Fall 2018

An algorithm to determine all odd primitive abundant numbers with d prime divisors

Jacob Liddy
jpl61@zips.uakron.edu

An algorithm to determine all odd primitive abundant numbers with d prime divisors

Jacob P. Liddy, Jeffrey M. Riedl

September 17, 2018

Abstract

For an integer n, if the sum of the proper divisors of n equals or exceeds n, then we say that n is an *abundant number*. An abundant number is said to be *primitive* if none of its proper divisors are abundant. An abundant number must have at least one primitive abundant divisor. In 1913, Dickson proved that for an arbitrary positive integer d there exists only finitely many odd primitive abundant numbers having exactly d distinct prime divisors. In 2017, all odd primitive abundant numbers with up to 5 distinct prime divisors have been found by Dičiūnas. In this paper, we describe a fast algorithm that finds all odd primitive abundant numbers with d distinct prime divisors. We use this algorithm to find all odd primitive abundant numbers with 6 distinct prime divisors. An abundant number n is said to be *weird* if no subset of the proper divisors of n sums to n. We use our algorithm to show that an odd weird number must have at least 6 prime divisors.

1 Introduction

The proper divisors of a positive integer n are the positive divisors of n that are less than n. We denote the set of proper divisors of n by A_n.

We define the function $\sigma : \mathbb{N} \to \mathbb{N}$ such that

$$\sigma(n) = \sum_{d|n} d$$

the sum taken over the divisors of n. We say n is *abundant* if $\sigma(n) \geq 2n$ [1], *perfect* if $\sigma(n) = 2n$, and *deficient* if $\sigma(n) < 2n$. We say that n is *pseudoperfect* if n is a non-perfect abundant and there exists a subset $S \subset A_n$ such that

$$n = \sum_{d \in S} d.$$

We say that n is *weird* if n is abundant but not pseudoperfect or perfect. The smallest weird number is 70, and all known weird numbers are even.

If n is abundant and all of its proper divisors are deficient, we say that n is a primitive abundant number. An abundant number must have at least one primitive abundant divisor. The smallest primitive abundant number is 6, by our definition. Some authors exclude perfect numbers from abundant numbers, in which case the smallest primitive abundant number is 20. This paper focuses on odd primitive abundant numbers, which we refer to as OPANs (Following the notation of [2] and [3]). The sequence of OPANs is available at `https://oeis.org/A006038`.[4]

[1] This is the definition Erdős gave for abundant numbers.[1]

It is not known whether any odd perfect or odd weird numbers exists. However, it is easy to show that any odd perfect number is an OPAN, if such a number exists. Likewise if an odd weird number w exists, it can be shown that w is an OPAN or w is the multiple of a weird OPAN.

Our purpose is to find an algorithm that enumerates all OPANs with a fixed number of unique prime divisors d. With this, we classify all OPANs with 5 prime divisors as weird, perfect, or pseudoperfect. We found that all OPANs with $3, 4$, or 5 divisors are pseudoperfect, which allows us to conclude that an odd weird number and an odd perfect number must have at least 6 unique prime factors.

In this paper, we succeed in finding the described algorithm. We prove that the algorithm does indeed find all OPANs with $3, 4, 5$, and 6 unique prime divisors. We wish to generalize the proof to d divisors. Finding all OPANs with 6 prime divisors was previously an unsolved problem[3].

Through implementing the algorithm in Section 5, we were able to find the number of OPANs with 6 divisors. We found that $|OPAN(6)| = 59687996404445$

The largest odd primitive abundant number with 6 divisors is: $3^{38}5^{28}17^{16}257^{8}65537^{4}4294967291^{2}$. It has 116 digits.

2 Preliminaries

We define the function $b : \mathbb{N} \mapsto \mathbb{Q}$ by

$$b(n) = \frac{\sigma(n)}{n}.$$

Hence n is abundant if and only if $b(n) \geq 2$

Note that if $n = p_1^{m_1} p_2^{m_2} p_3^{m_3} ... p_k^{m_k}$ for distinct primes $p_1, ..., p_k$, it can be shown that

$$\sigma(n) = \prod_{1 \leq i \leq k} \frac{p_i^{m_i+1} - 1}{p_i - 1}.$$

Thus, $b(n)$ can be expressed as:

$$b(n) = \frac{\sigma(n)}{n} = \prod_{1 \leq i \leq k} \frac{p_i^{m_i+1} - 1}{(p_i - 1)p_i^{m_i}} = \prod_{1 \leq i \leq k} b(p_i^{m_i}) \tag{1}$$

Also, if n and m are positive integers greater than 1, then

$$b(n) < b(nm). \tag{2}$$

To show this, we first suppose that m and n share no prime factors. Thus if the prime factorization of n is $p_1^{s_1} p_2^{s_2} ... p_i^{s_i}$ and the prime factorization of m is $r_1^{l_1} r_2^{l_2} ... r_j^{l_j}$, then the unique prime factorization of mn is

$$mn = p_1^{s_1} p_2^{s_2} ... p_i^{s_i} r_1^{t_1} r_2^{t_2} ... r_j^{t_j}.$$

Thus $b(mn) = b(n)b(m)$. Since $\sigma(m) \geq m + 1$(as m and 1 are divisors of m), we have that $b(m) \geq (m + 1)/m > 1$. Thus, $b(n) < b(n)b(m) = b(nm)$.

