Incongruent Restricted Disjoint Covering Systems

Laura Lyman, Tim Morris, and Bridget Toomey

September 17, 2012

1 Abstract

It is impossible for a covering of the integers to be both incongruent and disjoint [9]; however, systems of congruences that cover only finite intervals can satisfy both of these conditions. These systems are called incongruent restricted disjoint covering systems (IRDCS). They have the additional requirement that each congruence must cover at least two integers within the specified interval. Here, we present the results of our research on IRDCS.

2 Background

2.1 History

In 1956, Paul Erdős used covering sets of the integers to construct arithmetic progressions which created an infinite set of counterexamples to *de'Polignac's conjecture*. Since then, the study of these covering systems has developed into its own subject of mathematics.

Many open problems, such as the minimum modulus problem (Erdős) and the odd modulus problem, have been presented in this area of mathematics. A new type of covering system was developed by Myerson, Poon, and Simpson in 2007. These new systems are called IRDCS, and they will be the main focus of this paper.

2.2 Incongruent Restricted Disjoint Covering Systems

Consider a residue class $S(m_i, a_i) = \{a_i + m_i z : z \in \mathbb{Z}\}$. A covering system is a collection of residue classes $\{S(m_i, a_i)\}$ such that every integer is satisfied by at least one congruence. We note that there are infinite covering systems (see [1], [2], [5], [10]). However, for the purposes of this paper, we will only consider finite collections of congruences. Observe that systems of congruences with distinct moduli exist. Daveport constructed the following example of such a system in the 1930's:

$$\{S(2,0), S(3,0), S(4,1), S(6,1), S(12,11)\}.$$

It is left to the reader to verify that this is a covering system.

We now define an incongruent restricted disjoint covering system (IRDCS).

Definition 1. A system of linear congruences $\{S(m_i, a_i) \mid 1, \ldots i, \ldots t\}$ is *incongruent* if the moduli are distinct.

Definition 2. A system of linear congruences $\{S(m_i, a_i) \mid 1, \ldots i, \ldots t\}$ is *disjoint* if $S(m_i, a_i) \cap S(m_j, a_j) = \emptyset$ for $i \neq j$.

Note that the integer t is refered to as the *order* of an IRDCS $\mathcal{A}$. In other words, the order of a given IRDCS refers to the number of congruence classes which cover it.

Definition 3. A covering system is *restricted* if for every i in a finite interval $[1, n]$ of integers $1, 2, \ldots, n$, i is satisfied by at least one congruence.

Definition 4. A covering system $\mathcal{A}$ is an *incongruent restricted disjoint covering system (IRDCS)* on $[1, n]$ if it satisfies each of the following:

1. $\mathcal{A}$ is incongruent
2. $\mathcal{A}$ is disjoint
3. $\mathcal{A}$ is restricted
4. Every congruence class contains at least two integers in $[1, n]$

This last condition is enforced to avoid trivial coverings on $[1, n]$ such as $\{S(n, 1), S(n+1, 2), \ldots, S(2n, n)\}$. This condition also implies that on a given interval $[1, n]$, there is a maximum possible modulus, namely $n - 1$. Thus, there is a finite number of IRDCS on any given finite interval.

The Mirsky-Newman theorem states that it is impossible for a covering of the integers to be both incongruent and disjoint [9]. Thus, it is reasonable to question if IRDCS exist. However, the restricted aspect of the IRDCS makes the construction possible. Here is an example of an IRDCS on $[1, 11]$:

Example 1. $\{S(6, 1), S(9, 2), S(3, 0), S(4, 0), S(5, 0)\}$

Another representation of an IRDCS is in sequential notation as shown in example 2.

Example 2. $\{6, 9, 3, 4, 5, 3, 6, 4, 3, 5, 9, 3\}$

Hence, an IRDCS can be viewed as a sequence of integers $s_1, \ldots, s_n$ such that $s_i = m$ for some m if and only if $s_{i+km} = m$ for all k such that $i + km \in [1, n]$. This *sequential notation* will be used often throughout the paper, as it allows for interesting visual properties.

