Construction and Analysis of Greedy Sequences of Non-Negative Integers

Atanas Iliev

Under the direction of Dr. Katerina Velcheva

October - March 2021

Contents

Abstract

A "greedy" sequence is a number sequence of non-negative integer in which the first member is 0 every next member is defined as the smallest integer larger than the last term such that a certain equation is avoided for two or more terms. We start by analyzing the Stanley Sequence (avoiding arithmetic progression) beginning with $0, 1$, and construct a non-recursive formula for calculating its terms. We then define some notation and move on to analyzing different greedy sequences. We use different numerical systems to do so and try to conjecture what can be achieved by this approach.

1 Introduction

Let's define $\{a_n\}$ as a "greedy" sequence of integer numbers such that several terms $a_0...a_{t-1}$ are chosen, and every following term $a_i > a_{i-1}$, and a_i equals the smallest such integer for which a certain number of terms avoid a given equation or a set of equations. One such case is a sequence G where the first 2 terms are chosen (for example $a_0 = 0$, and $a_1 - 1$) in which no 3 terms (not necessarily consecutive) form an arithmetic sequence.

This is the same as saying that there are no three terms a_a, a_b, a_c for which $a_a + a_c = 2 \cdot a_b$ [1]

The first few terms of the sequence where $k = 1$ are: $0, 1, 3, 4, 9, 10, 12, 13, ...$

We first examine these sequences for different values of k. Then we provide a "nice" description for a_n, and construct a formula for calculating this. Then we also examine "greedy" sequences beginning in the same way as the original one but avoiding a different or several different equations.

2 The Case $k = 1$

This is, obviously, one of the more simple variations of a "greedy" sequence. In the following sections we will systematize our notation in order to provide a hierarchy for the different "greedy"" sequences we analyze but for now let's stick with the former notation. Let's denote a "greedy" sequence as $G(k)$ making the one produced by $k = 1$, $G(1)$.

2.1 Initial observations

It is a tedious task to calculate many terms of this sequence by hand, as indicated by the growth rate of the sequence. That's why it can be useful for a simple console application to be constructed. Its job won't be to calculate terms fast, but to calculate them for us the tedious way (by checking for an arithmetic sequence among all three terms combinations). One such is provided at the end

of this paper in $C\#$. [A.1] Please note that all the programs used in this paper were developed by myself.

The first 20 terms are provided in the table below [Table 1]:

Index:	0	1	2	3	4	5	6	7	8	9
Value:	0	1	3	4	9	10	12	13	27	28
Index:	10	11	12	13	14	15	16	17	18	19
Value:	30	31	36	37	39	40	81	82	84	85

Table 1: The first 20 terms of $G(1)$

2.1.1 Explanation of the brute force algorithm used for deriving terms.

Let us quickly explain how the program's algorithm works. First the program receives as input from the user two values - n and k - giving the number of terms desired and the second term of the sequence. Note that by definition the first term of G, $a_0 = 0$, and that the terms that will be received will have indexes ranging from 0 to $n-1$. An array that will store the terms of the sequence is then defined and a_0 and a_1 are defined such that $a_0 = 0$ and $a_1 = k$. Another variable br is then defined such that initially $br = k+1$ which will be used to calculate the next terms.

The program then uses three loops to calculate the terms from a_2 to a_{n-1}. The first is responsible for providing the index of the term - i - and ranges from 2 to $n-1$. The other two provide the other two indexes j and l so that a_i avoids forming an arithmetic progression with a_j and a_l. That being said j ranges from 0 to $i-1$ and l ranges from 1 to $i-1$. When we have chosen i, j, l we simply check if $2 \cdot a_l$ equals $a_j + br$. If that is the case we increase br by one and set j and l to their initial values - 0 and 1 respectively. If not then this is our next term. Therefore we define a_i such that $a_i = br$ and again increase the value of br by one. The value of i is then increased by one for the next term and the process starts again (which resets the values of j and l.

Last, note that this program is by no means fast and is really what one calls a "brute force" algorithm. Instead its goal is to find the terms of G without any equations we derive instead checking one by one if the equation we want to avoid is indeed avoided and picking the the next terms by this operation alone. Now that we made that clear and explained how the program works, let us move to some interesting observations.

2.1.2 Stanley's observation about numerical systems

R. P. Stanley, an emeritus professor at MIT, had an interesting idea that solved this case. In his honor such "greedy" sequences are sometimes called - Stanley

sequences.

Let us write the indexes of these terms in binary and their values in ternary. For the first 10 terms these are as follows:

Index(2):	0	1	10	11	100	101	110	111	1000	1001
Value(3):	0	1	10	11	100	101	110	111	1000	1001

Table 2: The first 10 indexes and terms of $G(1)$, written in binary and ternary

It can be seen that when expressed in this way the indexes and the values are the same (in terms of representation, and NOT value).[2]

This remains true, as this paper will prove, for $i \to \infty$ where i is the index of the term of $G(1)$. This means that one way of calculating a_i is writing i in binary and reading it in ternary. This paper considers such description a "nice" one. That being said, it is by no means an explicit formula yet. 1

2.2 Finding the terms of $G(1)$

One of the important results proved in this section (first noted by R. P. Stanley[3]) is the following theorem:

Theorem 1: *The members of* $G(1)$ *are the integers that does not include a* 2 *when written in ternary.*

Proof: To prove that the former observation is true for every i lets consider what happens when we multiply a number in ternary (by 2). There are three digits that can be multiplied. For each of them we receive:

$$0 \to 0$$

$$1 \to 2$$

$$2 \to 1$$

where the '$\to$' indicates "is overwritten by". As one can see the only way for a 2 to be obtained in this way is if the operand is 1.

There are now two things that will want to prove: 1) that the sequence we described above is indeed free of the equation $a_a + a_c = 2 \cdot a_b$ and 2) that there is no sequence lexicographically before the one we described above that also satisfy the former condition.

The first one is particularly easy to prove. First let's note that $a_a \neq a_c$ as the terms in the proposed sequence are by definition different and unique. We shall also note, again by definition, that both a_a and a_c are 2-free in their ternary representation and, hence does not produce carried 1s anywhere when added.

Therefore we can be sure that $a_a + a_c$ possesses at least one 1 in its ternary representation (at a place where a_a differs from a_c). The multiplication of a_b by 2 also does not produce carried 1s anywhere for the same reasons. Therefore it consists only of 0s and 2s in ternary proving the statement.

Now for the second part let's assume a lexicographically smaller such sequence $\{b_n\}$ exists. Let m be minimum where $b_m \neq a_m$ for the first time, and therefore $b_m < a_m$. Now consider that all 2-free ternary numbers smaller than a_m are exactly $a_0, a_1, ..., a_{m-1}$, so we may conclude that b_m contains at least a single 2 in its ternary representation. Let α be the number that in ternary has digit 0 where b_m has digit 0 and digit 1 in all other cases. Now let β be the number that in ternary has digit 1 where b_m has digit 1 and digit 0 in all other cases. Then both α and β are 2-free in ternary and both are smaller than b_m. This means they both must be terms of $\{b_n\}$. It is now easy to see that $b_m + \beta = \alpha$ as when we add each digit of β (in ternary) we turn every 1 in b_m into a 2, thus obtaining α.

As we have now proven this first theorem and obtained the values of each member of $G(1)$ we shall move on to constructing an explicit formula/function that results in the correct value for a_i when provided with input i (the index of the term). It is also important to note that while the observation was initially made by Professor Stanley, he did not provide an explicit proof of it in his paper, only suggesting how it can be done (a different method)[3].

2.3 Constructing a formula for the calculation of a_n

In order to do so, we will first construct a way to obtain the specific digits of the number in ternary and then convert them to the number system of base 10. As we have already established these digits (0s and 1s) are actually the same as in the binary representation of the index of the term we are looking for. This implies that the first part of this task is equivalent to finding a way to convert the index of the term in question from decimal to binary.

We will first explain the method of conversion and will do this by an example - converting 10(10) to 1010(2). We start by dividing 10 by 2 and obtaining 5. Since 5 is a whole number and thus we have found a degree of 2 in 10 we write a 0 at the end of the binary representation. Then 5 divided by 2 is 2.5 so we get the whole part of this and a 1 next to the 0. 2 divided by 2 is again whole - 1, so we add another 0. We finish by adding the remaining 1 at the front.

Let us now consider how many times should we perform the operation in order to find the binary representation of a_n where n is given. One trivial approach is to use $\log_2 n$ and since a number of operations must be a whole number to round it down, hence obtaining $\lfloor \log_2 n \rfloor$.