To show that that (2) is true in general, we first note that given any prime p and any positive integer m, we can show that $b(p^m) < b(p^{m+1})$. Let $n = p_1^{s_1} p_2^{s_2} ... p_d^{s_d}$. Let m be written in the prime factorization $m = q_1^{t_1} q_2^{t_2} ... q_i^{t_i} r_1^{e_1} r_2^{e_2} ... r_j^{e_j}$ where $q_1, q_2, ..., q_i$ are prime factors that n and m share, and each r is a prime factor that divides m but not n. Suppose that m and n

share at least one prime factor, as the other case was discussed in the previous paragraph. Let $n' = nq_1^{s_1}q_2^{s_2}...q_i^{s_i} = p_1^{s_1'}p_2^{s_2'}...p_d^{s_d'}$, where each $s_i' \geq s_i$. Hence $b(n) < b(n')$. Since n' and $r_1^{l_1}r_2^{l_2}...r_j^{l_j}$ share no prime factors, we have that $b(n') \leq b(n'r_1^{l_1}...r_j^{l_j}) = b(nm)$. Thus $b(n) < b(n') \leq b(mn)$. Hence, $b(n)$ is a multiplicatively increasing function.

Taking the limit of $b(p^m)$ as m goes to infinity, we have that $\lim_{m\to\infty} b(p^m) = \frac{p}{p-1}$. Applying this to the sequence of primes $P = \{p_1, p_2, ..., p_d\}$, we define $b_\infty(P)$ by

$$b_\infty(P) = \prod_{1\leq i\leq k} \lim_{m\to\infty} b(p_i^m) = \prod_{1\leq i\leq k} (\frac{p_i}{p_i - 1}). \tag{3}$$

Theorem 2.1. *Consider the set of primes $P = \{p_1, p_2, p_3, ..., p_k\}$. There exists infinitely many abundant integers n whose prime factors compose the set P if and only if $b_\infty(P) > 2$*

Proof. First we assume that $b_\infty(P) > 2$. Let $L = b_\infty(P)$. Note that

$$L = \prod_{1\leq i\leq k} \lim_{m_i\to\infty} b(p_i^{m_i}) = \prod_{1\leq i\leq k} \lim_{m_i\to\infty} (\frac{p_i^{m_i+1} - 1}{(p_i - 1)p_i^{m_i}}).$$

.

Hence , we can allow

$$\prod_{1\leq i\leq k} \frac{p_i^{m_i+1} - 1}{(p_i - 1)p_i^{m_i}}$$

to be arbitrarily close to L for large enough exponents m_i. Since L is greater than two, there exists a collection of positive integers $m_1, m_2, ..., m_k$ such that

$$L > \prod_{1\leq i\leq k} \frac{p_i^{m_i+1} - 1}{(p_i - 1)p_i^{m_i}} = \prod_{1\leq i\leq k} b(p^{m_i}) > 2.$$

Thus we let $n = p_1^{m_1}p_2^{m_2}p_3^{m_3}...p_k^{m_k}$. Since $b(n) > 2$, it follows that n is abundant. Because $b(n)$ is an increasing function, n can be multiplied by any one of its prime factors, and an abundant number will be the product. In other words,

$$2 < b(n) < b(np_1) < b(np_1^2) < ....$$

Thus there are infinitely many abundant integers n whose prime factors are in the set P.

Now we assume that there exists infinitely many abundant integers whose prime factors compose P. Let $p_i \in P$. Given any positive integer r and abundant integer n whose prime factors compose P, we have that that np_i^r is abundant. Hence,

$$b_\infty(P) = \prod_{1\leq i\leq k} \lim_{m_i\to\infty} (\frac{p_i^{m_i+1} - 1}{(p_i - 1)p_i^{m_i}}) > 2.$$

□

Theorem 2.2. *If $b_\infty(P) \leq 2$, then there does not exist an abundant number whose prime factors compose P.*

Proof. Let $P = \{p_1, p_2, ..., p_d\}$. Since $b(p_1^{m_1}p_2^{m_2}...p_d^{m_d})$ is strictly increasing as the exponents $m_1, m_2, ..., m_d$ increase while $b_\infty(P) \leq 2$, we have that $b(p_1^{m_1}p_2^{m_2}...p_d^{m_d}) < b_\infty(P) \leq 2$ for any given exponents $m_1, ..., m_d$. Thus given any positive integer n whose prime factors compose P, we will have that $b(n) < 2$. □

Theorem 2.3. *Let p and q be primes such that $p < q$. Then $b_\infty(\{q\}) \leq b(p^1)$. If $p \geq 3$, then $b_\infty(\{q\})$ is strictly less than $b(p^1)$.*

The proof of Theorem 2.3 is straightforward. It is worth noting that if q is prime and $q-1$ is treated as if it were prime, then $b_\infty(\{q\}) = b(q)$

Another important function we will need describes what happens when some exponents are fixed and some exponents are allowed to increase without limit. The function will take a set of primes and exponents, as well as the indices of certain prime factors which are not limited in exponent increase. Let P be the sequence of primes and E be the respective exponents for $p_1^{m_1}p_2^{m_2}...p_d^{m_d}$. Let I be a subset of $\{1, 2, ..., d\}$ describing the indices in which the corresponding exponents are desired to be raised without limit. We define mb as follows:

$$mb(P, E, I) = \prod_{i \in I} b_\infty(p_i) \prod_{1 \leq j \leq d | j \notin I} b(p_j^{m_j}). \tag{4}$$

Let P be a sequence of primes $P = \{p_1, p_2, ..., p_d\}$, and let $E = \{m_1, ..., m_d\}$ be a sequence of positive integers with the same cardinality as P. We define the positive integer $\nu(P, E)$ by $\nu(P, E) = p_1^{m_1}p_2^{m_2}...p_d^{m_d}$.

We now prove the following theorem, making use of the above definitions:

Theorem 2.4. *Let $P = \{p_1, p_2, ...p_d\}$ be a sequence of primes, and let $E = \{m_1, m_2, ..., m_d\}$ be a sequence of exponents. Let $n = \nu(P, E)$. Let I be a nonempty subset of $\{1, ..., d\}$. The condition $mb(P, E, I) \leq 2$ holds if and only if there exists an abundant number k such that n divides k and every prime divisor of k/n is a member of $\{p_i \mid i \in I\}$.*

We (currently) leave the proof to the reader. It is similar to the proof of Theorem 2.1.

