Improvement to Lotto Design Tables

by

Lutful Karim

A dissertation submitted in partial satisfaction of the
requirements for the degree of

Master of Science

Department of Computer Science

Faculty of Graduate Studies
University of Manitoba

Abstract

An (n, k, p, t) lotto design is a collection of k-subsets of a set X of n numbers wherein every p-subset of X must intersect at least one k-subset in t or more elements. $L(n, k, p, t)$ is the minimum number of k-subsets which guarantees an intersection of at least t numbers between any p-subset of X and at least one of the k-subsets. To determine $L(n, k, p, t)$ is the main goal of lotto design research. In previous work on lotto designs, other researchers used sequential algorithms to find bounds for $L(n, k, p, t)$. We will determine the number of non-isomorphic optimal lotto designs on 5 or 6 blocks for $n, k, p, t \leq 20$ and also improve lower bounds for $L(n, k, p, t) \geq 6$ if possible by a more efficient implementation of a backtracking algorithm.

Acknowledgements

I would like to express my sincere gratitude and appreciation to my supervisor Dr. G.H.J van Rees who made this masters thesis possible. He helped me greatly in the research and writing this thesis. Without his help, this work could not have been completed.

I am very grateful to Dr. P.C Li and Dr. R. Padmanabhan for being on my thesis committee.

I would like to thank my parents, brothers and sisters for their continuous support and encouragement to finish the thesis successfully.

I would also like to thank my friends for their valuable suggestions.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Most governments run lotteries for charity and to collect funds to develop different sectors of the government. For example, a lottery is run to collect funds to develop sports in Bangladesh. Most lotteries have the same functional scheme. The lotteries that casinos run are known as Keno whereas the lotteries run by governments are known as Lotto. However, both work in the same way. Each lottery ticket has k numbers selected from the set of n numbers (e.g., $1, 2, 3, \ldots, n$). People buy lottery tickets and pick k numbers per ticket. When ticket sales are closed, the government picks p numbers from the set of n numbers. If a ticket matches at least t numbers from p numbers that the government picks, the ticket holder wins a prize. The larger the value of t is, the greater the prize will be.

Lotto designs can best be understood by examining an example: the Canadian lotto 6/49. Here, $n = 49$ (i.e. the set consists of 49 numbers), $k = 6$, $p = 6$ and $t = 3$, 4, 5 or 6. Each ticket is a set of 6 numbers. The government picks 6 numbers randomly from the set of 49 numbers. A person wins a prize if one of his or her tickets matches at least 3 numbers of the numbers the government picks. To guarantee a win in lotto 6/49, the minimum number of tickets that must be bought is between 87

and 163 for $t = 3$ [14]. The minimum number of tickets that must be bought is 19 for $t = 2$ [14]. In order to win the big jackpot, a person needs to buy $\binom{49}{6} = 13983816$ tickets. The main task in this area of research is to find out the minimum number of tickets that guarantees a match of at least t numbers between one of the tickets and the government's pick for various values of n, k, p and t.

1.1 Definitions

In this section, we will define some important terms related to lotto designs.

Definition 1.1.1 *: An n-set is a set of n elements and thus, a k-set consists of k-elements.*

Definition 1.1.2 *: If A is a set of x elements and y is an integer such that $y \leq x$ then B is a y-subset of A if $B \subseteq A$*

We now formally define lotto designs.

Definition 1.1.3 *: Let n, k, p and t be positive integers. Suppose, X is a set of n numbers and $\mathcal{B}$ is a collection of k-subsets of X. Then $(X, \mathcal{B})$ is called an (n, k, p, t) lotto design if any p-subset of X must intersect some k-subset of $\mathcal{B}$ in at least t elements. The k-subsets of X are called the blocks of the lotto design.*

The minimum number of blocks in an (n, k, p, t) lotto design is denoted by $L(n, k, p, t)$. An (n, k, p, t) lotto design with b blocks is denoted by $LD(n, k, p, t; b)$. An (n, k, p, t) lotto design with $L(n, k, p, t)$ blocks is called an optimal or minimal design. An optimal lotto design is denoted by $LD^*(n, k, p, t; b)$, where $b = L(n, k, p, t)$.

In this thesis, we will try to find $\eta(n, k, p, t)$ on 5 or 6 blocks with n, k, p, t less than or equal to 20 (i.e. entries in Li's [15] table). For those cases that will take too

long and for which it is known only that $L(n, k, p, t)$ is greater than or equal to 6, we will determine if $L(n, k, p, t)$ is equal to 6 or not.

We will use the following definitions:

Definition 1.1.4 *: A p-set is represented by a k-set if the p-set intersects the k-set in at least t elements.*

To find an (n, k, p, t) lotto design, all p-sets must be represented by some k-set or block. Now we will give two examples of lotto designs.

Example 1.1.1 *: Let,* $X = \{1, 2, 3, 4, 5\}$ *and* $\mathcal{B} = \{\{1\ 2\ 3\}, \{1\ 2\ 4\}, \{1\ 2\ 5\}, \{3\ 4\ 5\}\}$*.* $(X, \mathcal{B})$ *forms a (5, 3, 2, 2) lotto design as any 2-subsets of* X *intersects at least a block of* $\mathcal{B}$ *in* $t = 2$ *elements.*

Example 1.1.2 *: Let,* $X = \{1, 2, 3, 4, 5, 6, 7\}$ *and* $\mathcal{B} = \{\{1\ 2\ 3\}, \{3\ 4\ 5\}, \{5, 6, 7\}\}$*.* $(X, \mathcal{B})$ *forms a (7, 3, 4, 2) lotto design as any 4-subset of* X *intersects some block of* $\mathcal{B}$ *in at least* $t = 2$ *elements. This is not a minimal lotto designs as (*X*,* $\{\{1, 2, 3\}, \{4, 5, 6\}\}$*) forms another (7, 3, 4, 2) lotto design with fewer blocks. Since no 3-set from* X *forms an* $LD(7, 3, 4, 2)$*, then* $L(7, 3, 4, 2) = 2$*.*

Isomorphism has an important role in this area. We will define isomorphism as follows:

Definition 1.1.5 *: An isomorphism between two lotto designs,is a one-to-one, onto mapping of the elements of one lotto design to the elements of another lotto design with the property that the elements in a block of the first design get mapped to the elements in a block of the second design.*

Definition 1.1.6 *: Two lotto designs are isomorphic if there exists an isomorphism from one to the other.*

A less powerful, but easier to implement idea is trivial isomorphism. We now define it.

Definition 1.1.7 *: Two elements are trivially isomorphic if when one of them occurs in a block so does the other.*

A block either contains both of them or neither of them.

Definition 1.1.8 *Two k-sets are trivially isomorphic or equivalent if one can be transformed into the other by replacing one or more elements by their trivially isomorphic elements.*

Definition 1.1.9 *A trivial isomorphism is a one-to-one,onto mapping of the elements of one lotto design to the trivially isomorphic elements of another lotto design with the property that the elements in a block of the first design get mapped to the elements in a block of the second kind.*

Definition 1.1.10 *Two lotto designs are trivially isomorphic if there exists a trivial isomorphism from one to the other.*

Clearly, all trivial isomorphisms between lotto designs are isomorphisms but the converse is not necessarily true.

1.2 Problem Statement

Researchers are working on lotto design tables to improve bounds using different algorithms or methods so that known upper bounds get closer to known lower bounds. Li [15] has large tables for upper and lower bounds of lotto designs. Recently, a new approach to finding good lotto designs was introduced by a group of South African

researchers A.P. Burger, W.R. Grundlingh and J.H. van Vuuren [5]. They developed a backtracking algorithm to find the number of non-isomorphic optimal lotto designs. The total number of non-isomorphic optimal lotto designs for a fixed n, k, p, t is denoted by $\eta(n, k, p, t)$. We also plan to find the number of non-isomorphic optimal lotto designs for $L(n, k, p, t)$ on 5 or 6 blocks with $n, k, p, t \leq 20$ from Li's table and Burger's result or if that is impossible find better lower bounds for these parameters.

In the past, others have also used exhaustive backtracking algorithms to solve the lotto problem. However, these algorithms require much time to traverse the state space tree or search tree. The search tree grows exponentially with the size of the parameters. Hence, the program that implements this backtracking algorithm may not compute $L(n, k, p, t)$ in a reasonable time. A rule of thumb for "reasonable time" is one hour of CPU time for each input. Therefore, we will develop an algorithm which will generate all possible non-isomorphic sets of three blocks using block intersection properties. By generating all non-isomorphic sets of three blocks, it is possible to avoid backtracking on the first three blocks. Backtracking to the first three blocks is very slow. Thus, these techniques will reduce the search time to a great extent. We will generate the rest of the blocks with trivial isomorphism rejection which will prune the search tree quickly, but not as completely as the much slower complete isomorphism rejection. We will further speed up the program by applying some new frequency results in lotto designs. The algorithm may still generate isomorphic designs for a particular set of parameters n, k, p, t, so we will run Kocay's [13] program to eliminate these and produce a list of all non-isomorphic designs.

1.3 Thesis Overview

The remaining chapters of this thesis are organized as follows:

In Chapter 2, some background information and history related to lotto designs will be presented. Firstly, we will briefly discuss Tuŕan and covering designs and Dan Gordon's [10] works on the upper bounds of covering designs. Then we will discuss the seminal work done by Bate [1, 2]. Furedi et al. [9] represent lotto designs using multigraphs and their lower bounds formula of lotto designs for $t = 2$ will be discussed. Some other works of De Caen [8], Brouwer and Voorhoeve [4] on lower bounds of lotto designs will be presented. We will then discuss the most extensive work on lotto designs done by Li and van Rees [16] which also appear in Li's thesis [14]. Finally, we will discuss recent work by A.P. Burger, W.R. Grundlingh and J.H. van Vuuren [5], a group of South African researchers on lotto designs

Using isomorphism rejection, basic backtracking algorithms can be made much more efficient. In Chapter 3, the isomorphism rejection techniques of Bate will be discussed. Then we will describe our basic serial backtracking algorithm and the improvement made to it. We will illustrate the improved backtracking algorithm with examples. We will also briefly describe the data structures used for the implementations of the improved backtracking algorithms.

In Chapter 4, the total number of non-isomorphic designs for $L(n, k, p, t)$ on 5 or 6 blocks for $n, k, p, t \leq 20$ and/or improved lower bounds for these parameters will be listed in a tabular form. We will also list designs in a condensed (encoded) form quite similarly to the form used by A.P. Burger et al.

In Chapter 5, we will summarize our works and results. We will also try to give further research directions.

By implementing our algorithm, we verified $\eta(n, k, p, t)$ for 14 lottery numbers of A.P Burger et al. and generated $\eta(n, k, p, t)$ for 112 new lottery numbers and improved lower bounds of $L(n, k, p, t)$ for 18 lottery numbers.

Chapter 2

Background

In this chapter, we will briefly discuss the research of several researchers working on lotto designs. We will start by defining covering and Tuŕan designs which are special cases of lotto designs. Then, we will mention some work of Bate's on lotto designs from his doctoral thesis. We will then state the lower bound formulas of Furedi et al. Hanani et al. De Caen, Brouwer and Voorhoeve. More extensive work on lotto designs were done by Li and van Rees. We will briefly discuss their theoretical results and computer algorithms. We will use lotto designs with 5 or 6 blocks from Li's tables and will try to find the number of non-isomorphic lotto designs for these parameters or try to improve the lower bounds of these designs. Recently, a group of South African researchers A.P. Burger, W.R. Grundlingh and J.H. van Vuuren introduced the concept of a "Lottery Graph" and got improved bounds. We will finish by introducing their results.

2.1 Covering and Tuŕan Designs

Lotto designs are a generalization of covering designs and Tuŕan designs. Lotto designs where $p = t$ are called (n, k, t) covering designs or coverings and the lotto designs where $k = t$ are called (n, p, t) Tuŕan designs. The minimum number of blocks in (n, k, t) covering designs is denoted by $C(n, k, t)$ and the minimum number of blocks in (n, p, t) Tuŕan designs is denoted by $T(n, p, t)$. Hence, $L(n, k, t, t) = C(n, k, t)$ and $L(n, t, p, t) = T(n, p, t)$. Tuŕan designs and covering designs are closely related as they have the following relationship:

$C(n, k, t) = T(n, n - k, n - t)$

A great deal of work has been done in covering designs and Tuŕan designs. Results on these designs are recorded in Gordon's tables. He has a large table [10] of (n, k, t)-coverings with and less than 50,000 blocks. He also presents some new constructions for coverings in [11]. These constructions include greedy coverings which do not depend on smaller coverings. In these greedy coverings, a k-subset from the list of k-sets will be added to the design if this k-set represents the maximum number of t-sets that are still unrepresented. This process is repeated until the design is found. Other constructions are finite geometry coverings, induced coverings and smaller coverings using dynamic programming method. We are not going to do much work with the minimum number of blocks $C(n, k, t)$ and $T(n, p, t)$, because these are well studied.

2.2 Bate's Work on Lotto Designs

One of the earliest works related to lotto design was done by Bate [1]. Bate describes generalization of covering designs which he calls a "generalized (T, K, L, V) design". (T, K, L, V) designs are equivalent to (V, K, L, T) lotto designs. Bate investigated

$(V, 2, L, 2)$ which is also known as the Tuŕan problem. He determines the minimum number of blocks of the design, $B(V, K, K, T)$, for $T = 2$, $K = 3$ and 4. He also determines $B(V, K, L, T)$ for $T = 2$, $K = 3$ or 4, and $L \leq 5$.

Bates states and proves the following theorems of about lotto designs [14].