What we want to do now is divide n by two $\log_2 n$ times and concluding

if we should write down a 0 or a 1 after each time. One way to do this is to subtract $\lfloor \frac{n}{2^i} \rfloor$ from $\frac{n}{2^i}$ where i is the degree on which we divide. If they are equal we will get 0 and if they are not some real number. Then we should just simply round the whole expression which will result in the wanted 1 if the two terms are different by more or equal to 0.5. Therefore:

$$\left[\frac{n}{2^i} - \left\lfloor \frac{n}{2^i} \right\rfloor \right] \tag{1}$$

Then we have to multiply the received digit by the correct power of 3 since ultimately we want to read the number in ternary to convert it into decimal. Since the first division produces the rightest most digit of the binary number we can conclude that reading it in ternary will correspond to the 0th power of 3. Hence we receive:

$$\left[\frac{n}{2^i} - \left\lfloor \frac{n}{2^i} \right\rfloor \right] \cdot 3^{i-1} \tag{2}$$

Finally, we want to add all powers of 3 to get the term in decimal. This summation has to consider the number of divisions we will perform on the index n of the number a_n we are looking for. We can conclude from equation (2) that we will always have to add the biggest number of 3 contained in n which as we have explained above will always be $3^{\lfloor \log_2 n \rfloor}$. This means the exponents of 2 on which we will divide n range from 1 to as already established $\lfloor \log_2 n \rfloor$. Therefore we can expand on (2) in order to conclude the formula as:

$$a_n = \sum_{i=1}^{\lfloor \log_2 n \rfloor} \left[\frac{n}{2^i} - \left\lfloor \frac{n}{2^i} \right\rfloor \right] \cdot 3^{i-1} + 3^{\lfloor \log_2 n \rfloor} \tag{3}$$

Notice that while this formula is not nearly as "nice" as we would have initially hoped as it relies pretty heavily on some piece-wise functions like the rounding and the floor ones, and involves a sum in it, the formula is still non-recursive. In other words, we are able to calculate a_n by using a formula that does not need to replace the initial input it is given (in our case - n) after each iteration of the sum.

This concludes the analysis of $G(1)$. In the next sections we will consider some other cases for k, and what happens when we want to avoid another equation instead of $a_a + a_c = 2 \cdot a_b$. First, however, let us get back to our initial definition of a "greedy" sequence, make some observations about the dependencies in the to-be-avoided equation(s), and define order of dependency.

3 Order of Dependency, Greedy Sequences Where $O(G) \leq 3$

3.1 Order of dependency $O(G)$

When we first explain what a greedy sequence (and more specifically what a Stanley sequence) is we mentioned that the conditions regarding their construc-

tions can be summarized in an equation or a set of equations that we want to avoid. In the case of the Stanley sequences this was the equation for three consecutive terms of an arithmetic progression. This equation was in the form: $x + z = 2 \cdot y$. We can rewrite this as $x - 2 \cdot y + z = 0$, and so we want to avoid this which is equivalent to following $x - 2 \cdot y + z \neq 0$ where x, y, and z are any 3 terms of the Stanley sequence.

Let us now formulate the term order of dependency. We will say that the Stanley sequences are of order of dependency 3 since the sum of 3 different terms multiplied by coefficients should not equal 0.

Now let us formally define the function for the order of dependency O for a greedy sequence G, so that $O(G) = t$ where t is the number of terms which when multiplied by various coefficients and added together should not equal 0, for them to be all terms of G. Therefore for $p_1...p_t \neq 0$:

$$p_1 \cdot a_1 + p_2 \cdot a_2 + ... + p_{t-1} \cdot a_{t-1} + p_t \cdot a_t \neq 0 \Rightarrow O(\{a\}) = t$$

It is important to understand that here the indexes of the terms are not necessarily corresponding to the actual terms with these indexes, but are used for illustrative purposes. Let us now note that these are the only type of equations we will follow with the exception of $a_n > a_{n-1}$ where a_n and a_{n-1} are members of the greedy sequence $\{a\}$. Another important note is that if we have more than 1 equation of this form concerning different terms we can simple add them together into one that should not equal 0 and use it to calculate the order of dependency of the sequence.

We will now analyze several easy to describe values of $O(G)$ and how knowing these values help us draw conclusions or find the members of G.

3.2 Greedy sequences G for which $O(G) < 3$

First let us identify what values can $O(G)$ where G is a greedy sequence can assume. Because it counts a number of terms we know immediately that either $O(G) = 0$ or $O(G) \in \mathbb{N}$. Now remember that by definition if a_i and a_j are terms of the greedy sequence G and $i < j$ we know that $a_i < a_j$ by definition.

3.2.1 Range of $O(G)$

The former observation, however is the same as following the equation $a_j - a_i \nleq 0$ which includes in itself the equation $a_j - a_i \neq 0$. Therefore, we can conclude that just by definition for a greedy sequence G, $O(G) \in \mathbb{N}$. Even more this implies that $\forall G$, $O(G) \geq 2$. We will now examine some easy to analyze cases.

3.2.2 Sequences G for which $O(G) = 2$, and p_i, p_j have the same sign

We have already figured that each greedy sequence has an order of dependency of at least 2. That being said, by the definition of order of dependency, a greedy

sequence G for which $O(G) = 2$ will have to follow the equation:

$$p_i \cdot a_i + p_j \cdot a_j \neq 0$$

Let us say without loss of generality that $j > i$. Then, we can rewrite this as:

$$p_i \cdot a_i \neq -p_j \cdot a_j \Leftrightarrow a_j \neq -\frac{p_i}{p_j} \cdot a_i$$

Notice that by definition a_i and a_j are strictly non-negative integer numbers, and when it comes to the coefficients we know that $p_i, p_j \in \mathbb{R}$, and $p_i, p_j \neq 0$.

Now first we will consider the possibility of p_i, p_j having the same sign (both being positive or both being negative). This immediately implies that a_j should just avoid being negative because $a_i \geq 0$ and $a_j > a_i$.

Let us summarize three distinct rules we will follow when constructing greedy sequences of a given order of dependency. First, the 0th term of a sequence will always be 0. Next, we will always define a_i as the smallest number bigger than a_{i-1}, and third, we will always "choose" and not construct $t - 1$ terms of the G sequence where $O(G) = t$. This is because we can rewrite the definition of $O(G)$ as a new equation:

$$p_1 \cdot a_1 + p_2 \cdot a_2 + ... + p_{t-1} \cdot a_{t-1} \neq -p_t \cdot a_t$$

This implies that we can first construct a term of the sequence by following a rule different then the second one we defined in the paragraph above for every greedy sequence only after we have $t-1$ terms already defined. Also because of the restriction $a_n > a_{n-1}$ we know that if $a_0 = 0$ which is our first restriction $a_j \neq 0$ for $j \neq 0$ which are all the other terms.

Because we know that the current sequences of concern are of order of dependency 2 we will not have to define any terms because $2 - 1 = 1$ except for the 0th which is, by definition, equal to 0. Now let us consider the equation given by the order of dependency. We have already established it is equivalent to a_j not being negative which it never is. Therefore by applying $a_n > a_{n-1}$ and the restriction about the smallest bigger number we can see that only one such sequence exists, hence:

$$G = \{0, 1, 2, ...\infty\}$$

Let us move to the other possible option about the coefficients before the terms in the equation given by the order of dependency.

3.2.3 Sequences G for which $O(G) = 2$, and p_i, p_j have different signs

Look again at the equation given by $O(G) = 2$ in it's already transformed version:

$$p_i \cdot a_i \neq -p_j \cdot a_j \Leftrightarrow a_j \neq -\frac{p_i}{p_j} \cdot a_i$$

Now we will consider what happens with the terms of G when exactly one of p_i and p_j is negative/positive and keep in mind we know that $a_j > 0$ since j_i and therefore $j \neq 0$. We also can deduce that $|p_i| > |p_j|$ because again $a_j > a_i$. If the opposite happens the only possible sequence G where only a_0 is chosen and not constructed is equivalent to the one provided when the coefficients are of the same sign. Another important remark is that if a_n is the general term of G because $O(G) - 1 = 1$ we again only have to choose the 0th which by definition equals 0.

Now we can move on to the construction of the general term of G. Let us choose three different integer numbers α, β, γ so that $\alpha, \beta, \gamma \geq 0$ and $\alpha < \beta < \gamma$. Now we ask if it is possible for a_γ and a_β to satisfy the equation $a_j > a_i$ without a_γ and a_α satisfying it. According to the restrictions discussed above it is not. Then we can conclude that we only need to consider the relationship between a_n and a_{n-1} in order to construct a_n.