For any given prime p and positive integer m, we define the rational number $\Delta_+(p, m)$ by

$$\Delta_+(p, m) = \frac{p^{m+2} - 1}{p(p^{m+1} - 1)}.$$

Theorem 2.5. *Suppose $n = p_1^{m_1}p_2^{m_2}...p_i^{m_i}...p_d^{m_d}$. For each $i \in \{1, ..., d\}$, we have*

$$b(n)\Delta_+(p_i, m_i) = b(p_i n).$$

The proof for 2.5 is straightforward.

3 Exponent Theorems and The Exponent Algorithm

Let r and n be positive integers such that r divides n. We say r is a divisor of n of order 0 if and only if $r = n$. For each $i \geq 1$, we say that r is a divisor of n of order i in case there exists a prime p such that pr is a divisor of n of order $i - 1$. We remark that each divisor r of n has a unique well-defined order k for some nonnegative integer k. For example, $3^2 5^1 13^1$ is a divisor of $3^3 5^2 13^1$ of order 2.

A related notion is *incomplete primes*. Suppose r divides n, and p is a prime. We say that p is incomplete in r with respect to n if pr divides n. In this case, if r is a divisor of n of order k, then pr is a divisor of n of order $k - 1$. 2 is incomplete in 18 with respect to 72.

Yet another notion is *crucial primes*. Let $n = \nu(P, E)$ be a positive integer. Suppose that $p \in P$. If pn is abundant, then we say that p is a crucial prime for n.

For any given sequence of primes P and corresponding sequence of nonnegative exponents E, whose cardinalities are both d, we wish to define $G_{\Delta_+}(P, E)$ as a particular ordering of $\{0, 1, ..., d - 1\}$. Given any distinct $i, j \in \{0, 1, ..., d - 1\}$, we order $G_{\Delta_+}(P, E)$ such that i precedes j in $G_{\Delta_+}(P, E)$ if $\Delta_+(p_i, e_i) \leq \Delta_+(p_j, e_j)$. In the case where $\Delta_+(p_i, e_i) = \Delta_+(p_j, e_j)$, it does not matter which element is the predecessor or successor in $G_{\Delta_+}(P, E)$.

Theorem 3.1. *Suppose $n = \nu(P, E)$, where $P = \{p_0, p_1, ..., p_{d-1}\}$ and $E = \{e_0, ..., e_{d-1}\}$. Let $G_{\Delta_+}(P, E) = \{g_0, g_1, ..., g_{d-1}\}$. Then $b(np_{g_i}) \leq b(np_{g_j})$ when $i \leq j$.*

The proof of 3.1 follows directly from inequality 2.
To demonstrate how $G_{\Delta_+}(P, E)$ is defined, we calculate $G_{\Delta_+}(P, E)$ for $P = \{3, 5, 11\}$ and $E = \{1, 2, 1\}$.

First we calculate each of $\Delta_+(3, 1)$, $\Delta_+(5, 2)$, and $\Delta_+(11, 1)$. We find that $\Delta_+(3, 1) = \frac{13}{12}$, $\Delta_+(5, 2) = \frac{156}{155}$, and $\Delta_+(11, 1) = \frac{133}{132}$. Listing these values in order from least to greatest, we have

$$\Delta_+(5, 2) < \Delta_+(11, 1) < \Delta_+(3, 1).$$

Note $p_0 = 3$, $p_1 = 5$, and $p_2 = 11$. Since $\Delta_+(5, 2)$ is the smallest while $p_1 = 5$, Our $G_{\Delta_+}(P, E)$ordering on $\{0, 1, 2\}$ begins with 1. Since $\Delta_+(11, 1)$ is the next smallest while $p_2 = 11$, the corresponding index 2 succedes 1 in our ordering on $\{0, 1, 2\}$. Finally, since $\Delta_+(3, 1)$ is the largest while $p_0 = 3$, the corresponding index 0 succedes 2 in our ordering. Hence our ordering is $\{1, 2, 0\}$ and we write $G_{\Delta_+}(P, E) = \{1, 2, 0\}$

Theorem 3.2. *Let $G_{\Delta_+}(P, E) = G$ for some integer $n = \nu(P, E)$. Suppose i precedes j in G. If p_i is crucial in n, then p_j is crucial in n. In other words, there exists some g such that all elements that precede $G[g]$ are indices of non-crucial primes and all elements that succeed $G[g]$ are indices of crucial primes.*

Informally, the above theorem states that $G_{\Delta_+}(P, E)$ ′ looks like ′

$$G_{\Delta_+}(P, E) = (\text{... non-crucial prime indices}|\text{crucial prime indices...}).$$

Proof. Suppose that p_i is crucial in n, so $2 \leq b(np_i)$. Since i precedes j in G, Theorem 3.1 tells us that $b(np_i) \leq b(np_j)$. Thus $2 \leq b(np_j)$, which tells us p_j is crucial in n. If no p_i is crucial in n, choose the highest possible index g in G. □

Theorem 3.3. *Suppose a positive integer n is abundant. Then n is primitive if and only if all of its first order divisors are deficient.*

Proof. First suppose that n is a primitive abundant and let r be a divisor of n of order 1. Since r is a proper divisor of n, indeed r cannot be abundant and is therefore deficient.

Suppose that every divisor of n order 1 is deficient. Let r be any proper divisor in n. Then there exists a divisor k of n of order 1 such that r divides k. Since k is deficient while r divides k, indeed r is deficient (Any divisor of a deficient number is deficient). Thus n is abundant and all proper divisors of n are deficient. By definition, n is a primitive abundant number. □

We now have enough tools to prove the theorem below.

