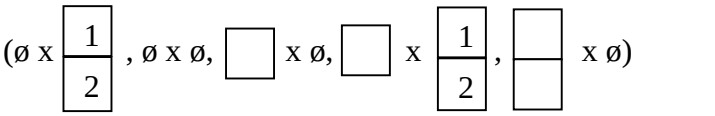


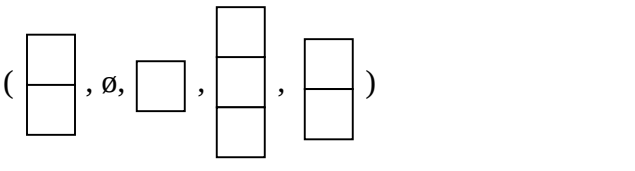








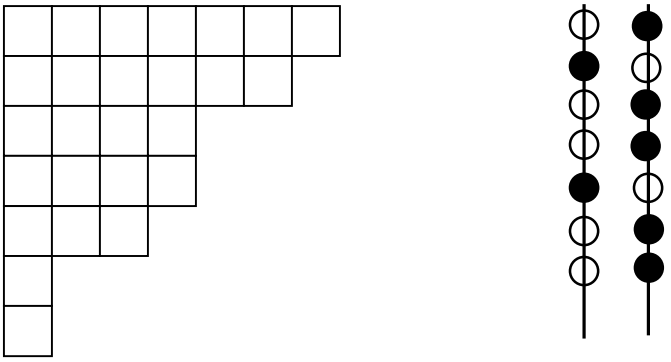




Clearly, there exists many different partitions that are connected to λ because there are many places where we make a choice of where to move boxes, in what shape to move them, and how to combine them with other boxes.

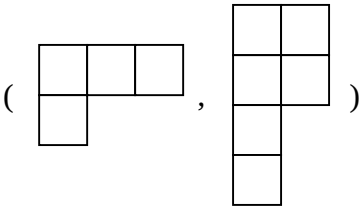
Section 2: p=2

Now that we have defined all of these terms and processes, we are ready to begin working towards our result. Suppose we choose λ = (7, 6, 4, 4, 3, 1, 1) and p=2. This partition looks like this:

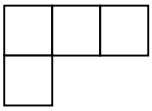
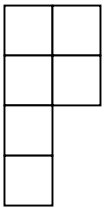


As described above, we can find that the 2-core of this partition to be as follows:

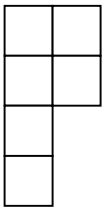
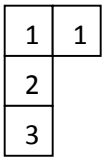
We can also find the quotient of λ as:



Now that we have the quotient of λ, we can start the linking process. First, we split each quotient into a left and right piece. Remembering our rules regarding the first and last terms of the quotient, we see that there is only one way to split these into pieces, namely:

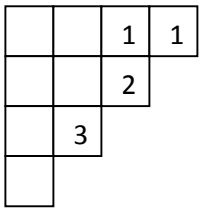
(ø x  ,  x ø)

Now we conjugate the right piece and move it one slot to the right, which bring us to this:

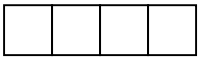
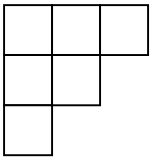
(ø,  x )

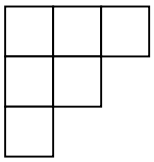
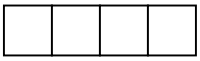
Note that we add the ballot sequence to the boxes on the right pieces here.

Next, we will follow the rules for Littlewood-Richardson to multiply these partitions together. We can do this in any number of ways; we will do it this way:

(ø, )

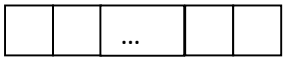
This is the simplest way to multiply two partitions; note this corresponds to the simple ballot sequence. We put all the boxes with ballot 1 in the first row, 2 in the second row, etc., until we run out of boxes. We now reverse the process like so:

(ø x ø,  x )

( , )

Again, if we call this new partition β, we can say that λ and β are connected. It is important to note that, when we fix p to be 2, every first step of the linking process will see the whole quotient in the first slot move to the right and combine with the whole quotient in the right slot. In that respect, we are limited in some ways while linking these quotients.