We first start according to our initial restriction with $a_0 = 0$. Now since 0 multiplied by any real number is 0 it is obvious that $a_1 = 1$. Let's consider an example equation with given values of p_i and p_j and try to construct a general formula for the terms of G based on this example.

Let us use $p_i = 5$ and $p_j = -3$ (notice that the example abides by the former restrictions for the values of p_i, p_j). This will make the equation given by the order of dependency equivalent to:

$$-\frac{5}{-3} \cdot a_i = \frac{5}{3} \cdot a_i \neq a_j$$

The first few terms of the sequence will then be: $0, 1, 2, 3, 4, 6, 7, 8, 9, 11, ...$ It is easy to see that the numbers excluded from this sequence are the natural numbers which divide 5. However if we had that $p_i = 6$ instead the terms would have become $0, 1, 3, 5, 7, 9, 11, ...$ which are all the odd natural numbers or the ones which do not divide $6/3 = 2$. This means that the numbers we will strive to avoid when constructing the terms of G are the multiples of:

$$\frac{p_i}{\gcd(p_i, p_j)} \tag{4}$$

Note that we will define the $\gcd(a, b)$ function to only have a positive range so that for any two real numbers x, y, we have $\gcd(x, y) > 0$.

Now we should come up with a non-recursive formula that takes an index n and calculates a_n, term of G, without using the value of a_{n-1}. This can simply be done by finding how many additional 1s we will have to add to the index n. This is trivial and equals to:

$$\left\lfloor \frac{n \cdot \gcd(p_i, p_j)}{p_i} \right\rfloor \tag{5}$$

Now to obtain the general term of G, a_n we simply add (5) to the index n:

$$a_n = n + \left\lfloor \frac{n \cdot \gcd(p_i, p_j)}{p_i} \right\rfloor \tag{6}$$

We have now finished the analysis of the most simple "greedy" sequences we move forward to the initial sequence we analyze and some other of the same order of dependency.

3.3 Greedy sequences G for which $O(G) = 3$

Let us start by defining some important notation that will utilize which along side the order of dependency will make it extremely easy to systematize all types of "greedy" system we will be analyzing as well as the ones we have already explained.

We will define $G_O(a_1, a_2, ...a_{t-2})$ as a "greedy" sequence G of order of dependency O and terms with indexes 0 to $t-2$ equal to $0, a_1, a_2, ...a_{t-2}$ respectively. This means that the sequence we analyzed in the second section of this paper can now be denoted as $G_3(1)$, and the ones we analyzed in the former subsections of this section as G_2.

First, we will again turn our attention to the equation provided by the order of the dependency of sequences G_3. This equation is in the form:

$$p_i \cdot a_i + p_j \cdot a_j + p_k \cdot a_k \neq 0$$

.

Notice that this is the first time when will have to "choose" a term of G instead of apply a formula to construct it. Technically we should have also done so when $O(G) = 2$ since $O(G) = t - 1 = 2 - 2 = 1$. Besides a_1 there are several things we can manipulate to achieve infinitely different variations of G_3. These are the different coefficients in front of $\{a_n\}$ denoted in the equation as p_i, p_j, p_k.

Let us again rewrite the equation given by the order of dependency $O(G)$ of G_3 so that exactly one term is on the right side of the equation:

$$p_i \cdot a_i + p_j \cdot a_j \neq -p_k \cdot a_k$$

We will now define a_k as the next term in the sequence we want to find. As one can see there are many different sequences we can make by changing the values of the coefficients $\{p_n\}$. In this subsection we consider two main types.

3.3.1 Stanley sequences

First, let us return our attention to Stanley sequences. These are sequences G with $O(G) = 3$ where $p_i, p_j = 1$ and $p_k = -2$. As noted at the beginning of this paper Stanley sequences are greedy sequences that avoid a three-term arithmetic progression.

Because the coefficients are set in the Stanley sequences the only thing we can modify is what in the beginning of this paper we denoted as k and is simply the value of a_1. In section 2 we proved that the terms of $G_3(1)$ are in decimal their indexes written in binary and read in ternary.

This can be summarized as a really interesting approach. We will attempt to check if two numbers x, y exists such that the terms of a Stanley sequence G in the form a_n are obtained by writing n in a number system of base x and reading it back into decimal using a number system of base y.

In order to do that we again construct a program in $C\#$. The source code is provided below in the appendix section of this paper [A.2]. We will now explain how the program works as well as the ranges of the different variables the algorithm is checking.

The first part of the code is essentially the same as the one we have explained at the beginning of chapter two and its goal is to calculate the necessary terms of $G_3(k)$ with respect to k which is inputted by the user alongside n - the number of terms to be outputted. Again we repeat that this is done by individually checking if each number that is potentially a term does not follow the equation the "greedy" sequence avoids, instead of applying some other algorithm we have suggested.

The latter part is new and has to be explained in a bit more detail. Its purpose is to calculate the terms of a sequence obtained by writing the index of the term in a given numerical system and reading it in another. Again three loops are used - one for the index of the term that is calculated, and two other for the bases of the two numerical systems used for constructing the said terms.

In order for the index to be transformed into the appropriate numerical system, the regular method of conversion is used. The number is continuously divided by the base of the "writing" numerical system denoted in the source code as "$j2$". This is done as long the number is bigger than 0. The remainders are combined into a single string of digits (and letters if the base is big enough), that is then in another loop converted into the "reading" numerical system of base "$i2$" which happens by multiplying each character (digit/letter) of the string by the appropriate power of $i2$. These multiplications are then summed to receive the appropriate term in decimal.

A counter variable that compares the results of the two algorithms that calculate the terms is used. The purpose of it is to calculate how many of the terms are the same as after all we expect to be calculating the terms of a single sequence in two different ways. We check for numerical systems with bases up to 30 and compare the first 100 terms of each sequence.

Unfortunately only when $k = 1$ a similarity of 100% is reached - which is also the sequence we already analyzed in section 2. Normally, the values found for $i2$ and $j2$ were 3 and 2 respectively. The cases when the counter variable was different then 1 (because obviously the term with index 0 is always 0) are such where the same variable is still to small to be considered a meaningful result where the error is technical and not rooted in the hypothesis. However, due to the small amount of data we tested, we are not yet able to conjecture that no two bases of a numerical system exist for which the terms of a Stanley sequence $G_3(k)$ can be calculated (in decimal) by writing the index of a term in one of them and reading it back through the other.

That being said, Stanley was able to come up with some interesting observations concerning his sequences. Namely in one of his papers, he state that it can be proven through a case by case analysis that the terms of Stanley sequences for which the second term is a power of 3 or 2 times a power of 3 follow certain restrictions when written in ternary.[3]

In this paper, we will not delve into them, and will instead consider some other "greedy" sequences. We do this in an attempt to find more general theories applying to a broader range of sequences.

3.3.2 "Stanley-like" sequences

Consider again the equation given by the order of dependency. We explained that the only variables beside the second term of G_3 we can manipulate are the coefficients beside each of the terms participating in this equation. This is how we rewrote the equation given by $O(G)$ in the beginning of this subsection:

$$p_i \cdot a_i + p_j \cdot a_j \neq -p_k \cdot a_k$$

Let us then from now on refer to p_k as p_{right} and to a_k as a_{right}.

Now in order to explain why a_{right} will not be the term we will be finding when applying the equation it is important to consider the sequences we will be analyzing now:

In this sub-subsection we will be changing the value of p_{right} and hence of $-p_{right}$. There are several reasons to decide changing exactly this. First, p_{right} is the only coefficient in the Stanley sequences different than 1 and Stanley sequences are the only "greedy" sequences where there is even one known example of the hypothesis we devised above working. Second, the value of p_{right}

in Stanley sequences (which is by definition -2) equals the base of the "writing" numerical system with opposite signs (binary). Third, in the same paper Stanley points out another interesting observation where $-p_{right} \neq 2$ which will consider in the next section, however.

What we now do is basically test the same hypothesis for this kind of sequences which we formally define like this:

Let G_O be a "Stanley-like" sequence if G_O is "greedy", has an order of dependency $O(G)$ and has all coefficients in the equation given by $O(G)$ to strictly equal 1 with exception for p_{right}.

Before we apply a computer program to test our hypothesis let us consider the special case where $p_{right} > 0$ too. This means that $-p_{right} < 0$ and therefore $-p_{right} \cdot a_{right} < 0$. However, since $p_i, p_j = 1$ it follows that $p_i \cdot a_i + p_j \cdot a_j > 0$. Therefore this sequence is once again:

$$G = \{0, 1, 2, ...\infty\}$$

There is one more interesting case deserving our attention: $p_{right} = -1$ and where we include additional criteria that a_{right} is always the number we are looking for next. This means that we want to construct a sequence in the opposite way of which the Fibonacci sequence is constructed since $-p_{right} = 1$ and $p_i, p_j = 1$ too.