Definition 2.2.1 *The complement of a k-set is the set of n-k elements which do not occur in the k-set. The complement of an (n, k, p, t) lotto design is obtained by complementing all the k-sets of the design.*

Theorem 2.2.1 *: The complement of an (n, k, p, t) lotto design is an $(n, n-k, n-p, n-k-p+t)$ lotto design. Thus $L(n, k, p, t) = L(n, n-k, n-p, n-k-p+t)$.*

Example 2.2.1 *: Let {{1 2 3 4 5 6},{1 2 3 4 7 8},{1 2 3 4 9 10},{5 6 7 8 9 10},{11 12 13 14 15 16}} be an $LD(16, 6, 11, 5; 5)$. If we complement each block, we get {{1 2 3 4 5 6 7 8 9 10}, {1 2 3 4 5 6 7 8 11 12}, {1 2 3 4 5 6 9 10 11 12}, {1 2 3 4 5 6 13 14 15 16}, {7 8 9 10 11 12 13 14 15 16}} which is an $LD(16, 10, 5, 4; 5)$. These are optimal so L(16 6 11 5) = L(16 19 5 4) = 5.*

Bate also states a theorem for finding $L(n, 3, 3, 2)$. Later, this theorem was independently presented by Brouwer [4]. In our terminology, it becomes:

Theorem 2.2.2 *:*
$$L(n,3,3,2) = \begin{cases} \lceil \frac{n^2-2n}{12} \rceil & if n \equiv 2,4,6 (mod 12) \\ \lceil \frac{n^2-2n}{12} \rceil + 1 & if n \equiv 0,8,10 (mod 12) \\ \lceil \frac{n^2-n}{12} \rceil & if n \equiv 1,3,5,7 (mod 12) \\ \lceil \frac{n^2-n}{12} \rceil + 1 & if n \equiv 9,11 (mod 12) \end{cases}$$

Bate also states the following theorems. We state them in our terminology:

Theorem 2.2.3 *:* $L(3n, 3, 4, 3) \leq 3\binom{n}{3} + 3n\binom{n}{2}$

Theorem 2.2.4 *:* $L(3n+1,3,4,3) \leq 2\binom{n}{3} + \binom{n+1}{3} + (2n+1)\binom{n}{2} + n\binom{n}{2}$

Theorem 2.2.5 *:* $L(3n+2,3,4,3) \leq \binom{n}{3} + 2\binom{n+1}{3} + (2n+1)\binom{n+1}{2} + (n+1)\binom{n}{2}$

Bate uses a backtracking algorithm along with some optimizations to compute many $L(V,K,L,T)$s. The basic backtracking algorithm of Bate begins with computing all L-sets. Then, the algorithm picks the first unrepresented L-set and adds a K-set to the design which represents that L-set. In this way, the algorithm recursively finds an unrepresented L-set and adds a K-set to the design which represents that L-set. If the design reaches the maximum number of K-sets (i.e. upper bound determined before hand) without representing all the L-sets, the algorithm backtracks to the previous level. A solution is found when all the L-sets have been represented and the design does not have more than the maximum number of K-sets. As the basic backtracking algorithm is slow, Bate uses some optimizations, namely preclusion and isomorphic rejection. In Chapter 3, we will present the pseudo-code of the basic backtracking algorithm of Bate.

Bate used the following formula to find the number of L-sets represented by a K-set:

$$C = \sum_{i=T}^{min(K,L)} \begin{pmatrix} K \\ i \end{pmatrix} \begin{pmatrix} V-K \\ L-i \end{pmatrix} \tag{2.1}$$

Bate also uses "LIMIT" at the beginning of his program, which determines the maximum number of blocks a design may have, and initially "LIMIT" is set to a large number. When a design is found, "LIMIT" is the number of blocks in the just found design minus one. He uses another variable "MAXPOS", which determines the maximum number of L-sets that can be represented by blocks yet to be taken. "MAXPOS" can be determined by combining two variables C and "LIMIT" as follows:

$$MAXPOS = C * (LIMIT - y) \tag{2.2}$$

where y is the number of blocks already in the design.

A variable "NLEFT" is also used to count the L-sets which have not been represented. The value of "NLEFT" decreases whenever L-sets are represented and increases whenever L-sets are unrepresented. Thus, for preclusion, these two variables "MAXPOS" and "NLEFT" are maintained with a few additional computations. When $MAXPOS < NLEFT$ then the program backtracks to the previous level. This means that whenever the maximum number of L-sets that can be represented by the additional blocks of the design is less then the number of L-sets left to be represented in the design, the program backtracks to the previous level. In this way, the program avoids traversing large areas of the search tree and thus, the program can be sped up to a great extent.

Isomorphic rejection techniques also significantly speed up backtracking algorithms. Using these techniques, a K-set A is rejected on a level X if the inclusion of A does not lead to a successful design. Any other block B is also rejected if the block B generates an isomorphic partial design to the partial design generated by A on level X. To implement isomorphic rejection, a rejection flag is used with each of the K-sets. This flag is set if the K-set cannot be added on a level X. Before adding a K-set to the design, the rejection flag of the K-set is checked. If the flag of a K-set is set, then K-set cannot be added to the design and is ignored. When the program backtracks to the previous level, X - 1, the rejection flags of all K-sets at the level X are reset. This type of complete isomorphism testing eliminates the duplication of all isomorphic subtrees. The cost of complete isomorphism testing is too great. So, Bate did a simplified form of isomorphism testing which eliminates

most duplication. In this simple form of isomorphism testing which he called trivial isomorphism testing, a table is used to maintain the list of isomorphic elements. The elements which occur either in all blocks already added in the design or in no block of the design are called trivially isomorphic elements. Two blocks are trivially isomorphic if one can be transformed into the other by replacing one or more elements by their corresponding trivially isomorphic elements. A rejection flag is used for each of the blocks to implement the trivial isomorphism rejection scheme. When a block is added to the design the table of trivially isomorphic elements is updated and all trivially isomorphic blocks of the added block are generated and the rejection flags of these blocks are set. A block can not be added to the design if its rejection flag is set. When the program backtracks to the previous level, all rejection flags are reset.

This simple form of isomorphism rejection reduces the size of the search tree and therefore also reduces the search time.

In Bate's algorithm, LIMIT is set to a large number. By using Li's table [16], LIMIT can be set to the upper bound found in Li's tables thus reducing the search time to a great extent.

Bate also states some theorems which will be useful later in the thesis. Again, we use our notation.

Theorem 2.2.6 *:* $L(n,k,p,t) = 1$ *iff* $t \leq k + p - n$

Bates [1] has also some results in lotto designs based on the frequency of elements.

Theorem 2.2.7 *There exists a minimal or optimal* $LD(n,k,p,t;b)$ *in which all elements occur at least once or the design contains no repeated elements.*

Proof:

Consider any minimal $LD(n,k,p,t;b)$, L. If there are no repeated elements, then we are done. So let a_1 be an repeated element in the minimal lotto design $LD(n,k,p,t;b)$. Let a_2 be an element that does not occur in this design. Again if no a_2 exists, we are done. Let B be a block containing a_1. Replace a_1 with a_2 in B. We claim the modified L is also a minimal $LD(n,k,p,t;b)$. Assume it is not. The only way it could not be a minimal lotto design if some p-set, P_1, containing a_1 is not represented in the modified lotto design. Then P_1 contains a_1 and $p-1$ other elements from X. Now, P_1 is not represented by any other block in L. So the p-set $(P_1 \setminus \{a_1\}) \cup \{a_2\}$ cannot be represented in L by B or by any of the other block in L as a_2 does not appear in L. But this contradicts that L is a lotto design.

Thus, we can modify any minimal $LD(n,k,p,t;b)$ by replacing repeated elements with non-occuring elements until all elements occur at least once.□

Corollary 2.2.8 *: If there exists a minimal $LD(n,k,p,t;b)$ with $b \geq \frac{n}{k}$ then there exists a minimal $LD(n,k,p,t;b)$ in which each element occurs in the design at least once. If there exists a minimal $LD(n,k,p,t;b)$ with $b \leq \frac{n}{k}$ then there exists a minimal $LD(n,k,p,t;b)$ in which the blocks are pair-wise disjoint and all elements in them have frequency 1.*

Proof:

In the first case there are enough repeated elements to ensure each non-occuring element can be traded into the design. In the second case, all the repeated elements can be traded with non-occuring elements until every element has frequency 0 or 1. Then the blocks must be pair-wise disjoint.□

This section can be finished with a lemma from [17].

Lemma 2.2.9 *: $p \leq (t-1-k)\lfloor \frac{n-1}{k} \rfloor + n$ if and only if there exists a minimal $LD(n,k,p,t;b)$ in which every element has frequency at least one.*

2.3 Work on Lower Bounds of Lotto Designs

Furedi et al. [9] give the lower bounds for $L(n,k,p,2)$. They represent $(n,k,p,2)$ lotto designs using multi-graphs and transform such a multi-graph into another multi-graph which contains p - 1 disjoint sub-graphs. Then they calculate lower bounds on $L(n,k,p,t)$ by analyzing the newly transformed multi-graph. They calculate $L(n,k,p,2)$ using the following formula:

$$L(n,k,p,2) \geq \frac{1}{k} \sum_{i=1}^{min(p-1)} a_i \left(\sum_{i=1}^{p-1} a_i \left\lceil \frac{a_i - 1}{k-1} \right\rceil\right) \tag{2.3}$$

Where,$a_1, a_2, \ldots, a_{p-1}$ are integers and $\sum_{i=1}^{p-1} a_i = V$, the set of vertices.

Hanani et al. [12] work on lottery designs and give the following lower bound formula:

$$L(n,k,p,t) = \frac{n(n-p+1)}{k(k-1)(p-1)} \tag{2.4}$$

De Caen [8] works on Tuŕan designs and gives the following lower bound formula for Tuŕan designs:

$$T(n,p,t) = \frac{\binom{n}{t}}{\binom{p-1}{t-1}} \cdot \frac{n-p+1}{n-t+1} \tag{2.5}$$

Brouwer and Voorhoeve [4] also work on Tuŕan designs and give the following lower bound formula for lotto designs:

$$L(n,k,p,t) \geq \frac{T(n,p,t)}{\binom{k}{t}} \tag{2.6}$$

The results of De Caen and Brouwer and Voorhoeve can be generalized to the following lower bound formula for lotto designs which was found in Li's doctoral thesis [14].

$$L(n,k,p,t) \geq \frac{\binom{n}{t}}{\binom{p-1}{k-1} \cdot \binom{k}{t}} \cdot \frac{n-p+1}{n-t+1} \tag{2.7}$$

2.4 Li's Work on Lotto Designs

Li [14] develops different methods for computing lower and upper bounds on $L(n,k,p,t)$. He also has a large table of $L(n,k,p,t)$ where, $5 \leq n \leq 20$, $2 \leq \{k,p\} \leq n$, $2 \leq t \leq \min\{k,p\}$.

Li presents constructions to generate upper bounds for $L(n,k,p,t)$. He defines balanced incomplete block designs (BIBD) and also states the conditions for a BIBD to be a lotto design.

Definition 2.4.1 *: An (v,b,r,k,λ) BIBD has a collection of b, k-sets of a set X(v) of v elements wherein each element occurs r times and each pair of elements in the k-sets occurs λ times.*

Theorem 2.4.1 *: If $\mathcal{B}$ is the collection of k-sets of a (v,b,r,k,λ) BIBD and p, t are two positive integers such that $\lfloor \frac{pr}{t-1} \rfloor C(t-1,2) + C(pr - \lfloor \frac{pr}{t-1} \rfloor (t-1),2) < C(p,2)\lambda$ then $\mathcal{B}$ is the set of blocks of an (v,k,p,t) lotto design and $L(v,k,p,t) \leq b$*

Corollary 2.4.2 *: If $r < 2\lambda$ in a (v,b,r,k,λ) BIBD then $L(v,k,5,4) \leq b$*

Corollary 2.4.3 *: If $r < \frac{5}{2}\lambda$ in a (v,b,r,k,λ) BIBD then $L(v,k,6,4) \leq b$*

Corollary 2.4.4 *: If $r < 3\lambda$ in a (v, b, r, k, λ) BIBD then $L(v, k, 7, 4) \leq b$*

Li also defines monotonicity formulas which relate the lower bounds or upper bounds of one lotto design to the lower bounds or upper bounds of another lotto design. Some of these monotonicity results will be used.

Theorem 2.4.5 *:*

$L(n, k, p, t) \leq L(n + 1, k, p, t)$
$L(n, k, p, t) \geq L(n, k, p + 1, t)$
$L(n, k, p, t) \geq L(n + 1, k + 1, p, t)$
$L(n, k, p, t) \geq L(n + 1, k + 1, p + 1, t)$
$L(n, k, p, t) \geq L(n, k + 1, p, t)$
$L(n, k, p, t) \leq L(n + 1, k, p, t + 1)$
$L(n, k, p, t) \leq L(n + 1, k + 1, p + 1, t + 1)$
$L(n, k, p, t) \leq L(n + 1, k + 1, p, t + 1)$
$L(n, k, p, t) \leq L(n + 1, k, p, t + 1)$
$L(n, k, p, t) \leq L(n, k + 1, p, t + 1)$
$L(n, k, p, t) \leq L(n, k, p + 1, t + 1)$
$L(n, k, p, t) \leq L(n + 1, k, p + 1, t + 1)$
$L(n, k, p, t) \geq L(n + 1, k, p + 1, t)$
$L(n, k, p, t) \leq L(n, k + 1, p + 1, t)$

Li described another construction technique namely "Semi-Direct Product" construction which constructs lotto designs based on smaller lotto designs. This construction is not described as we do not use it in this paper.

Li uses an exhaustive backtracking algorithm to calculate $L(n, k, p, t)$. In his backtracking algorithm, a k-set is added to the design recursively which represents the first unrepresented p-set. This backtracking algorithm can stop if a lotto design with b blocks is found or it can continue until a lotto design is found within the expected period of time. Li [14] also uses greedy algorithms, simulated annealing and heuristic techniques (hill-climbing) to get better speed rather than to use his exhaustive backtracking algorithm. Using greedy algorithms, all k-sets and p-sets are

ordered in any of the orderings, namely lexicographical, reverse lexicographical or co-lexicographical. Then these algorithms choose a k-set which represents the most p-sets and adds the k-set to the design. These algorithms stop until all p-sets are represented or the number of blocks reaches the maximal number. Greedy algorithms can generate better upper bounds by applying this logic recursively. However, these algorithms do not always guarantee a minimal design.

One of the major drawbacks of exhaustive search is that the search tree grows exponentially as the value of parameter increases. Li [16] uses some heuristics to reduce the size of the exhaustive search tree. In a (n, k, p, t) lotto design with b blocks, the total number of spots is kb. Thus the average occurrence of each element is kb/n. If $kb/n = nx + r$, where x is the quotient and r is the remainder, $n - r$ elements have the frequency x and r elements have the frequency $x + 1$. By applying this heuristic, the size of the search tree, the memory requirements and the execution time reduce to a great extent.

In comparison to Bate's algorithm which starts with the maximum number of blocks $b = C(n, k)$ in a design and then tries to decrease the number of blocks, Li uses a smaller number b as the starting number of blocks in the designs and then finds whether a design exists with b blocks and tries to find a design with b - 1 blocks and so on. This optimization also reduces the size of the search tree and thus the running time.