2.3 Reversals

For any IRDCS, there exists a reversal contruction defined as follows:

Definition 5. Given an IRDCS $\mathcal{A} = \{S(m_i, a_i) \mid 1, \ldots, i, \ldots, t\}$ on $[1, n]$, its *reversal* $\mathcal{A}' = \{S(m_i, n - a_i + 1) \mid 1, \ldots, i, \ldots, t\}$.

Thus we have an immediate theorem on the structure of IRDCS.

Theorem 1. *No IRDCS equals its reversal.*

A detailed proof is presented in [7]. This result implies that for any interval, there exists an even number of IRDCS on that interval.

Here, we note a result originally mentioned in [6]. Given an IRDCS $\mathcal{A}$ and its reversal $\mathcal{A}'$, the sequential notation of $\mathcal{A}'$ will be the reverse of the sequential notation of $\mathcal{A}$. Now, consider $\mathcal{A}$, a length 11 IRDCS we have already seen:

$$\begin{array}{rcl} n & \equiv & 1 \pmod{6} \\ n & \equiv & 2 \pmod{9} \\ n & \equiv & 0 \pmod{3} \\ n & \equiv & 0 \pmod{4} \\ n & \equiv & 0 \pmod{5} \end{array} \quad \Leftrightarrow \quad \{6, 9, 3, 4, 5, 3, 6, 4, 3, 5, 9\}$$

Then by the definition of the reversal construction, $\mathcal{A}'$ will be as follows:

$$
\begin{array}{rcl}
n & \equiv & 5 \pmod 6 \\
n & \equiv & 1 \pmod 9 \\
n & \equiv & 0 \pmod 3 \\
n & \equiv & 0 \pmod 4 \\
n & \equiv & 2 \pmod 5
\end{array}
\quad \Leftrightarrow \quad \{9,5,3,4,6,3,5,4,3,9,6\}
$$

2.4 Doubling and Families

Myerson, Poon, and Simpson have shown that there exist IRDCS for lengths 11 and $17-31$ [7]. Futhermore, there exist IRDCS for all lengths greater than 17. This result is shown through the doubling construction [7]. Considering an IRDCS $\mathcal{A} = \{S(m_i, a_i) \mid 1, \ldots, i, \ldots, t\}$ on $[1, n]$, we contruct its double in the following manner. The double of $\mathcal{A}$ is denoted by $D(\mathcal{A}) = \{S(2m_i, 2a_i) \mid 1, \ldots, i, \ldots, t\} \cup S(2,1)$. It is easily verified that this is an IRDCS on $[1, 2n+1]$. Omitting the last element, which is covered by $S(2,1)$, we obtain an IRDCS of length $2n$. Also, we can remove the first and last elements of the interval, which are both covered by $S(2,1)$. We can then shift the remaining residues down by 1 and generate an IRDCS on $[1, 2n-1]$. This allows us to double the IRDCS found in [7] and create IRDCS of all lengths greater than or equal to 17.

Doubles establish a family of IRDCS. A family is generated when we enforce another restriction on the set of all IRDCS. For instance, we can obtain IRDCS of all even moduli, odd moduli, doubles, etc. For the majority of this paper, we focus our research on a special family of IRDCS, the 9-6-3 construction.

Definition 6. A *9-6-3 construction* is an IRDCS of length 18 or greater with the sequence of moduli *9, 6, 3* somewhere in its sequential notation.

It follows directly from the definition of a 9-6-3 construction that we will have a length 18 pattern of this form in such an IRDCS:

$$\ldots, 9, 6, 3, \_, \_, 3, \_, 6, 3, 9, \_, 3, \_, 6, 3, \_, \_, 3 \ldots$$

Though we present this pattern beginning with the sequence *9, 6, 3*, the starting location of this sequence is irrelevant. As long as the length of our IRDCS is at least 18, the pattern will repeat in its entirety at least once, though possibly with a different starting position.

3 Preliminary Result on IRDCS

We begin by presenting a general theorem which places some important restrictions on valid choices for IRDCS moduli.

