ESTIMATING LANDAU'S FUNCTION

BY:

RICHARD GUNTER

SENIOR THESIS

Submitted in partial satisfaction of the requirements for the degree of

BACHELOR OF SCIENCE
in
MATHEMATICS

in the

COLLEGE OF LETTERS AND SCIENCE
of the
UNIVERSITY OF CALIFORNIA, DAVIS

Approved:

_____
José Simental Rodríguez

Contents

1. Introduction

The main focus of this paper is in studying Landau's Function $g : \mathbb{N} \to \mathbb{N}$ that gives the maximum order of an element of the symmetric group S_n:

$$g(n) = \max_{\sigma \in S_n} \mathrm{Ord}(\sigma).$$

The main result of this paper is the description of an algorithm for estimating $g(n)$ that is more accurate than the formula $F(n)$ derived in Miller (1987) [2]. We have also written MATLAB programs that provide multiple types of data discussed on the elements of S_n. Furthermore, we have derived a probability distribution over the partitions of n describing the probability of randomly choosing a cycle of a random cycle type in S_n. We use MATLAB to generate and analyze histograms of these probability distributions. Similarly, we use MATLAB to generate and analyze histograms of the orders of elements in S_n. Finally, we prove a relationship between the maximum order of S_n and $G(\ell, n)$ allowing our estimation methods to reach from S_n to a family of wreath-product groups.

1.1. Definitions.

Let us start with definitions that will be important and used throughout this work.

Definition 1.1 (Prime numbers)**.** *Primes are natural numbers $p \in \mathbb{N}$ such that if a natural number n divides p, then $n = p$ or $n = 1$.*

We will denote the set of all primes as $\mathbb{P}$ and $\mathbb{P}_n = \{p_1, p_2, ..., p_n\}$ be the set of the first n primes, with $p_1 = 2$.

Definition 1.2 (Permutations and the Symmetric group)**.** *The symmetric group is the group $(S_n, \circ)$ of bijective mappings $\sigma : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ together with composition of those functions.*

$(S_n, \circ)$ is most commonly denoted S_n for simplicity. The elements of the symmetric group are called permutations. A permutation

$$\sigma = \begin{pmatrix} 1 & 2 & \ldots & n \\ a_1 & a_2 & \ldots & a_n \end{pmatrix}$$

is a bijective mapping $\sigma : \{1, \ldots, n\} \longrightarrow \{1, \ldots, n\}$ given by $\sigma(i) = a_i \ \forall \ i \in \{1, \ldots, n\}$.

Permutations can be more compactly expressed in "one-line notation",

$$\sigma = [a_1 \ a_2 \ \ldots \ a_n]$$

such that we "drop the top row" of the permutation and read the ith component in the list as the output $\sigma(i) = a_i$.

An especially nice and useful class of permutations are the cycles.

Definition 1.3 (Cycles)**.** *A k-cycle $\sigma^{(k)}$ (not to be confused with σ to the k-th power of σ) is a permutation of the form*

$$a_1 \mapsto a_2 \mapsto \cdots \mapsto a_k \mapsto a_1$$

that fixes every other element of $\{1, \ldots, n\}$.

We will denote the k-cycle $\sigma^{(k)}$ by $(a_1\ a_2\ \dots\ a_k)$.

For example, if $\sigma = (123)$ and $\rho = (23)$, $\sigma \circ \rho = (123)(23) = (12)$. As we can see, $\rho(1) = 1$, $\sigma(1) = 2$, $\rho(2) = 3$, $\sigma(3) = 1$, $\rho(3) = 2$, $\sigma(2) = 3$. It is not important to continue to use $\circ$ to denote permutation composition, so for the rest of the paper

$$\sigma\rho$$

will denote the composition of σ and ρ, or product in S_n.

We say that two cycles $(a_1 a_2 \dots a_k)$ and $(b_1 b_2 \dots b_s)$ are *disjoint* if $\{a_1, \dots, a_k\} \cap \{b_1, \dots, b_s\} = \emptyset$. The following important result elaborates further on the importance of cycles.

We say the *order of a permutation* is the smallest positive integer b such that $\sigma(\sigma(\sigma(...\sigma) = \sigma^b = e$. Note that a k-cycle has order precisely k.
b compositions

Theorem 1.4. *Every cycle $\sigma \in S_n$ can be decomposed as a product of disjoint cycles.*

Proof. Let $\sigma \in S_n$ be an arbitrary permutation. If we can write σ as a cycle $(a_1\ a_2\ \dots\ a_n)$, we are done. If not, then we must have $\sigma(a_k) = a_1$ for some $k \neq n$. This is because all permutations are bijective and thus every output has a unique input. We can then "factor out" the cycle $(a_1\ \dots\ a_k)$ and if we can then write the remaining elements in a cycle $(a_i\ \dots\ a_n)$ such that $\sigma = (a_1 \dots\ a_k)(a_i a_{i+1} \dots\ a_n)$ then we are done. If $\sigma(a_j) = a_i$ for $j \neq n \in \{i, \dots\ n\}$ we can again factor another cycle and write $\sigma = (a_1\ \dots\ a_k)(a_i\ \dots\ a_j)(a_{j+1}\ \dots\ a_n)$. We can repeat this at most n times until we exhaust the factorable cycles in σ. The extreme case of this process is when $\sigma = (a_1)(a_2)\dots(a_n) = e$, the permutation that sends $a_i \mapsto a_i\ \forall i \in \{1, \dots, n\}$ □

Two permutations σ and ρ are said to be *conjugate* if there exists a permutation τ such that $\tau\sigma\tau^{-1} = \rho$. It is easy to see that being conjugate is an equivalence relation, and the corresponding equivalence classes are called *conjugacy classes.* These are the sets $\{\rho | \tau\sigma\tau^{-1} = \rho\}$ for some $\tau \in S_n$. The conjugacy classes can be expressed in terms of the cycle decomposition, as expressed by the following notion.

Definition 1.5 (Cycle-type)**.** *Let $\sigma \in S_n$ be a permutation with disjoint cycle decomposition $\sigma = \sigma_1^{(\ell_1)}\sigma_2^{(\ell_2)}\dots\sigma_k^{(\ell_k)}$. We see the cycle length of σ_i is ℓ_i for all i. By rearranging if necessary, we may assume that $\ell_1 \leq \ell_2 \leq \dots \leq \ell_k$. Then we say the* cycle type of σ *is the tuple of non-decreasing values $[\ell_1, \ell_2, \dots, \ell_k]$.*

Theorem 1.6. *Two permutations σ and ρ are conjugate if and only if they have the same cycle type.*

The proof of this theorem will depend on the following lemma.

Lemma 1.7. *Assume $\sigma = (a_1 \cdots a_k)$ is a k-cycle. Then, for any permutation τ we have $\tau\sigma\tau^{-1} = (\tau(a_1) \cdots \tau(a_k))$*

Proof. Let $\sigma, \tau \in S_n$ with $\sigma = (a_1\ \dots\ a_k)$ and τ arbitrary. Then

$$\tau\sigma\tau^{-1}(\tau(a_i)) = \tau\sigma(a_i) = \tau(a_{i+1})$$

for all i except $i = k$ in which case

$$\tau\sigma(a_k) = \tau(a_1).$$

So $\tau\sigma\tau^{-1}$ describes the mapping

$$\tau(a_1) \mapsto \tau(a_2) \mapsto \dots \ \tau(a_k) \mapsto \tau(a_1).$$

Thus $\tau\sigma\tau^{-1} = (\tau(a_1) \cdots \tau(a_k))$. □

Proof of Theorem 1.6. ($\Rightarrow$) By Lemma 1.7, if two permutations are conjugate, they are of the same cycle type.

($\Leftarrow$) If two permutations σ,ρ have the same cycle type, then simply choose a $\tau \in S_n$ such that $\tau(\sigma(i)) = \rho(i) \forall i \in \{1, \dots, n\}$. Then conjugation by τ gives $\tau\sigma\tau^{-1} = (\tau(\sigma(1)) \ \tau(\sigma(2)) \ \dots \ \tau(\sigma(n))) = (\rho(1) \ \rho(2) \ \dots \ \rho(n))$. Thus σ,ρ having equal cycle types is necessary and sufficient for their conjugacy. □

To finish this section, we elaborate on how the cycle-type of a permutation σ completely determines its order. Recall that the "least common multiple" (lcm for short) of a tuple of positive integers is the smallest integer divisible by each integer in the tuple. Equivalently, the lcm is a number divisible by every number in the tuple and that divides every other number also divisible by every number in the tuple.