Theorem 3.4. *Let P be an increasing sequence of d odd primes. The following algorithm will extract all primitive abundant odd numbers whose prime divisors compose P.*

In the below algorithm, the sequence of primes P is indexed starting at 0, not 1. In other words, $P = \{p_0, p_1, ..., p_{d-1}\}$. Also, when a function is called, the arguments are passed by reference. This means that a function may modify one of its inputs.

The Exponent Algorithm(main)
INPUT: The sequence of primes $P = \{p_0, p_1, ..., p_{d-1}\}$.
OUTPUT: All primitive abundant odds whose prime factors compose P

---

if $b_\infty(P) \leq 2$ **then**
    return $\{\}$
end if
$E := \{e_0 = 1, e_1 = 1, ..., e_n = 1\}$
$A := \{\}$
expAbundant(P, E, A)
return A

Algorithm: expAbundant *(Check if n is abundant with its current exponents)*
INPUT: The sequence of primes $P = \{p_0, p_1, ..., p_{d-1}\}$, the exponents to those primes E, and the primitive abundants found so far, A.
OUTPUT: No output. However, A will be modified.

---

if $b(P, E) \geq 2$ **then**
    if $\nu(P, E)$ is primitive abundant **then**
        $A := A\&\nu(P, E)$
        return
    end if
    return
end if
crucialAlg(P, E, A)
return

Algorithm: crucialAlg *(orders the Δ_+ values)*
INPUT: The sequence of primes $P = \{p_0, p_1, ..., p_{d-1}\}$, exponents $E = \{m_0, m_1, ...\}$, the set of primitive abundant numbers found A.
OUTPUT: True or False, depending on if the algorithm found any primitive abundant numbers.

---

e_{found} := FALSE
$G := G_{\Delta_+}(P, E)$
$g := |P| - 1$
$i := 0$
while $g \geq 0$ **do**
  $i := G[g]$
  if p_i is crucial in $\nu(P, E)$ **then**
    $\{E_{new}$ is the new exponents raising the exponent that allows the number to be abundant$\}$
    $E_{new} := E$
    $E_{new}[i_p] := E_{new}[i_p] + 1$
    if $\nu(P, E_{new})$ is primitive **then**
      $A := A \& \nu(P, E)$
      e_{found} := TRUE
    end if
    $g := g - 1$
  else
    break;
  end if
end while
if $g \geq 0$ **then**
  {Not all indexed exponents were able to make an abundant when raised one power}
  if $mbAlg(P, E, A, g)$ **then**
    e_{found} := TRUE
  end if
end if
return e_found

Algorithm: mbAlg *(exponent increasing algorithm)*
INPUT: The sequence of primes $P = \{p_0, p_1, ..., p_{d-1}\}$, the exponents to those primes E, the primitive abundants found so far A, and an index, s.
OUTPUT: True or False.

1: e_{found} := FALSE
2: $G = G_{\Delta_+}(P, E)$
3: **while** $g \geq 0$ **do**
4: $\quad G_{b<2} := \{G[0], G[1], ..., G[g], \}$
5: $\quad$ **if** $mb(P, E, G_{b<2}) > 2$ **then**
6: $\qquad E_{new} := E$
7: $\qquad i_p := G[g]$
8: $\qquad E_{new}[i_p] := E_{new}[i_p] + 1$
9: $\qquad$ **if** $crucialAlg(P, E_{new}, A)$ **then**
10: $\qquad\quad e_{found}$:= TRUE
11: $\qquad$ **end if**
12: $\qquad g := g - 1$
13: $\quad$ **end if**
14: **end while**
15: **return** e_{found}

Proof. Let $n = \nu(Q, F)$ be an odd primitive abundant number for a sequence of primes Q and a sequence of positive integers F, where Q and F have the same cardinality. We assume that F contains some element not equal to 1, otherwise the proof is trivial. We want to show that n is collected in A, as this would prove our theorem. Suppose that $m = \sum_{e \in F}(e)$

Let ℓ be a divisor of n. Then $\ell = \nu(Q, F')$ for some sequence of nonnegative integers F'. We say that ℓ is *passed* in the algorithm if the exponent sequence F' is stored in either E or E_{new} at some point when the algorithm is being executed. The *first appearance* of ℓ is the first time in which ℓ is passed during the execution of the algorithm.

We want to show that n is collected in this algorithm, and later in this proof we will show that passing n will force the algorithm to collect n. In order to show that n is passed in the algorithm, we shall establish a more general statement, namely that the algorithm will pass at least one divisor of n of the following orders: $\{m - d, m - d - 1, ..., 0\}$. Since n itself is the only divisor of n of order 0, this would be sufficient to establish the more general statement. We proceed by reverse induction.

Our base case is: *The algorithm passes some divisor of order $m - d$.* This is clearly true as $\ell = p_0 p_1 ... p_{d-1}$ is the first integer the algorithm passes, and this divisor has order $m - d$.

Our inductive step is: *Suppose the algorithm passes a divisor of order k, where $0 < k \leq m-d$. We want to show there exists a divisor of order $k - 1$ that is passed in this algorithm.*

Assume the inductive hypothesis is true for some divisor of n of order k. Denote this divisor $\ell_k = (Q, F_k)$. Note that ℓ_k must make its first appeance whenever E is modified or an E_{new} is initialized. Thus ℓ_k must make its first appearance in one of two places: It can appear in *main* in line 4 or *mbAlg* in line 8. Note that this divisor cannot appear in *crucialAlg* in line 10, for if ℓ_k did first appeared here, the if statement on line 7 would imply that ℓ_k is abundant.

We will now argue that regardless of where ℓ_k first appears, the algorithm will call *crucialAlg* with parameters $P := Q$, $E := F_K$, and $A := A^2$:

[2]The state of A does not matter

Suppose that ℓ_k first appears in *main* in line 4. The algorithm calls *expAbundant* on line 8. Note that, as $k \neq 0$, we have that $b(\ell_k) < 2$. Thus the algorithm moves on to $crucialAlg(Q, F_k, A)$.