Theorem 2. *Let p be a prime, $a \in \mathbb{Z}$ such that $p \nmid a$. Then in an IRDCS with a and p as moduli, the congruence class with a can cover at most $p-1$ integers.*

Proof. Suppose we have $n_1 \equiv k \pmod{a}$ and $n_2 \equiv l \pmod{p}$ as congruence classes. To derive a contradiction, suppose that $n_1 \equiv k \pmod{a}$ covers at least p integers without clashing. Then this congruence class covers at least these integers: $\{k, k+a, ..., k+(p-1)a\}$. By a lemma used to prove Fermat's Little Theorem, the set $\{0, a, ..., (p-1)a\} \pmod{p}$ is the same as the set $\{0, 1, ..., p-1\}$ (we're adding the the same element, 0, to each set in the original lemma, so the lemma is still valid). But $\pmod{p}$, the set $\{k, k+a, ..., k+(p-1)a\}$ is simply a translation (and therefore a re-ordering) of $\{0, a, ..., (p-1)a\}$. Thus, the set $\{k, k+a, ..., k+(p-1)a\} \pmod{p}$ is the same as $\{0, 1, ..., p-1\} = \mathbb{Z}_p$. But $n_2 \equiv l \pmod{p}$ covers all of the integers in the IRDCS equivalent to some element in $\mathbb{Z}_p$. Thus, since at least one element of $\{k, k+a, ..., k+(p-1)a\}$ will also cover an element of this form, we'll have a clash. □

4 Our Findings Regarding the 9-6-3 Construction

Now, we focus on a specific family of IRDCS we have already defined. As mentioned in our explanation of the 9-6-3 construction, the special pattern in the sequential notation of a 9-6-3 IRDCS is 18 digits long. Therefore, the minimum length of an IRDCS containing the pattern in its entirety is 18. However, we sharpen this lower bound on the length of possible 9-6-3 constructions even further.

Theorem 3. *There exists no IRDCS of length 18 that adheres to the 9-6-3 construction.*

Proof. To derive a contradiction, suppose that such an IRDCS is possible. Then the modulus 9 will cover 2 integers in the interval, the modulus 6 will cover 3 integers, and the modulus 3 will cover 6 integers. Thus, a total of $2+3+6 = 11$

integers will be covered, so we'll have 7 integers left. But since each congruence must cover at least 2 integers, we must have at least one congruence that covers at least 3 integers. However, since we've already used 3, 6, and 9 as moduli, our only choices of moduli remaining are 2, 4, 5, 7, and 8 (integers 10 and greater will cover at most 2 integers in the interval). But all of these moduli are relatively prime to 3, so if any of them cover 3 or more integers, we'll obtain a clash by Theorem 2. □

Following are some additional results on the possible structures of IRDCS with the 9-6-3 construction.

Lemma 1. *Given an IRDCS of length at least 18 with 3, 6, and 9 as moduli, there are 36 possible combinations of non-clashing residue choices for these moduli.*

Proof. Fix a residue choice for 3, and observe that this residue will cover 2 integers mod 6 and 3 integers mod 9. Thus, our 4 open slots mod 6 leave us with 4 residue choices. Choosing a residue mod 6 will cover 3 more integers mod 9, so we now have 9 - 6 = 3 uncovered integers. So 3 residue choices for modulus 9 and 4 residue choices for modulus 6 leave us with 12 total possibilities given a fixed modulus covering 3. Since there are 3 choices of residues mod 3, we have 36 total possibilities. □

At first glance, this lemma might not appear to apply to 9-6-3 constructions specifically. However, it is integral to one of two proofs we found for our next theorem, as detailed below.

Theorem 4. *Any IRDCS of length 18 or greater with 3, 6, and 9 as moduli must be either a 9-6-3 IRDCS or its reversal.*

Proof. Recall that the 9-6-3 pattern repeats after 18 spaces in the sequential notation of an IRDCS. Within one complete repetition of length 18, there are 18 possibilities for the placement of the initial 9 in the sequence. That is, viewing one subsequence of length 18, there are 18 places the 9-6-3 pattern could begin. For each of those possibilities, only one choice of residue for the moduli 6 and 3 will result in our having the sequence 9, 6, 3 appear in our sequential notation. Note that this is the definition of a 9-6-3 construction. Thus, there are 18 unique possible choices that will lead to a 9-6-3 construction. But each of these 9-6-3 constructions has a reversal which is distinct from the original IRDCS by the distinctness of reversals [7]. Thus, we have a total of 36 possible IRDCS with 3, 6, and 9 as moduli choices. So by Lemma 1, these are the only possible IRDCS with these moduli.