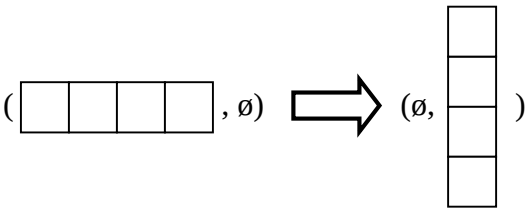
Now, we are trying to determine an upper bound on the distance apart two quotients of equal weight are. Given any two partitions λ and β, we will come up with an algorithm to connect the two consistently. We will show that both λ and β are connected to a special partition that we will name Ω for convenience. Ω looks like this:

Ω = (ø, )

The weight of Ω is equivalent to the weight of λ and β, but all of the boxes in the partition of Ω are in one row on the right side of the quotient set. If we try to connect λ and β through Ω, we can define a process that will connect λ to β via Ω. We will start by focusing on connecting λ to Ω as efficiently as we can.

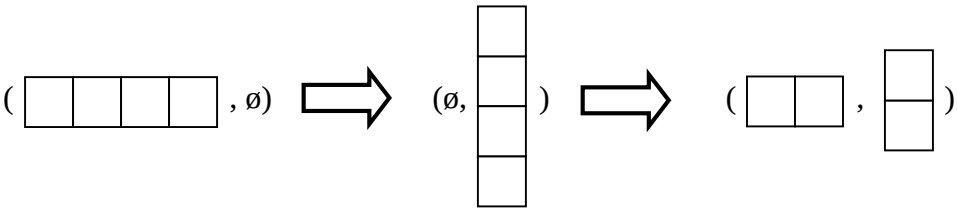
We will begin our investigation by considering the worst case scenario in terms of how far apart λ and Ω can be. Consider λ to be a row of boxes equal to its weight in the left quotient

spot and an empty set in the right. If we start our linking process from here, our first step forces us to conjugate the row into a column on the left side, like so:

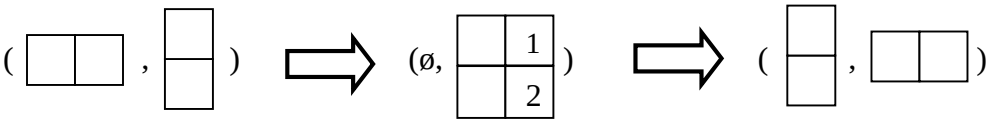


We can clearly see that this is the opposite of what we want: instead of a row on the right, we have a column. So, we need to define a process for rearranging this column into a row.

To accomplish this, we will be clever with the second half of the linking process. We will leave the top half of the column on the right side, and take the bottom half back over to the left. When we do this, we will end up with the following link:

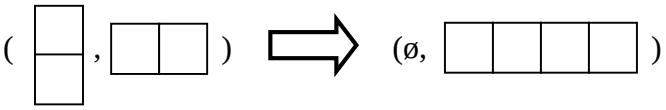


We will refer to this as our first "step," since we have completed the full linking process one time. We still do not have Ω, so we are going to do a similar process that will look like this:



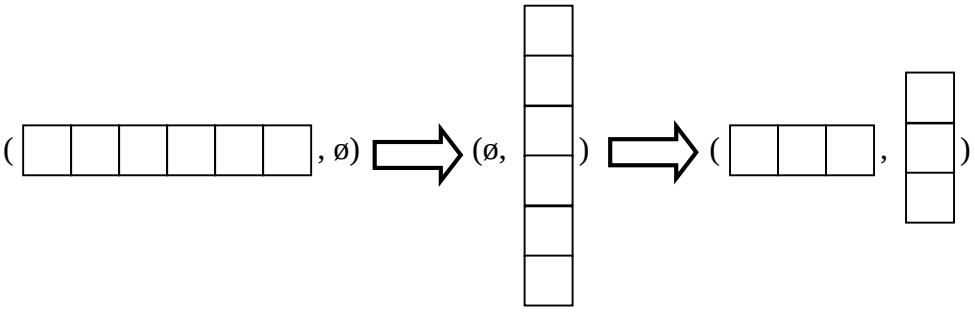
Notice that in this second step, we had to consider the Littlewood-Richardson multiplication. Since we are finding a general process, we are going to apply the Littlewood-Richardson process