Lemma 1.8. *Let $\sigma \in S_n$ be a permutation with cycle type $[\ell_1, \ell_2, \dots, \ell_k]$. Then, the order of σ is* $\operatorname{lcm}(\ell_1, \dots, \ell_k)$.

Proof. Let $\sigma = \sigma_1\sigma_2 \cdots \sigma_k$ be a decomposition of σ into disjoint cycles of length $\ell_1, \dots, \ell_k$. Since the cycles are disjoint, they commute. So for every $m > 0$

$$\sigma^m = \sigma_1^m \sigma_2^m \cdots \sigma_k^m$$

Note that $\sigma^m = e$ if and only if $\sigma_j^m = e$ for every $j \in \{1, \dots, k\}$.

First we prove that k-cycles have order k. Let $\tau = (a_1 \ a_2 \ \dots \ a_{k-1} \ a_k)$. Then $\tau(a_i) = a_{i+1}$ except $\tau(a_k) = a_1$. Then $\tau^2(a_i) = \tau(a_{i+1}) = a_{i+2}$ except $\tau^2(a_k) = \tau(a_1) = a_2$. Continue and we see that $\tau^k(a_i) = a_i$ for all $i \in \{1, \dots, k\}$ Thus the order of a k-cycle is k.

Since $\sigma_j^{\ell_j} = e$ for $j \in \{1, \dots, k\}$. Then $\sigma_j^{\ell_j b} = e^b = e$ for any integer $b \in \mathbb{N}$. So the order of σ needs to be the smallest integer b such that it is a multiple of all ℓ_1 through ℓ_k. In other words, the order of σ is $\operatorname{lcm}(\ell_1, \ell_2, \dots, \ell_k)$. □

2. Partitions

Definition 2.1 (Partitions)**.**

A *partition of n* is a tuple

$$\lambda = [\lambda_1, \lambda_2, \dots, \lambda_k]$$

of non-decreasing positive numbers such that

$$\sum_{i=1}^{k} \lambda_i = n$$

We will call the numbers $\lambda_1, \dots, \lambda_k$ the *parts of* λ.

We will also use an alternative equivalent notation for a partition λ:

$$\lambda = \langle 1^{a_1}, 2^{a_2}, \dots n^{a_n} \rangle$$

This notation is read "a_1 1s, a_2 2s, $\dots$ a_n ns."

So for example the partition

$$\lambda = \langle 1^1, 2^4, 3^0, 4^0, 5^3 \rangle$$

means one 1, four 2s, zero 3s, zero 4s, and three 5s. Then λ is a partition of $24 = 1 + 4 \times 2 + 3 \times 5$.

Definition 2.2 (Least Common Multiple of a Partition)**.** *Let λ be a partition. We call the least common multiple of λ the least common multiple of its parts.*

2.1. **The partition-permutation relation.** We will be exploring a deep connection between partitions of n and cycle types in S_n. Often, we will move interchangeably between talking about a cycle-type and the corresponding partition, or vice versa. The key observation is that the cycle type and permutation contain essentially the same information.

To elaborate this, consider the partition

$$n = 1 + \cdots + 1$$

Using non-overlapping parenthesis, we can group together any number of 1s in a sum to create any other partition. For example, to create the partition

$$\lambda = [1, k, (n - k - 1)]$$

construct the following non-overlapping parenthetical grouping:

$$n = 1 + \underbrace{(1 + \cdots + 1)}_{k \text{ ones}} + \underbrace{(1 + \cdots + 1)}_{n-k-1 \text{ ones}}.$$

For an arbitrary partition $\lambda = [\lambda_1, \ldots, \lambda_k]$, construct the grouping

$$n = \underbrace{(1 + \cdots + 1)}_{\lambda_1 \text{ ones}} + \cdots + \underbrace{(1 + \cdots + 1)}_{\lambda_k \text{ ones}} = \lambda_1 + \cdots + \lambda_k.$$

In the same way, we can take the trivial cycle in S_n

$$e = (1)(2)\ldots(n-1)(n)$$

and create any possible cycle type in S_n by joining adjacent 1-cycles into larger cycles. Again using the example partition

$$\lambda = [1, k, (n - k - 1)]$$

we can construct the corresponding permutation

$$\sigma = (1)(2\ldots(k+1))((k+2)\ldots n)$$

and for an arbitrary partition $\lambda = [\lambda_1, \ldots, \lambda_k]$ we construct an arbitrary permutation

$$\sigma = \sigma_{\lambda_1}\sigma_{\lambda_2}\ldots\sigma_{\lambda_k}$$

.

So in summary, we relate partitions and elements of S_n via Definition 1.5.

Corollary 2.3. *If two permutations σ and ρ have the same cycle type, they have the same order. Because σ and ρ have the same cycle type, the* lcm *of their cycle types is the same.*

Definition 2.4. *Let λ be a partition of n. We define the order of λ to be the order of any permutation in S_n that has cycle type λ. In other words, the order of λ is* $\mathrm{lcm}(\lambda)$

Note that the order of λ is nothing but the least common multiple of the parts of λ. The main problem that we will study in this work is the following.

Main Problem. *Given n, find a partition of n with maximal possible order.*

Definition 2.5 (Landau function)**.** *The Landau function of n is defined to be:*

$$g(n) := \max\{\mathrm{Ord}(\lambda) \mid \lambda \textit{ is a partition of } n\}$$

Our first goal is to restrict the partitions we should look at to find the value of $g(n)$. First, we find an expression for the order of a partition in terms of the prime decomposition of its parts. Note: We cite OEIS [6] for the true value of $g(n)$ for comparison to our estimates.

Lemma 2.6. *Let $(\alpha_1, \ldots, \alpha_m) \in \mathbb{Z}^m_{>0}$. Assume that α_i has a prime decomposition*

$$\alpha_i = \prod_{j=1}^{\infty} p_j^{a_{ij}}$$

(note that $a_{ij} = 0$ for $j \gg 0$). Then the lcm *of $\alpha_1, \ldots, \alpha_m$ is given by*

$$\mathrm{lcm}(\alpha_1, \ldots, \alpha_m) = \prod_{j=1}^{\infty} p_j^{\max_i a_{ij}}$$

Proof. We must show that (1) $\prod_{j=1}^{\infty} p_j^{\max_i a_{ij}}$ is divisible by every α_i and (2) that every other number divisible by every α_i is also divisible by $\prod_{j=1}^{\infty} p_j^{\max_i a_{ij}}$.

(1) Every number α_k in the m-tuple divides $\prod_{j=1}^{\infty} p_j^{\max_i a_{ij}}$ because $\max_i a_{ij} \geq a_{kj}$ for any $i, k \in \mathbb{N}$, and we see

$$\frac{\prod_{j=1}^{\infty} p_j^{\max_i a_{ij}}}{\alpha_k} = \prod_{j=1}^{\infty} p_j^{\max_i a_{ij} - a_{kj}} \in \mathbb{N}.$$

(2) Every number divisible by all $\alpha_1, \ldots, \alpha_m$ must also be divisible by $\prod_{j=1}^{\infty} p_j^{\max_i a_{ij}}$. Suppose we have a number β with prime decomposition $\prod_{j=1}^{\infty} p_j^{b_j}$ for some $b_j \in \mathbb{N}$ such that β is divisible by all $\alpha_1, \ldots, \alpha_m$. Then $b_j \geq \max_i a_{ij}$ for all j. If there was some j such that $b_j < \max_i a_{ij}$, then there is a k such that $a_{kj} = \max_i a_{ij}$ and

$$\frac{\beta}{\alpha_k} = \prod_{j=1}^{\infty} p_j^{b_j - a_{kj}} \notin \mathbb{N}$$

.