□

The following is an alternate proof to Theorem 4. We discover this proof with the assistance of our graduate student mentor, Elizabeth Wesson. This proof centers on properties of the sequential notation of a 9-6-3 construction.

Proof. Consider an IRDCS in sequential notation of length $n > 18$. When using moduli 3, 6, 9, without loss of generality, observe a possible portion of the sequence beginning with modulus 3. Note the following choices for the placements of the moduli 6 and 9:

$$\begin{array}{cccccccccccccccccccc}
\ldots & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & \ldots \\
\ldots & 3 & - & 6 & 3 & - & - & 3 & - & 6 & 3 & - & - & 3 & - & 6 & 3 & - & - & \ldots \\
\ldots & 3 & 9 & 6 & 3 & - & - & 3 & - & 6 & 3 & 9 & - & 3 & - & 6 & 3 & - & - & \ldots
\end{array}$$

or

$$\begin{array}{cccccccccccccccccccc}
\ldots & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & \ldots \\
\ldots & 3 & - & 6 & 3 & - & - & 3 & - & 6 & 3 & - & - & 3 & - & 6 & 3 & - & - & \ldots \\
\ldots & 3 & - & 6 & 3 & 9 & - & 3 & - & 6 & 3 & - & - & 3 & 9 & 6 & 3 & - & - & \ldots
\end{array}$$

or

$$\begin{array}{cccccccccccccccccccc}
\ldots & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & \ldots \\
\ldots & 3 & - & 6 & 3 & - & - & 3 & - & 6 & 3 & - & - & 3 & - & 6 & 3 & - & - & \ldots \\
\ldots & 3 & - & 6 & 3 & 9 & - & 3 & - & 6 & 3 & 9 & - & 3 & 9 & 6 & 3 & - & - & \ldots
\end{array}$$

Notice that all of these possibilities are 9-6-3 constructions. Similarly, the other choice of residue modulo 6 (relative to the residue modulo 3) yields the reversal of a 9-6-3 construction:

$$\begin{array}{cccccccccccccccccccc}
\ldots & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & \ldots \\
\ldots & 3 & 6 & - & 3 & - & - & 3 & 6 & - & 3 & - & - & 3 & 6 & - & 3 & - & - & \ldots \\
\ldots & 3 & 6 & 9 & 3 & - & - & 3 & 6 & - & 3 & - & 9 & 3 & 6 & - & 3 & - & - & \ldots
\end{array}$$

or

$$\begin{array}{cccccccccccccccccccc}
\dots & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & \dots \\
\dots & 3 & 6 & - & 3 & - & - & 3 & 6 & - & 3 & - & - & 3 & 6 & - & 3 & - & - & \dots \\
\dots & 3 & 6 & - & 3 & 9 & - & 3 & 6 & - & 3 & - & - & 3 & 6 & 9 & 3 & - & - & \dots
\end{array}$$

or

$$\begin{array}{cccccccccccccccccccc}
\dots & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & 3 & - & - & \dots \\
\dots & 3 & 6 & - & 3 & - & - & 3 & 6 & - & 3 & - & - & 3 & 6 & - & 3 & - & - & \dots \\
\dots & 3 & 6 & - & 3 & - & - & 3 & 6 & 9 & 3 & - & - & 3 & 6 & - & 3 & - & 9 & \dots
\end{array}$$

In order to prevent the moduli 3 and 6 from covering any of the same integers, a 3 must come either immediately after or immediately before a 6 every time a 6 appears in the sequential notation of a 9-6-3 construction. Thus, these are the only two possibilities for the placement of our residues for the moduli 3 and 6 relative to each other. Any valid choice of residue mod 9 will result in a 9-6-3 construction in the first case or the reversal of a 9-6-3 construction in the second case, as seen in the above arrays.

□

We now present an open question regarding the 9-6-3 construction. This question was first posed by Emanuel in [3].