in the simplest way possible, i.e. we put all the 1's in the first row, all the 2's in the second row, etc., until all the boxes are placed. Essentially, we push the right piece up against the left piece to form a new block. We can see that now we are getting close to Ω. If we do this one more time, we will get:

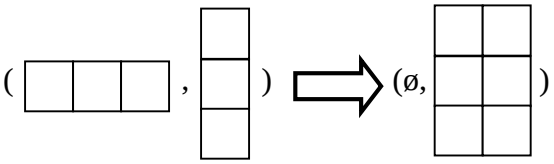


Now, on our third step, we have arrived at Ω. The pattern that we have established is that we take half of the left quotient and move it over each time. Because of this process, we can see that our formula for getting from any partition λ to Ω could be $\log_2(w) + 1$, since in this example we had $w = 4$, which gives $\log_2(4) + 1 = 2 + 1 = 3$, the number of steps that we took. Note that, since we took three steps to get from λ to Ω, we know that they are three edges apart.

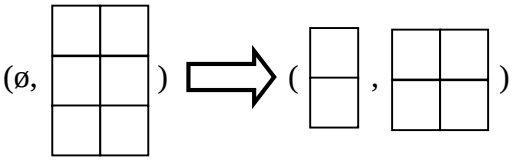
This is a fine process if the weight of λ is a power of 2, but what would happen if it was not? For example, what if the weight of λ was 13? If we run through the process, we will find that this partition will behave like the next largest weight of power 2; in this case, λ will behave like a partition with weight 16. Let us look at an example where the weight is 6. We will start the same way as before, by moving half the boxes:



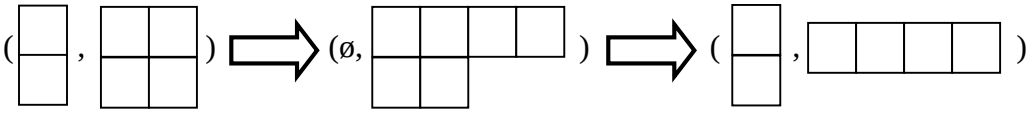
If we do this again, we get stuck:

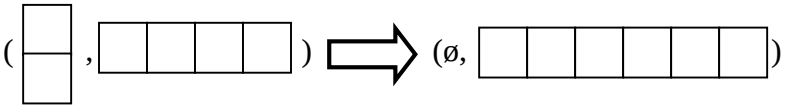


We cannot take half of the column height here because the column height c is odd and we cannot take half of a block. To rectify this, we are going to move (1/2)(c – 1). So, for our example, we will move (1/2)(3 – 1) = (1/2)(2) = 1 row, like so:



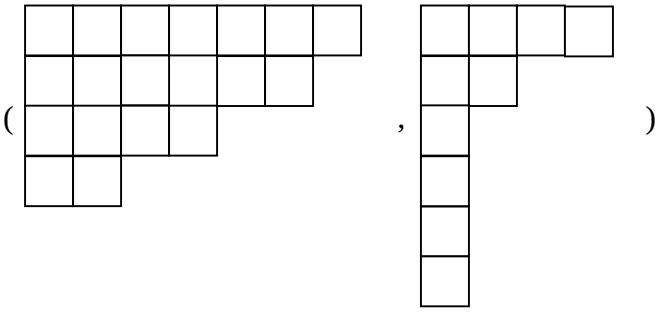
Now, when we proceed as normal, we will see this happen:



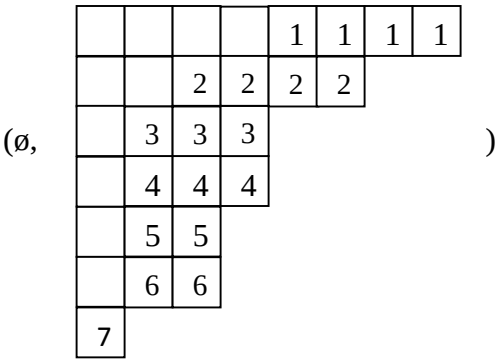


Our result is that this partition of weight 6 took 4 steps to get from λ to Ω, which is the same amount that would be calculated with the next weight of power 2, namely 8. So, our formula is a step function that increases the step count by one every time the weight surpasses a power of two.

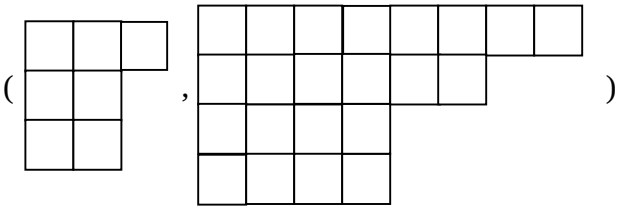
Suppose we had any partition λ of weight w for p=2. We must show that a similar process will work for transforming λ into Ω in at most $\log_2 w + 1$ steps. Let us, for instance, observe the following quotient:



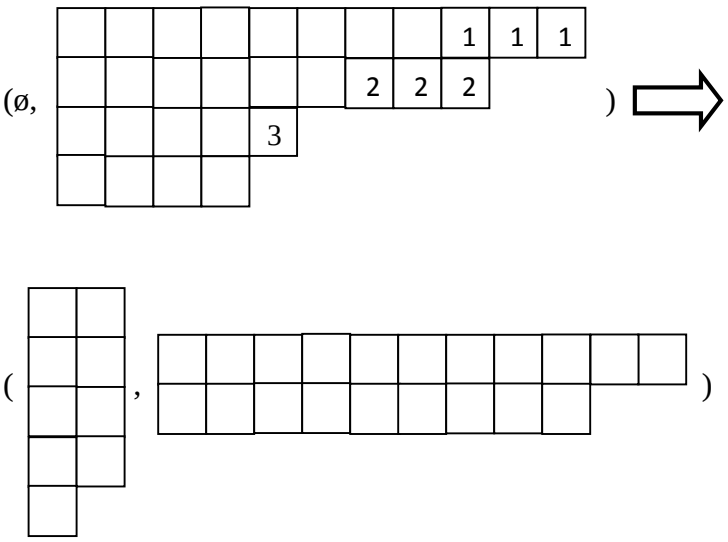
After we do our simple Littlewood-Richardson multiplication, we will get to here:



If we follow this algorithm, we will take the bottom three rows off this partition and move it back, leaving us with the following quotient:



Going through this process again, we will see this happen:



Once we repeat this process two more times, we will reach our desired quotient set, namely an empty partition and a row of length w. So, we can see that this process will work for any pair of quotients.

It is clear that the number of rows we move back to the first quotient spot depends on the number of rows in the second quotient. If the number of rows r we have is even, we will take r/2 of them off. If it is odd, we will take off (r-1)/2, leaving behind (r+1)/2. It is also clear that the weight of the partition is between two powers of 2, i.e. $2^{k-1} < w \leq 2^k$ for some positive integer k. If w is bounded by 2^k, then the weight of the first quotient after one step must be bounded by 2^{k-1}, since we have moved less than half of the weight. This is clear based on the construction of our algorithm. Therefore, the first quotient is always restrained by a power of two.