□

Theorem 2.7. *For any $n > 0$, there exists a partition λ^* whose parts are all are prime powers (or 1) such that $g(n)$ is the order of λ^*.*

Proof. Let $\mu = (\mu_1, \mu_2, \ldots, \mu_k)$ be a partition of n such that

$$\mathrm{lcm}(\mu_1, \ldots, \mu_k) = g(n)$$

Assume there exists an i such that the part μ_i is not a prime power nor 1. Let us take the prime decomposition of μ_i

$$\mu_i = \prod_{j=1}^{m} p_j^{a_{ij}}$$

We can define a new partition μ^* (with non-decreasing rearrangement) to be

$$\mu^* = [\mu_1, \ldots, \mu_{i-1}, p_{i_1}^{a_1} \cdots, p_{i_m}^{a_m}, \mu_{i+1} \ldots, \mu_k]$$

Then, by the Geometric Mean Inequality $\sum_{j=1}^{m} p_{i_j} \leq \prod_{j=1}^{m} p_{i_j}$ we find that

$$\sum_{i \text{ st } \mu_i^* \neq 1}^{\infty} \mu_i^* \leq \sum_{i \text{ st } \mu_i \neq 1}^{\infty} \mu_i$$

Finally, these two partitions μ and μ^* have the same order. This comes simply from the identity $\operatorname{lcm}(\mu) = \prod_{j=1}^{\infty} p_j^{\max_i a_{ij}}$

Since μ^* contains the same parts as μ, save for the inclusion of μ_i's parts, all powers of primes contained in μ are preserved in μ^*. Thus

$$\operatorname{lcm}(\mu) = \prod_{j=1}^{\infty} p_j^{\max_i a_{ij}} = g(n) = \operatorname{lcm}(\mu^*)$$

. □

Let us finish this section with some small examples of the value of $g(n)$ and the partition λ^* that achieves this order. All these values can be found by manual inspection.

n	$g(n)$	λ^*
2	2	[2]
3	3	[3]
4	4	[4]
5	6	[2, 3]
6	6	[1, 2, 3], [6]
7	12	[3, 4]
8	15	[3, 5]
9	20	[4, 5]
10	30	[2, 3, 5]

3. Budgeting methods for estimating $g(n)$

3.1. **The Simple Budgeting Method.** In estimating $g(n)$, it is useful to think of the quantity n as our 'budget' with which to construct a partition whose lcm is as large as possible. Because $g(n)$ is a maximal least common multiple, we need to avoid choosing highly composite parts, as much of our budget is "wasted". This is because highly composite parts of n will have a smaller lcm than more coprime parts of n Theorem 2.7.

For example, let $n = 30$, then the partition $2+4+8+16 = 30$ has $\operatorname{lcm}\{2, 4, 8, 16\} = 16$ whereas we know $g(30) = 4620$ given by the partition $[3, 4, 5, 7]$. We currently do not possess a closed form expression for $g(n)$, so our best tool for studying this

function is direct calculation and estimation. Our first method of estimation is described in [2], we call it the Simple Budgeting Method.

The Simple Budgeting Method works as follows: pick the first k primes whose sum is less than or equal to n,

$$\sum_{i=1}^{k} p_i \leq n.$$

This gives us a partition (omitting r 1s) $\lambda = [p_1, ..., p_k]$, and the order of this partition is given nicely by

$$\text{lcm}(p_1, ..., p_k) = \prod_{i=1}^{k} p_i$$

because all p_i are coprime. This gives us the nice expression

$$g'(n) := \prod_{i=1}^{k} p_i.$$

The paper [2] shows how this product (called $F(n)$) has the property that $\log F(n) \sim \sqrt{n \log(n)} \sim \log(g(n))$ where $a(n) \sim b(n)$ denotes $\lim_{n\to\infty} \frac{a(n)}{b(n)} = 1$ for some sequences a,b indexed by n. Thus $g'(n)$ has the same asymptotic behavior as $g(n)$. Note also that $g'(n)$ is increasing but not strictly increasing with respect to n.

If we take r to be the difference between $\sum_{i=1}^{k} p_i$ and n:

$$\sum_{i=1}^{k} p_i + r = n,$$

we note that r grows large as the gap between consecutive primes grows large. This is where the SBM fails, and in the next section we describe an improved budgeting method that takes advantage of the remainder r.

3.2. **Examples of the Simple Budgeting Method.**

Example 3.1. The first k primes that sum to 5 are p_1 and p_2, or 2 and 3. We see $5 = 2 + 3$ and thus we deduce that the partition set is $\{2, 3\}$ and that the order of the corresponding cycle type $(a_1\ a_2\ a_3)(b_1\ b_2) \in S_5$ is $2 \times 3 = 6$. Indeed $g(5) = 6$.

Example 3.2. Let $n = 10$. $10 = 2 + 3 + 5$, the first three primes. Thus we estimate that the maximal order of an element in S_{10} the order of the permutation $(a_1\ a_2)(b_1\ b_2\ b_3)(c_1\ c_2\ c_3\ c_4\ c_5)$ is $g'(n) = 2 \times 3 \times 5 = 30$. Indeed $g(10) = 30$.

Example 3.3. Let $n = 50$. We partition n as $50 = 2 + 3 + 5 + 7 + 11 + 13 + 9$, and thus we estimate the maximal order of an element in S_{50} is $g'(50) = 30030$. In fact, $g(50) = 180180$, but we note here that $g'(50) = 30030 = g(41)$, the maximal order of an element in S_{41} and of course $50 = 41 + 9$.

Example 3.4. Let $n = 100$. We find that 100 is equal to the sum of the first 9 primes, $100 = 2 + 3 + 5 + 7 + 11 + 13 + 17 + 19 + 23$. So we say $g'(n) = \prod_{i=1}^{9} p_i = 223092870$. Compared to the true value of $g(100)$, we see $\frac{g'(100)}{g(n)} = \frac{223092870}{232792560} = \frac{23}{24} \approx 95.83\%$.

3.3. **The Advanced Budgeting Method (ABM).** The Advanced Budgeting Method (ABM) takes advantage of the remainder

$$r = n - \sum_{i=1}^{k} p_k$$

that grows as the gap between primes gets large to improve estimations of $g(n)$.

From Theorem 2.7 we see that partitions with prime powered parts have a minimal sum relative to the largest lcm that that sum can achieve. This suggests that an optimal way to spend our remainder budget r is on the difference in powers of primes $p_i^c - p_i$. This is the motivation for the ABM The Advanced Budgeting Method works as follows: After applying the Simple Budgeting Method to n and obtaining the first k primes whose sum is less than or equal to n, we continue to budget the remainder amongst nearest prime powers one power at a time, scanning to find the next prime power our remainder can afford. Because the ABM is an extension of SBM, we have that

$$g'(n) \leq g''(n) \ \forall n \in \mathbb{N}.$$

We also know that

$$g''(n) \leq g(n) \ \forall n \in \mathbb{N}$$

because through this process we build a partition of n, whose order cannot exceed $g(n)$. Since $g'(n) \leq g''(n) \leq g(n)$ and $g'(n) \sim g(n)$, we get

$$g''(n) \sim g(n).$$

The Advanced Budgeting Method will have multiple different remainder values to keep track of, so call the first r_1, the difference between our total budget n and the first k_1 primes such that their sum does not exceed n.

$$r_1 = n - \sum_{i=1}^{k_1} p_i$$

Then take the remainder r_1, and pick the first k_2 differences between squares of primes and primes such that their sum to less than or equal to r_1.

$$\sum_{i=1}^{k_2} (p_i^2 - p_i) \leq r_1.$$

Now call r_2

$$r_2 = r_1 - \sum_{i=1}^{k_2} (p_i^2 - p_i).$$

Then take the remainder r_2, and pick the first k_3 differences between cubes and squares of primes such that their sum to less than or equal to r_2.

$$\sum_{i=1}^{k_3} (p_i^3 - p_i^2) \leq r_2$$

Now call r_3

$$r_3 = r_2 - \sum_{i=1}^{k_3} (p_i^3 - p_i^2).$$

This leads us to a recursion relation

$$r_t = r_{t-1} - \sum_{i=1}^{k_t} (p_i^t - p_i^{t-1}).$$

If at any point in this process, $r_t < 2^{t-1}$, then stop. At this point, it is helpful to denote the differences between next powers of primes $p_i^t - p_i^{t-1} = p_i^{[t]}$ so we can rewrite our general remainder term

$$r_t = r_{t-1} - \sum_{i=1}^{k_t} p_i^{[t]}.$$

We note that it is helpful to do this in the form of a tableau to keep track of the remainders r_t and the values you will need to finally compute $g''(n)$. Each entry in the bottom row is the sum of the column its in and the whole bottom row is λ^*, giving us $g''(n) = \text{lcm}(\lambda^*) = \prod_{i=1}^{k} \lambda_i$. Below is a general form of this tableau.

p_1	p_2	$\cdots$	$\cdots$	p_{k_1}	r_1
$p_1^{[2]}$	$p_2^{[2]}$	$\cdots$	$p_{k_2}^{[2]}$	0	r_2
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$p_1^{[j]}$	$\cdots$	$p_{k_i}^{[j]}$	0	0	r_j
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$p_1^{[t]}$	0	0	0	0	r_t
p_1^t	$\cdots$	$p_{k_i}^j$	$\cdots$	p_{k_1}	

Note that many of the entries will be 0 as you will find that the next prime difference is greater than your remainder, and you must move to the next row. It is reasonable to leave 0 cells empty.