Open Question. *Does there exist some $N \in \mathbb{N}$ such that there exists a 9-6-3 construction for all lengths $n \geq N$?*

Theorem 4 makes a contribution towards definitively answering the question. By Theorem 4, any length 18 or greater IRDCS which uses 3, 6, and 9 as moduli is a 9-6-3 construction or the reversal of a 9-6-3 construction. But even if such an IRDCS is the reversal of a 9-6-3 construction, by the definition of a reversal, its own reversal will be

- A valid IRDCS of the same length
- A 9-6-3 construction

Thus, Theorem 4 implies that the existence of an IRDCS of a given length with 3, 6, and 9 as moduli guarantees the existence of a 9-6-3 construction of the same length.

Based on this result, we have determined a sufficient condition that would guarantee the existence of infinitely many 9-6-3 constructions. This condition follows from a result originally given in [7]. This result is reminiscent of the doubling construction described earlier. Suppose we have two sets of congruences $\{S(m_1, a_1), ..., S(m_s, a_s)\}$ and $\{S(n_1, b_1), ..., S(n_t, b_t)\}$, each of which forms an IRDCS for $[1, n]$. Additionally, suppose their respective sets of moduli are distinct. Then the set $\{S(3m_i, 3a_1 + 1) \mid i = 1...s\} \cup \{S(3n_i, 3b_i + 2) \mid i = 1...t\} \cup \{S(3, 0)\}$ forms an IRDCS for $[1, 3n]$.

Therefore, our sufficient condition is as follows. By the doubling construction, we can create IRDCS of infinitely many lengths with 2 as a modulus. Furthermore, we wish to show that there exist infinitely many lengths for which there is at least one IRDCS with 3 as a modulus. Supposing we can prove this conjecture, let $\{m_i(2)\}$ be the collection of moduli for an IRDCS with modulus 2 on some arbitrary interval of length n. Similarly, let $\{n_i(3)\}$ be the collection of moduli for an IRDCS with modulus 3 on the same interval. If there are infinitely many n such that there exist $\{m_i(2)\}$, $\{n_i(3)\}$ with $\{m_i(2)\} \cap \{n_i(3)\} = \emptyset$, then we will have infinitely many 9-6-3 constructions. This is because applying the tripling construction to these IRDCS will generate an IRDCS with modulus 3 (from taking the union with 0 (mod 3)), modulus 6 (from tripling all of the moduli in the covering with modulus 2), and modulus 9 (from tripling all of the moduli in the other covering). If this IRDCS is a 9-6-3 construction, we're done. If not, it'll be the reversal of a 9-6-3 construction, by Theorem 4. Thus, its reversal will be both a valid IRDCS and a 9-6-3 construction.

On first glance, one might wonder why we chose to focus on this 9-6-3 structure. Why did we not study analagous sequences using the moduli 6, 4, and 2; 12, 8, and 4; or $3n$, $2n$, and n for any $n \geq 2$? The short answer to this question is that the 9-6-3 construction is exactly the right "size" for us to obtain convenient results.

First, a $\{6, 4, 2\}$ sequence is actually impossible to create without producing clashes. The congruence class with modulus 2 will already cover every other integer, so it is impossible to have 2 other integers appear in the sequential notation immediately before an integer covered by 2. We briefly considered IRDCS with a $\{4, 2\}$ pattern. However, this case is somewhat trivial, as we can produce it easily via doubling. Recall that doubling produces a new IRDCS with the congruence 1 (mod 2). Doubling twice will give us an IRDCS using

1 (mod 2) (again) and 2 (mod 4) (from doubling the 1 (mod 2) congruence we obtained in the first double). Thus, we have an IRDCS with the subsequence $\{2, 4\}$ in its sequential notation. Therefore, we can take the reversal of this IRDCS and obtain a $\{4, 2\}$ IRDCS.