2.5 Work of Bate and van Rees

Bate and van Rees [2] introduce optimal "nice" $L(n, k, p, t)$ designs wherein each element occurs at least once and the elements of frequency one occur only with other elements of frequency one. An *independent set* is a set of elements no pair of ele-

n	$L(n,6,6,2)$	n	$L(n,6,6,2)$
35	9	45	15
36	9	46	16
37	9	47	17
38	11	48	18
39	11	49	19
40	12	50	19
41	13	51	20
42	13	52	21
43	14	53	22
44	15	54	23

Table 2.1: $L(n,6,6,2)$ for $n \leq 54$ using "nice" design

ments of which occurs in any block of the design. The elements of that set are called independent elements and the blocks containing independent elements are called independent blocks. Bate and Van Rees show that in any $LD^*(n,k,p,t;b)$ with an independent set of size p - 1, every element must occur in the independent blocks. They also show that a nice $LD^*(n,k,p,t;b)$ exists if $L(n,k,p,t) = b$, and $b \geq \frac{n}{k}$ and $n \geq k(p-1)$. Using nice designs, they calculate $L(n,6,6,2)$ for $n \leq 54$ and got the results listed in the Table 2.1.

2.6 Results of Li and van Rees

In this section, we will present some of the frequency results of Li and van Rees. They state the following theorems to determine whether a lotto design has elements of a certain frequency.

Theorem 2.6.1 *: If there exists a lotto design $LD(n,k,p,t;b)$ with an element of frequency x then there exists an $LD(n-x(k-1)-1,k,p-1,t;b-x)$ lotto design.*

Proof:

Suppose, an element a_1 of $LD(n, k, p, t; b)$ has frequency x. Delete all k-sets from the design which have the element a_1 and in the remaining k-sets, replace all the elements that were in the deleted k-sets, with arbitrary elements. After the deletion and replacement of elements, we have $b - x$ blocks and $n_1 = n - kx + x - 1$ elements remaining. Let R be any p - 1 set chosen from the set of n_1 elements. Then, $R \cup \{a_1\}$ is a p-set represented by the original design $LD(n, k, p, t; b)$. As the elements of R and a_1 never occur together in the any blocks of the original design, $R \cup \{a_1\}$ must be represented by one of the $b - x$ blocks. Then R must intersect at least one of the $b - x$ blocks in t elements. Thus, the truncated design forms an $LD(n - x(k - 1) + 1, k, p - 1, t; b - x)$ lotto design.□

Since Theorem 2.6.1 is usually used with $x = 1$, we state it with $x = 1$ as the following Corollary 2.6.2.

Corollary 2.6.2 *: If there exists a lotto design $LD(n, k, p, t; b)$ with an element of frequency 1 then there exists an $LD(n - k + t - 2, k, p - 1, t; b - 1)$ lotto design.*

Corollary 2.6.3 *: If there exists a lotto design $LD(n, k, p, t; b)$ with x elements of frequency 1 in the same block where $n \geq 2k - t + x + 1$ then there exists an $LD(n - k + t - x - 1, k, p - x, t; b - 1)$ lotto design.*

Li and van Rees [16] also prove the following 2 lemmas.

Lemma 2.6.4 *: If an $(n, k, p, t; b)$ lotto design contains y elements such that each block contains exactly d ($d \leq t$) of the y elements then an $(n - y, k - d, p - y, t - d; b)$ lotto design exists.*

Lemma 2.6.5 *: If an $(n, k, p, t; b)$ lotto design contains y elements such that each*

(n, k, p, t)	Previous $L(n, k, p, t)$	New $L(n, k, p, t)$
(14, 4, 7, 3)	6:8	7:8
(17, 5, 7, 3)	6:7	7
(16, 4, 8, 3)	7:8	8
(19, 5, 8, 3)	5:7	7
(20, 5, 8, 3)	7:9	8:9
(17, 4, 9, 3)	6:7	7
(18, 4, 8, 3)	7:9	8:9
(19, 4, 9, 3)	8:11	9:11
(19, 4, 10, 3)	6:7	7
(20, 4, 10, 3)	7:9	9

Table 2.2: New $L(n, k, p, t)$s found by Li and van Rees

block contains at most one of the y elements then an $(n - y, k, p - y, t - 1; b)$ *lotto design exists.*

Definition 2.6.1 *: An element x is covered by another element y if x always occurs with y in the blocks of the design.*

Theorem 2.6.6 *: If a lotto design* $LD(n, k, p, t; b)$ *exists with x elements of frequency f and if an element of frequency f is covered by an element of frequency not less than f + 2 then a lotto design* $LD(n, k, p, t; b)$ *with x - 1 elements of frequency f exists.*

From the Theorem 2.6.6, Li and van Rees got the new results listed in Table 2.2:

Now, we will present some proofs of their results. The proof for (17, 5, 7, 3), (16, 4, 8, 3), (17, 4, 9, 3), (19, 4, 10, 3) and (20, 4, 10, 3) are similar. We will present the proof of only (17, 4, 9, 3) of this category.

Theorem 2.6.7 $L(17, 4, 9, 3) = 7$

Proof:

Assume there exists an $LD(17, 4, 9, 3; 6)$. Since, $9 \leq (3 - 1 - 4)\lfloor \frac{17-1}{4} \rfloor + 17 = 9$, every element has frequency at least one by Lemma 2.2.9. Thus, no element of frequency zero exists in $L(17, 4, 9, 3)$. Now, assume that there are at least two elements of frequency one in a block of an $LD(17, 4, 9, 3)$. Then, by Corollary 2.6.3, there should exist an $LD(13, 4, 7, 3; 5)$. But, Li's tables have that $L(13, 4, 7, 3) = 6$. This is a contradiction.

So, there are at most 6 elements of frequency 1. But there are at least 2(17) - 6(4) = 10 elements of frequency 1 in an $LD(17, 4, 9, 3; 6)$ when there are no elements of frequency 0. This is a contradiction. Therefore, $L(17, 4, 9, 3) = 7$.□

The proofs for (14, 4, 7, 3), (18, 4, 8, 3), (20, 5, 8, 3) and (19, 4, 9, 3) are also similar. We will present the proof of only (14, 4, 7, 3) of this category.

Theorem 2.6.8 $7 \leq L(14, 4, 7, 3) \leq 8$

Proof:

Assume there exists an $LD(14, 4, 7, 3; 6)$. Since, $7 \leq (3 - 1 - 4)\lfloor \frac{14-1}{4} \rfloor + 14 = 8$, every element has the frequency at least one by Lemma 2.2.9. Thus, no element of frequency zero exists in $L(14, 4, 7, 3)$. Now, we assume that at least two elements of frequency one exists in a block of $LD(14, 4, 7, 3)$. Thus, according to the Corollary 2.6.3, there should exist a $LD(14 - 4 + 3 - 3, 4, 7 - 2, 3) = LD(10, 4, 5, 3)$ lotto design with $L(10, 4, 5, 3) \leq 5$. From Li's table we get $L(10, 4, 5, 3) = 7$. This is a contradiction. On the other hand, if there exists an $LD(14, 4, 7, 3; 6)$, there should be at least 14(2) - 6(4) = 4 elements of frequency 1.

Assume there are 5 elements of frequency 1. Thus, in Lemma 2.6.5 we can put $y = 5$ and we will get $L(9, 4, 2, 2) \leq 6$. But, from Li's table we get $L(9, 4, 2, 2) = 8$. This is a contradiction. Therefore, there are exactly 4 elements of frequency 1. Let the elements of frequency 1 be 1, 2, 3 and 4 and let these elements occur in the block A_1, A_2, A_3, A_4 respectively. All other elements must have frequency 2. Suppose element 5 occurs in blocks A_5 and A_6. In this case, we can apply Lemma 2.6.4 with $y = 5$ and $d = 1$. Thus, we get $L(9, 3, 2, 2) \leq 6$. But, from Li's table we get $L(9, 3, 2, 2) = 12$. This is also a contradiction. Therefore, element 5 occurs in A_5 but not in A_6. Similarly, element 6 occurs in A_6 but not in A_5. Now, we have 4 blocks containing one of the elements 1, 2, 3, 4, 5, 6. Now, other element, say 7, repeats in the (4 - 1) 4 = 12 spaces for 8 elements. Thus, {1, 2, 3, 4, 5, 6, 7} is not represented in the lotto design. This is a contradiction. Thus, there is no (14, 4, 7, 3) with 6 blocks.

Therefore, $7 \leq L(14, 4, 7, 3) \leq 8$. □

2.7 Other Significant Work on Lotto Designs

Colbourn [7] has an extensive survey on lottery designs. He lists most lotteries (e.g. Lotto, Keno, Quick Cash, Supercash) in the real world, some of which have 'quick pick' and 'bonus number' options. He gives the formal definition of Lotto designs and Tuŕan designs with examples and theorems of [12, 9]. He also defines lottery wheels (systems for buying multiple tickets). He gives some example of lottery wheels.

Nurmela and Östergard [18] generated a program named "Cover" to find upper bounds based on simulated annealing. In simulated annealing, b random k-sets are picked. If these k-sets do not represent all p-sets, one of the k-sets is chosen randomly. Then the k-set is replaced by one of its neighboring k-set. The neighboring k-set is obtained by replacing a point of a k-set by a point which is not in the k-set or in the design already found. This replacement may reduce the number of unrepresented p-sets. The design may be obtained by applying the process a certain number of times. If the design comes up with b blocks, simulated annealing then tries to find a design with b -1 blocks and so on. Simulated annealing requires huge amount memory and time. However, it is a useful technique to compute an upper-bound for $L(n, k, p, t)$.

Bluskov et al. [3] uses techniques similar to Nurmela and Östergard. They present some constructions producing the best known upper bounds on $C_\lambda(v, k, t)$ where, $k = 6$ and the subscript λ denotes that every t-set occurs in at least λ of the blocks of the design.

2.8 Current Research

Recently, one group of South African researchers, A.P. Burger et al. [5] improved the bounds and generated new lottery designs. They found all lottery design numbers

where $L(n,k,p,t) = 1$, 2 or 3.

Theorem 2.8.1 *:*

$L(n,k,p,t) = 1$ *if and only if* $k+p \geq n+t$

$L(n,k,p,t) = 2$ *if and only if* $2t-1+max\{n-2k,0\} \leq p \leq n+t-k-1$

$L(n,k,p,t) = 3$ *if and only if* $p \leq min\{2t-2+max\{n-2k,0\}, n-k+t-1\}$

and

$$t \geq \begin{cases} 3t-2+max\{n-3k,0\} & if n \geq 2k \\ \frac{3}{2}t-1+max\{n-\frac{3}{2}k,0\} & if n < 2k \end{cases}$$

They also introduced lottery graphs.

Definition 2.8.1 *: Let,* $G(V)$ *be the vertex set of a graph* $G(V,E)$*. A subset* D *of* $G(V)$ *is called the dominating set of* $G(V,E)$ *if each vertex of* $G(V,E)$ *which is not in* D *is adjacent to a vertex in* D*. The dominating set* D *is minimal if no subset of* D *is also a dominating set. The lower bound of a lotto design is determined by the minimal dominating set.*

A lotto design $L(n,k,p,t)$ *can be defined by a bipartite graph* $G(n,k,p,t)$ *whose vertex set can be partitioned into two subsets* $V(k)$ *and* $V(p)$*.* $V(k)$ *consists of all* k*-sets and* $V(p)$ *consists of all* p*-sets. A vertex* $a \in V(k)$ *is adjacent to a vertex* $b \in V(p)$ *if* $V(k) \cap V(p) = t$*. This graph is known as a lottery graph. The minimal dominating set* D *is the subset of* $V(k)$ *which dominates every vertex of* $V(p)$

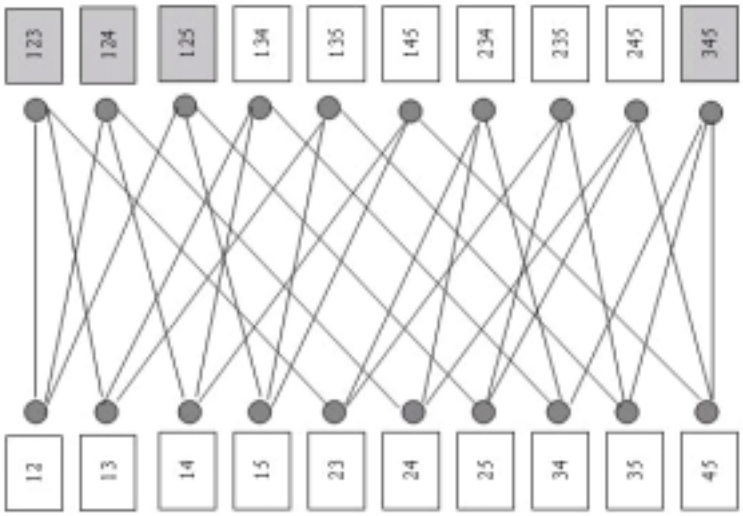

Figure 2.1: The Lottery Graph $G(5, 3, 2, 2)$. The shaded nodes are the minimal dominating set

Let us consider, the lottery scheme for $G(5, 3, 2, 2)$. The order of the lottery graph is $\binom{5}{3} + \binom{5}{2} = 10 + 10 = 20$. The nodes are partitioned into two subsets: 10 k-sets and 10 p-sets. From the figure, we find the lottery set = {{1 2 3} {1 2 4} {1 2 5} {3 4 5}} which dominates all the nodes of $V(p) = V(2)$. Thus, $L(5, 3, 2, 2) = 4$.

Properties of Lottery Graph

The *diameter* of a graph is the longest distance in the graph and the *radius* of a graph is the length of the shortest minimal path in the graph. Now, we will present some properties of lottery graphs for which $p = k$

(i) For any $1 \leq t \leq k \leq n$

The lottery graph $G\langle n, k, k, t\rangle$ is r-regular where

$$r = \sum_{i=t}^{k-1} \begin{pmatrix} k \\ i \end{pmatrix} \begin{pmatrix} n-k \\ k-i \end{pmatrix} \tag{2.8}$$

(ii) For all $1 \leq t \leq k \leq n$,

$$Radius(G\langle n,k,k,t\rangle) = diameter(G\langle n,k,k,t\rangle) = \begin{cases} \lceil k/k-t \rceil & if n \geq 2k \\ \lceil (n-k)/(k-t) \rceil & if n < 2k \end{cases} \tag{2.9}$$

(iii) For $k = t$,

$$Radius(G\langle n,k,k,t\rangle) = diameter(G\langle n,k,k,t\rangle) = \alpha \tag{2.10}$$

A.P. Burger et al. present the following formula for lower bounds and upper bounds.