Lemma 3.5. *Let λ be a partition of n obtained by the Simple Budgeting Method such that $\lambda_i = p_i$ or 1 and let r be the remainder $r = n - \sum_{\lambda_i \neq 1} \lambda_i$. Also let c_i be appropriately chosen integers such that the difference $n - \sum_{i=1}^{k} p_i^{c_i}$ is minimized. Then for*

$$r = \sum_{i=1}^{k} (p_i^{c_i} - p_i),$$

we construct our prime powered partition through the ABM:

$$\lambda^* = [p_1^{c_1}, p_2^{c_2}, \ldots, p_k^{c_k}]$$

and thus the order of the corresponding cycle type, our estimate for $g(n)$, is

$$g''(n) = \text{lcm}(\lambda^*) = \prod_{i=1}^{k} p_i^{c_i}.$$

Proof. The final row in the ABM tableau represents the entries of the prime powered partition λ^*, which are obtained by summing the column above. Since the entries of the column are either 0 or $p_1, \ldots, p_i^{[j]}$. Then due to our choice of $p_i^{[t]} = p_i^t - p_i^{t-1}$ the i'th column will have the value $\sum_{i=1}^{j} p_i^{[j]} = p_i + (p_i^2 - p_i) + \cdots + (p_i^j - p_i^{j-1}) = p_i^j$.

We have $r = \sum_{i=1}^{k}(p_i^{c_i} - p_i)$, after obtaining the first row (SBM). So adding $\lambda_i = p_i$ to $(p_i^{c_i} - p_i)$ gives us $\lambda_i^* = p_i^{c_i}$ for all $i \in \{1, \ldots, k\}$ which shows $\lambda^* = [p_1^{c_1}, p_2^{c_2}, \ldots, p_k^{c_k}]$ and completes the proof. □

3.4. **Examples of the ABM.**

Example 3.6. *Let $n = 50$.* Then, first we find the first k primes from n such that the remainder is nonnegative.

$$50 = 2 + 3 + 5 + 7 + 11 + 13 + 9.$$

So with a remainder r_1 of 9, we find the first k differences of squared primes and primes

$$9 = 2 + 6 + 1.$$

Then we are done, and the partition λ^* is $[4, 5, 7, 9, 11, 13]$ and our estimate $g''(50) = 4 * 5 * 7 * 9 * 11 * 13 = 180180$. Indeed, $g(50) = g''(50)$ wheras $g'(50)/g(50) = 30030/180180 = 1/6 \approx 16.67\%$ accurate. the following table helps visualize this process and result*:*

2	*3*	*5*	*7*	*11*	*13*	*9*
2	*6*	*1*				
4	*9*	*5*	*7*	*11*	*13*	

Example 3.7. *Let $n = 70$.* Then $70 = 2 + 3 + 5 + 7 + 11 + 13 + 17 + 12$. With $r_1 = 12$ we find $12 = 2+6+4$. With $r_2 = 4$ we find $4 = 2^3 - (2^2 - 2) - 2 = 8 - 2 - 2$. Then we are done, and the partition λ^* is $[5, 7, 8, 9, 11, 13, 15]$ and our estimate $g''(70) = 5 * 7 * 8 * 9 * 11 * 13 * 15 = 6126120$. Indeed $g(70) = 6126120$, wheras $g'(70) = 510510$ and thus is $g'(70)/g(70) = 510510/6126120 = 1/12 \approx 8.33\%$ accurate. The following table again illustrates this method*:*

2	*3*	*5*	*7*	*11*	*13*	*17*	*12*
2	*6*	*4*					
4							
8	*9*	*5*	*7*	*11*	*13*	*17*	

Example 3.8. *Let $n = 99$.* Then $99 = 2 + 3 + 5 + 7 + 11 + 13 + 17 + 19 + 22$. With $r_1 = 22$, we find $22 = 2 + 6 + 14$. With $r_2 = 14$, we find $14 = 4 + 10$. With $r_3 = 8 + 2$. Since $r_4 = 2 \leq 2^3$, we cannot continue. Tallying our primes and prime remainders, we get $\lambda^* = [5, 7, 9, 11, 13, 16, 17, 19]$ and finally $g''(99) = \text{lcm}(\lambda^*) = 5 * 7 * 9 * 11 * 13 * 16 * 17 * 19 = 232792560$. Indeed, $g(99) = 232792560$.

2	*3*	*5*	*7*	*11*	*13*	*17*	*19*	*22*
2	*6*	*0*	*0*	*0*	*0*	*0*	*0*	*14*
4	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*10*
8	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*2*
16	*9*	*5*	*7*	*11*	*13*	*17*	*19*	

Example 3.9. *Let $n = 100$.* Then we partition $100 = 2+3+5+7+11+13+17+19+23$, thus $r_1 = 0$ and we are done. Thus we say $\lambda^* = [2, 3, 5, 7, 11, 13, 17, 19, 23]$ and thus $g''(100) = 2*3*5*7*11*13*17*19*23 = 223092870$ which is $\frac{23}{24} \approx 95.83\%$ accurate. Below we give the rather uninspiring ABM tableau for $n = 100$.

2	*3*	*5*	*7*	*11*	*13*	*17*	*19*	*23*	*0*
2	*3*	*5*	*7*	*11*	*13*	*17*	*19*	*23*	

Here we run into a fascinating problem,

$$g''(99) > g''(100),$$

meaning our estimate is not non-decreasing for all n like $g'(n)$ is. More importantly,

$$g''(99) = g(99) = g(100) > g''(100),$$

meaning the ABM was 100% accurate for a given some n and then the accuracy decreases for $n + 1$ even though the real value of $g(n)$ remained constant. It is unknown how often $g''(n) > g''(n + 1)$, but the first occurrence is at $n = 99$.

3.5. **Comparing $g(n)$, SBM, and ABM.** Below is a MATLAB generated figure containing plots for all $g(n)$, $g'(n)$, $g''(n)$ from $n = 1$ to $n = 100$.

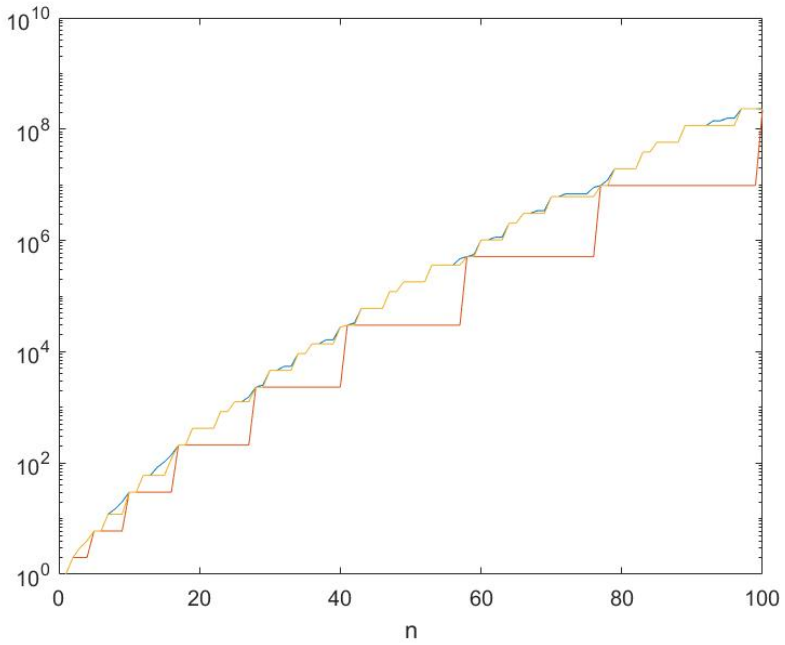

FIGURE 1. The graphs of the Landau function $g(n)$ and its approximations $g'(n), g''(n)$ on a log y-axis. The blue graph is $g(n)$, the yellow graph is $g''(n)$ and the red graph is $g'(n)$.