Additionally, 3 is a better choice for our smallest modulus in our special construction than is 4 or any other larger number. This is because choosing larger moduli yields more residue choices, thus preventing us from obtaining results similar to the conclusion of Theorem 4. To illustrate this claim, consider the pattern $\{12, 8, 4\}$ in sequential notation (essentially analogous to our 9-6-3 construction, but we replace each modulus $3n$ with $4n$). Since the least common multiple of these moduli is 24, we will obtain a pattern of length 24 in the same manner as we obtained a length 18 pattern with the 9-6-3 construction. This implies that there are 24 possible arrangements that will lead to this construction. However, fixing a residue for the modulus 4 in general leaves us with 6 choices for our residue modulo 8. Fixing a residue for the moduli 4 and 8 leaves us with 6 choices for our residue moduluo 12. Thus, taking into consideration our initial 3 choices for our residue modulo 4, we have $3 * 6 * 6 = 108$ total choices. Since $108 > 2 * 24$, the existence of an IRDCS with 4, 8, and 12 as moduli does not guarantee the existence of a 12-8-4 IRDCS. We obtain similar results for other constructions of the form $3n, 2n, n$, where $n \geq 5$.

5 Overview of the Algorithm

For IRDCS of longer lengths, it quickly becomes infeasible to compute all 9-6-3 constructions manually. We therefore create a computer algorithm to compute all IRDCS of a given length based on the algorithm detailed in [3] and [7]. The program builds an IRDCS in sequential notation through *backtracking*. Backtracking refers to the process of exhaustively attempting to cover a position in $[1, n]$ with all possible moduli and if necessary, transitioning back to the most recently covered position to update there.

The program contains three major arrays: x builds our IRDCS in sequential notation, *modusage* stores what moduli have been used, and *primary* specifically indicates how to backtrack. We modify the program to print only 9-6-3 constructions and their reversals by nesting the *print* statement within a loop only entered when moduli 3, 6, and 9 are used.

6 The Algorithm, Technically

The program begins by setting the x array of length n to have 1's at every position. This will represent the IRDCS in sequential notation, built one modulus at a time. We define a Boolean variable *clash* that is set to true when the program attempts to cover a single position with two separate moduli (doing so would contradict the *disjoint* property of the IRDCS). Consequently, the program backtracks to the position most recently covered and attempts to use a larger modulus to cover it, repeating as necessary.

The *modusage* array of length n has a 0 or 1 in position p if the modulus p can or cannot be selected at that point in the algorithm, respectively. We immediately exclude modulus 1 to avoid trivialities. We also exclude modulus n to ensure that each congruence covers at least two integers. The *modusage* array is therefore initialized as $[1, 0, 0, ..., 0, 1]$.

The *primary* array determines the position to which the algorithm must backtrack. It puts a 1 in position p if p is the primary integer that it is attempting to cover. It then puts 0s in all positions corresponding to integers covered *as a secondary result* of covering p with a particular choice of modulus. The positions corresponding to uncovered integers have 1's, since they have not yet been covered by any congruences. The algorithm continues to backtrack until primary[i] = true.

The variable *position* stores the current integer we are trying to cover. This position is where the primary modulus will be inserted. We always commence in the middle of the interval by initializing *position* at the floor of $(n/2) + 1$. Additionally, position is adjusted by the variables *increment* and *polarity*. The variable *increment* is first set to 1 and then increased by 1 with every change in position. *Polarity* oscillates in value between ± 1, beginning with -1 and switching sign with every change in position. This setup enables the algorithm to fill the x array by reflecting around the middle position. For example, an IRDCS of length 11 is built by covering the middle position, followed by the 5th position, the 7th position, etc. This ordering optimizes the algorithm's efficiency, as stated in [3].

The variable *maxmodulus* is constantly updated to calculate the maximum possible modulus that can be used to cover the current position. Given a typical IRDCS of length n, *maxmodulus* = max(n -*position*, *position* - 1).