Lower Bound

A general lower bound for lottery number based on the properties of lottery graph given above:

$$L(n,k,k,t) \geq \left\lceil \frac{\binom{n}{k}}{r+1} \right\rceil \qquad for \quad all \qquad 1 \leq t \leq k \leq n \tag{2.11}$$

Upper Bound

If G is a connected p-order graph the lower domination number of G is denoted by

$$\gamma(G) \leq \frac{2}{5}p \tag{2.12}$$

This number is improved for a connected p-order graph with minimal vertex degree $\delta(G)$ at least 2 by $\gamma(G) \leq \frac{3}{8}p$

Again, for maximal vertex degree $\Delta(G)$, $\gamma(G) \leq p - \Delta(G)$ and $\gamma(G) \leq \delta(G)$ where, vertex degree is the number of edges a vertex is adjacent to.

From all these equation, they deduce the following formula for upper bounds

$$L(n,k,k,t) \leq min\left\{\frac{3}{8}\binom{n}{k}, \binom{n}{k} - r\right\} \tag{2.13}$$

If diameter $(G\langle n,k,k,t\rangle) = 2$ then $L(n,k,k,t) \leq r$

Then they present the known values of lottery numbers L, and also values of the lower and upper bounds calculated from the formula given above.

A.P. Burger et al. also developed a new backtracking algorithm to find the number of non-isomorphic optimal lotto designs $LD(n,k,p,t)$. This algorithm produces all possible non-isomorphic sets of 5 blocks and then checks to see if these 5 blocks forms a lotto design. The number of non-isomorphic lottery designs or the number of overlapping structures of minimum cardinality for (n,k,p,t) lottery designs is denoted by $\eta(n,k,p,t)$. $\eta(n,k,p,t)$ is important for recursive construction of lotto designs. Furthermore, they got 19 new lottery numbers and improved bounds for 20 lottery numbers. Their results are listed in the Table 2.3.

Belic R. [19] has a large list of upper bounds for lotto designs on a website. A.P. Burger et al. [6] consider the optimality of the 302 cardinality 7 (or less) lotto designs of Belic's R. list with $n \geq 20$. They found that 192 of these designs are optimal and they improved 78 designs. They also improved 429 upper bounds of the Belic's table.

Improvements of Upper Bounds

Li and van Rees [17] improved the upper bounds of lotto designs. These are actually a generalization obtained from the data given by A.P. Burger et al.

Upper bounds can be improved by applying the following Lemma 2.8.2 and Lemma 2.8.3 [17].

(n, k, p, t)	Previous $L(n, k, p, t)$	New $L(n, k, p, t)$	$\eta(n, k, p, t)$
(17, 7, 5, 3)	5:7	6:7	-
(18, 7, 5, 3)	5:8	6:8	-
(19, 8, 5, 3)	5:6	6	≥ 1
(20, 8, 5, 3)	5:7	6:7	-
(18, 6, 6, 3)	6:7	7	≥ 1
(19, 7, 6, 3)	4:5	5	2
(20, 7, 6, 3)	4:7	6:7	-
(19, 5, 8, 3)	5:7	6:7	-
(18, 4, 10, 3)	5:6	6	≥ 1
(16, 10, 5, 4)	4:5	5	1
(17, 10, 5, 4)	5:8	6:8	-
(17, 11, 5, 4)	4:5	5	2
(18, 11, 5, 4)	4:7	6:7	-
(19, 11, 5, 4)	5:10	6:10	-
(19, 12, 5, 4)	4:5	5	2
(20, 12, 5, 4)	5:7	6:7	-
(14, 6, 7, 4)	5:8	6:8	-
(15, 7, 7, 4)	4:5	5	26
(16, 7, 7, 4)	4:7	6:7	-
(17, 7, 7, 4)	5:11	6:11	-
(17, 8, 7, 4)	4:5	5	67
(18, 8, 7, 4)	4:6	5	1
(19, 8, 7, 4)	4:9	6:9	-
(20, 8, 7, 4)	4:10	6:10	-
(19, 9, 7, 4)	4:6	5	154
(20, 9, 7, 4)	4:6	5	3
(16, 6, 8, 4)	4:7	6:7	-
(18, 7, 8, 4)	4:6	6	≥ 1
(19, 7, 8 , 4)	4:9	6:9	-
(15, 5, 9, 4)	5:8	6:8	-
(17, 6, 9, 4)	4:6	6	≥ 1
(18, 6, 9, 4)	5:6	6	≥ 1
(19, 6, 9, 4)	5:10	6:10	-
(19, 7, 9, 4)	4:5	5	20
(20, 7, 9, 4)	4:6	6	≥ 1
(16, 5, 10, 4)	5:6	6	≥ 1
(19, 6, 10, 4)	5:6	6	≥ 1
(20, 6, 10, 4)	5:8	6:8	-
(18, 5, 11, 4)	5:7	6:7	-

Table 2.3: New Results found by A.P. Burger et al.

As we will not do much work with upper bounds, we will not proof the lemma. These are only presented with examples.

Lemma 2.8.2 *: A lotto design* $LD(|A|+|B|+|C|+|D|, k, 7, 4; r+3)$ *has the following blocks:*
$A \cup B$
$A \cup C$
$B \cup C$
$A_1 \cup D$
$A_2 \cup D$
$\vdots$
$\vdots$
$A_r \cup D$
where $A_1 \cup A_2 \cup A_3 \cup \ldots \cup A_r = A$ *and* $|A_1| = |A_2| = \ldots = |A_r| = w$ *and all the blocks have the same size.*

Some results found from the Lemma 2.8.2 are as follows. The asterisk(*) represents the results already found by A.P. Burger et al.

When $|A| = |B| = |C| = 3$, $|D| = 5$, r=3 Then $LD(14, 6, 7, 4; 6)$ So, $L(14, 6, 7, 4) = 6$ (new)

When $|A| = |B| = |C| = 4$, $|D| = 5$, r=2 Then $LD(17, 8, 7, 4; 5)$ So, $L(17, 8, 7, 4) = 5^*$

When, $|A| = |B| = |C| = 4$, $|D| = 6$, r=2 Then $LD(18, 8, 7, 4; 5)$ So, $L(18, 8, 7, 4) = 5^*$

When, $|A| = |B| = |C| = 4$, $|D| = 6$, r=4 Then $LD(19, 8, 7, 4; 7)$ So, $6 \leq L(19, 8, 7, 4) \leq 7^*$ (new)

When, $|A| = |B| = |C| = 3$, $|D| = 5$, r=2 Then $LD(15, 7, 7, 4; 5)$ So, $L(15, 7, 7, 4) = 5^*$

When, $|A| = |B| = |C| = 3$, $|D| = 6$, r=3 Then $LD(16, 7, 7, 4; 6)$ So,$L(16, 7, 7, 4) = 6^*$(new)

When, $|A| = |B| = |C| = 4$, $|D| = 6$, $r = 2$ Then $LD(19, 9, 7, 4; 5)$ So, $L(19, 9, 7, 4) = 5^*$

When, $|A| = |B| = |C| = 4$, $|D| = 7$, r=2 Then $LD(20, 7, 7, 4; 5)$ So, $L(20, 7, 7, 4) = 5^*$

Lemma 2.8.3 *:* $LD(|A| + |B| + |C| + |D| + |E|, k, 5, 4; 5)$ *has the following block:*
$A \cup B$
$A \cup C \cup D$
$A \cup C \cup E$
$A \cup D \cup E$
$B \cup C \cup D \cup E$
where all blocks have the same size

Some results found from the Lemma 2.8.3:

When, $|A| = 6, |B| = 4, |C| = |D| = |E| = 2$ Then $LD(16, 10, 5, 4; 5)$ So, $L(16, 10, 5, 4) = 5$

When, $|A| = 7, |B| = 4, |C| = |D| = |E| = 2$ Then $LD(17, 11, 5, 4; 5)$ So, $L(17, 11, 5, 4) = 5$

When, $|A| = 7, |B| = 5, |C| = 3, |D| = |E| = 2$ Then $LD(19, 12, 5, 4; 5)$ So, $L(19, 12, 5, 4) = 5$

Chapter 3

Algorithms

3.1 Introduction

The main task of our thesis is to develop backtracking algorithms that will find the number of non-isomorphic optimal lotto designs for $L(n, k, p, t)$ on 5 or 6 blocks or if this is impossible, try to improve lower bounds for $n, k, p, t \leq 20$ by a backtracking algorithm with isomorphism rejection. In this chapter, we will first present the backtracking algorithm of Bate along with his optimization techniques; preclusion and isomorphism rejection. Then, we will present our basic sequential algorithm based on the idea of Bate's algorithm. Then, we will present an improved backtracking algorithm which will optimize the basic sequential algorithm by pruning the search tree to a great extent. We will give an example of the implementations and briefly describe the data structures.

3.2 Basic Backtracking Algorithm of Bate

Bate uses a backtracking algorithm along with some optimizations to compute $L(n, k, p, t)$. In his algorithm, Bate sets the maximum allowable number of k-sets in the design to a large number. The algorithm begins by computing all p-sets. Then the algorithm finds the first unrepresented p-set and generates all k-sets that represent the p-set. These k-sets are a choice set for the first unrepresented p-set on the first level. The algorithm adds to the design a k-set from the choice set and keeps track (marks) of the index of all p-sets represented by the added k-set using flags. The algorithm then finds the next unrepresented p-set and adds a k-set which represents the p-set. If the algorithm reaches the maximum number of k-sets, b, on the level $X + 1$, then the algorithm backtracks to the previous level X and takes other possible k-sets from the choice set of the level X and proceeds in the same fashion.

Before backtracking to the previous level X, this algorithm resets all the flags of the p-sets represented by the k-set on the level $X + 1$. If the algorithm finds a design with b blocks, the algorithm stores the results and tries to find a design with b - 1 blocks using the same procedure. This process is repeated until all possibilities have been checked.

Now, we will present the pseudo-code of Bate's basic backtracking algorithm.

Pseudo-code of Bate's Algorithm

Find the first unrepresented p-set in the design.

if No such p-set is found **then**

  A design is found

else

  if Design contains the maximum number of k-sets **then**

Backtrack to the previous level

else

Adds a k-set to the design and sets the flags of the p-sets represented by the added k-set

Apply this procedure to complete the design

if A design is found on a level **then**

Resets the flags of the p-sets represented by the k-set at this level and removes the k-set from the design

Return to the previous level

end if

end if

end if

3.2.1 Preclusion

Bate uses an optimization, preclusion, to speed up the basic backtracking algorithm. Using Preclusion, the number of search nodes is reduced to a great extent.

In preclusion, the maximum number of p-sets, C, represented by any k-set of the design is calculated using the Equation 3.1:

$$C = \sum_{i=t}^{k} \begin{pmatrix} k \\ i \end{pmatrix} \begin{pmatrix} n-k \\ p-i \end{pmatrix} \tag{3.1}$$

A depth limit "LIMIT" is also defined at the beginning of the program. "LIMIT" is the maximum number of blocks a lotto design may have and initially, it is set to a large number. "MAXPOS" is the maximum numbers of p-sets that can be represented by any set of $(LIMIT - d)$ blocks where, d is the number of blocks in partial lotto

design. Then "MAXPOS" is calculated using the Equation 3.2:

$$MAXPOS = C * (LIMIT - d) \tag{3.2}$$

Bate uses "NLEFT" to count the p-sets which have not been represented yet. If $MAXPOS < NLEFT$, the program backtracks to the previous level and takes other k-sets from the choice set. This means that the maximum number of p-sets that can be represented by the additional blocks the design may have, is less than the number of p-sets left in the design. In this case, the program backtracks to the previous level.

Preclusion works well at the lower level of the search tree. However, we did not use preclusion in our algorithm because preclusion does not prune the search tree much in our instances of the problem and preclusion increases computation time.

3.2.2 Isomorphism Rejection

Isomorphic rejection techniques significantly speed up backtracking algorithms. Using these techniques, a k-set A is rejected on a level X of the search tree if the inclusion of A does not lead to a successful design. Any other k-set B is also rejected if B generates an isomorphic partial design to the partial design generated by A on level X. Suppose, A is a block rejected at the $(i+1)$th level i.e. $(X_1, X_2, X_3, \ldots, X_i, A)$ is the rejected partial design. Thus, a set of blocks $(B_1, B_2, B_3, \ldots, B_r)$ are also rejected at the level $i + 1$ if the partial designs $(X_1, X_2, X_3, \ldots, X_i, B_1), (X_1, X_2, X_3, \ldots, X_i, B_2)$ $, \ldots, (X_1, X_2, X_3, \ldots, X_i, B_r)$ are isomorphic to $(X_1, X_2, X_3, \ldots, X_i, A)$. This type of isomorphic rejection is called complete isomorphic rejection. Complete isomorphic rejection reduces the size of the search tree to a great extent. However, this kind of complete isomorphism rejection is expensive in the context of search time. Thus,

Bate introduces a simplified form of isomorphism testing [1].

The simplified form of isomorphism testing is called *trivial isomorphism* testing. In this test, a table is used to maintain the list of trivially isomorphic elements. Recall the definitions of Definition 1.1.7 - 1.1.10.

Rejection flags are used for blocks to implement the trivially isomorphic rejection scheme. When a block is added to the design, the table of trivially isomorphic elements is updated and all trivially isomorphic blocks of the added block are generated and the rejection flags of these blocks are set. A block cannot be added to the design if its rejection flag is set. After trying all trivially non-isomorphic blocks at a level, the program backtracks to the previous level and all rejection flags are reset.

This simple form of isomorphism rejection techniques works best at levels with few blocks or at the top levels of the tree, thus reducing the size of the search tree and also the search time to a great extent.

Of course, using the trivial isomorphism rejection test does not necessarily eliminate all non-isomorphic copies but they do not affect the value of $L(n, k, p, t)$.

3.3 Basic Sequential Algorithm

As backtracking on the first three blocks requires much search time, we will generate all non-isomorphic set of three blocks for a set of parameters using block intersection properties. This technique will be explained with an example in a later section. Then, for each set of the three blocks, we will take the next k-set in lexicographical order. In this way, we will try to find a design by checking all possibilities until the design reaches the maximal k-sets. Now, we will present our basic sequential algorithm which we call the '3-block sets Algorithm'. This algorithm finds the number of non-isomorphic designs of $L(n, k, p, t)$ on 5 or 6 blocks or, with a slight change, find if

$L(n,k,p,t) = 6$ for n, k, p, t less than or equal to 20. Any 3 blocks, B_1, B_2, B_3 can be arranged so that $|B_1 \cap B_2| \geq |B_1 \cap B_3| \geq |B_2 \cap B_3|$. So, our algorithm finds the blocks this way. In the 3-blocks set and improved 3-blocks set algorithms, the variables $pSet$ stores all the p-sets, $pSetFlag$ stores the flags of the p-sets, $nTotalPset$ is the total number of p-sets, $nTotalThreeBlocks$ is the total number of pre-computed sets of 3-blocks. We will store the p-sets that are unrepresented by the first 3 blocks into the variable $newPset$ and the flags of these p-sets are stored into the variable $newPsetFlag$. $nextKset$ is a k-set in lexicographical order whereas, $nonIsoKset$ are the trivially non-isomorphic k-sets generated at the level 4, 5 and 6.