The first few terms of this sequence will then be: $0, 1, 2, 4, 7, 10, 13, 16, ...$ It is easy to see that every term after the 2 will equal:

$$1 + 3 \cdot (n - 2) \tag{7}$$

.

This allows us to easily construct a formula for $\{a_n\}$.To do so first consider the following two equations:

$$\left\lceil \frac{n-2}{n+4} \right\rceil \tag{8}$$

The equation inside the rounding brackets gives -0.5, -0.2, and 0 when n equals $0, 1, 2$ respectively. Also

$$\lim_{n \to \infty} \frac{n-2}{n+4} = 1$$

This implies that if we round it up we will always get a 1 except in the cases described above in which we will get a 0. Now for the second equation we have:

$$\left\lfloor 1 - \frac{n-2}{n+4} \right\rfloor = \left\lfloor \frac{6}{n+4} \right\rfloor \tag{9}$$

Like the previous one we will use (8) for the special cases of $0, 1, 2$. This equation returns $1.5, 1.2, 1$ respectively. Also

$$\lim_{n\to\infty}\left(1-\frac{n-2}{n+4}\right)=\lim_{n\to\infty}\frac{6}{n+4}=1-1=0$$

The implication being that if we round it down we will always get a 0 except for the special cases we discussed above.

Now we need to multiply n by (9) in order to get the value of the special cases and (7) by (8) for the ones with index bigger or equal to 3. This gives us the final formula:

$$a_n = n\cdot\left\lfloor\frac{6}{n+4}\right\rfloor+(1+3\cdot(n-2))\cdot\left\lceil\frac{n-2}{n+4}\right\rceil \tag{10}$$

3.3.3 "Stanley-like" sequences G where a_{right} is always the next term

For the sake of explicitness let us introduce the notation a_{next} for the term we are finding when checking in the equation given by $O(G)$ for a "Stanley-like" greedy sequence G at a certain point in the generation of members. In this part of the section we will analyze the "Stanley-like" sequences for which it is a criteria that $a_{right} = a_{next}$.

Please note that a_{right} is defined by $pright$ which is defined as the only non-one coefficient in the equation given by $O(G)$ for a certain greedy sequence G.

Last, we will always "choose" the smallest possible option for the terms we need to define before we can apply the equation given by $O(G)$ so in our case the first two terms will always be $0, 1$ respectively.

First, let us consider one more example for such sequence G where $O(G) = 3$. This time let $p_{right} = p_{next} = -2$. We will have for the equation given by $O(G)$:

$$a_i + a_j \neq 2\cdot a_{next}$$

We will now find the first few terms of this sequence while keeping in mind that the $a_0 = 0$ and that $a_1 = 1$. Therefore, we get: $0, 1, 2, 3, 4, 5, 6, ...$ It is easy to see that this sequence is equivalent to:

$$G = \{0, 1, 2, ...\infty\}$$

However, now we will try to prove it. We are suggesting that $a_n = n$ and, in order to do so, we will observe that the following inequality is true for $\forall n \in \mathbb{N}$ bigger or equal to 2:

$$n-2+n-1<2\cdot n \Leftrightarrow 2\cdot n-3<2\cdot n \Leftrightarrow 0<3$$

Now a quick explanation of why this is enough to prove the initial hypothesis can be derived by looking into three things. First, the possible values of n in this inequality are all the values of n for the terms a_n we could be searching since $a_0 = 0$ and $a_1 = 1$ by definition. Second, strict smallness implies that the two side of the inequality are not equal as desired in the equation given by $O(G)$ and, third, that proving that $a_{right} = a_{next} = a_n$ is strictly smaller than a_{n-1} and a_{n-2} is enough to say the equation given by the order of dependency is met for every a_i, a_j, a_{next} since the sequence G is, by definition, increasing.

We can now definitively prove that the sequence G as defined in this part of the chapter is one an the same for every value of $p_{right} = p_{next} \leq -2$. We do this in the following theorem:

Theorem 2: *The members of the "Stanley-like" greedy sequence $G_3(0, 1)$ where $a_{right} = a_{next} = a_n$ equal there indexes if $p_{right} \leq -2$.*

Proof: Let n be the index of a_{next} such that $a_{next} = a_n$ and is the term we are finding next. Note that the theorem is always true for a_0 and a_1 since they are, by definition, equal to $0, 1$ respectively. This means that $n \in \mathbb{N}$ and that $n \geq 2$.

Now we consider the equation given by the order of dependency $O(G) = 3$ rewritten in the usual way:

$$a_i + a_j \neq -p_{right} \cdot a_n$$

We will now assume the theorem to be correct which would mean that $a_n = n$ for $\forall n$. Then look at the following inequality:

$$(n-2) + (n-1) < -p_{right} \cdot n$$

Now we explain why proving this inequality is sufficient to prove the theorem. First, please note that the sequence is defined as an increasing one because it is greedy. This implies that proving the inequality for the last two elements before the one we are looking for in enough to prove it for every too and this same one we are searching since $a_{x-1} < a_x$ for $\forall x \in \mathbb{N}$. Then it is also important that "$<$" implies "$\neq$" since it is strict. Finally, we want to prove the inequality for $\forall n$.

That being said, the left side of the equation is equivalent to $2 \cdot n - 3$. On the other hand, for the right side we have that $-p_{right} \cdot n \geq -(-2) \cdot n = 2 \cdot n$. This means we can now state that:

$$2 \cdot n - 3 \leq 2 \cdot n \Leftrightarrow -3 \leq 0 \Leftrightarrow 0 < 3$$

This concludes the prove of the inequality and, therefore, the theorem itself.

We will now continue with a program that tests our previous hypothesis about the Stanley sequences for the "Stanley-like" sequences where $a_{right} \neq$

a_{next}. In the next chapter we will revisit the greedy sequences with this additional constraint as well for larger values of $O(G)$.

3.3.4 Testing a hypothesis for sequences G_3

Let us now consider what happens when $-p_{right} \in \mathbb{N}$ and $p_{right} \neq -1$. We will once more try our hypothesis using a console application because after all the initial case we analyzed in section 2 is both a Stanley and a "Stanley-like" greedy sequence. The source code of the program we use is again written in $C\#$ and is provided in the appendix [A.3]. We will now explain how the program works. Before that, however, please note that we considered $-p_{right} = 1$ to be a special case not just because it is easy to describe but also because we initially expected that is a "writing" and "reading" bases exist, the "writing" one will equal $-p_{right}$ as is the case when $-p_{right} = 2$.

The program we use to check if suitable bases of numerical systems describing the terms of "Stanley-like" sequences exist is pretty similar to the one we used to check if such bases exist for the different Stanley greedy sequences. Namely the method of calculating the "true" terms is the same as the one described in the very first program we discussed but another variable for $-p_{right}$ is introduced instead of 2 which also means there is a loop responsible for the varying range of its value as well as another responsible for checking how many numbers on the left side of the equation must be summed when checking. The way we check for suitable systems is also the same since it only depends on the index of the term we are looking for - n.

The program uses the same ranges for the two bases but now checks values for p_{right} ranging from -999 to -2. Unfortunately again there are no complete matches except for the case we solved in the second section of the paper.

Despite the fact the hypothesis was definitively proven wrong for both Stanley and "Stanley-like" greedy sequences there is still more we can achieve by using numerical systems as a method of describing greedy sequences. It will be shown in the next section that there are still certain points that can be made about the terms of a greedy sequence with respect to numerical systems of different bases. The theory that will form around that, however, is relevant for only one greedy sequence G_O for each value of $O(G)$. That being said, we move to different values of the order of dependency.

4 Greedy Sequences Where $O(G) > 3$

In this section we will consider greedy sequences G_O where $O(G) > 3$. Until now we have only been able to analyze specific cases for different values of $O(G)$ (with the exception of G_2 which is particularly easy to describe) despite making attempts to generalize our observation and hypothesizing about describing

the terms of various greedy sequences using numerical systems of different bases.

We will now try a different approach. We will define a very specific type of greedy sequences building on definitions and insights from previous sections and try to make general observations and inquiries with respect to $O(G)$ in other words we will try to give a specific value for everything but $O(G)$ when analyzing and proving observations about G_O.

4.1 "Stanley-like" sequences where $a_{right} = a_{next}$

In this part of the chapter we revisit the additional constraint we add to the "Stanley like" sequences such that $a_{right} = a_{next} = a_n$ for $\forall n$ keeping in mind that a_{right} is defined by p_{right} - the only non-one coefficient in the equation given by $O(G)$ of a greedy sequence G.

