THE $\Gamma_0$ GRAPH OF A $P$-REGULAR PARTITION

A Thesis

Presented to

The Graduate Faculty of The University of Akron

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

Corey Francis Lyons

May, 2010
THE $\Gamma_0$ GRAPH OF A $P$-REGULAR PARTITION

Corey Francis Lyons

Thesis

Approved: 

Advisor
Dr. James P. Cossey

Faculty Reader
Dr. Stuart Clary

Faculty Reader
Dr. Jeffrey Riedl

Dean of the College
Dr. Chand Midha

Dean of the Graduate School
Dr. George Newkome

Date

Department Chair
Dr. Joseph Wilder

ii
ABSTRACT

We investigate the $p$-regularization function from the set $\Gamma_0(n)$ to the set of $p$-regular partitions of $n$. This function induces a natural directed graph for a given $p$-regular partition $\lambda$, called the $\Gamma_0$ graph of $\lambda$. We demonstrate how MATLAB can be used to help create the $\Gamma_0(n)$ graph and develop examples with complicated structure. Then we narrow our view to one-ladder partitions to see how the $\Gamma_0$ graph can be arbitrarily complicated.
ACKNOWLEDGEMENTS

I would first like to thank Professor J.P. Cossey for all of his help and patience throughout my graduate studies. The time and effort J.P. spent helping me was enormous. I would also like to thank all of my professors at the University of Akron for all of their assistance.

I would like to thank Professor Jeffrey Riedl and Professor Stuart Clary for their useful input and productive discussions, with a special thanks to Dr. Riedl, for his individual attention he gave me during the three classes I’ve taken from him.

Of course, I am grateful to my parents, John and Liz Lyons, and all my siblings, for all of their support.

I would like to thank all of my friends for being patient. Finally, I would like to thank the good folks at spell check, without them this thesis would be incomprehensible.
TABLE OF CONTENTS

LIST OF FIGURES ................................................................. vi

CHAPTER

I. INTRODUCTION ................................................................. 1

II. DEFINITIONS AND EXAMPLES ........................................... 3

III. MATLAB ................................................................. 20

IV. MAIN RESULTS ............................................................. 29

BIBLIOGRAPHY ................................................................. 38
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>$\Gamma_0(10, C)$ and $P(10, C)$</td>
<td>16</td>
</tr>
<tr>
<td>2.2</td>
<td>Example of 5-regularization</td>
<td>19</td>
</tr>
<tr>
<td>3.1</td>
<td>$\Gamma_0$ graph of $\lambda$</td>
<td>27</td>
</tr>
<tr>
<td>4.1</td>
<td>A non-$p$-regular section of $\lambda$ before one step of $p$-regularization on the abacus</td>
<td>32</td>
</tr>
<tr>
<td>4.2</td>
<td>After one step of the $p$-regularization of Figure 4.1 on the abacus</td>
<td>33</td>
</tr>
<tr>
<td>4.3</td>
<td>3-regularization of $\lambda = (7^36^25^243^22^31)$</td>
<td>34</td>
</tr>
<tr>
<td>4.4</td>
<td>The $\Gamma_0$ graph of Figure 4.3</td>
<td>35</td>
</tr>
<tr>
<td>4.5</td>
<td>Abacus construction</td>
<td>36</td>
</tr>
<tr>
<td>4.6</td>
<td>$\Gamma_0$ graph for $\lambda$</td>
<td>37</td>
</tr>
</tbody>
</table>
CHAPTER I
INTRODUCTION

We define a partition of $n \in \mathbb{N}$ to be a non-increasing, sequence of positive integers $\lambda = (\lambda_1, \lambda_2, ..., \lambda_r)$ such that $\sum_{i=1}^{r} \lambda_i = n$ [1]. A partition is said to be $p$-regular in case none of its parts is used $p$ or more times. The set of all $p$-regular partitions has significance in representation theory. In this thesis we shall discuss another set of partitions of $n$, called $\Gamma_0(n)$, sometimes denoted $\Gamma_0$ when $n$ is known, that is connected to deep problems in the representation theory of the symmetric group [2]. In particular $\Gamma_0$ is relevant to the study of the Alperin weight conjecture for $S_n$ [2].

It is known that the set of all the $p$-regular partitions of $n$ has the same cardinality as the set $\Gamma_0(n)$ [2]. There is a natural mapping from the set of all partitions of $n$ to the set of all $p$-regular partitions of $n$ known as $p$-regularization. We would hope this function would be a natural bijection from $\Gamma_0(n)$ to the $p$-regular partitions of $n$, but unfortunately this is not a bijection. Furthermore there is no known natural bijection.

We study the directed graph of all partitions in $\Gamma_0$ that $p$-regularize to a given partition $\mu$. We call this the $\Gamma_0$ graph of $\mu$. We also say that $\lambda$ semi-$p$-regularizes to $\sigma$ if $\sigma$ is an intermediate step between $\lambda$ and its image under the map of $p$-regularization. Given two nodes $\lambda$ and $\sigma$ on the graph we write $\lambda \looparrowright \sigma$ if node $\lambda$ semi-$p$-regularizes to
\(\sigma\), where nodes \(\lambda\) and \(\sigma\) are partitions that \(p\)-regularize to \(\mu\). We will later show that, given integers \(n, m > 0\), there exists a \(p\)-regular partition \(\mu\) such that the \(\Gamma_0\) graph of \(\mu\) is “arbitrarily complicated” with at least \(nm\) nodes on it. We will use an object called an abacus of a partition, which is another way to view a partition, to help us achieve our goal. We also narrow our sights to a class of partitions called one-ladder partitions to make our arbitrarily complicated graph look like lattice points on the directed graph \([n] \times [m] \subseteq \mathbb{Z}^2\) from the point \((0,0)\) to \((m,n)\), where \([m]\) denotes \(\{0,1,\ldots,m\}\), and similarly for \([n]\). In this graph, each point is connected to the points immediately to the right or above the point by a directed edge.
CHAPTER II
DEFINITIONS AND EXAMPLES