3.6. **Formulation as a Nonlinear Optimization Problem.** Here we use what we have learned to formulate the Main Problem as a Nonlinear Optimization problem. The many methods of solving NLPs should be employed towards further increasing our ability to approximate this puzzling function.

Main Problem: Given $n \in \mathbb{N}$, find a partition of n such that its lcm is maximal.
As an NLP:

$$\max_{x \in \mathbb{R}^k} \operatorname{lcm}(x)$$

$$s.t. \sum_{i=1}^{k} x_i = n$$

The solution x^* to this NLP is clearly a partition of n corresponding to the cycle type giving the maximum order in S_n.

$$\operatorname{lcm}(x^*) = g(n)$$

4. Probability of guessing λ^*

At one point we found it interesting to ask, "what is the probability of randomly guessing what cycle type in S_n gives you maximal order?" Below we explore and solve this problem.

First we define a probability distribution P over the partitions of n in order to correspond to the cycle types in S_n.

Definition 4.1. *Let $\Lambda(n)$ be the set of all partitions $\lambda = \langle 1^{a_1}, 2^{a_2}, \dots n^{a_n} \rangle$ of n. Define the probability distribution $P : \Lambda(n) \to [0,1] \cap \mathbb{Q}$*

$$P(\lambda) = \frac{1}{\prod_{i=1}^{n} i^{a_i}(a_i)!}$$

If we permit a partition of 0 to be λ_0, then we define $P(\lambda_0) = 0$

This probability distribution describes the probability of randomly choosing an element in S_n of a given cycle type corresponding to λ. This was derived through the following argument, that makes use of the orbit-stabilizer formula, cf. [1]. There are $n!$ permutations in S_n. Each permutation can be decomposed into a product of disjoint cycles, giving a cycle type. For each cycle of length i, there are a_i equivalent cycles.For each repeated cycle type, there are $a_i!$ ways of arranging those because their multiplicative order does not matter. So the size of the conjugacy class corresponding to λ is $\frac{n!}{\prod_{i=1}^{n} i^{a_i}(a_i)!}$. If we sum over all conjugacy classes, we recover the size of S_n. So since

$$\sum_{\lambda \in \Lambda(n)} \frac{n!}{\prod_{i=1}^{n} i^{a_i}(a_i)!} = n!,$$

dividing by $n!$ yields

$$\sum_{\lambda \in \Lambda(n)} \frac{1}{\prod_{i=1}^{n} i^{a_i}(a_i)!} = 1.$$

Theorem 4.2. *Let λ^* be the partition whose corresponding cycle type gives the maximal* lcm*. Then $P(\lambda^*) = \frac{1}{g(n)}$*

Proof. Recall the form of an all prime powered partition $\lambda^* = [p_1^{c_1}, p_2^{c_2}, \dots, p_k^{c_k}]$. This means it is of the form $\lambda^* = \langle \dots (p_1^{c_1})^1 \dots (p_2^{c_2})^1 \dots (p_k^{c_k})^1 \dots \rangle$ and all zero elsewhere. So for λ^*, $a_i = 0$ or 1 for all $i \in \{1, \dots, n\}$. Likewise $i = p_i^{c_i}$ for all i. Combining these facts, we get

$$P(\lambda^*) = \frac{1}{\prod_{i=1}^{n} p_i^{c_i}} = \frac{1}{g(n)}.$$

□

In the appendix, we showcase some plots of these probability distributions for several n

5. Wreath-product groups

In this section we examine a class of groups closely related to the symmetric group, that of *wreath product groups*. To that end, fix natural numbers n and ℓ. Let $\zeta := e^{2\pi\sqrt{-1}/\ell} \in \mathbb{C}$ be a primitive ℓ-th root of unity, and consider the set

$$X(n,\ell) := \{\zeta^i j \mid i = 0, \dots, \quad \ell - 1, j = 1, \dots, n\}$$

Note that the set $X(n,\ell)$ is closed under multiplication by ζ.

Definition 5.1. *The group $G(\ell, n)$ consists of all bijections $\sigma : X(n,\ell) \to X(n,\ell)$ that are ζ-linear, that is, satisfying*

$$\sigma(\zeta^j i) = \zeta^j \sigma(i)$$

for every $i = 1, \dots, n$ and $j = 0, \dots, \ell - 1$.

For example, $G(1, n) = S_n$. Note that, since $G(\ell, n)$ consists of bijections on a set of $n\ell$ elements, we have a natural embedding $G(\ell, n) \hookrightarrow S_{n\ell}$.

Definition 5.2. *Let $\ell, n > 0$. We denote by $g_\ell(n)$ the maximal order of an element in $G(\ell, n)$ so that, for example, $g_1(n)$ is the Landau function.*

Note that, since we have an embedding $G(\ell, n) \hookrightarrow S_{n\ell}$, we immediately obtain the inequality

$$g_\ell(n) \leq g(n\ell)$$

Our main result regarding wreath-product groups is that, in fact, the function $g_\ell(n)$ has a very simple expression in terms of the function $g(n)$.

Theorem 5.3. *For every $n, \ell > 0$ we have*

$$g_\ell(n) = \ell g(n)$$

In order to prove Theorem 5.3 we will establish a connection between the groups $G(\ell, n)$ and S_n.

Lemma 5.4. *Let $\sigma \in G(\ell, n)$, and assume that $\sigma(i) = \zeta^{m_i}\pi(i)$. Then, the map $\pi : \{1, \dots, n\} \to \{1, \dots, n\}$ is bijective.*

Proof. It is enough to show that π is surjective. Let $i \in \{1, \dots, n\}$. Since σ is bijective, there exist $k \in \{0, \dots, \ell - 1\}$ and $j \in \{1, \dots, n\}$ such that

$$\sigma(\zeta^k j) = i$$

But then, by the linearity of σ:

$$\sigma(j) = \sigma(\zeta^\ell j) = \zeta^{\ell-k}\sigma(\zeta^k j) = \zeta^{\ell-k} i$$

Thus, $\pi(j) = i$ and π is surjective. □

Let us denote by $\pi(\sigma) \in S_n$ the permutation obtained in Lemma 5.4.

Lemma 5.5. *The map $\pi : G(\ell, n) \to S_n$, $\sigma \mapsto \pi(\sigma)$ is a group homomorphism.*

Proof. Let σ,$\rho \in G(\ell, n)$. We want to show that $\pi(\sigma\rho) = \pi(\sigma)\pi(\rho)$. More explicitly, we want to show that for every $i \in \{1, \ldots, n\}$

$$\pi(\sigma\rho)(i) = \pi(\sigma)\pi(\rho)(i)$$

By definition, $\rho(i) = \zeta^{m_i}\pi(\rho(i))$ for some m_i. Then, by linearity of σ:

$$\sigma(\rho(i)) = \sigma(\zeta^{m_i}\pi(\rho)(i)) = \zeta^{m_i}\sigma(\pi(\rho)(i)) = \zeta^{m_i}\zeta^{m_{\pi(\rho)(i)}}\pi(\sigma)\pi(\rho)(i)$$

Which by definition of π gives

$$\pi(\sigma(\rho(i))) = \pi(\sigma)\pi(\rho)(i).$$

□

Lemma 5.6. *Let $\sigma \in G(\ell, n)$. Let o_σ be the order of σ and $o_{\pi(\sigma)}$ be the order of $\pi(\sigma) \in S_n$. Then*

$$o_\sigma \leq \ell o_{\pi(\sigma)}$$

Proof. Note that $\pi(\sigma^{o_{\pi(\sigma)}}) = e$ so, by definition of π:

$$\sigma^{o_{\pi(\sigma)}}(i) = \zeta^{m_i} i$$

Raise this to the ℓ

$$\sigma^{\ell o_{\pi(\sigma)}}(i) = \zeta^{\ell m_i} i = i$$

For all $i \in \{1, \ldots, n\}$.