The variable *finished* has the value *false* until the algorithm has exhaustively found all IRDCS. We now present the algorithm in Sage construct, with assistance from E. Wesson.

```
%First, we create functions that alias indexing.

def place(vec,k):
    return vec[k-1]

def setplace(vec,k,num):
    vec[k-1] = num


% Length of desired IRDCS.

n = 19

% X Array default: fill every position
    with 1 (IRDCS in sequential notation).

x = [1]*n

% Modusage Array default: fill every
  available modulus to 0 except the
  moduli 1 and n which are set to 1.

% Used to calculate the next available modulus.

modusage = [0]*n
setplace(modusage,1,1)
setplace(modusage,n,1)

% Primary array default: every position to 1.
  Used when backtracking, calculates the correct
  position to backtrack.
```

```
primary = [1]*n

% Current integer which is being covered, congruence is
      calculated with respect to this variable.

position = floor(n/2)+1

% Used in calculating the next position.

polarity = −1

% Used to move through all the integers. Is
    a sum or difference by 1 each change of position.

increment = 1

% Used to exit WHILE A.

finished = False

% Variable which changes to true when two
  congruences are not disjoint. Forces a
  backtrack.

clash = False



% WHILE A: Main Loop

while not finished:

    % The calculation of a simple IRDCS
      with no further restrictions.
```

```
maxmodulus = max(n-position, position-1)

% Calculation of m(current modulus)
  with respect to position.

m = place(x,position)+1

% IF/ELSE B: Calculates next
  available modulus.

if m < n-1:
    while (place(modusage,m) == 1):
        m += 1
else:
    m = maxmodulus+1

% END IF/ELSE B


% IF C: Fills the interval with
  congruences using available moduli.

if m <= maxmodulus:

    % Calculation of the congruence
      which covers position.

    setplace(x,position,m)
    i = mod(position,m).lift()
    if i == 0: i += m

    % WHILE D: Fills the X array with
      proper residues. Evaluates clashes.
```

```
while i <= n and not clash:
    setplace(modusage,m,1)

    %IF E: Saves the moduli
      at position.

    if i != position:

        % IF F

        if place(x,i) == 1:
            setplace(x,i,place(x,position))
            setplace(primary,i,0)
            i += m
        else:
            i -= m
            clash = True

        % END IF/ELSE F

    else:
        i += m

     % END IF/ELSE E

% END WHILE D

% IF/ELSE G: When clash has been
  detected, backtrack to the current
  position.

if clash:

    setplace(modusage,m,0)
```

```
% WHILE H:

while i >= 1:

    %IF I

    if place(primary,i)==0:

            % IF J

            if i != position:
                setplace(x,i,1)
                setplace(primary,i,1)

            % END IF J

            i -= m

    else:
            i -=m

    % END IF/ELSE I


% END WHILE H

clash = False


else:

% WHILE K: Check to see if the
  current X is an IRDCS.

while 1 <= position and position <= n and
```

```
place(x,position) > 1:
    position += increment*polarity
    increment += 1
    polarity *= -1

% END WHILE K

% IF L: Prints current X if done, resets X to make a
new IRDCS through the backtracking proccess.

if position < 1 or position > n:
    print x
    setplace(modusage,m,0)
    increment -= 1
    polarity *= -1
    position -= increment*polarity

    % WHILE M

    while place(primary,position) == 0:
        increment -= 1
        polarity *= -1
        position -= increment*polarity

    % END WHILE M

    i = mod(position,m).lift()
    if i == 0: i += m

    % WHILE N

    while i <= n:

        % IF O
```

```
            if i != position:
                setplace(x,i,1)
                setplace(primary,i,1)
            i += m

            % END IF O

        % END WHILE N

    % END IF L

  % END IF/ELSE G

% ELSE C: Backtracking to change the previously used moduli,
 since all m have been exhausted for current position.

else:

  % IF P: If position is the initial position it is a
   complete system of IRDCS for length n, else backtrack.

  if position == floor(n/2)+1:
      finished = True
      print "No more solutions"

  else:
      setplace(modusage,place(x,position),0)
      setplace(x,position,1)
      increment -= 1
      polarity *= -1
      position -= increment*polarity



    % WHILE Q
```

```
            while place(primary, position) == 0:
                increment -= 1
                polarity *= -1
                position -= increment*polarity

            % END WHILE Q

            m = place(x, position)
            i = mod(position, m).lift()
            if i == 0: i += m
            setplace(modusage, place(x, position), 0)

            % WHILE R

            while i <= n:

                % IF S

                if i != position:
                    setplace(x, i, 1)
                    setplace(primary, i, 1)
                i += m

                % END IF S

            % END WHILE R

        % END IF/ELSE P

    % END IF/ELSE C

% END WHILE B
```

The pseudocode presented in Emanuel's thesis [3] had to be modified in the following ways:

(1) The *modusage* array was never set to *true*, meaning it never indicated when a modulus was used.