Definition 2.1. A partition \( \lambda \) of a positive integer \( n \) is a non-increasing sequence of positive integers \( \lambda_1, \lambda_2, \lambda_3, ..., \lambda_r \) such that

\[
\sum_{i=1}^{r} \lambda_i = n.
\]

We call \( \lambda_1, \lambda_2, ... \) the parts of \( \lambda \), and we will call \( n \) the size of \( \lambda \). We will say that 0 is the unique partition of 0. The number of parts \( \lambda \) has is called the length of \( \lambda \) and denoted by \( l(\lambda) \), in this case \( l(\lambda) = r \). For example, with \( n = 42 \) one partition is \( \lambda = (8, 8, 7, 7, 7, 3, 2) \). When the parts of \( \lambda \) are understood, we may drop the commas and write \( \lambda = (8877732) \); alternatively \( \lambda \) can be written as \( \lambda = (8^27^33^2) \).

The irreducible representations of the symmetric group \( S_n \) are indexed by partitions (see Chapter 2 of [3]). Also conjugacy classes of \( S_n \) are determined completely by the cycle types of the elements, which are partitions. These are two of many reasons to study partitions. For more see Kevin Kreighbaum’s thesis [4].

We define \( p(n) \) to be the number of distinct partitions of an integer \( n \). Surprisingly there is no known simple closed form for \( p(n) \) but we do know [1] the generating function for \( p(n) \), denoted \( g(x) \):
\[ g(x) = \prod_{k=1}^{\infty} \frac{1}{1 - x^k}. \]  

Let us take another look at the above equation to see why \( g(x) \) is in fact the generating function for \( p(n) \).

\[
g(x) = \prod_{k=1}^{\infty} \frac{1}{1 - x^k} = \frac{1}{1 - x} \cdot \frac{1}{1 - x^2} \cdot \frac{1}{1 - x^3} \cdots \]

Looking at this expansion, we can see that in order to find the all the partitions of 3 we would just have to find all the ways to get an \( x^3 \) term. For example we could choose: \( (x^3)^1, (x^2)^1(x^1)^1, (x^1)^3 \). This corresponds respectively to choosing 3 once, choosing 2 once and 1 once, and choosing 1 three times. These correspond to all of the possible partitions of 3.

When dealing with partitions of an integer \( n \), it is often helpful to classify them with respect to a prime \( p \). A characteristic that is commonly used to classify partitions is \( p\)-regularity. There are two definitions of \( p\)-regular that are completely different.

**Definition 2.2.** A partition \( \lambda \) is \( p\)-regular of type one if no part of \( \lambda \) is divisible by \( p \).

This is completely different from \( p\)-regular of type two.
Definition 2.3. A partition $\lambda$ is said to be $p$-regular of type two if no part of $\lambda$ is used $p$ or more times.

Notice that a partition can be $p$-regular of the type one but not of the second type, and vice versa. For example, the partition $\lambda = (6531)$ is not 3-regular of type one because $\lambda_1$ and $\lambda_3$ are divisible by 3, but it is 3-regular of type two because no part of $\lambda$ is used 3 or more times. On the other hand, the partition $\lambda = (544441)$ is 3-regular of type one since no part of $\lambda$ is divisible by 3, but it is not 3-regular of type two because $\lambda_2 = \lambda_3 = \lambda_4 = \lambda_5 = 4$. These definitions create two separate sets of partitions, but we can show that these sets are the same size by examining a generating function in different ways. The generating function for each type of $p$-regular partition is:

$$g(x) = \prod_{k=1}^{\infty} \frac{1 - x^{kp}}{1 - x^{k}}$$

(2.0.2)

Expanding this product yields:

$$g(x) = \prod_{k=1}^{\infty} \frac{1 - x^{kp}}{1 - x^{k}}$$

$$= \left( \frac{1 - x^{p}}{1 - x}(1 - x^2) \cdots (1 - x^p) \right) \left( \frac{1 - x^{2p}}{1 - x^{2p-1}}(1 - x^{2p+2}) \cdots (1 - x^{2p}) \right) \cdots$$

$$= \left( \frac{1}{1 - x} \right)(1 - x^2) \cdots \left( \frac{1}{1 - x^{p-1}} \right)(1 - x^{p+1}) \cdots \left( \frac{1}{1 - x^{2p-1}} \right)(1 - x^{2p+1}) \cdots$$

$$= (1 + x^1 + (x^1)^2 + \ldots) \cdots (1 + (x^{p-1})^1 + (x^{p-1})^2 + \ldots)$$

$$\cdots (1 + (x^{2p-1})^1 + (x^{2p-1})^2 + \ldots) \cdots$$
With this expansion, we can see that this generating function counts the number of partitions that are $p$-regular of type one. No factor has a power of $x$ whose exponent is a multiple of $p$, hence no part of $\lambda$ is divisible by $p$. Expanding this product in another way we can see that it counts the number of partitions that are $p$-regular partitions of the second type. For instance, if we examine the first two factors in our product, we see that:

$$g(x) = \prod_{k=1}^{\infty} \frac{1-x^{pk}}{1-x^k}$$

$$= \left(\frac{1-x^p}{1-x}\right)\left(\frac{1-x^{2p}}{1-x^2}\right)\cdots$$

$$= (1 + (x^1)^1 + (x^1)^2 + \ldots)(1 - x^p)(1 + (x^2)^1 + \ldots)(1 - x^{2p})\cdots$$

$$= (1 + (x^1)^1 + (x^1)^2 + \ldots + (x^1)^{p-1})(1 + (x^2)^1 + (x^2)^2 + \ldots + (x^2)^{p-1})\cdots$$