So

$$o_\sigma \leq \ell o_{\pi(\sigma)}.$$

□

Corollary 5.7. We have

$$g_\ell(n) \leq \ell g(n)$$

Proof. Let $\sigma \in G(\ell, n)$ be an element of maximal order. Note that $o_{\pi(\sigma)} \leq g(n)$ as well as $g(n) = o_\sigma$. Then

$$g_\ell(n) = o_\sigma \leq \ell o_{\pi(\sigma)} \leq \ell g(n)$$

□

Thus, to show that $g_\ell(n) = \ell g(n)$, we need to produce an element of $G(\ell, n)$ whose order is $\ell g(n)$. To this end, define an element of $G(\ell, n)$ as follows.

Let $\Omega : S_n \to G(\ell, n)$ be defined over the cycle decomposition of $\sigma = \sigma_1\sigma_2 \ldots \sigma_m \in S_n$ with

$$\Omega(\sigma_i^{(k)}) = (a_1\ a_2\ \ldots\ a_k\ \zeta a_1\ \zeta a_2\ \ldots\ \zeta a_k\ \zeta^2 a_1\ \zeta^2 a_2\ \ldots\ \zeta^2 a_k\ \ldots\ \zeta^{\ell-1} a_1\ \zeta^{\ell-1} a_2\ \ldots\ \zeta^{\ell-1} a_k\ )$$

and

$$\Omega(\sigma) = \Omega(\sigma_1)\Omega(\sigma_2)\ldots\Omega(\sigma_m).$$

Note that if σ_i is a k-cycle, then $\Omega(\sigma_i)$ is an ℓk-cycle. Note also that Ω is not a group homomorphism, as

$$\Omega(e) = (e)(\zeta e)(\zeta^2 e)\ldots(\zeta^{\ell-1} e) \neq e$$

for $\ell \neq 1$. Then if k_i is the order of σ_i, the order of $\Omega(\sigma)$ is the order of the product of ℓk_i-cycles

$$\operatorname{lcm}(\ell k_1, \ell k_2, \dots \ \ell k_m) = \ell \operatorname{lcm}(k_1, k_2, \dots k_m) = \ell o_\sigma = \ell g(n).$$

This completes the proof of Theorem 5.3.

References

[1] M. Artin, *Algebra*

[2] W. Miller, *The maximum order of an element of a finite symmetric group.* Amer. Math. Monthly **94**, issue 6 (1987), 497–506 https://doi.org/10.1080/00029890.1987.12000673

[3] John D'Errico (2020). Partitions of an integer (https://www.mathworks.com/matlabcentral/fileexchange/12009-partitions-of-an-integer), MATLAB Central File Exchange. Retrieved May 15, 2020.

[4] Josh (2020). Least Common Multiple Set (https://www.mathworks.com/matlabcentral/fileexchange/24670-least-common-multiple-set), MATLAB Central File Exchange. Retrieved May 15, 2020.

[5] OEIS Foundation Inc. (2020), The On-Line Encyclopedia of Integer Sequences, http://oeis.org/A000793

[6] OEIS Foundation Inc. (2020), The On-Line Encyclopedia of Integer Sequences, http://oeis.org/A000793/b000793.txt

APPENDIX

Appendix A. Matlab Code

A.1. **Code for creating distributions of orders of elements in S_n.** This is my MATLAB code for creating figures for the distribution of orders of elements in the Symmetric Group.It relies on two functions [3],[4] found online and the authors of those two functions are credited as well as the links to their files.

```
%This code relies on the >partitions function by John D'Errico (2020) found at
%https://www.mathworks.com/matlabcentral/fileexchange/12009-partitions-of-an-integer
%And the >lcms function by Josh (2020) found at
%https://www.mathworks.com/matlabcentral/fileexchange/24670-least-common-multiple-set
function n = LPP(n)
P = partitions(n);
%matrix partitions of n where the 1st column represents 1s & the last column represents n's
lP = length(P);
%Total number of partitions
lambda = zeros(1,n);
y = zeros(1,lP);

for i = 1:lP
lambda = P(i,:); %lambda assigned value of ith row of P%

Plambda = lcms(sign(lambda).*[1:n]);
%Calculates the Order of the Cycle this partition represents
y(i) = Plambda; %vector y given order value

end
bar(1:lP,y)
```

A.2. **Code for the Simple budgeting method.** This is my MATLAB code for running the Simple Budgeting Method

```
   function gprime = SBM(n)
p = primes(n);
budget = n;
numprimes = length(p);
r = 0;
glamb = zeros(numprimes);
gprime = 0;

%This loop simply checks if the sum of primes exceeds n and updates gprime
for i = 1:numprimes
    if sum(p(1:i))<= n
        gprime = prod(p(1:i));
    end
end
```

This is my MATLAB code for producing the figures for the Probability distributions over the partitions of n, revealing their fractal-like appearance. If you copy-paste it into MATLAB, it should work. This code relies on a function [3] that is cited and linked in the code.

```
%This code relies on the >partitions function by John D'Errico (2020) found at
%https://www.mathworks.com/matlabcentral/fileexchange/12009-partitions-of-an-integer
function n = PPP(n)
P = partitions(n);
lP = length(P); %This is the famous pi(n) function, counting # of partitions of n
lambda = zeros(1,n);
y = zeros(1,lP);

for i = 1:lP
lambda = P(i,:);
%Below is the formula derived for choosing a random cycle-type from S_n%
Plambda = 1/(prod(factorial(lambda)))*1/((prod((1:n).^lambda)));
y(i) = Plambda;
%This line assigns the components of a vector to the Plambda value in the loop
end
bar(1:lP,y)
```

Appendix B. Figures

The following are figures of orders of elements in S_n for some specified n generated by my MATLAB code "LPP". After that there are figures showing the Probability Distributions derived in the Probability section for a given n.