Suppose that ℓ_k first appears in *mbAlg* in line 8. Notice that the algorithm will immediately call $crucialAlg(Q, F_k, A)$ on the very next line.

As desired, $crucialAlg(Q, F_k, A)$ will be called.

At this point we split off into cases, the case where $k = 1$ and the case where $2 \leq k \leq m - d$. This is important for our inductive step.

Case 1: Suppose that $2 \leq k \leq m - d$. In $crucialAlg(Q, F_k, A)$, notice that $G_{\Delta_+}(Q, F_k)$ is calculated on line 2. Since $k \geq 2$, we claim that all incomplete primes in ℓ_k are not crucial primes. Let q be an incomplete prime in ℓ_k. Then $\ell_k q \mid n$, and in fact $\ell_k q$ has order $k - 1$. Since $k \geq 2$, we have that $k - 1 \geq 1$. Thus $\ell_k q$ is a proper divisor of n and is therefore deficient. Thus q is not a crucial prime in ℓ_k.

For convenience, we will refer to $G_{\Delta_+}(Q, F_k)$ by G. By Theorem 3.2, there exists a positive integer h such that all elements that precede $G[h]$ are not indices of crucial primes and all elements that succede $G[h]$ are indices of crucial primes. Since ℓ_k contains at least one non-crucial prime (as it contains at least incomplete prime), we know that this h can be chosen such that $G[h]$ is the index of some non-crucial prime.

Informally, we can see that G is ordered in a way where all crucial prime indices are on the right (Theorem 3.2), and all incomplete prime indices are on the left (as $k \geq 2$)

$$G = \{\text{... other indices and incomplete prime indices} \mid \text{crucial prime indices...}\}$$

We know that the while statement on line 5 in $crucialAlg(Q, F_k, A)$ will terminate once $g = h$, as this point is the first time where g is the index of a non-crucial prime. When the loop terminates, the algorithm will call $mbAlg(Q, F_k, A, g)$, as $g \geq 0$.

In $mbAlg(Q, F_k, A, g)$, we have that $g \geq 0$. Thus we will descend into the while loop at line 3. Notice that $G_{b<2} = \{G[0], G[1], ..., G[g]\}$. Thus $G_{b<2}$ contains all incomplete prime indices of ℓ_k. This allows us to state that $mb(Q, F_k, G_{b<2}) > 2$.

Consider the first time that $i = G[g]$ is the index of some incomplete prime p_i of ℓ_k. At this point, all incomplete prime indices still exist in $G_{b<2}$. Thus $mb(Q, F_k, G_{b<2}) > 2$. Thus the execution of this algorithm will reach line 8 in *mbAlg*. Now $p_i \ell_k$ will be passed in the algorithm, and $p_i \ell_k$ is a divisor of order $k - 1$. Thus the inductive step has been proven.

Case 2: Suppose that $k = 1$. Recall that we are starting from $crucialAlg(Q, F_k, A)$. In this case, we note that there is only one incomplete prime p and it must be a crucial prime(as $l_k p = n$ is abundant) As all crucial prime indices will pass the if statement on line 7 in *crucialAlg*, we have that our incomplete prime's exponent will be increased and the result will be n. Therefore n will be collected in A. Thus the inductive step has been proven. We have also shown that n has been collected in A by the algorithm, proving the theorem. □

4 Prime Sequences: Theorems and Definitions

We start this section with some useful definitions to simplify notation for the theorems that follow. Let $P = \{p_0, p_1, ..., p_d\}$ be an increasing sequence of prime numbers while q is a prime number greater than p_d. We shall write $P + q$ to denote the sequence $P + q = \{p_0, p_1, ..., p_d, q\}$. If n is an integer having exactly k prime divisors such that $k > d$ and P composes the smallest i prime divisors of n, then n is referred to as "P-initiated". For example, $3^2 7^{11} 17^1 29^3$ is $\{3, 7\}$-initiated.

Theorem 4.1. *Suppose that $b_\infty(\{p_1, p_2, ..., p_i\}) > 2$, where $p_1 < p_2 < p_3 < ... < p_i$. Then there exists some positive integer ℓ less than or equal to i such that $\{p_1, p_2, p_3, ..., p_\ell\}$ composes the prime factors of some OPAN.*

Proof. Let ℓ be the smallest positive integer such that $b_\infty(\{p_1, p_2, ..., p_\ell\}) > 2$. Note that $2 = b_\infty(\{2\}) \geq b_\infty(\{p\})$ for any prime p. By Theorem 2.2, there does not exist an abundant with exactly one prime divisor. Thus $\ell \geq 2$.

For any positive integer k such that $k \leq \ell - 1$, we have that $b_\infty(p_k) > b_\infty(p_l)$. Thus for any proper subsequence S of $P' = \{p_1, p_2, ..., p_\ell\}$, we have that $b_\infty(S) \leq 2$. Using Theorem 2.2, we can see that there does not exist an abundant number whose prime divisors compose S. Since $b_\infty(P') > 2$, there exists an abundant number whose prime divisors compose P'. Suppose m is such an integer. Then m is an OPAN or a divisor of m is an OPAN. Note that any divisor of m which is an OPAN must have prime factors that compose P'. Thus there exists an OPAN with divisors composing P'. □

Theorem 4.2 (Prime Divisibility of Abundant Numbers)**.** *Let P be an increasing sequence of odd primes. Let r and s be primes such that $max(P) < r < s$. Suppose there exists a $(P+s)$-initiated OPAN with d divisors. Then there exists a $(P+r)$-initiated OPAN with at most d divisors.*

Proof. By hypothesis, there exists a $(P+s)$-initiated OPAN with d divisors. Suppose n is such an $OPAN$. Note that n is in the form $n = \nu(P+s, M_{(P+s)})\nu(Q, M_Q)$ for sequences of positive integers $M_{(P+s)}, M_Q$, and some increasing sequence of primes Q such that $s < min(Q)$.