Here we see that no part of $\lambda$ will be used more than $p$ times. Thus $g(x)$ also counts the $p$-regular partitions of the second type. Hence these two sets have the same size. From now on when we talk about $p$-regular we will be talking about the second type of $p$-regular. For a more in-depth discussion see James and Kerber [5].

When working with a partition there is a convenient way to visualize it as a diagram. This diagram is called the Young diagram of $\lambda$, denoted $[\lambda]$. Start for instance with a partition $\lambda = (8, 7, 3, 2)$. To make a Young diagram we will have $l(\lambda) = 4$ rows. The first row will have $\lambda_1 = 8$ “boxes” or “nodes”. Our second row will have $\lambda_2 = 7$ boxes and in general our $i$th row will have $\lambda_i$ boxes. Below is the Young diagram for $\lambda = (8732)$. 6
Definition 2.4. The conjugate of the partition \( \lambda \) is the partition \( \lambda' \) whose parts are the lengths of the columns of \( \lambda \).

Intuitively, we are switching rows and columns. Note \( \lambda_i' = |\{j : \lambda_j \geq i\}| \). Using our previous example we have \( \lambda_1' = |\{j : \lambda_j \geq 1\}| = |\{8,7,3,2\}| = 4 \). Similarly we find \( \lambda' = (4,4,3,2,2,2,2,1) \).

We define the node \( x = (i, j) \in [\lambda] \) to be the box in the \( i \)th row and \( j \)th column of \( \lambda \).

We will often abuse notation and write \( (i,j) \in \lambda \).

Definition 2.5. The hook length of a node \( (i, j) \in \lambda \) is the number of nodes directly below \( (i, j) \) and directly to the right of \( (i, j) \) including the node \( (i, j) \).

We denote the hook length of \( x \) by \( h(x) \) and it can be found by the formula

\[
h(x) = h(i,j) = \lambda_j - j + \lambda'_i - i + 1. \tag{2.0.3}
\]
Below is $\lambda$ with the hook lengths filled in:

\[
\begin{array}{ccccccc}
11 & 10 & 8 & 6 & 5 & 4 & 3 & 1 \\
9 & 8 & 6 & 4 & 3 & 2 & 1 \\
4 & 3 & 1 \\
2 & 1 \\
\end{array}
\]

If a node $x = (i, j)$ has hook length $h(i, j) = k$, then we define the rim hook of the node $x$ to be the connected path of $k$ nodes starting from node $(\lambda_j', j)$ to node $(i, \lambda_i)$ going along the right edge of the partition. For example, the rim hook of length 11 of the node $(1, 1)$ is shown below.

\[
\begin{array}{cccccccccc}
11 & & & & & & & & & \\
& & & & & & & & * & * \\
& & & & * & * & * & * & & \\
& & * & * & * & * & & & & \\
& * & * & & & & & & & \\
& * & * & & & & & & & \\
\end{array}
\]

**Definition 2.6.** We define the $p$-core of a partition $\lambda$ to be the partition remaining after all rim hooks of length divisible by $p$ have been successively removed. If $\lambda$ is a partition of $n$ we define the block containing $\lambda$ to be the set of all partitions of $n$ with the same core as $\lambda$.

**Example 2.7.** Given the partition $\lambda = (8, 7, 3, 2)$, we will find the 3-core of $\lambda$ denoted $\tilde{\lambda}$. If the hook length of every node is not divisible by 3, then the 3-core of the partition $\lambda$ is $\lambda$ itself. On the other hand, if this is not the case the core may found by the following sequence of steps. First we remove the 3-hook of length 6 located at node $(1, 4)$:
This leaves us with

Then we can remove the 3-hook of length 9 located at node (1,1).

This yields the partition

which has one more 3-hook at node (2,1). After the final removal, we are left with the 3-core of \( \lambda = (8732) : \)
Clearly, there can be multiple ways to remove the \( p \)-hooks from a Young diagram. A natural question is, will we always finish with the same \( p \)-core? In other words, is finding the \( p \)-core well defined? This question will be answered in the affirmative with the aid of an abacus.