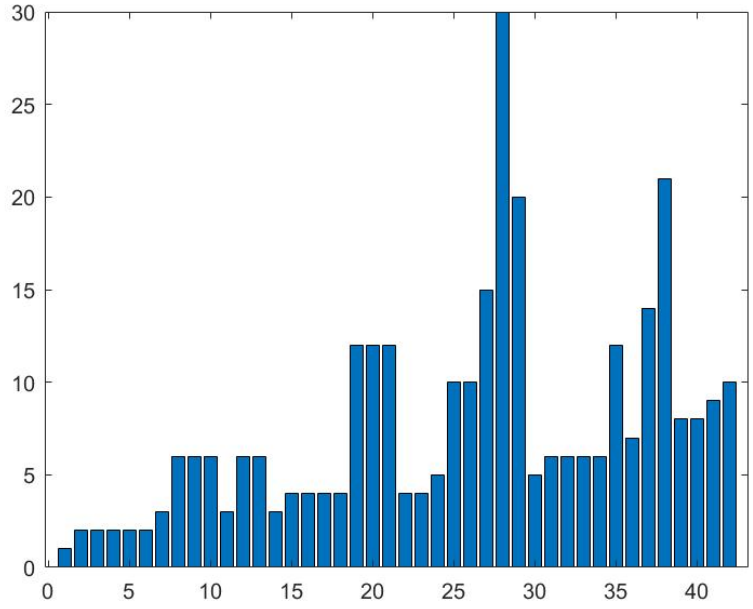

FIGURE 2. Orders of elements in S_{10} Group

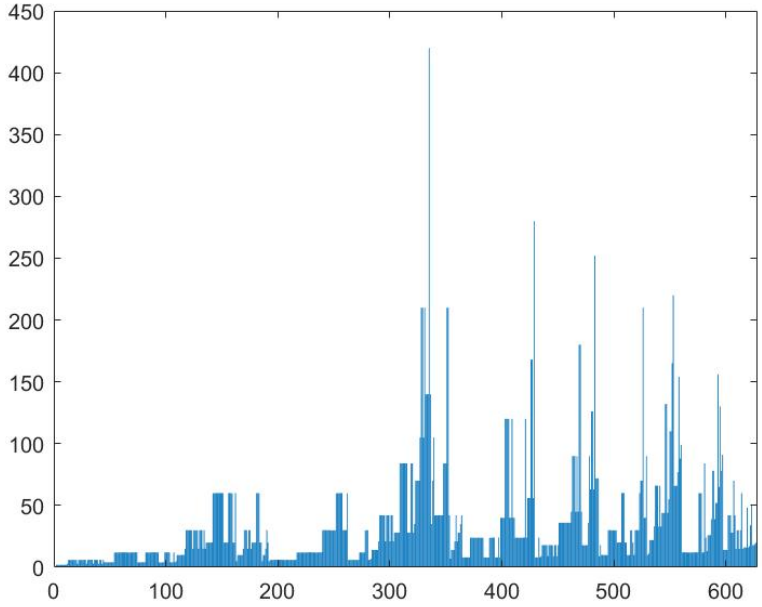

FIGURE 3. Orders of elements in S_{20} Group

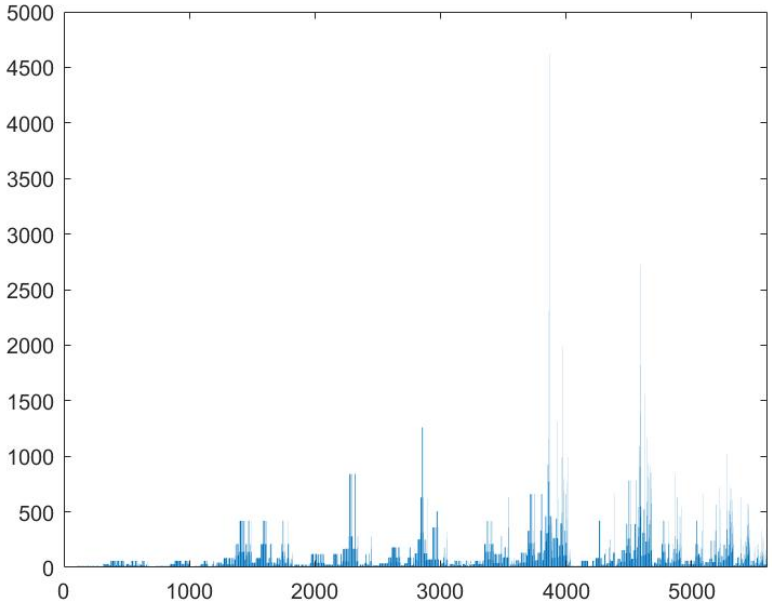

FIGURE 4. Orders of elements in S_{30}

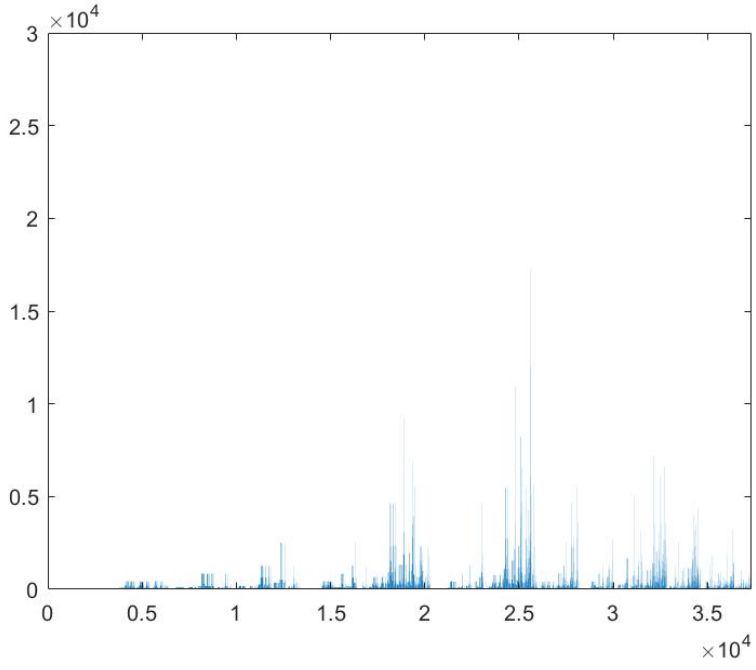

FIGURE 5. Orders of elements in S_{40}

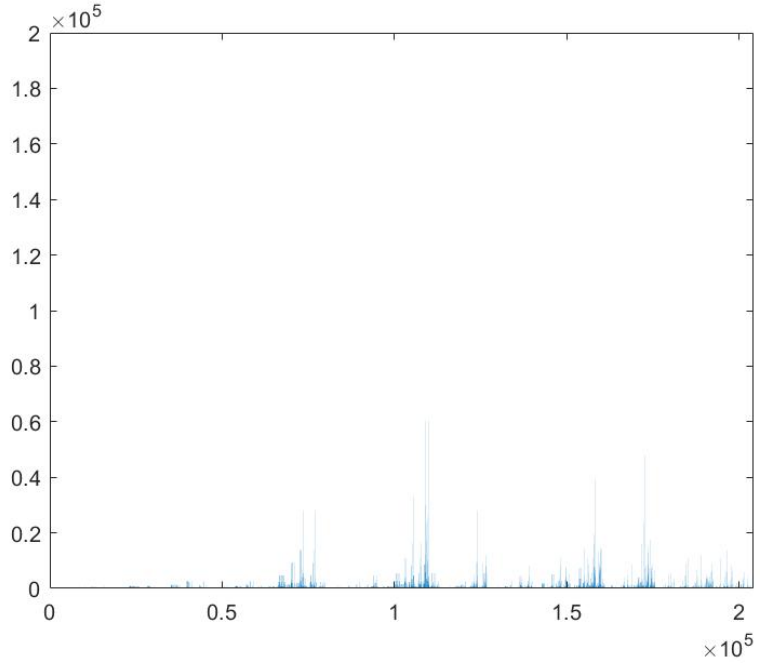

FIGURE 6. Orders of elements in S_{50}. As you can see, it becomes very difficult for MATLAB to render all of the thousands of bars representing the order of elements in S_{50}. In fact, the graph appears much taller than it needs to be, but thats because the blue bar representing $g(50)$ is so small it was not rendered at that zoom level.

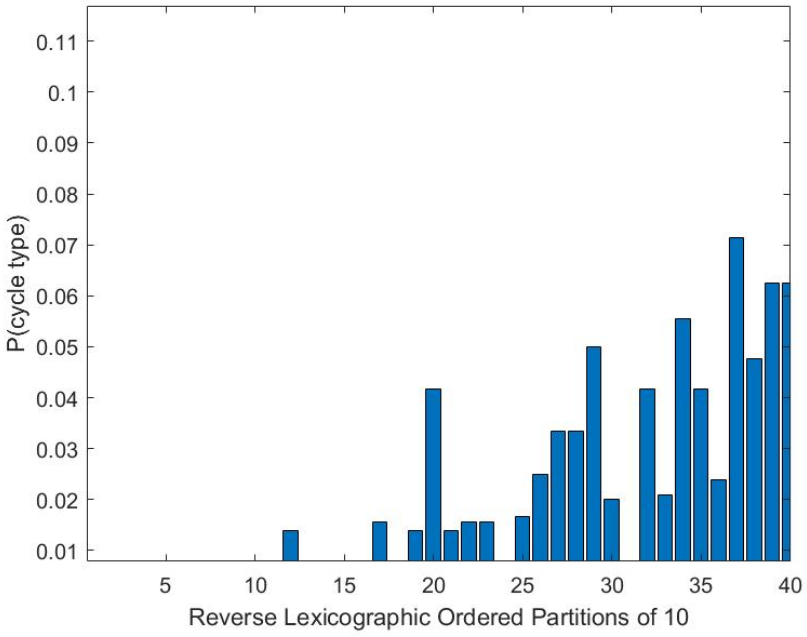

FIGURE 7.