4.1.1 The case $O(G) = 4$ for a greedy sequence G

First, we will, as usual, consider the next possible smallest value (this time in terms of $O(G)$ and not p_{right} necessarily) in an attempt to gain some additional insight before trying to make a general statement with regards to $O(G)$. In our case this is $O(G) = 4$. Therefore, we have:

$$a_i + a_j + a_k \neq -p_{right} \cdot a_n$$

Let us attempt now to use a similar inequality to the one we use to prove that $G_3 = \{0, 1, 2, ..., \infty\}$ when $p_{right} \leq -2$. Taking the terms right next to a_n and trying $a_n = n$ we get:

$$(n-3) + (n-2) + (n-1) < -p_{right} \cdot n \Leftrightarrow 3 \cdot n - 6 < -p_{right} \cdot n$$

Now, it is easy to see that for all values of $p_{right} \geq 3$ the inequality is equivalent to $-6 < k \cdot n$ where $n \in \mathbb{N}$ and $n \geq 3$ and $k \in \mathbb{Z}$ and $k \geq 0$. We now know that, for $\forall p_{right} \leq -3$, $G_4 = \{0, 1, 2, ..., \infty\}$ keeping in mind that the first three terms are by definition respectively equal to $0, 1, 2$.

We move to considering some more interesting values of p_{right} specifically $-1, -2$ (remember that we know what happens when $p_{right} > 0$ and that because it is a coefficient in the equation given by $O(G)$ it cannot equal zero).

For $p_{right} = -1$ we have that:

$$a_i + a_j + a_k \neq a_n$$

Consider the first several terms starting from index 0 of the sequence: $0, 1, 2, 4, 8, 15, 22, 29, 36, 43$.

There are several things to notice here. First notice what happens when we have already constructed the terms up to the "8" (index 4) inclusive. We can't

have the next term equal 9 because of $0+1+8$, neither 10 because of $0+2+8$ or 11 because of $1+2+8$. $12, 13, 14$ are out too because of $0+4+8$, $1+4+8$, and $2+4+8$ respectively so it have to be 15.

For the next term we can't have $16, 17, 18, 20, 21$ because we can combine 15 with $0+1$, $0+2$, $1+2$, $0+4$, $1+4$, and $2+4$ respectively. However, $22 = 15+7$ is working since neither can seven be constructed by two of the previous term or it is larger or equal to 8 - the term after 4. This is now sufficient information to say that every next term will equal to 7 plus the previous one which starts to resemble the sequence we had for $O(G) = 3$ when $p_{right} = -1$. We can also now construct a formula about this sequence too:

$$a_n = \left(2^n - n - \left\lfloor \frac{1}{n+1} \right\rfloor\right) \cdot \left\lfloor \frac{12}{n+8} \right\rfloor + (1 + 7 \cdot (n-3)) \cdot \left\lceil \frac{n-4}{n+8} \right\rceil \tag{11}$$

This is basically a slightly modified formula to the one we use for G_3. We use $\frac{n-4}{n+8}$ because it approaches 1 when rounded up (and therefore when subtracted from 1 and rounded down approaches 0) and has a shifting point at $n = 5$ instead of the one we used earlier. Also for the first few terms we express them as a power of 2 equaling their index minus their index with the exception of 0 for which we need too subtract additional 0 which we do by rounding down $\frac{1}{n+1}$ as this approaches 0 as $n \to \infty$ but equals 1 when $n = 0$,

We now move to our last case for G_4 where $p_{right} = -2$. For the equation given by $O(G) = 4$ we will have that:

$$a_i + a_j + a_k \neq 2 \cdot a_n$$

Again we consider the first several terms of the sequence beginning with the one with index equal to zero: $0, 1, 2, 3, 4, 5, 7, 9, 11, 13$. This is a much more interesting case. It seems that after the four the terms are constructed as the smallest next odd number. Now let us try and see why exactly this happen. For the term equal to three we have that $(0+1+2)/2 = 1.5$ which even rounded up is smaller than three. Then for the term equal to four we have that $(0+1+3)/2 = 2$ and that $(0+2+3)/2 = 2.5$, and that $(1+2+3)/2 = 3$ which again are all smaller or equal to the last element so we choose the next bigger integer which in this case is 4. Then $(4+3+2)/2 = 4.5$ which is smaller than 5 and equal to the last defined element when rounded down.

Now we have that $(3+4+5)/2 = 6$ so we are forced to skip it and pick 7 instead. We can now clearly see that $(3+4+5)/2 + 1 = (3+4+7)/2$ and that therefore we will have to skip every odd number. It is easy to prove that every odd number is a term given that we checked for 3 and 5 and that 1 was "chosen" in the very beginning.

Let a number $x = 2 \cdot y + 1$ where $y \in \mathbb{N}$ and $x \geq 7$ is such that $x \notin G_4$. This means that $x = (a_i + a_j + a_k)/2$. We have proven that no number z such

that $2|z$ such that $z \geq 6$ and $z \in G_4$. Therefore the only even numbers in G_4 are $0, 2, 4$. If we sum them up and divide them by two we get 3 but 3 is already in the sequence because of the $a_{right} = a_{next} = a_n$ criteria. The only other way to receive a sum such that it is even is if we only use one of these even terms.

If this is 0 then the sum divided by 2 is a number between the odd terms and, therefore, is of no interest because of the criteria above. If it is 2 then even if the two odd numbers are the biggest so far we will have $(2 + (x-4) + (x-2))/2 = (2x-4)/2 = x-2$ which is smaller than x. If it is 4 then the sum divided will equal $x-1$ which is both smaller than x and even which concludes the proof. We can now construct a very simple formula:

$$a_n = n \cdot \left\lfloor \frac{12}{n+8} \right\rfloor + (1 + 2 \cdot (n-3)) \cdot \left\lceil \frac{n-4}{n+8} \right\rceil \tag{12}$$

The formula, we believe, is clear enough to not waste time and space explaining the minor changes. This, however, is not the best results we could have hoped for because it does not make it possible to give an easy description for every value of p_{right}. In the next part of this section we will attempt to derive a method and a formula for the construction for the terms of this sequences G_O for a given value of $O(G)$ and of p_{right}.

4.1.2 Terms of the sequences G_O where $a_{right} = a_{next}$

Let us begin by describing the usual equation given by $O(G)$. We will take $O(G) = t$. Therefore, for G_t we have that:

$$p_{i_1} \cdot a_{i_1} + p_{i_2} \cdot a_{i_2} + ... + p_{i_{t-1}} \cdot a_{i_{t-1}} \neq -p_{right} \cdot a_n$$

We will look at several different values of p_{right} beginning with the simplest case when $p_{right} > 0$. Keep in mind that $p_{right} \neq 0$ since p_right is a coefficient in the equation given by $O(G)$ of G_t and that, by definition, $a_0, a_1, ..., a_{t-2}$ equal $0, 1, ..., t-2$ respectively. Then since a_n where $n \geq t-1$ is positive we have that $-p_{right} \cdot a_n < 0$ and that the left side of the equation is strictly positive. We now have that $G_O = G_t = \{0, 1, 2, ..., \infty\}$.

Proceed by considering the following inequality which earlier in this paper we proved when proven is sufficient to state that the sequence G_t is again equivalent to all non-negative integers:

$$(n-t+1) + (n-t) + ... + (n-1) < -p_{right} \cdot n$$

We now expand the left side so that we get:

$$(t-1) \cdot n - (t-1+t-2+...+1) < -p_{right} \cdot n$$

Even more simplified we get that:

$$(t-1) \cdot n - \frac{t \cdot (t-1)}{2} < -p_{right} \cdot n$$

Now we can rearrange the inequality so that we can get a clear relationship between p_{right} and t:

$$(t - 1 + p_{right}) \cdot n < \frac{t \cdot (t-1)}{2}$$

Keep in mind that $t = O(G_t)$ and, hence $t \geq 2$ making the right side of the inequality positive for $\forall t$.

There are several ways in which this inequality can prove useful. First notice that since the inequality needs to be true only for positive values of n if $(t - 1 + p_{right} \leq 0$ the inequality is true for $\forall n, t$. Therefore for $\forall p_{right}, t$ such that $p_{right} \leq 1 - t$ we have that $G_O = G_t = \{0, 1, 2, ..., \infty\}$.

Now let's consider what happens when $p_{right} > 1 - t$. First, we consider the biggest negative value of p_{right} which is -1. When $O(G) = 3$ we got: $0, 1, 2, 4, 7, 10, 13, 16, ...$, and when $O(G) = 4$ we got that: $0, 1, 2, 4, 8, 15, 22, 29, 36, 43$.