We now discuss another useful way of representing partitions, the abacus. Fix a prime \( p \). We can represent a partition by means of an abacus consisting of beads and runners. We pick an integer \( m \geq l(\lambda) \) and create a set of positive integers \( \beta = \{\beta_1, \beta_2, \ldots, \beta_m\} \) where

\[
\beta_i = \lambda_i + (m - i). \tag{2.0.4}
\]

Each beta number corresponds to a bead on the abacus [6]. We will demonstrate how to make an abacus given the partition \( \lambda = (9, 8, 7, 7, 6, 5, 5, 5, 3, 2) \). We will use \( p \) runners on our abacus. Suppose \( p = 3 \), so that we have 3 runners labeled left to right 0, 1, 2 and suppose that we choose \( m = 12 \geq l(\lambda) = 10 \). Since we made \( m \) to be more than the length of \( \lambda \) we append zeros to the end of the partition to make \( \lambda \) have \( m \) parts, that is \( \lambda = (9, 8, 7, 7, 6, 5, 5, 5, 3, 2, 0, 0) \). The bead corresponding to \( \beta_i \) is placed in the \( \beta_i \) spot if we read our abacus left to right, top to bottom. That is, the upper leftmost spot of the abacus is the zero spot, the spot directly below that is the \( p^{th} \) spot or in this case the third spot. Or alternatively we can express each beta number in terms of 3 in this case. In general if \( \beta_n = k = pi + j \) (where \( 0 \leq j < p \)) then the \( n^{th} \) bead is on the \( j^{th} \) runner and is \( i \) rows down. We compute the beta
Here is the abacus for our partition $\lambda$:

\begin{align*}
\beta_1 &= \lambda_1 + 12 - 1 = 9 + 12 - 1 = 20 = 3(6) + 2 \\
\beta_2 &= \lambda_2 + 12 - 2 = 8 + 12 - 2 = 18 = 3(6) + 0 \\
\beta_3 &= \lambda_3 + 12 - 3 = 7 + 12 - 3 = 16 = 3(5) + 1 \\
\beta_4 &= \lambda_4 + 12 - 4 = 7 + 12 - 4 = 15 = 3(5) + 0 \\
\beta_5 &= \lambda_5 + 12 - 5 = 6 + 12 - 5 = 13 = 3(4) + 1 \\
\beta_6 &= \lambda_6 + 12 - 6 = 5 + 12 - 6 = 11 = 3(3) + 2 \\
\beta_7 &= \lambda_7 + 12 - 7 = 5 + 12 - 7 = 10 = 3(3) + 1 \\
\beta_8 &= \lambda_8 + 12 - 8 = 5 + 12 - 8 = 9 = 3(3) + 0 \\
\beta_9 &= \lambda_9 + 12 - 9 = 3 + 12 - 9 = 6 = 3(2) + 2 \\
\beta_{10} &= \lambda_{10} + 12 - 10 = 2 + 12 - 10 = 4 = 3(1) + 1 \\
\beta_{11} &= \lambda_{11} + 12 - 11 = 0 + 12 - 11 = 1 = 3(0) + 1 \\
\beta_{12} &= \lambda_{12} + 12 - 12 = 0 + 12 - 12 = 0 = 3(0) + 0.
\end{align*}
We point out that the definition of the beta numbers given here is equivalent to the definition given on page 30 of [4] when $m = l(\lambda)$.

A more practical way to make an abacus is placing the beads in reverse order. That is bead $i$ must have $\lambda_i$ empty spots before it. The 12th bead must have zero empty spots before it since $\lambda_{12} = 0$. Likewise the 11th bead must have zero empty spots before it, since $\lambda_{11} = 0$ and so forth, until the first bead, which has nine empty spots before it. These two ways of making an abacus are not obviously equivalent. Let us take another look at the beta numbers to see why these two methods of creating an abacus yield the same abacus. Suppose we wanted to place the first bead before any of the other beads. We know we need to have some empty spots and some beads before it. More specifically we need to have $\lambda_1$ empty spots before it and $m - 1$ beads before it. Recall that $\beta_i = \lambda_i + (m - i)$. Looking at the formula for $\beta_i$ we can see that the beta numbers are really counting how many beads and empty spots come before each bead. Thus the two methods are really equivalent and must necessarily yield the same abacus.
If we look at the runner labeled 0 as if it were an abacus in itself we would get the partition \( q_0 = (2211) \), which is called the *quotient* of \( \lambda \) on runner zero. Likewise we have \( q_1 = (111) \) and \( q_2 = (53) \).

**Definition 2.8.** The weight, \( w(q_i) \), of a quotient \( q_i \) is the size of the partition \( q_i \).

For example the weight of \( q_0 \) is \( 6 = 2 + 2 + 1 + 1 \). Also notice \( w(q_1) = 3 \) and \( w(q_2) = 8 \).

We say in this case the weight of \( \lambda \) is \( w(q_0) + w(q_1) + w(q_2) = 6 + 3 + 8 = 17 \). Notice it is no coincidence that \( |\lambda| = 57 \) and if \( C \) is the core of \( \lambda \) then \( |C| = 6 = 57 - 17 \times 3 \).

This follows because, as we will soon see, each time we move a bead up \( k \) spots on a runner, we are reducing the weight of \( \lambda \) by \( k \) and the size of \( \lambda \) by \( 3k \).

Another way to draw the abacus of a partition is to use the Young diagram and the “road map” technique. To do this, we are going to use the border of a Young diagram as a road map for the abacus. Going north on the border of the Young diagram is equivalent to placing a bead on the abacus and going east is equivalent to an empty space. We start at the bottom left of our Young diagram and work our way to the top right. The bottom of the partition corresponds to the top of the abacus, so we start placing beads on the top of the abacus and work our way right and down, just like a book. Every time we move right (east) we put an empty space on our abacus. Every time we move up (north) we put a bead on our abacus. Notice adding extra beads to the beginning of the abacus really just cycles the abacus without altering the corresponding partition. This corresponds to increasing the value of \( m \).
The question of whether the $p$-core is well defined reduces to whether the order we remove the $p$-hooks changes the $p$-core. When we remove a $p$-hook of length $kp$, the road map changes in exactly two places that are precisely $kp$ steps apart. Specifically, the first change will be a north movement instead of an east movement at the first node we remove, and the second change will be a move east instead of north at the last node we remove. On the abacus, this is equivalent to changing an empty space to a bead on one runner and changing a bead to an empty space $kp$ spaces later. Since our abacus has $p$ runners, this means the beads we are changing are on the same runner and are $k$ vertical spaces away from each other. Thus, removing a $p$-hook of length $kp$ is equivalent to shifting a bead up on a runner $k$ places. Here we remove a rim 6-hook which is equivalent to one bead up two places.