3.3.1 3-Blocks set Algorithm

Pre-compute all non-isomorphic set of 3-blocks for a set of parameters n, k, p, t

Generate all p-sets

for $i = 0$ to $nTotalPset - 1$ **do**

$pSetFlag[i] \leftarrow 0$

{initializing the flags of p-sets}

end for

for $i = 0$ to $nTotalThreeBlocks - 1$ **do**

for $j = 0$ to $nTotalPset - 1$ **do**

if $((|pSet[j] \cap firstBlock|) or (|pSet[j] \cap secondBlock|) or (|pSet[j] \cap thirdBlock|) = t)$ **then**

set $pSetFlag[j] \leftarrow 1$ {represents p-sets by the first three blocks }

end if

end for

$l = 0$

for $j = 0$ to $nTotalPset - 1$ **do**

if $pSetFlag[j] = 0$ **then**

$newPset[l] \leftarrow pSet[j]$

$newPsetFlag[l] \leftarrow 0$ {generate new list of p-sets still unrepresented by the first 3-blocks}

$l = l + 1$

end if

end for

for The level $L = 4$ to 6 **do**

$partialDesign[L-1] \leftarrow nextKset$ where, $nextKset$ is a k-set in lexicographical order

for Each of the index i_1 of $newPset$ represented by $nextKset$ **do**

$newPsetFlag[i_1] \leftarrow 1$

end for

if All p-sets are represented and number of blocks of the design ≤ 6 **then**

A lotto design is found, display the design

Reset $newPsetFlag$ and take the next k-set at level L in lexicographical order.

else if $L = 6$ and all k-sets have been tried at the level L without all p-sets being represented **then**

Backtrack to the Level $L - 1$

Reset $newPsetFlag[i] \leftarrow 0$, where i is the index of each p-set represented by the k-set at level L

end if

end for

end for

3.4 Improved Algorithm

Taking all the possible k-sets in lexicographical order at the level $4 \leq L \leq 6$ is slow, because, it requires a large time to search the state space tree. Thus, to speed up our basic sequential algorithm, we will generate all non-trivially isomorphic k-sets at the level $4 \leq L \leq 6$. We will further optimize our program by applying the fact that there exists an optimal lotto design wherein each element occurs at least once. Firstly, we will present our improved algorithm which we call 'Improved 3-blocks set Algorithm'. Then we will present the algorithm to pre-compute all non-isomorphic set of three blocks for a set of parameters n, k, p, t. Finally, we will describe improvements in a subsection.

3.4.1 Improved 3-blocks set Algorithm

Pre-compute all non-isomorphic set of 3-blocks for a set of parameters n, k, p, t

Generate all p-sets

for $i = 0$ to $nTotalPset - 1$ **do**

  $pSetFlag[i] \leftarrow 0$ {initializing the flags of p-sets}

end for

for $i = 0$ to $nTotalThreeBlocks - 1$ **do**

  for $j = 0$ to $nTotalPset - 1$ **do**

    if $((|pSet[j] \cap firstBlock|) or (|pSet[j] \cap secondBlock|) or (|pSet[j] \cap thirdBlock|) = t)$ **then**

      set $pSetFlag[j] \leftarrow 1$ {represents p-sets by the first three blocks }

    end if

  end for

  $l = 0$

for $j = 0$ to $nTotalPset - 1$ **do**

if $pSetFlag[j] = 0$ **then**

$newPset[l] \leftarrow pSet[j]$

$newPsetFlag[l] \leftarrow 0$ {generate new list of p-sets still unrepresented by the first 3-blocks}

$l = l + 1$

end if

end for

for The level $L = 4$ to 6 **do**

Generate trivially isomorphic list of elements based on the first L - 1 blocks

Generate all trivially non-isomorphic k-sets, $nonIsoKset$ at the level L

$j = 0$

while $j \leq nTotalNonIsoKset - 1$ **do**

$partialDesign[L-1] \leftarrow nonIsoKset[j]$

$j = j + 1$

if $L = 6$ and still frequency 0 elements left **then**

Take the next $nonIsoKset$ in $partialDesign$

else

for Each of the $newPset$ of index l where, $newPset[l] \cap nonIsoKset[j] = t$ **do**

$newPsetFlag[l] \leftarrow 1$

end for

if All $newPsetFlag = 1$ and $nTotalBlocks \leq 6$ **then**

A lotto design is found, display the design {$nTotalBlocks$ is the number of blocks added in the design }

Reset $newPsetFlags$ for the k-set at the level L and

Try to find another non-isomorphic lotto design with other $nonIsoKset$ at the level L

else if $L = 6$ **then**

Reset $newPsetFlag$ for the k-set at level L and backtrack

end if

end if

end while

end for

end for

3.4.2 Pseudo-code to pre-compute non-isomorphic set of Three Blocks

for $i = 0$ to $k - 1$ **do**

$firstBlock[i] \leftarrow i + 1$ {$firstBlock$ stores the first block}

end for

for $i = 0$ to $k - 1$ **do**

$firstUndifferentiatedList1[i] \leftarrow firstBlock[i]$ {Stores two lists of undifferentiated elements based on the first block into $firstUndifferentiatedList1$ and $secondUndifferentiatedList2$}

end for

for $i = k$ to $n - 1$ **do**

$firstUndifferentiatedList2[i] \leftarrow i + 1$

end for

for $i = 0$ to $k - 1$ **do**

$secondBlock[i] \leftarrow$ {first $(k-i-1)$ elements of $firstUndifferentiatedList1$}∪

{first $(i+1)$ elements of $firstUndifferentiatedList2$}

{Generates all set of 2-blocks}

end for

for $i = 0$ to $nTotalSecondBlock - 1$ **do**

$intersec1 \leftarrow |firstBlock \cap secondBlock[i]|$

{$intersec1$ is the intersection between the first and second block}

{Generates all lists of undifferentiated elements based on the first 2-blocks into $secondUndifferentiatedList1$, $secondUndifferentiatedList2$, $secondUndifferentiatedList3$ and $secondUndifferentiatedList4$}

$secondUndifferentiatedList1[i] \leftarrow a$,where $a \in firstBlock$ and $secondBlock$

$secondUndifferentiatedList2[i] \leftarrow b$,where $b \in firstBlock$, $b \notin secondBlock$

$secondUndifferentiatedList3[i] \leftarrow c$,where $c \notin firstBlock$, $c \in secondBlock$

$secondUndifferentiatedList4[i] \leftarrow d$,where $d \notin firstBlock$, $d \notin secondBlock$

for $index1 = TotalElementsInList1 - 1$ to 0 **do**

$totalElement = 0$ {$TotalElementsInList1$ is total number of elements in $undifferentiatedList1$}

for $j = 0$ to $index1 - 1$ **do**

$thirdBlock[j] \leftarrow secondUndifferentiatedList1[j]$

$totalElement = totalElement + 1$

end for

$remainingElement = k - totalElement$ {$totalElement$ is the total number of elements added in the thirdBlock and $remainingElement$ is additional elements can be added in the thirdBlock}

if $totalElement < k$ **then**

for $index2 = TotalElementsInList2 - 1$ to 0 **do**

if $index2 < remainingElement$ **then**

$nElement1 \leftarrow index2$ {$TotalElements2$ is total number of elements in $undifferentiatedList2$}

else

$nElement1 \leftarrow remainingElement$

end if

for $index3 = 0$ to $nElement1 - 1$ **do**

$thirdBlock[totalElement + index3] \leftarrow$

$secondUndifferentiatedList2[index3]$

$totalElement = totalElement + 1$

end for

$remainingElement = k - totalElement$

if $totalElement < k$ **then**

for $index4 = TotalElementsInList3 - 1$ to 0 **do**

if $index4 < remainingElement$ **then**

$nElement2 \leftarrow index4$

else

$nElement2 \leftarrow remainingElement$

end if

for $index5 = 0$ to $nElement2 - 1$ **do**

$thirdBlock[totalElement + index5] \leftarrow$

$secondUndifferentiatedList3[index5]$

$totalElement = totalElement + 1$

end for

$remainingElement = k - totalElement$

if $totalElement < k$ **then**

for $index6 = 0$ to $remainingElement - 1$ **do**

$thirdBlock[totalElement + index6] \leftarrow$

$secondUndifferentiatedList4[index6]$

end for

else if $totalElement = k$ **then**

$intersec2 \leftarrow |firstBlock \cap thirdBlock|$

$intersec3 \leftarrow |secondBlock[i] \cap thirdBlock|$ {$intersec2$ is the intersection between the first and third blocks and $intersec3$ is the intersection between the second and third block}

if $intersec1 \geq intersec2 \geq intersec3$ **then**

store the $thirdBlock$

end if

end if

end for

else if $totalElement = k$ **then**

$intersec2 \leftarrow |firstBlock \cap thirdBlock|$

$intersec3 \leftarrow |secondBlock[i] \cap thirdBlock|$

if $intersec2 \geq intersec2 \geq intersec3$ **then**

store the $thirdBlock$

end if

end if

end for

else if $totalElement = k$ **then**

$intersec2 \leftarrow |firstBlock \cap thirdBlock|$

$intersec3 \leftarrow |secondBlock[i] \cap thirdBlock|$

if $intersec2 \geq intersec2 \geq intersec3$ **then**

Stores the *thirdBlock*

end if

end if

end for

end for

3.4.3 Improvements

Firstly, we will generate all possible sets of three blocks by using block intersection properties. It is easy to use these properties to generate non-isomorphic sets of three blocks. Backtracking on the first three blocks is very slow. But by generating only non-isomorphic sets of three blocks, it is possible to avoid backtracking at these upper levels. Thus, this technique will speed up our program to a great extent. Secondly, we will apply trivial isomorphism rejection to generate the rest of the blocks. This technique is not as costly as the complete isomorphism rejection but does almost as well in pruning the search tree for a little extra time. In the case where we are not finding $\eta(n, k, p, t)$ but only finding if $L(n, k, p, t) = 6$, we can further optimize our program by checking whether any frequency 0 element is left at the last level. If so, the algorithm backtracks immediately to the previous level. These frequency results will prune the search tree to a great extent. We will explain the frequency results with an example in the next section. The algorithm may still generate isomorphic designs for a particular set of parameters n, k, p, t, so we will run Kocay's [13] program to eliminate these and produce a list of all non-isomorphic designs.

Why are we not generating all non-isomorphic sets of 4-blocks? Actually, generating non-isomorphic set of 4-blocks increases the computations greatly and produces a much larger list of starting designs which would ultimately decrease the performance.

3.5 Example

We will show how to generate all non-isomorphic sets of 3-blocks using block intersection properties for (18, 8, 7, 4). We will also show how to generate all trivially non-isomorphic blocks at the level $4 \leq L \leq 6$.

3.5.1 Generation of non-isomorphic sets of 3 blocks using block intersections

Let us consider the lotto design (18, 8, 7, 4). Clearly we can let {1 2 3 4 5 6 7 8} be the starting or the first block of the design because at this stage, all blocks are isomorphic. To generate all non-isomorphic sets of 3-blocks, we first generate all non-isomorphic sets of 2-blocks using block intersection properties.

Definition 3.5.1 *In simple isomorphism, the undifferentiated elements are the elements that appear in the same set of blocks. In isomorphism, the undifferentiated elements are the elements for which there exists an automorphism such that one undifferentiated element is mapped onto another undifferentiated elements in the list.*

The lists of undifferentiated elements after the first block {1 2 3 4 5 6 7 8} has been chosen are as follows:

{1 2 3 4 5 6 7 8} are undifferentiated
{9 10 11 12 13 14 15 16 17 18} are undifferentiated

Now, all possible non-isomorphic sets of 2-blocks are:

{1 2 3 4 5 6 7 8} {1 2 3 4 5 6 7 9}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 6 9 10}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11}
#Intersection = 7	#Intersection = 6	#Intersection = 5
{1 2 3 4 5 6 7 8} {1 2 3 4 9 10 11 12}	{1 2 3 4 5 6 7 8} {1 2 3 9 10 11 12 13}	{1 2 3 4 5 6 7 8} {1 2 9 10 11 12 13 14}
#Intersection = 4	#Intersection = 3	#Intersection = 2
{1 2 3 4 5 6 7 8} {1 9 10 11 12 13 14 15}	{1 2 3 4 5 6 7 8} {9 10 11 12 13 14 15 16}	
#Intersection = 1	#Intersection = 0	

Table 3.1: All non-isomorphic 2-blocks for $LD(18, 8, 7, 4; 5)$

The second block of each of the 2-blocks set is generated by taking different number of elements from the two lists of undifferentiated elements. For example, in the first 2-blocks set of the Table 3.1, 7 elements and 1 element come from the first and second lists of undifferentiated elements respectively, whereas in the second 2-blocks set, 6 and 2 elements come from the first and second list of undifferentiated elements. Hence, the number of intersections between the first and second blocks for each set of 2-blocks is different. Therefore, the 2-blocks sets are non-isomorphic.

Theorem 3.5.1 *: Any set of 2-blocks is isomorphic to one of the listed starting 2-blocks.*

Proof:

Let, $\beta = \begin{pmatrix} \{\alpha_1, \alpha_2, \ldots, \alpha_s, \alpha_{s+1}, \ldots, \alpha_k\} \\ \{\alpha_1, \alpha_2, \ldots, \alpha_s, \alpha_{k+1}, \ldots, \alpha_{2k-s}\} \end{pmatrix}$ be any 2-blocks with intersection s.
Let ϕ be the mapping function such that $\alpha_i \rightarrow i$. Then ϕ maps β to the 2-set starting blocks with intersection s. So we can state the following lemma.

Lemma 3.5.2 *: The algorithm '3-blocks set algorithm' generates all non-isomorphic sets of 2 blocks.*

Now, for each set of 2-blocks, we will generate all non-isomorphic set of 3-blocks using block intersection properties.

Now consider, the 2-block:

{1 2 3 4 5 6 7 8}
{1 2 3 4 5 9 10 11}

For this set of 2-blocks, the undifferentiated elements are:

{1 2 3 4 5} are undifferentiated
{6 7 8}, {9 10 11} are undifferentiated
{12 13 14 15 16 17 18} are undifferentiated

Note that {6 7 8} can be interchanged with {9,10,11}. Now, all non-isomorphic 3-blocks coming from the above 2-block are as follows. The pair (a_1, a_2) under each of the 3-block of the table represents the number of intersections between the first and third blocks with a_1 and denotes the number of intersections between the second and third blocks with a_2.