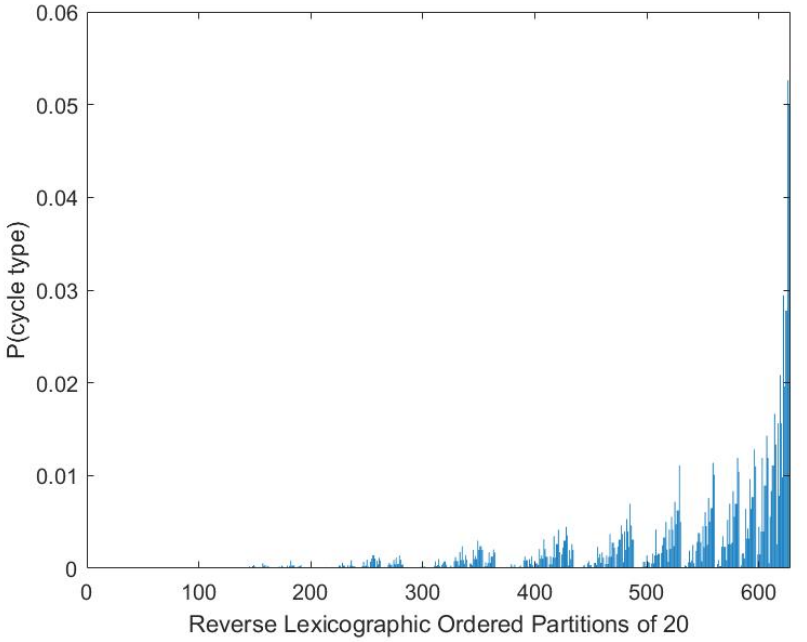

FIGURE 8.

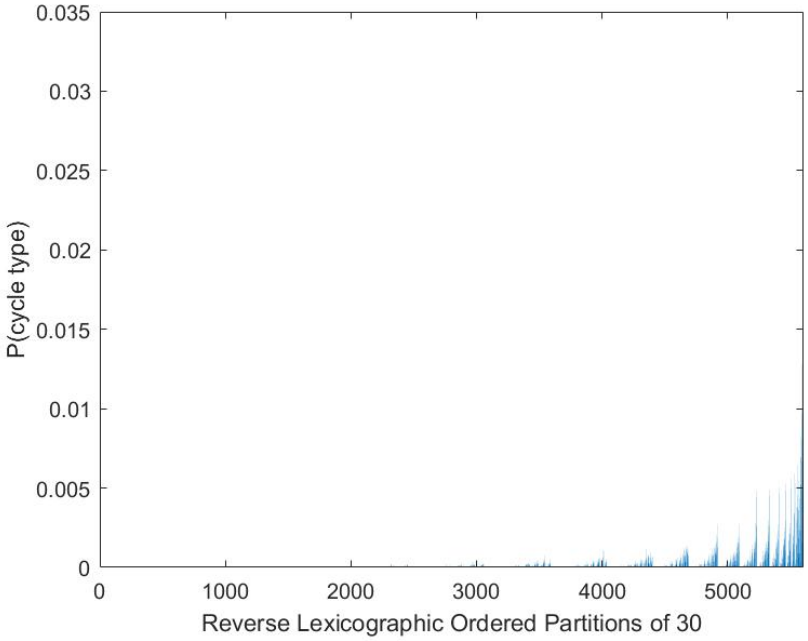

FIGURE 9.

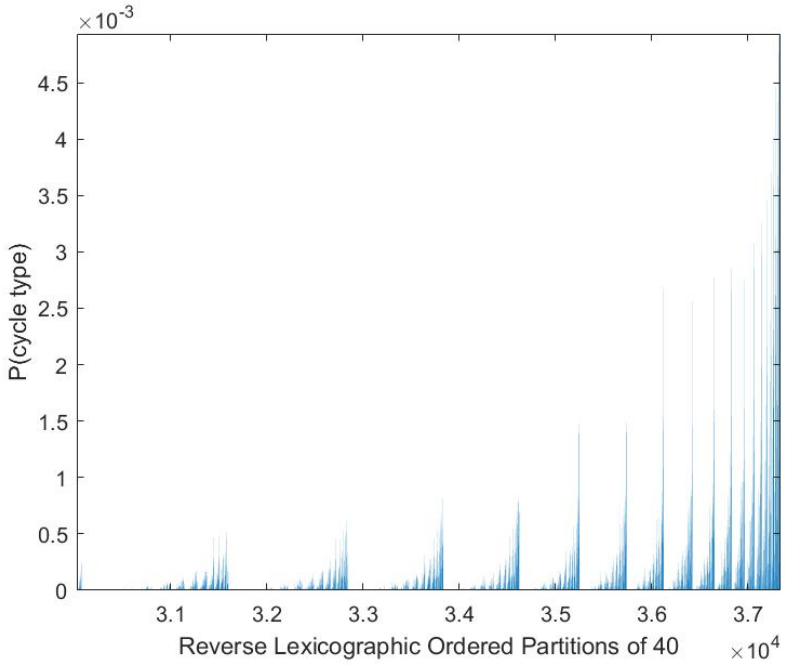

FIGURE 10. This figure was zoomed in in order to make out some detail. The maximum probability not shown in this frame is .03

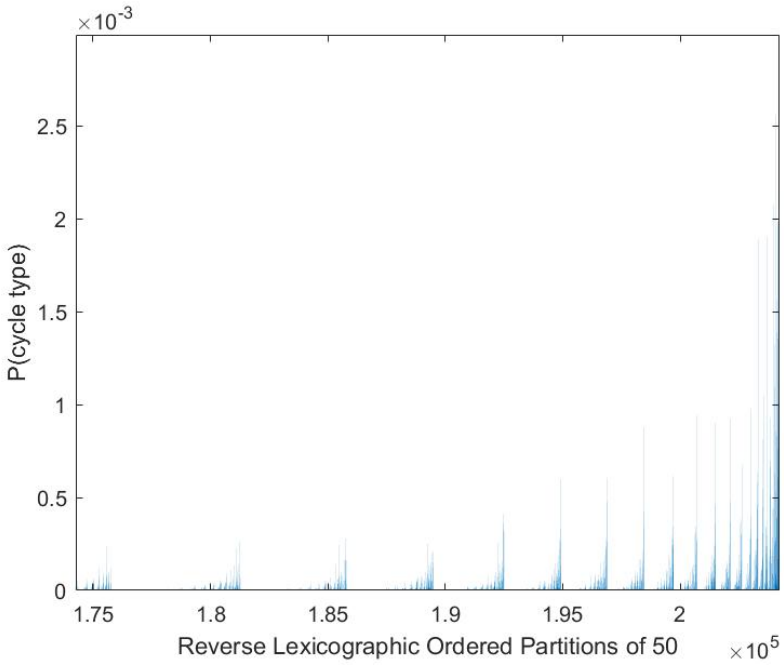

FIGURE 11. This figure was zoomed very far in to find the detail in this distribution. The maximum probability not shown in frame is .025

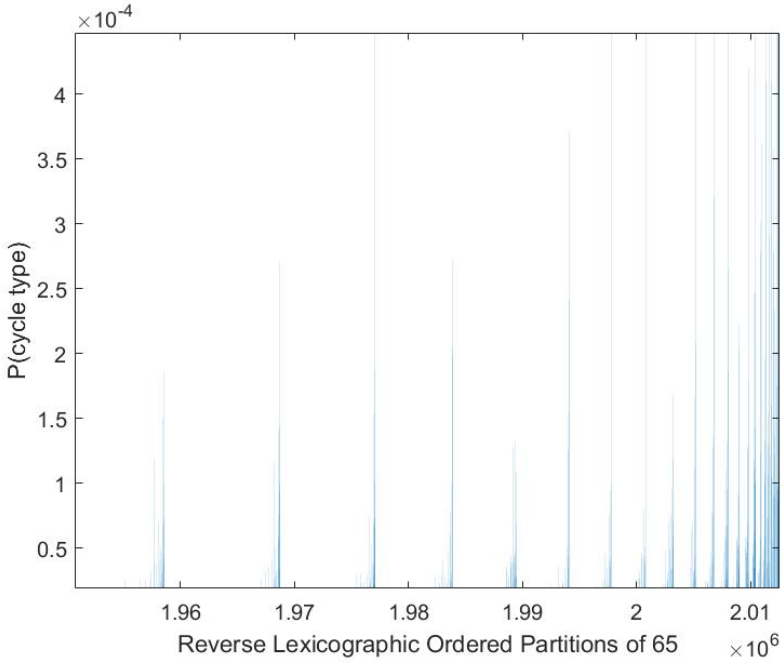

FIGURE 12. This is the distribution of probabilities of randomly choosing cycle types in S_{65}, but zoomed in several orders of magnitude such that the detail of this distribution is visible.