Removing a 3–hook of length 6
Therefore, by removing every $p$-hook we simply shift every bead as far up as it can travel on the corresponding runner. Hence, the order in which the $p$-hooks are removed does not matter. Thus the $p$-core is well defined, since it is equivalent to moving all of the beads up as far as possible on the abacus. The following is our resulting Young diagram and abacus.

When looking at the abacus of $\lambda$ without any extra beads, the runner that has the fewest beads on it is said to be $\lambda$'s first shortest runner. If multiple runners have the least number of beads we choose the first when reading left to right. This runner is sometimes called the $\Gamma_0$ runner.

**Definition 2.10.** Let $\lambda$ be a partition of $n$. We say that $\lambda$ is in $\Gamma_0(n)$ if the first shortest runner on the abacus display of $\lambda$ has empty quotient. If $\lambda$ has core $C$ we say that $\lambda$ belongs to the set $\Gamma_0(n, C)$.

**Example 2.11.** Notice the first shortest runner, runner one, on $\lambda$ and $\mu$ both have empty quotients. Thus $\lambda, \mu \in \Gamma_0(43, C')$.  

15
Interestingly the cardinality of $\Gamma_0(n)$ is the same as the number of $p$-regular partitions of $n$. Moreover, this can be shown “one core at a time”.

**Theorem 2.12.** [2] *The cardinality of $\Gamma_0(n, C)$ is the same as the number of $p$-regular partitions of $n$ with core $C$.***

**Definition 2.13.** *Define $P(n, C)$ be the set of all $p$-regular partitions of $n$ that have core $C$.***

**Example 2.14.** Below we see all of the partitions in $\Gamma_0(10, C)$ and $P(10, C)$ where $C = (3, 1)$ and $p = 3$. Note that the intersection is not trivial and neither is contained in the other.

Figure 2.1: $\Gamma_0(10, C)$ and $P(10, C)$
The following definition comes from a paper of Fayers [7]. A procedure called $p$-regularization induces a function $\rho$, from the set of partitions of $n$ to the set of $p$-regular partitions of $n$. In order to define this procedure we must first define the $l$th ladder in $\mathbb{N}^2$ to be the set

$$L_l = \{(i, j) \in \mathbb{N}^2 | i + (j - 1)(p - 1) = l\}$$

(2.0.5)

for any $l \geq 1$. Notice that $(i, j)$ and $(i - (p - 1), j + 1)$ are on the same ladder, and we say that a movement from $(i, j)$ to $(i - (p - 1), j + 1)$ is a step on the ladder. Given a non-$p$-regular partition $\lambda$, we $p$-regularize $\lambda$ by moving all the nodes of $\lambda$ as far up as they can go on their respective ladders such that the resulting figure is still a partition. This procedure yields a $p$-regular partition. This new partition is said to be the $p$-regularization of $\lambda$. When any of the intermediate steps yields a partition we call the intermediate step a semi-$p$-regularization of $\lambda$. Also note that any intermediate step may or may not be a partitions, some non-partitions may be completely disconnected.

**Example 2.15.** Let $p = 3$:

![Diagram](image)
This first move results in a new partition so this is an example of a semi-3-regularization.

However, the resulting partition is not (yet) 3-regular.

![Diagram](image1)

Notice this move does not result in a partition. This is a non-example of semi-$p$-regularizing.

![Diagram](image2)

Here the node labelled * has been moved as far up as possible on its ladder. To finish the 3-regularization, we need to move ** as far as possible up its ladder.
Notice that we now have a 3-regular partition.

**Example 2.16.** We also give an example of $p$-regularization when $p=5$ with one intermediate step.

**Figure 2.2:** Example of 5-regularization
Definition 3.1. For a $p$-regular partition $\mu$ of $n$, with core $C$, we define the $\Gamma_0$-graph of $\mu$ to be the directed graph with nodes $\{ \lambda \in \Gamma_0(n, C) | \lambda$ $p$-regularizes to $\mu \} \cup \{ \mu \}$, with directed edge $\lambda_1 \mapsto \lambda_2$ implying $\lambda_1$ semi-$p$-regularizes to $\lambda_2$.

In order to understand the $p$-regularization map better we wrote a program in MATLAB to give us many examples. Our end goal was to write a program that would give us all of the $\Gamma_0$ graphs of the $p$-regular partitions of a given weight and core. In order for us to work with partitions, abacuses, and Young diagrams in MATLAB we needed a way to enter the information into MATLAB in an organized way. To accomplish this, we used vectors for partitions. We used matrices as Young diagrams with hook lengths. Since Young diagrams are rarely rectangular we used zeros as non-boxes in the Young diagram. We also used matrices to represent the abacus. For instance if we had 3 runners on the abacus we now had 3 columns in our matrix for the abacus. We used ones for beads and zeros for empty.