Let us make some observations about these sequences. At first glance we can divide the terms in three categories: $a_0 = 0$, $a_i = 2^{i-1}$ where $i = \{1, 2, ..., t\}$ and the rest $a_n = 1 + (n - 2) \cdot (2^{t-1} - 1)$ for $\forall n > t$. Before rushing into an attempt to prove this, however, we will for sure have to consider at least one more example where we have more terms "chosen" since even the smallest next will be 3 (when $t = 5$) which is not a power of two.

Therefore, for the equation given by $O(G) = t = 5$ for greedy sequence G_5 (such that $p_{right} = -1$) we have:

$$a_i + a_j + a_k + a_l \neq a_n$$

And, hence the first several terms are: $0, 1, 2, 3, 4, 5, 15, 16, 17, 30, 31, ...$ These seem to break any pattern we have found so far. One interesting, though simple, thing we can prove is that $a_n = n$ for $\forall n \leq t$ where $t \geq 5$. This can be done easily by observing when the first possible sum is bigger than the next term to be found a_n:

$$0 + 1 + 2 + ... + (t - 2) > a_n$$

Now we want to know when the inequality is true if $a_n = n$. Therefore:

$$(0 + 1 + 2 + ... + (t - 2)) - n > 0$$

The inequality above is equivalent to the following one:

$$\frac{(t-2) \cdot (t-1)}{2} - n > 0$$

We have hypothesized that this is true for $\forall n \leq t$ so we are going to substitute n with t:

$$\frac{(t-2) \cdot (t-1) - 2 \cdot t}{2} > 0 \Leftrightarrow (t-2) \cdot (t-1) - 2 \cdot t > 0 \Leftrightarrow t^2 - 5 \cdot t + 2 > 0$$

It is easy to see that the following inequality is true for every integer bigger than 4 which completes the proof. We can also try and see if we can know exactly how many consecutive numbers starting from the beginning of the sequences will equal their indexes. To do that consider that all possible summations must be strictly larger than the index of the term we are looking for. This however is equivalent to the smallest sum being strictly larger to the index. This sum, by definition, includes $t-1$ terms in it and we know that the largest of them will equal $t-2$ given that we are finding a_n.

With all these in mind we return to the inequality above before substituting n with t and rearrange:

$$n < \frac{(t-2)\cdot(t-1)}{2}$$

This leaves us with:

$$a_n = n \text{ for } \forall n \in \left\{0, ..., \frac{(t-2)\cdot(t-1)}{2} - 1\right\}$$

There are two exceptions for this equation when t equals $2, 3$. When we have that $O(G) = 2$ the inequality gives us that in order for the index to equal the value we should have that $n < 0$ which is impossible since n is a non-negative integer. However, this is because the inequality was defined for constructed values of n only and the zero element $a_0 = 0$ was excluded. When $t = 3$ we have the same thing with $n \in 0, ..., 0$ again not accounting only for a "chosen" and not constructed element $a_1 = 1$.

We will now continue the section with some analysis on "Stanley-like" sequences for which there is no additional constraint to the term we are finding next.

4.2 "Stanley-like" sequences where $p_{right} = 1 - O(G)$

Let us start with defining the sequences we are going to observe in this section. Let us start with a greedy sequence G_t. Look at the order of dependency given by $O(G) = t$:

$$p_{i_1} \cdot a_{i_1} + p_{i_2} \cdot a_{i_2} + ... + p_{i_{t-1}} \cdot a_{i_{t-1}} + p_{i_t} \cdot a_{i_t} \neq 0$$

We rewrite the equation so only one term is on the right side and we substitute p_{i_t} with p_{right} and a_{i_t} with a_{right}:

$$p_{i_1} \cdot a_{i_1} + p_{i_2} \cdot a_{i_2} + ... + p_{i_{t-1}} \cdot a_{i_{t-1}} \neq -p_{right} \cdot a_{right}$$

The sequences we will be looking into are such sequences for which in the above equation $p_{i_1}, p_{i_2} ... p_{i_{t-1}} = 1$ and $p_{right} = -(O(G) - 1) = 1 - t$. The last is equivalent to $-p_{right} = t - 1$ making the above equation:

$$a_{i_1} + a_{i_2} + ... + a_{i_{t-1}} \neq (t-1) \cdot a_{right}$$

The motivation behind this choice (about the value of p_{right}) is based on the fact that was briefly mentioned in section 3 about the fact that the absolute value of p_{right} may equal what we called the base of the "writing" numerical system and this indirectly implies it may equal the base of the system in which we "read" the number when 1 is subtracted. This is at least the case when $O(G) = 3$ as we have proved in section 2.

Last, we will always "choose" the smallest possible option for the terms we need to define before we can apply the equation given by $O(G)$.

Keep in mind that in the previous part of this chapter we have proven that if a_{right} must always equal the next term we are looking for we will have a sequence G_t equivalent to $\{0, 1, 2, ..., \infty\}$. Because of this we now remove this criteria and examine the more general case.

4.2.1 Sequence G where $O(G) = 4$

Consider the sequence we described right above if $O(G) = 4$. It will be as follows:

$$a_i + a_j + a_k \neq 3 \cdot a_{right}$$

It is always useful to have a handy tool for calculating the terms of such sequences so we will use a very slightly modified version of the first program we considered A.1. The only notable difference will be an additional loop for assigning values for the "chosen" members of the sequence and a replacement of the "2" in the checking section of the inner most loop with a variable equal to $-p_{right} = O(G) - 1 = t - 1$. The source code for this program is provided in the appendix bellow as well.

4.2.2 Observation about the terms of $G_4(1, 2)$

The first 20 terms of this sequence are provided in a table below [Table 3]. Keep in mind that the first ones are by definition $0, 1, 2$.

Index:	0	1	2	3	4	5	6	7	8	9
Value:	0	1	2	3	4	12	13	14	15	16
Index:	10	11	12	13	14	15	16	17	18	19
Value:	48	49	50	51	52	60	61	62	63	64

Table 3: The first 20 terms of $G_4(1, 2)$

Let us now consider using the numerical system approach that helped us solving G_3. Sticking with the motivation we explained earlier we will try to write the values of the terms in $-p_{right} + 1 = 4$. The results are found in the following table:

Index:	0	1	2	3	4	5	6	7	8	9
Value:	0	1	2	3	10	30	31	32	33	100
Index:	10	11	12	13	14	15	16	17	18	19
Value:	300	301	302	303	310	330	331	332	333	1000

Table 4: The first 20 terms of $G_4(1,2)$ written in base 4

There is an interesting observation to be made hare again first noted by Stanley. According to him, the terms of $G_4(1,2)$ can be shown to be the numbers which in base 4 satisfy two conditions. First, only the last digit in the expansion can be a 2. Second, if a digit anywhere in the expansion equals 1 every digit that follows should equal 0 [3].

We now attempt to prove his conjecture about the terms of G_4. First, we shall prove that these terms satisfy the equation given by $O(G) = 4$ and then that they are the smallest such terms.

4.2.3 Proving the terms Stanley observed does not violate the equation given by $O(G)$

For the first part of the proof consider that, by definition, $a_i \neq a_j \neq a_k$ since the sequence is "greedy" and, hence strictly increasing. Now consider that a_{right} contains only zeroes after a 1 if it contains a 1 in its expansion in base 4. Now let us consider how the different digits change when multiplied by three in base 4:

$$0 \to 0$$

$$1 \to 3$$

$$2 \to 2$$

$$3 \to 1$$

Note that the last two result in a carry (of 1 and 2 respectively). Now let us consider how a term may look like in base 4. We know that only the last digit can be a 2 and if it is no 1 is present in the term because of the second criteria. The same goes if the number ends with a 3 and in that case it will consist only of digits 0 and 3. Such combination is also a part of every other such in which a 1 is present as the part before the 1. Also a maximum of one 1 or 2 is present in a term and never both. Let us now consider what happens when we multiply these types of base 4 numbers by three.

First, consider the terms that does not contain a 2. If such term contains a digit 1 when multiplied by three it will end with a 3 and a tail of several digits 0 equal to the original such present. Now we should look at how can this be achieved by summing up three other terms. One such is to combine three numbers that also have a 1 at this certain position making the two numbers the

same from that point until their ends since only digits equal to 0 can follow a 1. Now, however, consider that the terms must have only digits 0 and digits 3 before the 1. Therefore, the only way to achieve a multiplication of a 3 by three by summing three terms will also require them to have a 3 and not a 0 at the specific position implying that the three terms were from the start the same and, hence do not exist.