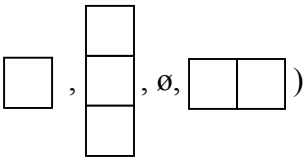
If our weight is restricted by 2^k, then it will take k+1 steps to get the first part of the quotient to be empty. The same argument, keeping track of the number of rows in the second component (which is of course bounded above by w) shows that after at most k+1 steps the second quotient will have only one row. Hence, the steps will be at most $\log_2 w + 1$ to reach Ω

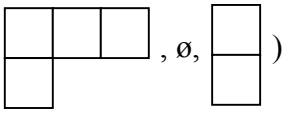
from λ. Similarly, it will take at most $\log_2 w + 1$ steps to reach Ω from β. With this, we have shown the following theorem:

Theorem (2): Given p=2, any two partitions of weight w must be at most $2\log_2 w + 2$ edges apart.

Section 3: p even

In this section, we will let p be an even integer bigger than two. In this case, our partition will have multiple quotients. If we are trying to connect two partitions when p is bigger than two, there is a possibility that, for each quotient, the quotient in one partition could have a smaller, bigger, or the same weight as the corresponding quotient in the other partition. For example:

Let λ = ()

Let α = (ø,  , ø,)

We see that the weight in the first quotient of λ is greater than the first quotient in α. The weight of the second quotient in λ is smaller than the corresponding quotient in α, and the last two quotients in both partitions have the same weight.

We will define the weight distribution of a quotient set to be the size of the partition of each quotient position. For the above λ, the weight distribution would be (1,3,0,2). Our first goal is to make the weight distribution of λ match the weight distribution of α. To make the weights match, we can go through the linking process in such a way that we ignore trying to make the

partitions match. We simply move enough boxes to make the weight distributions match. In our previous example, we can see that the easy thing to do is move the one box in the left quotient of λ one slot over to the second quotient while leaving the others alone. In this way, we make all the weights match.

For example, let p=8 and suppose the weight distribution of λ is given by (5, 4, 10, 2, 5, 1, 1, 4) and the weight distribution of α is (13, 2, 3, 5, 4, 0, 2, 3). Note that the actual quotients do not matter here; we are only concerned with the total weight in the quotients. To make these weights match up, we first want to make the first quotients match. So, our first step will be to move 8 boxes from the first quotient of α to the second. This will make a new partition α_1 with weight distribution (5, 10, 3, 5, 4, 0, 2, 3). We then move 6 boxes from the second quotient to the third quotient so that the second quotients will have matching weight, making α_2 with weight distribution (5, 4, 9, 5, 4, 0, 2, 3). Now, we see that λ has a larger third quotient than α_2. Therefore, we will move one box to make λ_3 with weight distribution (5, 4, 9, 3, 5, 1, 1, 4). We will continue this process in a similar manner until after p-2 steps we have λ_k with weight distribution (5, 4, 9, 3, 5, 1, 1, 4) and α_j with weight distribution (5, 4, 9, 3, 5, 1, 2, 3). Note that the last two quotients do not match exactly between λ_k and α_j. However, the total weight between the two quotients is the same. This will be addressed later. We have thus demonstrated the following result:

Lemma (1): Let λ and α be partitions of weight w for the integer p. Then λ and α can be made to have the same weight distribution in at most p-1 steps. If we only require that λ and α have the same weight distribution in the first p-2 positions, then we can achieve this in at most p-2 steps.

Notice that if λ and α have the same weight distribution for the first p-2 positions, then necessarily the total weight of λ on the last two positions is equal to the total weight of α on the last two positions. We will call such λ and α almost weight equivalent.

By the above discussion, we can transform any given λ and α into an almost weight equivalent pair in at most p-2 steps. Once λ and α have the almost the same weight distributions, we can then "pair off" the quotients: we will consider the first and second quotient of λ together, the third and fourth together, etc., until all the quotients are paired. Notice that the total weight on each pair (i, i+1) is equal on λ and α. We then run the p=2 algorithm previously discussed for each pair simultaneously until λ(i)=α(i) for all i. Notice this uses at most $2(\log_2(w) + 1) + p\text{-}2 = 2\log_2(w) + p$ steps.