First we made a few very simple programs, among them a program that yields the conjugate of a partition and a program whose input is a partition and whose output is the corresponding Young diagram with the hook lengths. For example, given the partition $\lambda = (8, 7, 3, 2)$ one program consisted of a nested for-loop that
went through the rows and columns of a matrix that had size $8 \times 4$ and computed all the hook lengths. If the Young diagram did not have a box, we told the program to fill in the entry with a zero. So here is the matrix our program would give us for $\lambda$.

$$
\begin{pmatrix}
11 & 10 & 8 & 6 & 5 & 4 & 3 & 1 \\
9 & 8 & 6 & 4 & 3 & 2 & 1 & 0 \\
4 & 3 & 1 & 0 & 0 & 0 & 0 & 0 \\
2 & 1 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
$$

In order to achieve our end goal we needed an index of all the partitions in a given block with a given weight. To accomplish this we needed a list of all the partitions for any $n$ within reason. For integers $n$ from 1 to 20 we wanted our list to include all of the partitions of $n$. Manually listing all the partitions of 20 would be a very time consuming task since there are 627 partitions of 20. So we needed a program to generate a list of every partition for every $n$ from 1 to 20.

**Remark 3.2.** Every partition of $n$ can be found by looking at the Young diagram of a partition of $n - 1$ and adjoining a node in one of the rows or by adjoining a node to a new row.

We wrote a program that, given all of the partitions of $n$, would generate all of the partitions of $n + 1$. We accomplished this by writing a nested for-loop with nested if statements. The first for-loop would run through our list of partitions of $n$. Our nested for-loop would then go down all of the rows of the Young diagram of the
partitions of \( n \) and add a box to the end of each of the rows, one at a time, thus creating a possible partition of \( n \). Obviously this will not always give us a partition. This issue was easily resolved with an if statement, checking whether this process gave us an actual partition of \( n + 1 \). If this process did indeed yield a partition, it was stored into a matrix in an organized way, if it was not already in the list.

Before we discuss how to generate all of the partitions in a given block with a given weight, we first take a necessary detour. Let \( b(w, p) \) be the number of partitions in a block \( C \) with weight \( w \) on \( p \) runners. Then we have an equation for \( b(w, p) \):

\[
 b(w, p) = \sum_{p\text{-tuples}} p(w_0)p(w_1) \cdot ... \cdot p(w_{p-1}) \quad (3.0.1)
\]

where \( p\text{-tuples} = \{ (w_0, w_1, ..., w_{p-1}) \mid w_0 + w_1 + ... + w_{p-1} = w \} \). Notice that this counting function \( b \) is completely independent of the core. Since the quotient on runner \( i \) can be any partition of \( w_i \) then it is clear that this is the counting function for any block with weight \( w \) on \( p \) runners. Looking at the equation for \( b(w, p) \) shines light on how to write the program to generate all of the partitions in a given block. We wrote this program for \( p = 3 \) but the same concept can be extended for larger \( p \). Essentially the program takes a given weight and a given core and looks at each runner, individually assigning weights in a organized fashion. It has nested for-loops that first go through the possible weights each runner can have. For instance, given some core and \( w = 5 \) it would first assign the following weights, \( (w_0, w_1, w_2) = (0, 0, 5) \). Then it would go through all the possible partitions for each individual weight one at
a time and put these partitions on each runner such that it preserved the given core. This iteration would create $p(w_0)p(w_1)p(w_2) = p(0)p(0)p(5) = 7$ partitions. The next iteration of weights would be $(0, 1, 4)$ and this would create $p(0)p(1)p(4) = 5$ new partitions in our block.

Next we needed a program that would $p$-regularize any given partition. The first thing our program did was use our previous program to create the “hook matrix”. When $p$-regularizing by hand, we usually look for a non-$p$-regular part and shift each block to the right one and up $p - 1$. Alternatively we could look at each node in the Young diagram and ask if this node could move to the right one and up $p - 1$. This is more time consuming, but for a computer it takes a moment. To say this more formally we wrote our program with a nested for-loop that would go through every row and every column and check if any entry in the matrix could move to the right $1$ and up $p - 1$ spots. So for any given entry $(i, j)$, for $i \geq p - 1$, the program would test if entry $(i - (p - 1), j + 1)$ was a zero, and if so it would change entry $(i, j)$ into a zero and change entry $(i - (p - 1), j + 1)$ into a one. One iteration of this is usually not enough. So at the end of the for-loop we had a small program that tested if the partition was now $p$-regular and if it was an actual partition. If it failed either of the two previous tests it would start at the beginning of the nested for-loop again. Finally we have another program that takes a “hook” matrix and converts it into a partition that is independent of hook lengths.
Example 3.3. To 3-regularize $\lambda = (332222)$ with the following hook matrix,

$$
\begin{pmatrix}
8 & 7 & 2 \\
7 & 6 & 1 \\
5 & 4 & 0 \\
4 & 3 & 0 \\
3 & 2 & 0 \\
2 & 1 & 0
\end{pmatrix},
$$

our nested for-loop would first find the entry $(5, 2)$, since entry $(5 - (3 - 1), 2 + 1) = (3, 3)$ is zero. It would then change entry $(3, 3)$ into a one, then change entry $(5, 2)$ into a zero. After this the program would continue finding the next entry that could move, which is entry $(6, 2)$ since entry $(6 - (3 - 1), 2 + 1) = (4, 3)$ is zero, and change the corresponding entries to zero and one. This yields our new matrix:

$$
\begin{pmatrix}
8 & 7 & 2 \\
7 & 6 & 1 \\
5 & 4 & 1 \\
4 & 3 & 1 \\
3 & 0 & 0 \\
2 & 0 & 0
\end{pmatrix}.
$$

At the end of the for-loop the program would check to see if the result was a partition
and whether it was $p$-regular. Since it still fails to be $p$-regular it would start the for-loop again. Notice that the program does not depend on any of the hook lengths. After a second iteration of this it would yield the following matrix.

\[
\begin{pmatrix}
8 & 7 & 2 & 1 \\
7 & 6 & 1 & 1 \\
5 & 4 & 0 & 0 \\
4 & 3 & 0 & 0 \\
3 & 0 & 0 & 0 \\
2 & 0 & 0 & 0
\end{pmatrix}
\]

Finally the program converts our matrix back into a partition using a process that is independent of hook lengths, i.e., when determining the corresponding partition it looks at whether an entry is zero rather than looking at the hook length and working backwards.