Another ways we can get a 3 are adding $(0+0+3)$, $(0+1+2)$, $(1+3+3)$ and $(2+2+3)$. Beginning with $(0+0+3)$ both the multiplication and the addition result in a 3 without carry so a contradiction may come from another point mainly two of these terms being the same. If we use the options with three different digits we will get no carry again and will achieve a contradiction by proving such an element none existent when adding other digits. First if the digit before the 1 is a 0 we will have to use either $(0+0+0)$ or $0+1+3$ since $0+2+2$ does not work for obvious reasons.

This is the way in which the analysis is done notably looking at a very tedious case by case analysis resulting in either a contradiction due to a lack of difference between elements or lack of options in which to add. The whole analysis is not included because we have shown examples of how cases are discarded and because of their large number it becomes an inconvenience to go over each one of them. We will now focus on showing how to prove that there are no any smaller terms satisfying the equation given by $O(G)$.

4.2.4 Proving the terms Stanley observed are the terms of G_4

Suppose a lexicographically smaller sequence H_4 exists where $\{b_n\}$ is the general term. Then let m be the smallest number for which $b_m \neq a_m$ which implies that $b_m < a_m$. Now because all the numbers smaller than a_m satisfying the two criteria set by Stanley are exactly $a_0, a_1, ..., a_{m-1}$ we can conclude that b_m does not satisfy one of them. If b_m has a digit 2 not on the very end let us have such three terms of H_4, α, β, γ such that α has a digit 1 where b_m has a digit 1 and β has a digit 1 where b_m has a digit 2 and γ has a digit 3 where b_m has a digit 3 and has a digit 1 where b_m has a digit 2 unless $b_m = 20$ in which case the example is simply $10+3+1$ or similar where the first digit is a 2 and the others are 0 we simply add the needed digits 0 to $10, 3, 1$.

The other possible criteria to be broken is if b_m has a digit different then 0 after a digit 1. In this case we will construct α, β, γ in the same way by swapping the digit that will cause the example to fail with either 3 or 0 and swapping the following digit with either an appropriate carry that is obtained if in need of 1 by $(3+3+3)$ and if in need of two by $3+3+0$ keeping the 0 on α such that we move the 1 of α to the leftest 1 in b_m. and swapping a 0 in β, γ to the two most right digits 1 in b_m An exception to this method is $b_m = 21$ when the combination we use is $12+2+0$ and any of the form where b_m begins with 21 and only digits 0 follow in which case we add the needed digits equal to 0 to

$12, 2, 0$.

We now take some time for some general remarks about how numerical systems can be used to derive terms or at least gain insight about them of the "Stanley-like" greedy sequences.

4.3 Numerical systems used for derivation of the terms of G_O

There are more things that can be said about G_4 and G_O in general and is very possible that there are ways to incorporate numerical systems in deriving properties of their terms. Based on the data we received from the programs we used it may be sufficient to conjecture that apart from G_3 there exist no other "Stanley-like" sequence for which a "writing" and "reading" bases of numerical systems exist.

It also seems by the examples we considered that the values of $O(G)$ and hence $-p_{right}$ are clear indicators of the bases in which to try and represent the values if not the indexes of the terms of such sequences. Finally, it is interesting if there exist any other values of t for which there exist a certain digit not present in the said expansion (of a certain base) for such greedy sequence.

5 Observations About the Length of Greedy Sequences

In this chapter we will focus on some common properties among the various sequences we've discussed so far. We will also define and investigate some intriguing theorems that may help us with the further analysis goals we set ourselves in previous sections. First, let us look at the general Stanley sequence.

5.1 Length of the Stanley sequence

First let us return to the definition of a Stanley sequence: an increasing greedy sequence $\{a\}$ such that no three elements a_a, a_b, a_c exists so that $a_a + a_b = 2 \cdot a_c$.

We can set an arbitrary restriction on the sequence by artificially bounding it above. Let us define M as an upper bound of $\{a\}$. Then we can define N as a "maximum" Stanley sequence - a bounded above greedy sequence such that it has a maximum number of elements equal to $n(M)$. It is interesting to see if it is possible to come up with some estimate for the value of $n(M) = n$ for a given M.

We will begin with the following theorem which is true for $\forall M \geq 8$. Please note that theorem is also true for the values $4 - 6$ but wrong when $M = 7$.

5.1.1 Theorem about $2 \cdot M$

We formulate:

Theorem 3: $n(2 \cdot M) \leq M$ *for* $\forall M \geq 8$ *such that* $M \in \mathbb{N}$

Proof: It is quite easy to prove the observation by induction. First, we derive that because $a_1 < a_2 < a_3 < ... < a_{n-1} < a_n$ is a N sequence, then:

$$N + 1 - a_n < N + 1 - a_{n-1} < ... < N + 1 - a_2 < N + 1 - a_1 \tag{13}$$

will also be a N sequence.

Accordingly for any integer $w < a_1$ we will have that:

$$a_1 - k < a_2 - k < a_3 - k < ... < a_{n-1} - k < a_{n-2} - k \tag{14}$$

will also be a N sequence.

Therefore, it is easy to conclude that:

$$n(X + Y) \leq n(X) + n(Y) \tag{15}$$

Now let us move our attention to the inductive part of the proof. The base we will be considering is $M = 4$. Striving for contradiction, let us assume that $n(8) \geq 5$. From (13) and (14) it follows that at least the number 1 and two larger integers less or equal to 4 are contained in $\{a\}$. This leaves room for just two possible sets - $1, 2, 4$ or $1, 3, 4$. Evidently, none of these combinations leads to a case such that $n(8) \geq 5$ and it is easy to see that $n(8) = 4$ because of the sequence $1, 2, 4, 5$.

We have now established the base for our proof. First, we observe some examples. Consider $n(10)$ for a moment. Let us assume that $n(10) \geq 6$. Then, from (14) and $n(8)$ it follows that $1, 2, 9, 10$ are contained in $\{a\}$ and because the sequence is greedy the only possible combination that remains is: $1, 2, 4, 7, 9, 10$ (as $3, 5, 6, 8$ cannot occur because of the original 4 members) but this is also impossible because of $1, 4, 7$ which is an arithmetic sequence. Therefore, because of $1, 2, 4, 9, 10$ we can conclude that $n(10) = 5$. The next smallest possible example will be $n(12)$. If $n(12) \geq 7$ we know by the same logic that $1, 2, 11, 12$ are present in the collection. from (14) and $n(8) = 4$ we know that $4, 9$ occur too. It is easy to see that this does not leave place for other members. It is possible to conclude that $r(12) = 6$ because of the previous sequence. Now, hence $1, 2, 4, 5, 10, 11, 13, 14$ are in the N sequence we know that $r(14) = 7$. Therefore, it follows from (15) that $n(16) \leq 8$, $n(18) \leq 9$, $n(20) \leq 10$, $n(20) \leq 10$, $n(22) \leq 11$.

From here it is relatively easy to build up the induction. We first assume that the theorem is true for $2 \cdot M - 8$. Hence, from (15) it follows that:

$$n(2 \cdot M) \leq n(2 \cdot M - 8) + n(8) < M - 3 + 4 = M + 1$$

, which is enough to conclude the theorem because the special cases for this theorem $16, 18, 20, 22$ have already been established.

6 Conclusions

In this section we are going to summarize what has been shown, proven, constructed, and conjectured in this paper.

For the Stanley sequence $G(1)$ it was proven that the terms are the numbers which ternary expansion does not include a digit equal to 2 as first noted by R. P. Stanley. In other words one way to derive them is writing their indexes in binary and reading them in ternary. Based on that a formula for calculating a_n for a given $n \geq 0$ has been constructed.

We then introduced useful notation for characterizing greedy sequences by inventing a property of the greedy sequence, called order of dependency denoted by $O(G)$ and have shown it has a range of bigger or equal to 2.

A complete analysis of $O(G) = 2$ followed alongside a formula for constructing it's terms when the two coefficients present in the equation are of different signs since the sequence when they are of the same is simply all non-negative integers.

We then considered what has been done by Stanley for $O(G) = 3$ and introduced different classes of greedy sequences: Stanley sequences, "Stanley-like" sequences, and "Stanley-like" sequences where $a_{right} = a_{next}$. For all of them we ran a computer simulation to test if two numerical systems exist that can be used to derive the terms of G_3 as in the original sequence we considered. Based on the results and ranges we used, we conjectured at the end of the paper that none other such pair exists. For "Stanley-like" sequences we shown the terms of the sequences for positive values of p_{right} independent of $O(G)$ and for "Stanley-like" sequences where $a_{right} = a_{next}$ we have shown and proven a method for deriving its terms when $O(G) = 3$ and constructed a formula for that depending on the value of p_{right}.