We want to consider $m = \nu(P+r, M_{(P+s)})\nu(Q, M_Q)$. The integer m is abundant as $r < s$. If m is primitive, we are done. Henceforth suppose m is not primitive. We will show that every primitive abundant divisor of m must be $(P+r)$-initiated(Proving the theorem).

Consider the maximum integer k such that $r^k | m$. Notice that $d = m/r^k$ is a divisor of n as $m/r^k = n/s^k$. Thus d is deficient and every divisor of d is therefore deficient. Thus an abundant divisor of m must be divisible by r.

Let $p \in P$. We now show that $2 > b(m/r^k) > b(m/p^t)$, where t is the maximum integer such that $p^t | m$. Note that $b(m/r^k) = b(m)/b(r^k)$, and likewise $b(m/p^t) = b(m)/b(p^t)$. Also, $b(r^k) < b(p) \leq b(p^k)$ by Theorem 2.3. Hence, $2 > b(m/r^k) > b(m/p^t)$. Therefore any primitive abundant divisor of m must be divisible by every prime in $P+r$. Since there must exist at least one odd primitive abundant divisor of m, there must exist a $(P+r)$-initiated OPAN with at most d divisors. □

Corollary 4.1 (Continuity of Primitive Abundant Numbers)**.** *Let $P = \{p_0, p_1, ..., p_{d-2}, s\}$ be an increasing sequence of primes. Suppose that $n = \nu(P, M)$ is a primitive abundant number for some sequence of positive integers M. Let s be a prime such that $p_{d-2} < r < s$, and let $P' = \{p_0, p_1, ..., p_{d-2}, r\}$. Then there exists some sequence of positive integers E such that $\nu(P', E)$ is a primitive abundant number.*

Proof. This is a special case of Theorem 4.2, where $d = |(P+s)|$. □

We now define a certain class of primes. Suppose P is an ordered sequence of i primes. Suppose d is an integer greater than i. Let p be the largest prime such that there exists a $(P+p)$-initiated OPAN having fewer than d prime divisors. We know this prime p exists since there exist only finitely many primitive abundant numbers having fewer than d prime divisors [5]. We denote this prime p as $Cap_d(P)$.

Theorem 4.3. *Let P be an ordered sequence of primes, and let r and s be primes such that $max(P) < r < s$. Suppose d is an integer such that $d > |P|$. Finally, suppose that $r > Cap_d(P)$. If there does not exist any $(P+r)$-initiated OPAN's with d divisors, then there does not exist any $(P+s)$-initiated OPAN's with d divisors.*

Proof. Assume, to the contrary, that there does exist a $(P+s)$-initiated OPAN with d divisors. By Theorem 4.2, there exists a $(P+r)$-initated OPAN having d or fewer divisors. However, $r > Cap_d(P)$, and by definition of $Cap_d(P)$, there cannot exist a $(P+r)$-initiated OPAN having fewer than d divisors. This is a contradiction. □

To calculate $Cap_d(P)$, we simply rely on knowing all OPANs with $d-1$ divisors.

5 The Main Algorithm

In order to describe the algorithm in this section, we need to develop more definitions and theorems. Let d be an integer greater than 2. We define the set $\mathcal{P}_d$ as follows: Suppose that P is an increasing sequence of odd primes with d elements. Such a sequence P is a member of $\mathcal{P}_d$ if and only if there exists some OPAN n in the form $n = \nu(P, E)$ for some sequence of positive integers E. Note that the set $\mathcal{P}_d$ is finite as there exists only finitely many OPANs with d divisors[5].

Let P be an element of $\mathcal{P}_d$ for some integer $d \geq 3$. We define the set of positive integer sequences $\mathcal{E}_P$ as

$$\mathcal{E}_P = \{E \,|\, \nu(P, E) \text{ is an OPAN}\}.$$

Now we want to develop an ordering on $\mathcal{P}_d$. We will use a lexicographical ordering, comparing each element sequentially. Let $P, Q \in \mathcal{P}_d$. Write $P = \{p_0, p_1, ..., p_{d-1}\}$ and $Q = \{q_0, q_1, ..., q_{d-1}\}$. We say that $P < Q$ if there exists some index i such that $p_i < q_i$ and for each index $k < i$ we have that $p_k = q_k$. Naturally, we say that $P \leq Q$ if $P < Q$ or $P = Q$.

Recall that we defined an integer n to be P-initiated if the smallest $|P|$ prime factors of n compose P. We now generalize this notation to include both integers and sequences. Suppose Q is a sequence of d elements and P is a sequence of $i < d$ elements. We say that Q is P-initiated if the first i elements of Q is the sequence P.

As $\mathcal{P}_d$ is a finite set with a total ordering, $\mathcal{P}_d$ has a minimum and a maximum element. We now want to find the minimum of $\mathcal{P}_d$ for $d = 3, 4, 5, 6$ as this will help us in the base case of Theorem 5.1. Dickson showed that $min(\mathcal{P}_3) = \{3, 5, 7\}$ and $min(\mathcal{P}_4) = \{3, 5, 7, 11\}$. This leaves us to show $min(\mathcal{P}_5) = \{3, 5, 7, 11, 13\}$ and $min(\mathcal{P}_6) = \{3, 5, 7, 11, 389, 397\}$.

To show that $min(\mathcal{P}_5) = \{3, 5, 7, 11, 13\}$, we use the fact that $3^1 5^1 7^1 11^1 13^1$ is an OPAN. There cannot exist a set of 5 odd primes $P < \{3, 5, 7, 11, 13\}$, and therefore we are done.