Thus we have proven the following theorem:

Theorem (3): For any even integer p, two partitions of weight w are at most $2\log_2 w + p$ edges apart.

Notice that doing this method merely requires the first p-2 positions to have matching weights between λ and α; they do not have to match exactly. This is the reason why we left the last two quotients unequal the way we did. As long as the total weight for the pairs match, we are not concerned with what the individual quotients look like.

Section 4: p odd

Now we must consider any odd p greater than 1. The problem we have here is that we cannot "pair off" the quotients like we could when p was even. If we tried to pair off the quotients, we would end up with one quotient left over. To rectify this, we are going to put the first three quotients into one set. By doing this, we are going to have a set of three quotients followed by pairs of quotients.

To start, we will make the weights of each quotient in λ match each quotient's weight in α, which we know will take at most p-2 steps, as stated earlier. Then, we will put the first three quotients into a set and pair off the rest of the quotients. All of the pairs of quotients will behave just like we described in section 2, so we know each of them have the bound of $2\log_2(w) + 2$. Let us focus on the set of three.

To connect the first three quotients in λ to the three in α, we are going to take one step in λ to move the weight from the first quotient into the second quotient. This will cause us to have a set like (ø, λ′(1), λ(2)), where λ′(1) is any partition obtained by combining the conjugate of λ(0) with λ(1). We can do the same thing to the three quotients in α, making (ø, α′(1), α(2)). Now we can ignore the first quotient, making the set of three quotients into a pair, which will also follow our formula. So, when we have a set of three quotients, we can connect them in $(2\log_2 w + 2) + 2$ steps, one extra for each movement of the first quotient.

By doing this process, we can see that we add two extra steps to our formula when p is odd. Therefore, our upper bound for the diameter will be $2\log_2 w + p + 2$.

Theorem (4): For any odd integer p, two partitions of weight w are at most $2\log_2 w + p + 2$ edges apart.

Section 5: General p

By combining Theorem (3) and Theorem (4), we can see that our Theorem (1) result follows trivially for any p value.

Conclusion

While we have found a good bound on the efficient path between two partitions, there are several ways in which we might be able to improve upon our result. For example, when we are smoothing out the weights of the quotients, during steps in which we are moving weights to the right, we could be taking pairs that are already weight-equivalent and running them through our algorithm. This way, by the time we have finished smoothing out the weights, some of the quotients will already be the same between λ and α. This could lead to losing several steps in any case where the weights are spread out across the partition, as opposed to being entirely inside one quotient as we considered in our worst case. There also may be a different, more clever way in which we can combine two partitions during linking such that we save a few steps. We merely did the uncreative method of pushing one partition onto the other, both because it is a method that will work with any two partitions and because it is simple. Also, it is clear that quotient pairs with smaller weights will complete the algorithm faster than larger quotient pairs. Thus, for two particular partitions, the bound generated by our algorithm may be significantly smaller than the general bound of $2\log_2 w + p + 2$.

There are other aspects of the graph that we may be interested in besides the diameter. For instance, one might ask what vertices have smallest or largest degree in the graph, or what the average degree of the graph is. There are also measures of connectivity that could be considered. Finally, we have only considered Rouquier blocks, but there are other blocks that could be considered. However, the combinatorial rules for constructing edges in these blocks are not well understood.

Works Cited

[1] Fayers, Matthew. "Irreducible Specht modules for Hecke algebras of type A." *Advances in Mathematics*. **193** (2005), 438-452.

[2] Kerber, Adalbert and Gordon James. *The Representation Theory of the Symmetric Group*. Reading, MA: Addison-Wesley Publishing Company, Inc., 1981. Print.

[3] Sagan, Bruce. *The Symmetric Group: Representations, Combinatorial Algorithms, and Symmetric Functions*. New York: Springer, 2001. Print.

[4] Turner, Will. *Rock Blocks*. Providence, RI: American Mathematical Society, 2009. Print.