For value of $O(G) > 3$ we have completely analyzed the "Stanley-like" sequences where $a_{right} = a_{next}$ when $O(G) = 4$ and have constructed a formula for calculating its terms. For values of the order bigger than 4 we have shown how many consecutive terms starting from a_0 have values equal to their indexes and have proposed a skeleton for a proof of Stanley's observation about how numerical system of base 4 can be use to describe the terms of the "Stanley-like" sequence $G_4(1, 2)$. We have also proposed and refuted a simple hypothesis about the first several terms of the "Stanley-like" sequences where $a_{right} = a_{next}$ dealing with their seeming formation's relationship with powers of 2.

7 Future Work

There are many more insights to be acquired when it comes to the various greedy sequences we have now analyzed. We divide our propositions based on the classification of the said greedy sequence:

For Stanley sequences regular values of k found by professor Stanley can be analyzed to see if a relationship between $-p_{right}$ or k and a base in which members of the sequence have clear properties apart from ternary can be found.

For "Stanley-like" sequences a much wider analysis using a console application must be performed after a suitable program is developed to see if our hypothesis about the connection between $-p_{right}$ and the base of the "reading" numerical system actually holds any truth. After this is completed a general formula may be constructed given an index n, a value of $O(G) = t$, and a value of p_{right} to derive the terms of the said sequence $G_O(0, 1, ..., t-2)$. It is also important to check if there exist a way to remove the constraint about the non p_{right} indexes and still have a clear connection to a numerical system which can perhaps be done by using the floor function and setting up a common denominator.

For "Stanley-like" sequences with the additional constraint imposed on a_{next} a similar computer simulation with bigger ranges must be performed. After that is done we can consider using math with combinations to try and see how exactly new terms are defined and probably construct a formula using a combination of t elements where t equals the $O(G)$ value of the sequence to a certain power of which to choose n where n equals the index of the term to be found.

In co conclusion, some general remarks that can be helpful with further research. It is interesting to try and prove our conjecture at the end of the paper about the "writing" and "reading" bases which may also need some clarification or even reformatting as it is now to abstract in giving any important information. It is also interesting to consider that the "clarity" with which one is able to derive term of a given sequence G_O decreases really fast with the increase of $O(G)$ given the examples $O(G) = 3$ and $O(G) = 4$. That being said, we will think of additional constraints that can help increase the "clarity" and when impose may even present a possibility for the disproving of the conjecture.

Finally, I want to thank my mentor, Katerina Velcheva, for the support she has been providing me, the time she has devoted for my many questions, and all the other ways in which she has been most helpful and irreplaceable. I want to say that I think I have learned much about a topic I am genuinely interested in and the field of research in general despite this being the first such project of my own, and am really eager to continue developing it in the following months. Thank you too for devoting your time and reading this paper to the end!

A C# Programs Source Code

A.1 Brute force algorithm for calculating the terms of $G(k)$

```
using System;

namespace GreedySequences
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Input a natural number n:");
            int n = int.Parse(Console.ReadLine());
            Console.WriteLine("Input a natural number k:");
            int k = int.Parse(Console.ReadLine());
            Console.WriteLine("Calculating the first n terms of the greedy" +
            "sequence beginning with 0,"+ k.ToString() +" :");
            int[] a = new int[1000];
            a[0] = 0;
            a[1] = k;
            Console.Write(a[0].ToString() + " ");
            Console.Write(a[1].ToString() + " ");
             int br = k + 1;
            for(int i = 2; i < n; i++)
            {
                for (int j = 0; j < i; j++)
                    {
                        for (k = j + 1; k < i; k++)
                        {
                            if ((2 * a[k]) == (a[j] + br))
                            {
                                br++;
                                j = 0;
                                k = j + 1;
                            }
                        }
                    }
                a[i] = br;
                br++;
                Console.Write(a[i].ToString() + " ");
            }
            Console.WriteLine();
        }
```

```
    }
}
```

A.2 Source code for a program finding numerical systems for Stanley sequences $G(k)$

```
using System;
using System.Numerics;
using System.Threading;

namespace GreedySequences
{
    class Program
    {
        static void Main(string[] args)
        {
            int k = 1;
            for (k = 1; k < 1000; k++)
            {
                int[] a = new int[1000];
                int[] b = new int[1000];
                a[0] = 0;
                a[1] = k;
                b[0] = 0;
                b[1] = k;
                int br = k + 1;
                int equals = 2;
                for (int i = 2; i < 100; i++)
                {
                    for (int j = 0; j < i; j++)
                    {
                        int c = k;
                        for (c = j + 1; c < i; c++)
                        {
                            if ((2 * a[c]) == (a[j] + br))
                            {
                                br++;
                                j = 0;
                                c = j + 1;
                            }
                        }
                    }
                    a[i] = br;
                    br++;
                }
```

```
        int i2 = 3;
        int j2 = 2;
        for ( i2 = 3; i2 < 30; i2++)
        {
        for( j2 = 2; j2 < i2; j2++)
        {
        for(int k2 = 2; k2 < 100; k2++)
        {
            int remainder;
            string result = string.Empty;
            int o = k2;
            while(o > 0)
            {
                remainder = o % j2;
                o /= j2;
                result = remainder.ToString() + result;
            }
            int resultIndex = int.Parse(result);
            int sum = 0;
            int w2;
            int y = (int)(Math.Floor(Math.Log(resultIndex, 10)));
            for (int w = 0; w <= y; w++)
            {
                w2 = resultIndex % 10;
                resultIndex /= 10;
                sum += (int)(w2 * Math.Pow(i2, w));
            }
            b[k2] = sum;
            if (b[k2] == a[k2]) equals++;
        }
        if (equals >= 90)
        {
        Console.WriteLine("The equation to be avoided has a" +
        " second term: " + k);
        Console.WriteLine("The index numerical system is of base: " +
        i2);
        Console.WriteLine("The term numerical system is of base: " +
        j2);
        Console.WriteLine();
        }
        equals = 2;
        }
    }
    a[0] = 0;
    a[1] = k;
    b[0] = 0;
```

```
            b[1] = k;
            br = k + 1;
        }
        }
    }
}
```

A.3 Source code for a program finding numerical systems for "Stanley-like" sequences

```
using System;
using System.Numerics;
using System.Threading;

namespace GreedySequences
{
    class Program
    {
        static void Main(string[] args)
        {
            int k = 1;
            int[] a = new int[1000];
            int[] b = new int[1000];
            a[0] = 0;
            a[1] = k;
            b[0] = 0;
            b[1] = k;
            int br = k + 1;
            int equals = 2;
            for (int x = 2; x < 1000; x++)
            {
                for (int i = 2; i < 100; i++)
                {
                    for (int j = 0; j < i; j++)
                    {
                        for (k = j + 1; k < i; k++)
                        {
                            if ((x * a[k]) == (a[j] + br))
                            {
                                br++;
                                j = 0;
                                k = j + 1;
                            }
                        }
                    }
```

```
                a[i] = br;
                br++;
            }
            int i2 = 3;
            int j2 = 2;
            for ( i2 = 3; i2 < 30; i2++)
            {
            for( j2 = 2; j2 < i2; j2++)
            {
            for(int k2 = 2; k2 < 100; k2++)
            {
                int remainder;
                string result = string.Empty;
                int o = k2;
                while(o > 0)
                {
                    remainder = o % j2;
                    o /= j2;
                    result = remainder.ToString() + result;
                }
                int resultIndex = int.Parse(result);
                int sum = 0;
                int w2;
                int y = (int)(Math.Floor(Math.Log(resultIndex, 10)));
                for (int w = 0; w <= y; w++)
                {
                    w2 = resultIndex % 10;
                    resultIndex /= 10;
                    sum += (int)(w2 * Math.Pow(i2, w));
                }
                b[k2] = sum;
                if (b[k2] == a[k2]) equals++;
            }
            if (equals >= 90)
            {
            Console.WriteLine("The equation to be avoided has a" +
            " coefficient value of: " + x);
            Console.WriteLine("The index numerical system is of base: "
            + i2);
            Console.WriteLine("The term numerical system is of base: "
            + j2);
            Console.WriteLine();
            }
            equals = 2;
        }
    }
```

```
            k = 1;
            a[0] = 0;
            a[1] = k;
            b[0] = 0;
            b[1] = k;
            br = k + 1;
            }
            }
        }
    }
```

References

[1] 18.821 project laboratory in mathematics: Project descriptions. 2016.

[2] N. J. A. Sloane and Simon Plouffe. The encyclopedia of integer sequences, academic press. 1995.

[3] R. P. Stanley and A. M. Odlyzko. Some curious sequences constructed with the greedy algorithm. 1978.