To show that $min(\mathcal{P}_6) = \{3, 5, 7, 11, 389, 397\}$, we note that $3^1 5^1 7^1 11^1 p^1$ is an OPAN when p is a prime such that $13 \leq p \leq 383$. Therefore, any lexicographical predecessor of $\{3, 5, 7, 11, 389, 397\}$ cannot exist in $\mathcal{P}_6$ as it would not meet the requirement of being associated with an OPAN with 6 divisors. Since $3^1 5^1 7^1 11^1 389^1 397^1$ is an OPAN, we have that $\{3, 5, 7, 11, 389, 397\}$ is the minimum element in $\mathcal{P}_6$.

To generalize Theorem 5.1, it would be useful if we could find the minimum of $\mathcal{P}_d$ for any d. This, however, goes beyond our scope of research.

Suppose $P \in \mathcal{P}_d$. We define $lex(P)$ to be the set of all $Q \in \mathcal{P}_d$ such that $Q < P$. For example, $lex(\{3, 5, 13\}) = \{\{3, 5, 7\}, \{3, 5, 11\}\}$.

Theorem 5.1. *The following algorithm determines all primitive abundant odd numbers with d divisors for $d \in \{3, 4, 5, 6\}$.*

Conjecture 1. *The following algorithm determines all primitive abundant odd numbers with d divisors, for any integer $d \geq 3$.*

The Main Algorithm
INPUT: A positive integer, d. **OUTPUT:** All OPAN's with d prime factors.

```
For all Q, set T(Q) = false
P := {}
running := TRUE
while running do
  if |P| ≠ d then
    if b_1(P) ≥ 2 then
      Remove the last element of P
      Let q be the minimum prime s.t. b_1(P + q) < 2
      P := P + q
    end if
    Let s be the minimum odd prime such that s > max(P) and P + s has never been
    stored in P before.
    P := P + s
    continue
  end if
  if b_∞(P) ≤ 2 then
    fail(P, running)
    continue
  end if
  if E_P is not empty then
    Store (P, E_P)
    r = nextprime(max(P))
    Replace the last prime in P with r
    continue
  end if
  s = backup(P)
  if |P| = d − 1 then
    fail(P, running)
    continue
  end if
  if s ≤ Cap_d(P) then
    P := P + nextprime(s)
    contiue
  end if
  fail(P, running)
end while
```

fail
INPUT: An increasing sequence of primes P and a boolean variable *running*.
OUTPUT: No output. Modifies inputs.

```
P' := P
while P' ≠ {} do
   Remove the last prime from P'
   if there exists A P'-initiated OPAN in lex(P) then
      break
   end if
   T(P') := TRUE
end while
if P' = {} then
   running := FALSE
   return
end if
while T(P') do
   Remove the last prime from P'.
end while
P := P'
return
```

backup
INPUT: An increasing sequence of primes P.
OUTPUT: The last prime removed from P.

```
P' := P
while P' ≠ {} do
   q := Max(P')
   Remove the last prime from P'
   if there exists A P'-initiated OPAN in lex(P) then
      break
   end if
end while
return q
```

Proof. It is noteworthy that this algorithm works in the lexicographical order defined on $\mathcal{P}_d$. We will leverage this so that we can induct on $\mathcal{P}_d$, showing that each member $Q \in \mathcal{P}_d$ will be recorded with its associated exponent sequences $\mathcal{E}_Q$.

Our base case for the proof is to show that $min(\mathcal{P}_d)$ is stored with its associated exponent sequences. For the cases $d = 3, 4, 5, 6$, this can be done computationally.

Our inductive step is as follows. Suppose Q, R are elements in $\mathcal{P}_d$ such that Q is the unique predecessor to R. If $(Q, \mathcal{E}_Q)$ is collected in the algorithm, then $(R, \mathcal{E}_R)$ is collected in the algorithm.

Suppose that Q and R are the same sequence apart from their respective last primes q and r respectively. By Corollary 4.1, we have r is the prime successor to q. When the algorithm collects $(Q, \mathcal{E}_Q)$ in line 20, the algorithm will immediately move on to transform $P := Q$ into $P := R$. Following the algorithm step by step will show that $(R, \mathcal{E}_R)$ is collected.

Now suppose that Q and R are not the same sequence when ignoring the last element of Q

and R. Thus Q and R are in the form

$$\begin{aligned} Q &= \{q_0, q_1, ..., q_{c-1}, q_c, q_{c+1}, ..., q_d\} \\ R &= \{q_0, q_1, ..., q_{c-1}, r_c, r_{c+1}, ..., r_d\} \end{aligned} \tag{5}$$

where $q_c < r_c$. As there is no $S \in \mathcal{P}_d$ such that $Q < S < R$, we can see that the algorithm will keep failing on line 34, until $P = \{q_0, q_1, ..., q_{c-1}\}$ The algorithm will then try the next prime s after q_c on line 12. Suppose $r = r_c$. It can be shown that from here that the algorithm will collect $(R, \mathcal{E}_R)$.

Henceforth assume that $nextprime(q_c) < r_c$. By Theorem 4.3, we have that $r_c \leq cap_d P$. Thus the algorithm will increment its last element as it will continue to execute the line 31 until the last prime of P is equal to r_c It can be shown that from here that the algorithm will collect $(R, \mathcal{E}_R)$.