(2) The backtracking section did not clear the appropriate modulus and primary data for the current position and the position to which it was backtracking. Furthermore, it contained an infinite loop.

Since our research centers around the 9-6-3 construction, we modify the program to output only 9-6-3 constructions and their reversals. Furthermore, we streamline our algorithm by applying Theorem 2. This theorem permits us to exclude several moduli that are guaranteed to clash with the prime 3 in our construction. Thus, we set *modusage* equal to true for these moduli.

The algorithm has provided us with 9-6-3 constructions of all lengths from 19-23 and 25-490. Some selected results are presented in the following table.

Selected Results for 9-6-3 IRDCS

Length (n)	Number of 9-6-3 IRDCS	Order
18	0	-
19	4	7
20	3	7
21	1	7
22	1	8
23	1	8
24	0	-
25	6	8
26	7	8
27	5	8
28	8	8,9
29	17	9
30	19	9
31	16	9
32	70	9,10
33	103	9,10
34	75	9,10

Table 1: Number of 963 IRDCS of various lengths and their respective orders.

Thus, we conjecture the following answer to Emanuel's open question.

Conjecture 1. *There exist 9-6-3 constructions* $\forall\ n \in \mathbb{N}$ *such that* $n \geq 25$.

7 Conclusion

To summarize, we establish a restriction on the maximum number of integers covered by a given congruence with relation to a prime (Theorem 2). We sharpen the lower bound on the possible lengths for a 9-6-3 construction. Also, we propose that 25 is the minimum possible value of N for which we can obtain 9-6-3 constructions for all $n \geq N$. Additionally, we speculate that we can answer a weaker version of the open 9-6-3 question in the affirmative through the "tripling construction." Our modification of the original IRDCS algorithm will determine whether a given length will have any 9-6-3 constructions. More processing speed would enable future researchers to definitively compute the number of 9-6-3 constructions for more lengths.

8 Acknowledgements

We would like to thank our advisor, Dr. Mark Kozek, and our TA, Elizabeth Wesson. Their guidance was instrumental throughout this project. Additionally, we would like to acknowledge the programming assistance provided by Dr. Damien Martin. We would also like to express our gratitude to Dr. Edward Swartz, the director of SMI, as well as the Cornell Mathematics Department and the Center for Applied Mathematics. Finally, we are very grateful for the financial support provided by the National Science Foundation.

References

[1] J. Beebe, Examples of infinite, incongruent exact covers, *The American Mathematical Monthly.* **95** (1988) 121–123.

[2] M.A. Berger, A. Felzenbaum, A.S. Fraenkel, R. Holzman, On Infinite and Finite Covering Systems, *The American Mathematical Monthly.* **98** (1991) 739–742.

[3] P. R. Emanuel, Covering Systems, *Thesis: Macquarie University.* (2011)

[4] P. Erdős, On integers of the form $2^k + p$ and some related problems, *Summa Brasil. Math.* **2** (1950) 113–123.

[5] E. Lewis, Infinite covering systems of congruences which don't exist, *Proceedings of the American Mathematical Society.* **124** (1996) 355–360.

[6] G. Myerson, J. Poon, J. Simpson, Computing incongruent restricted disjoint covering systems, *International Workshop on Combinatorial Algorithms 07.* (2007) 137–142.

[7] G. Myerson, J. Poon, J. Simpson, Incongruent restricted disjoint covering systems, *Discrete Mathematics.* **309** (2009) 4428–4434.

[8] G. Myerson, J. Simpson, in *Western Number Theory Problems, 18 and 20 Dec 2006.* (2006) 4.

[9] Z. W. Sun, On the Herzog-Schőnheim conjecture for uniform covers of groups, *Journal of Algebra* **273** (2004) 153175.

[10] C. Vanden Eynden, On a problem of Stein concerning infinite covers, *The American Mathematical Monthly.* **99** (1992) 355–358.