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 4 5 12 13 14}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 4 6 9 12 13}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 6 7 9 10 12}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 7 8 9 10 11}
(5,5)	(5,5)	(5,5)	(5,5)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 4 6 12 13 14}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 6 7 9 12 13}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 7 8 9 10 12}
(5,4)	(5,4)	(5,4)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 6 7 12 13 14}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 7 8 9 12 13}
(5,3)	(5,3)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 7 8 12 13 14}
(5,2)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 4 12 13 14 15}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 6 9 12 13 14}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 7 9 10 12 13}
(4,4)	(4,4)	(4,4)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 7 8 9 10 11 12}
(4,4)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 6 12 13 14 15}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 7 9 12 13 14}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 7 8 9 10 12 13}
(4,3)	(4,3)	(4,3)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 7 12 13 14 15}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 7 8 9 12 13 14}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 7 8 12 13 14 15}
(4,2)	(4,2)	(4,1)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 3 12 13 14 15 16}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 9 12 13 14 15}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 7 9 10 12 13 14}
(3,3)	(3,3)	(3,3)
{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 7 8 9 10 11 12 13}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 6 12 13 14 15 16}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 7 12 13 14 15 16}
(3,3)	(3,2)	(3,2)
{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 7 8 9 10 12 13 14}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 7 11 12 13 14 15}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 7 8 9 12 13 14 15}
(3,2)	(3,1)	(3,1)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 7 8 12 13 14 15 16}
(3,0)

{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 2 12 13 14 15 16 17}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 9 12 13 14 15 16}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 7 9 10 12 13 14 15}
(2,2)	(2,2)	(2,2)
{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 6 12 13 14 15 16 17}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 7 9 12 13 14 15 16}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 7 12 13 14 15 16 17}
(2,1)	(2,1)	(2,0)
{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {1 12 13 14 15 16 17 18}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 9 12 13 14 15 16 17}	{1 2 3 4 5 6 7 8} {1 2 3 4 5 9 10 11} {6 12 13 14 15 16 17 18}
(1,1)	(1,1)	(1,0)

Table 3.2: All non-isomorphic 3-blocks for the 2-block {{1 2 3 4 5 6 7 8}{1 2 3 4 5 9 10 11}}

Though the number of intersections (a_1, a_2) of two or more sets of 3-blocks is the same they can be non-isomorphic if different number of elements comes from the lists of undifferentiated elements in the third block of the 3-blocks sets. For example, the number of intersections of the first four sets of 3-blocks of the Table 3.2 is (5, 5); they are non-isomorphic. As in the third block of the first 3-blocks set, 5 elements come from the undifferentiated elements list {1 2 3 4 5} whereas in the third block of the second 3-blocks set, 4 elements come from the undifferentiated elements list {1 2 3 4 5}.

Theorem 3.5.3 *Any lotto design must have 3 blocks isomorphic to the one of the starting 3 blocks.*

Proof:

Without loss of generality, let the largest intersection is between the first and second blocks. Also without loss of generality, let the size of the intersection between block 1 and block 3 be equal to or larger than the size of the intersection between block 2 and block 3. Then the blocks can be written as follows:

$$\beta = \begin{pmatrix} \{\alpha_1, \alpha_2, \ldots, \alpha_s, \alpha_{s+1}, \ldots, \alpha_k\} \\ \{\alpha_1, \alpha_2, \ldots, \alpha_s, \alpha_{k+1}, \ldots, \alpha_{2k-s}\} \\ \{\alpha_1, \alpha_2, \ldots, \alpha_r, \alpha_{s+1}, \ldots, \alpha_{s+v}, \alpha_{k+1}, \ldots, \alpha_{k+w}, \alpha_{2k-s+1}, \ldots, \alpha_{3k-s-r-v-w}\} \end{pmatrix}$$

Where, $0 \leq s \leq k-1$, $0 \leq r \leq s$, $0 \leq v \leq k-s$, $0 \leq w \leq k-s$. Now, $\alpha_i \rightarrow i$ maps β onto the (v, w) 3-block where $|B_1 \cap B_2 \cap B_3| = r$, $|B_1 \cap B_2| = s$, $|B_1 \cap B_3 - B_2| = v$, $|B_2 \cap B_3 - B_1| = w$.□

3.5.2 Find the fourth block using trivial isomorphism rejection

For each set of the non-isomorphic 3-blocks, our program will find the fourth block using trivial isomorphism rejection.

Let us consider the following set of 3-blocks.

{1 2 3 4 5 6 7 8}
{1 2 3 4 5 9 10 11}
{1 2 3 4 5 12 13 14}

The lists of trivially isomorphic elements for this set of 3-blocks are as follows:

{1 2 3 4 5} are trivially isomorphic
{6 7 8} are trivially isomorphic
{9 10 11} are trivially isomorphic
{12 13 14} are trivially isomorphic
{15 16 17 18} are trivially isomorphic

Although {6 7 8} and {9 10 11} are indistinguishable, they are trivially non-isomorphic.

Now, we will generate all trivially non-isomorphic sets of fourth blocks for this set of 3-blocks and try to find partial lotto designs with four blocks. The following Table 3.3 lists all trivially non-isomorphic fourth blocks for this set of 3-blocks.

{1 2 3 4 5 15 16 17}	{1 2 3 4 6 9 12 15}	{1 2 3 4 6 9 15 16}
{1 2 3 4 6 12 15 16}	{1 2 3 4 6 15 16 17}	{1 2 3 4 9 12 15 16}
{1 2 3 4 9 15 16 17}	{1 2 3 4 12 15 16 17}	{1 2 3 4 15 16 17 18}
{1 2 3 6 7 9 10 12}	{1 2 3 6 7 9 10 15}	{1 2 3 6 7 9 12 13}
{1 2 3 6 7 9 12 15}	{1 2 3 6 7 9 15 16}	{1 2 3 6 7 12 13 15}
{1 2 3 6 7 12 15 16}	{1 2 3 6 7 15 16 17}	{1 2 3 6 9 10 12 13}
{1 2 3 6 9 10 12 15}	{1 2 3 6 9 10 15 16}	{1 2 3 6 9 12 13 15}
{1 2 3 6 9 12 15 16}	{1 2 3 6 9 15 16 17}	{1 2 3 6 12 13 15 16}
{1 2 3 6 12 15 16 17}	{1 2 3 6 15 16 17 18}	{1 2 3 9 10 12 13 15}
{1 2 3 9 10 12 15 16}	{1 2 3 9 10 15 16 17}	{1 2 3 9 12 13 15 16}
{1 2 3 9 12 15 16 17}	{1 2 3 9 15 16 17 18}	{1 2 3 12 13 15 16 17}
{1 2 3 12 15 16 17 18}	{1 2 6 7 8 9 10 11}	{1 2 6 7 8 9 10 12}
{1 2 6 7 8 9 10 15}	{1 2 6 7 8 9 12 13}	{1 2 6 7 8 9 12 15}
{1 2 6 7 8 9 15 16}	{1 2 6 7 8 12 13 14}	{1 2 6 7 8 12 13 15}
{1 2 6 7 8 12 15 16}	{1 2 6 7 8 15 16 17}	{1 2 6 7 9 10 11 12}
{1 2 6 7 9 10 11 15}	{1 2 6 7 9 10 12 13}	{1 2 6 7 9 10 12 15}
{1 2 6 7 9 10 15 16}	{1 2 6 7 9 12 13 14}	{1 2 6 7 9 12 13 15}
{1 2 6 7 9 12 15 16}	{1 2 6 7 9 15 16 17}	{1 2 6 7 12 13 14 15}
{1 2 6 7 12 13 15 16}	{1 2 6 7 12 15 16 17}	{1 2 6 7 15 16 17 18}
{1 2 6 9 10 11 12 13}	{1 2 6 9 10 11 12 15}	{1 2 6 9 10 11 15 16}
{1 2 6 9 10 12 13 14}	{1 2 6 9 10 12 13 15}	{1 2 6 9 10 12 15 16}
{1 2 6 9 10 15 16 17}	{1 2 6 9 12 13 14 15}	{1 2 6 9 12 13 15 16}
{1 2 6 9 12 15 16 17}	{1 2 6 9 15 16 17 18}	{1 2 6 12 13 14 15 16}
{1 2 6 12 13 15 16 17}	{1 2 6 12 15 16 17 18}	{1 2 9 10 11 12 13 14}
{1 2 9 10 11 12 13 15}	{1 2 9 10 11 12 15 16}	{1 2 9 10 11 15 16 17}
{1 2 9 10 12 13 14 15}	{1 2 9 10 12 13 15 16}	{1 2 9 10 12 15 16 17}
{1 2 9 10 15 16 17 18}	{1 2 9 12 13 14 15 16}	{1 2 9 12 13 15 16 17}
{1 2 9 12 15 16 17 18}	{1 2 12 13 14 15 16 17}	{1 2 12 13 15 16 17 18}
{1 6 7 8 9 10 11 12}	{1 6 7 8 9 10 11 15}	{1 6 7 8 9 10 12 13}
{1 6 7 8 9 10 12 15}	{1 6 7 8 9 10 15 16}	{1 6 7 8 9 12 13 14}
{1 6 7 8 9 12 13 15}	{1 6 7 8 9 12 15 16}	{1 6 7 8 9 15 16 17}
{1 6 7 8 12 13 14 15}	{1 6 7 8 12 13 15 16}	{1 6 7 8 12 15 16 17}
{1 6 7 8 15 16 17 18}	{1 6 7 9 10 11 12 13}	{1 6 7 9 10 11 12 15}
{1 6 7 9 10 11 15 16}	{1 6 7 9 10 12 13 14}	{1 6 7 9 10 12 13 15}
{1 6 7 9 10 12 15 16}	{1 6 7 9 10 15 16 17}	{1 6 7 9 12 13 14 15}
{1 6 7 9 12 13 15 16}	{1 6 7 9 12 15 16 17}	{1 6 7 9 15 16 17 18}
{1 6 7 12 13 14 15 16}	{1 6 7 12 13 15 16 17}	{1 6 7 12 15 16 17 18}
{1 6 9 10 11 12 13 14}	{1 6 9 10 11 12 13 15}	{1 6 9 10 11 12 15 16}
{1 6 9 10 11 15 16 17}	{1 6 9 10 12 13 14 15}	{1 6 9 10 12 13 15 16}
{1 6 9 10 12 15 16 17}	{1 6 9 10 15 16 17 18}	{1 6 9 12 13 14 15 16}
{1 6 9 12 13 15 16 17}	{1 6 9 12 15 16 17 18}	{1 6 12 13 14 15 16 17}
{1 6 12 13 15 16 17 18}	{1 9 10 11 12 13 14 15}	{1 9 10 11 12 15 16 17}
{1 9 10 11 15 16 17 18}	{1 9 10 12 13 14 15 16}	{1 9 10 12 13 15 16 17}

{1 9 10 12 15 16 17 18}	{1 9 12 13 14 15 16 17}	{1 9 12 13 15 16 17 18}
{1 12 13 14 15 16 17 18}	{6 7 8 9 10 11 12 13}	{6 7 8 9 10 11 12 15}
{6 7 8 9 10 11 15 16}	{6 7 8 9 10 12 13 14}	{6 7 8 9 10 12 13 15}
{6 7 8 9 10 12 15 16}	{6 7 8 9 10 15 16 17}	{6 7 8 9 12 13 14 15}
{6 7 8 9 12 13 15 16}	{6 7 8 9 12 15 16 17}	{6 7 8 9 15 16 17 18}
{6 7 8 12 13 14 15 16}	{6 7 8 12 13 15 16 17}	{6 7 8 12 15 16 17 18}
{6 7 9 10 11 12 13 14}	{6 7 9 10 11 12 13 15}	{6 7 9 10 11 12 15 16}
{6 7 9 10 11 15 16 17}	{6 7 9 10 12 13 14 15}	{6 7 9 10 12 13 15 16}
{6 7 9 10 12 15 16 17}	{6 7 9 10 15 16 17 18}	{6 7 9 12 13 14 15 16}
{6 7 9 12 13 15 16 17}	{6 7 9 12 15 16 17 18}	{6 7 12 13 14 15 16 17}
{6 7 12 13 15 16 17 18}	{6 9 10 11 12 13 14 15}	{6 9 10 11 12 13 15 16}
{6 9 10 11 12 15 16 17}	{6 9 10 11 15 16 17 18}	{6 9 10 12 13 14 15 16}
{6 9 10 12 13 15 16 17}	{6 9 10 12 15 16 17 18}	{6 9 12 13 14 15 16 17}
{6 9 12 13 15 16 17 18}	{6 12 13 14 15 16 17 18}	{9 10 11 12 13 14 15 16}
{9 10 11 12 13 15 16 17}	{9 10 11 12 15 16 17 18}	{9 10 12 13 14 15 16 17}
{9 10 12 13 15 16 17 18}	{9 12 13 14 15 16 17 18}	

Table 3.3: All trivially non-isomorphic 4th blocks for the 3-block {{1 2 3 4 5 6 7 8}{1 2 3 4 5 9 10 11}{1 2 3 4 5 12 13 14}}

Suppose, we try to find a design with the fourth block {1 2 3 4 5 15 16 17}. If we do not find any design containing these 4 blocks then obviously we will not find any design with a fourth block trivially isomorphic to {1 2 3 4 5 15 16 17}. Thus, we generate the list of trivially non-isomorphic fourth blocks by rejecting any trivially isomorphic blocks to the fourth blocks in the table. For example, in the Table 3.3, all the blocks between {1 2 3 4 5 15 16 17} and {1 2 3 4 6 9 12 15} in lexicographical order are trivially isomorphic to {1 2 3 4 5 15 16 17}. Hence, we have discarded all the blocks between {1 2 3 4 5 15 16 17} and {1 2 3 4 6 9 12 15} in the list of fourth blocks.

3.5.3 Finding Blocks at the level 5

For each set of trivially non-isomorphic 4-blocks, we will find fifth and sixth block using trivial isomorphism rejection.

Let us consider the following set of 4-blocks:

{1 2 3 4 5 6 7 8}
{1 2 3 4 5 9 10 11}
{1 2 3 4 5 12 13 14}
{1 2 3 4 5 15 16 17}

The lists of trivially isomorphic elements for this set of 4-blocks are as follows:

{1 2 3 4 5} are trivially isomorphic
{6 7 8} are trivially isomorphic
{9 10 11} are trivially isomorphic
{12 13 14} are trivially isomorphic
{15 16 17} are trivially isomorphic
{18}

Now, we will generate all trivially non-isomorphic sets of fifth blocks for this set of 4-blocks and try to find partial lotto designs with five blocks. The following table lists all trivially non-isomorphic fifth blocks for this set of 4-blocks.