THIS PROOF NEEDS WORK. □

6 Increasing Computational Efficiency

The exponent algorithm is a very computationally expensive algorithm. If we can avoid executing the exponent algorithm, then the amount of time it takes to find all OPAN's with d divisors will be drastically reduced. We make the following claim:

Lemma 6.1 (Efficiency Lemma). *Let P be an increasing sequence of odd primes. Suppose that p is a prime greater than the last prime of P and q a prime greater than p. If $\mathcal{E}_{(P+p)} \neq \emptyset$ and $\mathcal{E}_{(P+p)}$ is a subset of $\mathcal{E}_{(P+q)}$, then $\mathcal{E}_{(P+p)} = \mathcal{E}_{(P+q)}$.*

Proof. Suppose, to the contrary, that the theorem is not true. Then there exists some $E \notin \mathcal{E}_{P+p}$ such that $\nu(P+q, E)$ is an OPAN. As E is not in $\mathcal{E}_{P+p}$, we have that $\nu(P+p, E)$ is not an OPAN. However, $\nu(P+p, E)$ is abundant as $p < q$. Hence there exists a divisor of $\nu(P+p, E)$ which is an OPAN. Suppose a is such a divisor.

We will now show that the prime divisors of a compose $P+p$. We know that p must divide a, for if p does not divide a, then a is an abundant proper divisor of $\nu(P+q, E)$(a contradiction). Let k be the maximum integer such that p^k divides a. We know that a/p^k is deficient as it is a divisor of $\nu(P+q, E)$. Using a similar argument, we can show that all primes of the set P must divide a. Thus the prime factors of a compose $P+p$.

We have shown that each prime number in $P+p$ divides a, and therefore a can be written in the form $a = \nu(P+p, E')$ for some sequence of positive integers E'. As a is an OPAN, we have that $E' \in \mathcal{E}_{(P+p)}$. By hypotheses, $\mathcal{E}_{(P+p)}$ is a subset of $\mathcal{E}_{(P+q)}$. This implies that $E' \in \mathcal{E}_{(P+q)}$. Therefore $a_q = \nu(P+q, E')$ is abundant. However, a_q is a divisor of $\nu(P+q, E)$, and therefore a_q is deficient, a contradiction.

□

Theorem 6.2 (Efficiency Theorem). *Let P be an increasing sequence of odd primes. Suppose that p is a prime greater than the last prime of P and q a prime greater than p. If $\mathcal{E}_{(P+p)} \neq \emptyset$ and $\mathcal{E}_{(P+p)}$ is a subset of $\mathcal{E}_{(P+q)}$, then for any prime r such that $p < r < q$, we have that $\mathcal{E}_{(P+p)} = \mathcal{E}_{(P+r)} = \mathcal{E}_{(P+q)}$.*

Theorem 6.2 seems to apply to a massive amount of cases, especially as the last prime grows larger and larger. Because of this, Theorem 6.2 allows us to avoid the exponent algorithm, which is a very computationally expensive algorithm. This theorem allows all primitive abundant numbers with 5 divisors to be computed in just a couple of minutes, where previously it took about an hour. Using multithreading abilities on modern processors, the time can be cut even further.

Proof. If $\mathcal{E}_{(P+p)} = \{\{1,1,1,...,1\}\}$, then the theorem is trivial. Henceforth we shall assume that for each element E in $\mathcal{E}_{(P+p)}$, E contains at least one term greater than or equal to 2.

Suppose that $\mathcal{E}_{(P+p)} = \mathcal{E}_{(P+q)}$. Suppose, to the contrary, that $\mathcal{E}_{(P+r)} \neq \mathcal{E}_{(P+p)}$. Then there exists some $E \in \mathcal{E}_{(P+r)}$ such that $E \notin \mathcal{E}_{(P+p)}$, or there exists some $E \notin \mathcal{E}_{(P+r)}$ such that $E \in \mathcal{E}_{(P+p)}$.

Suppose the former case is true. Let E be an element of $\mathcal{E}_{(P+r)}$ such that $E \notin \mathcal{E}_{(P+p)} = \mathcal{E}_{(P+q)}$. It is easy to show that $\nu(P+q, E)$ is not abundant. If $\nu(P+q, E)$ were abundant, then $\nu(P+q, E)$ would be forced to be primitive abundant which would imply $E \in \mathcal{E}_{(P+q)}$. It is also easy to show $n_p = \nu(P+p, E)$ is abundant. Since $d = \nu(P+p, \{1,1,1,...,1\})$ is deficient and divides the abundant number n_p, there exists a primitive abundant odd which is a multiple of d and a divisor of n_p. Such a divisor would be in the form $\nu(P+p, E')$ for some sequence of positive integers E'. This would imply $E' \in \mathcal{E}_{(P+p)}$. However, this would create a contradiction as $E' \notin \mathcal{E}_{(P+q)}$ and $\mathcal{E}_{(P+p)} = \mathcal{E}_{(P+q)}$.

Now suppose there exists some $E \notin \mathcal{E}_{(P+r)}$ such that $E \in \mathcal{E}_{(P+p)}$. From this we know that $E \in \mathcal{E}_{(P+q)}$ and therefore $n = \nu(P+r, E)$ is abundant. We also know that for each divisor $\nu(P+r, E')$ of n, we have that $2 > \nu(P+p, E') > \nu(P+r, E')$. Thus n is a primitive abundant number, as each of its divisors are deficient. This implies that $E \in \mathcal{E}_{(P+r)}$, a contradiction. □

[5]

References

[1] S. J. Benkoski and P. Erdös. On weird and pseudoperfect numbers. *Mathematics of Computation*, 28(126):617–623, 1974.

[2] Valdas Dičiūnas. On the number of odd primitive abundant numbers with five and six distinct prime factors. Vilinus Conference in Combinatorics and Number Theory, page 12, 2017.

[3] Gianluca Amato, Maximilian F. Hasler, Giuseppe Melfi, and Maurizio Parton. Primitive abundant and weird numbers with many prime factors, 2018.

[4] N.J.A. Sloane. The on-line encyclopedia of integer sequences. `www.oeis.org`.

[5] Leonard Eugene Dickson. Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors. *American Journal of Mathematics*, 35(4), 1913.