So far we have a program that can list every partition in a given block with a given weight. Now all we have to do is find a way to sort the partitions. We really care only about the partitions in the symmetric difference. That is, we want a way to sort our block into two sets: the first set is the set of partitions that are in $\Gamma_0$ but not $p$-regular and the second set is the set of partitions that are $p$-regular but not in $\Gamma_0$. Since we are looking at the $p$-regularization map we are not interested in the partitions that are $p$-regular and in $\Gamma_0$ since $p$-regularization will be the identity.
map for these. So we insert a test that can tell us whether a partition is $p$-regular or whether it is in $\Gamma_0$ whenever our block program comes up with a new partition. So now our program generates two matrices, each of which contains a numbered list of our two sets of partitions. Now our program $p$-regularizes all of our non-$p$-regular partitions and looks at what partitions they $p$-regularize into, meanwhile keeping a list of all the partitions that get sent to another partition. Finally our program prints out our list of partitions in $\Gamma_0$ that get $p$-regularized to other partitions. One thing this program falls short of doing is telling us the directed graph of what partitions get sent to what. That is, our program prints out a list of partitions $\{\mu, \lambda, \sigma\}$ that all get $p$-regularized to $\gamma$, but we do not know what the directed graph looks like. We cannot say that $\mu \mapsto \lambda \mapsto \sigma \mapsto \gamma$ or anything along those lines. So we need to analyze each of the outputs to determine the directed graphs.

**Example 3.4.** Given $w = 15$ and the core $\mu = (642211)$ and

$$\lambda = (12, 10, 8, 7, 6, 5, 4, 3, 2, 2, 1, 1),$$

here is the Young digram and abacus of the core $\mu$ of $\lambda$:

```
11 8 5 4 2 1
8 5 2 1
5 2
4 1
2 1
```

We have the following $\Gamma_0$ graph of $\lambda$.
Let us see why $\lambda_1 \mapsto \lambda_2$. They differ in only two places, namely $5 = (\lambda_1)_4 \neq (\lambda_2)_4 = 6$ and $2 = (\lambda_1)_{12} \neq (\lambda_2)_{12} = 1$ and notice the nodes $(12, 2) \in [\lambda_1]$ and $(4, 6) \in [\lambda_2]$ are both on the same ladder, namely $L_{14}$, where $L_{14}$ is defined in equation 2.0.5. Thus $\lambda_1$ clearly semi-$p$-regularizes to $\lambda_2$. 

$$\lambda_1 = (12, 10, 7, 5, 4, 4, 3, 3, 2, 2, 2, 1, 1, 1, 1, 1, 1)$$
$$\lambda_2 = (12, 10, 7, 6, 4, 4, 3, 3, 2, 2, 2, 1, 1, 1, 1, 1, 1)$$
$$\lambda_3 = (12, 10, 8, 5, 4, 4, 3, 3, 2, 2, 2, 1, 1, 1, 1, 1, 1)$$
$$\lambda_4 = (12, 10, 8, 6, 4, 4, 3, 3, 2, 2, 2, 1, 1, 1, 1, 1, 1)$$
$$\lambda_5 = (12, 10, 8, 5, 4, 4, 3, 3, 2, 2, 2, 2, 2, 2)$$
$$\lambda = (12, 10, 8, 7, 6, 5, 4, 3, 2, 2, 1, 1).$$
Let us take a closer look at the relationship between $\lambda_2$ and $\lambda_3$. Clearly $\lambda_3$ does not semi-$p$-regularize to $\lambda_2$ since $l(\lambda_2) > l(\lambda_3)$. To see why $\lambda_2$ does not map to $\lambda_3$, we notice that node $(4, 6) \in \lambda_2$ and node $(12, 2) \in \lambda_3$ are on $L_{14}$. However node $(12, 2) \in \lambda_3$ is lower on the ladder than $(4, 6) \in \lambda_2$ and in order for $\lambda_2 \mapsto \lambda_3$, node $(4, 6) \in \lambda_2$ must move up the ladder to $(12, 2) \in \lambda_3$, which is ridiculous.
CHAPTER IV

MAIN RESULTS

We saw in the last chapter an example of a complicated $\Gamma_0$ graph. In this chapter we will make precise the notion of an arbitrarily complicated graph and then show that the $\Gamma_0$ graphs can be arbitrarily complicated. We will restrict our view to a better behaved set of partitions, namely the one-ladder partitions.

**Definition 4.1.** A partition is said to be a one-ladder partition if all of its $p$-regularization moves occur on the same ladder.

**Example 4.2.** For $p = 3$, the partition $\lambda = (665544332211)$ is one-ladder, which is easily seen from the Young diagram.
If we take a closer look at $\lambda$ we might notice that there are two ways of $p$-regularizing $\lambda$.