{1 2 3 4 6 9 12 15}	{1 2 3 4 6 9 12 18}	{1 2 3 4 6 9 15 18}
{1 2 3 4 6 12 15 18}	{1 2 3 4 9 12 15 18}	{1 2 3 6 7 9 10 12}
{1 2 3 6 7 9 10 15}	{1 2 3 6 7 9 10 18}	{1 2 3 6 7 9 12 13}
{1 2 3 6 7 9 12 15}	{1 2 3 6 7 9 12 18}	{1 2 3 6 7 9 15 16}
{1 2 3 6 7 9 15 18}	{1 2 3 6 7 12 13 15}	{1 2 3 6 7 12 13 18}
{1 2 3 6 7 12 15 16}	{1 2 3 6 7 12 15 18}	{1 2 3 6 7 15 16 18}
{1 2 3 6 9 10 12 13}	{1 2 3 6 9 10 12 15}	{1 2 3 6 9 10 12 18}
{1 2 3 6 9 10 15 16}	{1 2 3 6 9 10 15 18}	{1 2 3 6 9 12 13 15}
{1 2 3 6 9 12 13 18}	{1 2 3 6 9 12 15 16}	{1 2 3 6 9 12 15 18}
{1 2 3 6 9 15 16 18}	{1 2 3 6 12 13 15 16}	{1 2 3 6 12 13 15 18}
{1 2 3 6 12 15 16 18}	{1 2 3 9 10 12 13 15}	{1 2 3 9 10 12 13 18}
{1 2 3 9 10 12 15 16}	1 2 3 9 10 12 15 18	1 2 3 9 10 15 16 18
{1 2 3 9 12 13 15 16}	{1 2 3 9 12 13 15 18}	{1 2 3 9 12 15 16 18}
{1 2 3 12 13 15 16 18}	{1 2 6 7 8 9 10 11}	{1 2 6 7 8 9 10 12}
{1 2 6 7 8 9 10 15}	{1 2 6 7 8 9 10 18}	{1 2 6 7 8 9 12 13}
{1 2 6 7 8 9 12 15}	{1 2 6 7 8 9 12 18}	{1 2 6 7 8 9 15 16}
{1 2 6 7 8 9 15 18}	{1 2 6 7 8 12 13 14}	{1 2 6 7 8 12 13 15}
{1 2 6 7 8 12 13 18}	{1 2 6 7 8 12 15 16}	{1 2 6 7 8 12 15 18}
{1 2 6 7 8 15 16 17}	{1 2 6 7 8 15 16 18}	{1 2 6 7 9 10 11 12}
{1 2 6 7 9 10 11 15}	{1 2 6 7 9 10 11 18}	{1 2 6 7 9 10 12 13}
{1 2 6 7 9 10 12 15}	{1 2 6 7 9 10 12 18}	{1 2 6 7 9 10 15 16}
{1 2 6 7 9 10 15 18}	{1 2 6 7 9 12 13 14}	{1 2 6 7 9 12 13 15}
{1 2 6 7 9 12 13 18}	{1 2 6 7 9 12 15 16}	{1 2 6 7 9 12 15 18}
{1 2 6 7 9 15 16 17}	{1 2 6 7 9 15 16 18}	{1 2 6 7 12 13 14 15}
{1 2 6 7 12 13 14 18}	{1 2 6 7 12 13 15 16}	{1 2 6 7 12 13 15 18}
{1 2 6 7 12 15 16 17}	{1 2 6 7 12 15 16 18}	{1 2 6 7 15 16 17 18}
{1 2 6 9 10 11 12 13}	{1 2 6 9 10 11 12 15}	{1 2 6 9 10 11 12 18}
{1 2 6 9 10 11 15 16}	{1 2 6 9 10 11 15 18}	{1 2 6 9 10 12 13 14}
{1 2 6 9 10 12 13 15}	{1 2 6 9 10 12 13 18}	{1 2 6 9 10 12 15 16}
{1 2 6 9 10 12 15 18}	{1 2 6 9 10 15 16 17}	{1 2 6 9 10 15 16 18}
{1 2 6 9 12 13 14 15}	{1 2 6 9 12 13 14 18}	{1 2 6 9 12 13 15 16}
{1 2 6 9 12 13 15 18}	{1 2 6 9 12 15 16 17}	{1 2 6 9 12 15 16 18}
{1 2 6 9 15 16 17 18}	{1 2 6 12 13 14 15 16}	{1 2 6 12 13 14 15 18}
{1 2 6 12 13 15 16 17}	{1 2 6 12 13 15 16 18}	{1 2 6 12 15 16 17 18}
{1 2 9 10 11 12 13 14}	{1 2 9 10 11 12 13 15}	{1 2 9 10 11 12 13 18}
{1 2 9 10 11 12 15 16}	{1 2 9 10 11 12 15 18}	{1 2 9 10 11 15 16 17}
{1 2 9 10 11 15 16 18}	{1 2 9 10 12 13 14 15}	{1 2 9 10 12 13 14 18}
{1 2 9 10 12 13 15 16}	{1 2 9 10 12 13 15 18}	{1 2 9 10 12 15 16 17}
{1 2 9 10 12 15 16 18}	{1 2 9 10 15 16 17 18}	{1 2 9 12 13 14 15 16}
{1 2 9 12 13 14 15 18}	{1 2 9 12 13 15 16 17}	{1 2 9 12 13 15 16 18}
{1 2 9 12 15 16 17 18}	{1 2 12 13 14 15 16 17}	{1 2 12 13 14 15 16 18}
{1 2 12 13 15 16 17 18}	{1 6 7 8 9 10 11 12}	{1 6 7 8 9 10 11 15}
{1 6 7 8 9 10 11 18}	{1 6 7 8 9 10 12 13}	{1 6 7 8 9 10 12 15}
{1 6 7 8 9 10 12 18}	{1 6 7 8 9 10 15 16}	{1 6 7 8 9 10 15 18}

{1 6 7 8 9 12 13 14}	{1 6 7 8 9 12 13 15}	{1 6 7 8 9 12 13 18}
{1 6 7 8 9 12 15 16}	{1 6 7 8 9 12 15 18}	{1 6 7 8 9 15 16 17}
{1 6 7 8 9 15 16 18}	{1 6 7 8 12 13 14 15}	{1 6 7 8 12 13 14 18}
{1 6 7 8 12 13 15 16}	{1 6 7 8 12 13 15 18}	{1 6 7 8 12 15 16 17}
{1 6 7 8 12 15 16 18}	{1 6 7 8 15 16 17 18}	{1 6 7 9 10 11 12 13}
{1 6 7 9 10 11 12 15}	{1 6 7 9 10 11 12 18}	{1 6 7 9 10 11 15 16}
{1 6 7 9 10 11 15 18}	{1 6 7 9 10 12 13 14}	{1 6 7 9 10 12 13 15}
{1 6 7 9 10 12 13 18}	{1 6 7 9 10 12 15 16}	{1 6 7 9 10 12 15 18}
{1 6 7 9 10 15 16 17}	{1 6 7 9 10 15 16 18}	{1 6 7 9 12 13 14 15}
{1 6 7 9 12 13 14 18}	{1 6 7 9 12 13 15 16}	{1 6 7 9 12 13 15 18}
{1 6 7 9 12 15 16 17}	{1 6 7 9 12 15 16 18}	{1 6 7 9 15 16 17 18}
{1 6 7 12 13 14 15 16}	{1 6 7 12 13 14 15 18}	{1 6 7 12 13 15 16 17}
{1 6 7 12 13 15 16 18}	{1 6 7 12 15 16 17 18}	{1 6 9 10 11 12 13 14}
{1 6 9 10 11 12 13 15}	{1 6 9 10 11 12 13 18}	{1 6 9 10 11 12 15 16}
{1 6 9 10 11 12 15 18}	{1 6 9 10 11 15 16 17}	{1 6 9 10 11 15 16 18}
{1 6 9 10 12 13 14 15}	{1 6 9 10 12 13 14 18}	{1 6 9 10 12 13 15 16}
{1 6 9 10 12 13 15 18}	{1 6 9 10 12 15 16 17}	{1 6 9 10 12 15 16 18}
{1 6 9 10 15 16 17 18}	{1 6 9 12 13 14 15 16}	{1 6 9 12 13 14 15 18}
{1 6 9 12 13 15 16 17}	{1 6 9 12 13 15 16 18}	{1 6 9 12 15 16 17 18}
{1 6 12 13 14 15 16 17}	{1 2 3 4 6 9 12 15}	{1 2 3 4 6 9 12 18}
{1 2 3 4 6 9 15 18}	{1 2 3 4 6 12 15 18}	{1 2 3 4 9 12 15 18}
{1 2 3 6 7 9 10 12}	{1 2 3 6 7 9 10 15}	{1 2 3 6 7 9 10 18}
{1 2 3 6 7 9 12 13}	{1 2 3 6 7 9 12 15}	{1 2 3 6 7 9 12 18}
{1 2 3 6 7 9 15 16}	{1 2 3 6 7 9 15 18}	{1 2 3 6 7 12 13 15}
{1 2 3 6 7 12 13 18}	{1 2 3 6 7 12 15 16}	{1 2 3 6 7 12 15 18}
{1 2 3 6 7 15 16 18}	{1 2 3 6 9 10 12 13}	{1 2 3 6 9 10 12 15}
{1 2 3 6 9 10 12 18}	{1 2 3 6 9 10 15 16}	{1 2 3 6 9 10 15 18}
{1 2 3 6 9 12 13 15}	{1 2 3 6 9 12 13 18}	{1 2 3 6 9 12 15 16}
{1 2 3 6 9 12 15 18}	{1 2 3 6 9 15 16 18}	{1 2 3 6 12 13 15 16}
{1 2 3 6 12 13 15 18}	{1 2 3 6 12 15 16 18}	{1 2 3 9 10 12 13 15}
{1 2 3 9 10 12 13 18}	{1 2 3 9 10 12 15 16}	{1 2 3 9 10 12 15 18}
{1 2 3 9 10 15 16 18}	{1 2 3 9 12 13 15 16}	{1 2 3 9 12 13 15 18}
{1 2 3 9 12 15 16 18}	{1 2 3 12 13 15 16 18}	{1 2 6 7 8 9 10 11}
{1 2 6 7 8 9 10 12}	{1 2 6 7 8 9 10 15}	{1 2 6 7 8 9 10 18}
{1 2 6 7 8 9 12 13}	{1 2 6 7 8 9 12 15}	{1 2 6 7 8 9 12 18}
{1 2 6 7 8 9 15 16}	{1 2 6 7 8 9 15 18}	{1 2 6 7 8 12 13 14}
{1 2 6 7 8 12 13 15}	{1 2 6 7 8 12 13 18}	{1 2 6 7 8 12 15 16}
{1 2 6 7 8 12 15 18}	{1 2 6 7 8 15 16 17}	{1 2 6 7 8 15 16 18}
{1 2 6 7 9 10 11 12}	{1 2 6 7 9 10 11 15}	{1 2 6 7 9 10 11 18}
{1 2 6 7 9 10 12 13}	{1 2 6 7 9 10 12 15}	{1 2 6 7 9 10 12 18}
{1 2 6 7 9 10 15 16}	{1 2 6 7 9 10 15 18}	{1 2 6 7 9 12 13 14}
{1 2 6 7 9 12 13 15}	{1 2 6 7 9 12 13 18}	{1 2 6 7 9 12 15 16}
{1 2 6 7 9 12 15 18}	{1 2 6 7 9 15 16 17}	{1 2 6 7 9 15 16 18}

{1 2 6 7 12 13 14 15}	{1 2 6 7 12 13 14 18}	{1 2 6 7 12 13 15 16}
{1 2 6 7 12 13 15 18}	{1 2 6 7 12 15 16 17}	{1 2 6 7 12 15 16 18}
{1 2 6 7 15 16 17 18}	{1 2 6 9 10 11 12 13}	{1 2 6 9 10 11 12 15}
{1 2 6 9 10 11 12 18}	{1 2 6 9 10 11 15 16}	{1 2 6 9 10 11 15 18}
{1 2 6 9 10 12 13 14}	{1 2 6 9 10 12 13 15}	{1 2 6 9 10 12 13 18}
{1 2 6 9 10 12 15 16}	{1 2 6 9 10 12 15 18}	{1 2 6 9 10 15 16 17}
{1 2 6 9 10 15 16 18}	{1 2 6 9 12 13 14 15}	{1 2 6 9 12 13 14 18}
{1 2 6 9 12 13 15 16}	{1 2 6 9 12 13 15 18}	{1 2 6 9 12 15 16 17}
{1 2 6 9 12 15 16 18}	{1 2 6 9 15 16 17 18}	{1 2 6 12 13 14 15 16}
{1 2 6 12 13 14 15 18}	{1 2 6 12 13 15 16 17}	{1 2 6 12 13 15 16 18}
{1 6 12 13 14 15 16 18}	{1 6 12 13 15 16 17 18}	{1 9 10 11 12 13 14 15}
{1 9 10 11 12 13 14 18}	{1 9 10 11 12 13 15 16}	{1 9 10 11 12 13 15 18}
{1 9 10 11 12 15 16 17}	{1 9 10 11 12 15 16 18}	{1 9 10 11 15 16 17 18}
{1 9 10 12 13 14 15 16}	{1 9 10 12 13 14 15 18}	{1 9 10 12 13 15 16 17}
{1 9 10 12 13 15 16 18}	{1 9 10 12 15 16 17 18}	{1 9 12 13 14 15 16 17}
{1 9 12 13 14 15 16 18}	{1 9 12 13 15 16 17 18}	{1 12 13 14 15 16 17 18}
{6 7 8 9 10 11 12 13}	{6 7 8 9 10 11 12 15}	{6 7 8 9 10 11 12 18}
{6 7 8 9 10 11 15 16}	{6 7 8 9 10 11 15 18}	{6 7 8 9 10 12 13 14}
{6 7 8 9 10 12 13 15}	{6 7 8 9 10 12 13 18}	{6 7 8 9 10 12 15 16}
{6 7 8 9 10 12 15 18}	{6 7 8 9 10 15 16 17}	{6 7 8 9 10 15 16 18}
{6 7 8 9 12 13 14 15}	{6 7 9 12 13 14 18}	{6 7 8 9 12 13 15 16}
{6 7 8 9 12 13 15 18}	{6 7 8 9 12 15 16 17}	{6 7 8 9 12 15 16 18}
{6 7 8 9 15 16 17 18}	{6 7 8 12 13 14 15 16}	{6 7 8 12 13 14 15 18}
{6 7 8 12 13 15 16 17}	{6 7 8 12 13 15 16 18}	{6 7 8 12 15 16 17 18}
{6 7 9 10 11 12 13 14}	{6 7 9 10 11 12 13 15}	{6 7 9 10 11 12 13 18}
{6 7 9 10 11 12 15 16}	{6 7 9 10 11 12 15 18}	{6 7 9 10 11 15 16 17}
{6 7 9 10 11 15 16 18}	{6 7 9 10 12 13 14 15}	{6 7 9 10 12 13 14 18}
{6 7 9 10 12 13 15 16}	{6 7 9 10 12 13 15 18}	{6 7 9 10 12 15 16 17}
{6 7 9 10 12 15 16 18}	{6 7 9 10 15 16 17 18}	{6 7 9 12 13 14 15 16}
{6 7 9 12 13 14 15 18}	{6 7 9 12 13 15 16 17}	{6 7 9 12 13 15 16 18}
{6 7 9 12 15 16 17 18}	{6 7 12 13 14 15 16 17}	{6 7 12 13 14 15 16 18}
{6 7 12 13 15 16 17 18}	{6 9 10 11 12 13 14 15}	{6 9 10 11 12 13 14 18}
{6 9 10 11 12 13 15 16}	{6 9 10 11 12 13 15 18}	{6 9 10 11 12 15 16 17}
{6 9 10 11 12 15 16 18}	{6 9 10 11 15 16 17 18}	{6 9 10 12 13 14 15 16}
{6 9 10 12 13 14 15 18}	{6 9 10 12 13 15 16 17}	{6 9 10 12 13 15 16 18}
{6 9 10 12 15 16 17 18}	{6 9 12 13 14 15 16 17}	{6 9 12 13 14 15 16 18}
{6 9 12 13 15 16 17 18}	{6 12 13 14 15 16 17 18}	{9 10 11 12 13 14 15 16}
{9 10 11 12 13 14 15 18}	{9 10 11 12 13 15 16 17}	{9 10 11 12 13 15 16 18}
{9 10 11 12 15 16 17 18}	{9 10 12 13 14 15 16 17}	{9 10 12 13 14 15 16 18}
{9 10 12 13 15 16 17 18}	{9 12 13 14 15 16 17 18}	