We could take node $(13, 1)$ and move it to spot $(13 - 6(3 - 1), 1 + 6) = (1, 7)$.

Or we can take a different approach, that is move node $(3, 6)$ one step up the ladder to spot $(3 - 1(3 - 1), 6 + 1) = (1, 7)$.

Continuing this route we could then move node $(5, 5)$ one step up the ladder to spot $(3, 6)$. 
Continuing this process we can easily see that we will eventually move 6 boxes along the same ladder, $\mathcal{L}_{13}$.

In order to achieve our main result we need to know how to $p$-regularize on the abacus. We first point out that a non-$p$-regular section of a partition corresponds to $p$ or more consecutive beads on the abacus. Since we are dealing with one-ladder partitions, one can easily see that the non-$p$-regular sections on our abacus will have exactly $p$ consecutive beads. Suppose $\lambda = (\lambda_1^{m_1}, \lambda_2^{m_2}, ..., \lambda_k^{m_k})$, where some $m_i = p$. Now suppose $i < k$. (The case where $i = k$ is similar.) Then let us make an abacus with $l(\lambda)$ beads and focus on a part of the abacus where $\lambda$ is not $p$-regular. Note we are using $l(\lambda)$ beads in order to ensure that the beta numbers are indeed the first column hook lengths. Let us also look at the corresponding Young diagram of $\lambda$ at the non-$p$-regular section.
What happens when we \( p \)-regularize this section of \( \lambda \)? Node \( x \), which corresponds to the bead labeled \( \beta_i \), will move to the right one spot and up \( p - 1 \) spots. This changes exactly two of the first column hook lengths, and thus two of the beta numbers: \( \beta_i \) will be decreased by one and \( \beta_j \) will be increased by one. Decreasing a beta number by one moves a bead to the left one position on the abacus. Likewise if we increase a beta number by one it moves the bead to the right by one position on the abacus. As an aside, we note that this is one way to see that the \( p \)-regularization process is core-preserving, since these beads are exactly \( p \) apart, thus each step in the \( p \)-regularization process does not change how many beads are in each runner. To summarize we see the net result of one step of the \( p \)-regularization process is to shift the first bead in the non-\( p \)-regular section to the left one position and the last bead of the non-\( p \)-regular section to the right one position. We mention in passing that this same method can be extended to all partitions using beads “with multiplicity”, but we will not develop that here.
Example 4.3. With $p = 3$ and $\lambda = (776554433)$, we give the abacus and the $p$-regularization.

Example 4.4. For $\lambda = (7^36^25^243^22^31)$ we see below that $\lambda$ regularizes to $\mu = (87^26^25^243^22^31^2)$ in two different ways.
Notice that $\lambda_{a_2b_3} = \lambda_{b_3a_2}$, which is not a coincidence. In fact we have $\lambda_{a_ib_j} = \lambda_{b_ja_i}$ for $0 \leq i \leq 2$ and $0 \leq j \leq 3$. Notice that $\lambda$ has two non-$p$-regular sections, and the previous discussion shows that the net result of one step in the $p$-regularization process is to move beads on just one of the non-$p$-regular sections. We see then that 3-regularizing the bottom portion of the abacus is completely independent of 3-regularizing the top portion of the abacus. Let us look at the $\Gamma_0$ graph of $\mu$ in Figure 4.4. This is a directed graph where you can only move up or to the right. Moving one step to the right on the graph is equivalent to one step of the 3-regularization, which corresponds to moving beads on the top non-$p$-regular section of the abacus.
Similarly, moving right on the graph is equivalent to one step of the 3-regularization, which corresponds to moving beads on the bottom non-$p$-regular section of the abacus.

Recall the definition of $[n] \times [m]$ from Chapter 1.

**Theorem 4.5.** Given positive integers $n$ and $m$ and a prime $p$ there exists a $p$-regular partition $\mu$ such that the $\Gamma_0$ graph of $\mu$ has a subgraph $[n] \times [m]$.

*Proof.* Given $m, n$, and $p$ we can construct a partition $\lambda$ such that the abacus of $\lambda$ has the following form:
Note that by the previous discussion of $p$-regularization of a one-ladder partition, we see that $p$-regularizing the top portion of the abacus of $\lambda$ is completely independent of $p$-regularizing the lower portion of the abacus. When we perform one step of the $p$-regularization on the top portion of the abacus the bead at the end of the non-$p$-regular section shifts to the beginning of the next row, forming a new non-$p$-regular section in the next row. Similarly, when we perform one step of the $p$-regularization on the bottom portion of the abacus the bead at the beginning of the non-$p$-regular section shifts to the end of the previous row, forming a new non-$p$-regular section in the previous row. Thus by construction the $\Gamma_0$ graph of $\lambda$ will contain a subgraph of the form in Figure 4.6.
As this abacus stands it is not the case that the first shortest runner of \( \lambda \) has empty quotient. We can construct a new partition \( \hat{\lambda} \), for which the parts of \( \lambda \) are a subset of the parts of \( \hat{\lambda} \). We accomplish this by adding enough beads to the bottom of the abacus on all of the runners except runner one, thus making runner one be the shortest and have empty quotient. Therefore \( \hat{\lambda} \) is in \( \Gamma_0(s) \) for some integer \( s \). Moreover, the \( \Gamma_0 \) graph of the \( p \)-regularization of \( \hat{\lambda} \) has a subgraph with the same structure as the graph in Figure 4.6.

Figure 4.6: \( \Gamma_0 \) graph for \( \lambda \)
BIBLIOGRAPHY