Table 3.4: All trivially non-isomorphic 5th blocks for the 4-block {{1 2 3 4 5 6 7 8}{1 2 3 4 5 9 10 11}{1 2 3 4 5 12 13 14}{1 2 3 4 5 15 16 17}}

In the Table 3.4, all the blocks in lexicographical order between any two blocks are trivially isomorphic to the first of the two blocks thus are discarded from the list of fifth blocks. For example, all the blocks between {1 2 3 4 6 9 12 15} and {1 2 3 4 6 9 12 18} in lexicographical order are trivially isomorphic to {1 2 3 4 6 9 12 15}. Hence, we have discarded all the blocks between {1 2 3 4 6 9 12 15} and {1 2 3 4 6 9 12 18} from the list of fifth blocks.

3.5.4 Finding blocks at the level 6

Now, considering the following set of 5-blocks:

{1 2 3 4 5 6 7 8}
{1 2 3 4 5 6 7 9}
{1 2 3 4 5 6 7 10}
{1 2 3 4 5 6 7 11}
{1 2 3 4 5 6 7 12}

The lists of trivially isomorphic elements for this set of 5-blocks are as follows:

{1 2 3 4 5 6 7} are trivially isomorphic
{8}
{9}
{10}
{11}
{12}
{13 14 15 16 17 18} are trivially isomorphic

We get the following list of all trivially non-isomorphic sixth blocks.

{1 2 3 4 5 6 7 13}	{1 2 3 4 5 6 8 9}	{1 2 3 4 5 6 8 10}
{1 2 3 4 5 6 8 11}	{1 2 3 4 5 6 8 12}	{1 2 3 4 5 6 8 13}
{1 2 3 4 5 6 9 10}	{1 2 3 4 5 6 9 11}	{1 2 3 4 5 6 9 12}
{1 2 3 4 5 6 9 13}	{1 2 3 4 5 6 10 11}	{1 2 3 4 5 6 10 12}
{1 2 3 4 5 6 10 13}	{1 2 3 4 5 6 11 12}	{1 2 3 4 5 6 11 13}
{1 2 3 4 5 6 12 13}	{1 2 3 4 5 6 13 14}	{1 2 3 4 5 8 9 10}
{1 2 3 4 5 8 9 11}	{1 2 3 4 5 8 9 12}	{1 2 3 4 5 8 9 13}
{1 2 3 4 5 8 10 11}	{1 2 3 4 5 8 10 12}	{1 2 3 4 5 8 10 13}
{1 2 3 4 5 8 11 12}	{1 2 3 4 5 8 11 13}	{1 2 3 4 5 8 12 13}
{1 2 3 4 5 8 13 14}	{1 2 3 4 5 9 10 11}	{1 2 3 4 5 9 10 12}
{1 2 3 4 5 9 10 13}	{1 2 3 4 5 9 11 12}	{1 2 3 4 5 9 11 13}
{1 2 3 4 5 9 12 13}	{1 2 3 4 5 9 13 14}	{1 2 3 4 5 10 11 12}
{1 2 3 4 5 10 11 13}	{1 2 3 4 5 10 12 13}	{1 2 3 4 5 10 13 14}
{1 2 3 4 5 11 12 13}	{1 2 3 4 5 11 13 14}	{1 2 3 4 5 12 13 14}
{1 2 3 4 5 13 14 15}	{1 2 3 4 8 9 10 11}	{1 2 3 4 8 9 10 12}
{1 2 3 4 8 9 10 13}	{1 2 3 4 8 9 11 12}	{1 2 3 4 8 9 11 13}
{1 2 3 4 8 9 12 13}	{1 2 3 4 8 9 13 14}	{1 2 3 4 8 10 11 12}
{1 2 3 4 8 10 11 13}	{1 2 3 4 8 10 12 13}	{1 2 3 4 8 10 13 14}
{1 2 3 4 8 11 12 13}	{1 2 3 4 8 11 13 14}	{1 2 3 4 8 12 13 14}
{1 2 3 4 8 13 14 15}	{1 2 3 4 9 10 11 12}	{1 2 3 4 9 10 11 13}
{1 2 3 4 9 10 12 13}	{1 2 3 4 9 10 13 14}	{1 2 3 4 9 11 12 13}
{1 2 3 4 9 11 13 14}	{1 2 3 4 9 12 13 14}	{1 2 3 4 9 13 14 15}
{1 2 3 4 10 11 12 13}	{1 2 3 4 10 11 13 14}	{1 2 3 4 10 12 13 14}
{1 2 3 4 10 13 14 15}	{1 2 3 4 11 12 13 14}	{1 2 3 4 11 13 14 15}
{1 2 3 4 12 13 14 15}	{1 2 3 4 13 14 15 16}	{1 2 3 8 9 10 11 12}
{1 2 3 8 9 10 11 13}	{1 2 3 8 9 10 12 13}	{1 2 3 8 9 10 13 14}
{1 2 3 8 9 11 12 13}	{1 2 3 8 9 11 13 14}	{1 2 3 8 9 12 13 14}
{1 2 3 8 9 13 14 15}	{1 2 3 8 10 11 12 13}	{1 2 3 8 10 11 13 14}
{1 2 3 8 10 12 13 14}	{1 2 3 8 10 13 14 15}	{1 2 3 8 11 12 13 14}
{1 2 3 8 11 13 14 15}	{1 2 3 8 12 13 14 15}	{1 2 3 8 13 14 15 16}
{1 2 3 9 10 11 12 13}	{1 2 3 9 10 11 13 14}	{1 2 3 9 10 12 13 14}
{1 2 3 9 10 13 14 15}	{1 2 3 9 11 12 13 14}	{1 2 3 9 11 13 14 15}
{1 2 3 9 12 13 14 15}	{1 2 3 9 13 14 15 16}	{1 2 3 10 11 12 13 14}
{1 2 3 10 11 13 14 15}	{1 2 3 10 12 13 14 15}	{1 2 3 10 13 14 15 16}
{1 2 3 11 12 13 14 15}	{1 2 3 11 13 14 15 16}	{1 2 3 12 13 14 15 16}
{1 2 3 13 14 15 16 17}	{1 2 8 9 10 11 12 13}	{1 2 8 9 10 11 13 14}
{1 2 8 9 10 12 13 14}	{1 2 8 9 10 13 14 15}	{1 2 8 9 11 12 13 14}
{1 2 8 9 11 13 14 15}	{1 2 8 9 12 13 14 15}	{1 2 8 9 13 14 15 16}
{1 2 8 10 11 12 13 14}	{1 2 8 10 11 13 14 15}	{1 2 8 10 12 13 14 15}
{1 2 8 10 13 14 15 16}	{1 2 8 11 12 13 14 15}	{1 2 8 11 13 14 15 16}
{1 2 8 12 13 14 15 16}	{1 2 8 13 14 15 16 17}	{1 2 9 10 11 12 13 14}
{1 2 9 10 11 13 14 15}	{1 2 9 10 12 13 14 15}	{1 2 9 10 13 14 15 16}

{1 2 9 11 12 13 14 15}	{1 2 9 11 13 14 15 16}	{1 2 9 12 13 14 15 16}
{1 2 9 13 14 15 16 17}	{1 2 10 11 12 13 14 15}	{1 2 10 11 13 14 15 16}
{1 2 10 12 13 14 15 16}	{1 2 10 13 14 15 16 17}	{1 2 11 12 13 14 15 16}
{1 2 11 13 14 15 16 17}	{1 2 12 13 14 15 16 17}	{1 2 13 14 15 16 17 18}
{1 8 9 10 11 12 13 14}	{1 8 9 10 11 13 14 15}	{1 8 9 10 12 13 14 15}
{1 8 9 10 13 14 15 16}	{1 8 9 11 12 13 14 15}	{1 8 9 11 13 14 15 16}
{1 8 9 12 13 14 15 16}	{1 8 9 13 14 15 16 17}	{1 8 10 11 12 13 14 15}
{1 8 10 11 13 14 15 16}	{1 8 10 12 13 14 15 16}	{1 8 10 13 14 15 16 17}
{1 8 11 12 13 14 15 16}	{1 8 11 13 14 15 16 17}	{1 8 12 13 14 15 16 17}
{1 8 13 14 15 16 17 18}	{1 9 10 11 12 13 14 15}	{1 9 10 11 13 14 15 16}
{1 9 10 12 13 14 15 16}	{1 9 10 13 14 15 16 17}	{1 9 11 12 13 14 15 16}
{1 9 11 13 14 15 16 17}	{1 9 12 13 14 15 16 17}	{1 9 13 14 15 16 17 18}
{1 10 11 12 13 14 15 16}	{1 10 11 13 14 15 16 17}	{1 10 12 13 14 15 16 17}
{1 10 13 14 15 16 17 18}	{1 11 12 13 14 15 16 17}	{1 11 13 14 15 16 17 18}
{1 12 13 14 15 16 17 18}	{8 9 10 11 12 13 14 15}	{8 9 10 11 13 14 15 16}
{8 9 10 12 13 14 15 16}	{8 9 10 13 14 15 16 17}	{8 9 11 12 13 14 15 16}
{8 9 11 13 14 15 16 17}	{8 9 12 13 14 15 16 17}	{8 9 13 14 15 16 17 18}
{8 10 11 12 13 14 15 16}	{8 10 11 13 14 15 16 17}	{8 10 12 13 14 15 16 17}
{8 10 13 14 15 16 17 18}	{8 11 12 13 14 15 16 17}	{8 11 13 14 15 16 17 18}
{8 12 13 14 15 16 17 18}	{9 10 11 12 13 14 15 16}	{9 10 11 13 14 15 16 17}
{9 10 12 13 14 15 16 17}	{9 10 13 14 15 16 17 18}	{9 11 12 13 14 15 16 17}
{9 11 13 14 15 16 17 18}	{9 12 13 14 15 16 17 18}	{10 11 12 13 14 15 16 17}
{10 11 13 14 15 16 17 18}	{10 12 13 14 15 16 17 18}	{11 12 13 14 15 16 17 18}

Table 3.5: All trivially non-isomorphic 6th blocks for the 5-block {{1 2 3 4 5 6 7 8}{1 2 3 4 5 6 7 9}{1 2 3 4 5 6 7 10}{1 2 3 4 5 6 7 11}{1 2 3 4 5 6 7 12}}

We can only add a block as the sixth block from this list. For example, if we cannot add {1 2 3 4 5 6 7 13} as the sixth block then we cannot add any of the blocks in lexicographical order between {1 2 3 4 5 6 7 13} and {1 2 3 4 5 6 8 9} as the sixth block as these blocks are trivially isomorphic to {1 2 3 4 5 6 7 13}.

3.5.5 Frequency Results

In the searches where we try to prove that $L(n, k, p, t)$ is equal to 6 or not, we use the frequency results of Li and van Rees [14] to prune the search tree. The result states that there is an optimal lotto design wherein each element occurs at least once.

Our algorithm backtracks to the previous level immediately if it finds that there are more than k-elements left with frequency 0 at the last level. Let us consider, a lotto design (13, 4, 4, 2) and also consider the following 4-blocks.

{1 2 3 4}
{1 2 3 5}
{1 2 4 5}
{1 2 4 6}

If we try to find a lotto design with 5 blocks, then we still have $7 > 4$ elements of frequency 0 (i.e. 7, 8, 9, 10, 11, 12, 13) to be added in the fifth blocks. In this case, the program backtracks to the previous level.

Our algorithm also prunes the search tree by generating only the blocks containing the frequency 0 elements at the last level. Let us consider another 4-block of (13, 4, 4, 2):

{1 2 3 4}
{1 2 5 6}
{3 4 5 7}
{6 8 9 10}

Now, we have only 3 elements of frequency 0 (i.e. 11 12 13) left to be added in the third block. Hence, we can generate only the blocks containing the elements 11, 12, 13 as the fifth blocks and thus, reduces the size of the search tree.

In general, frequency results are of use at the bottom of the search tree. They are of no use when calculating $\eta(n, k, p, t)$.

3.6 Major Components and Data Structure

The programs determine the number of non-isomorphic optimal lotto designs for $L(n, k, p, t)$ on 5 or 6 blocks or try to improve the lower bounds for $n, k, p, t \leq 20$.

The major components (i.e. data structure and functions) of the programs are as follows:

Firstly, we will define variables for p-sets, k-sets, flags for p-sets, and variables for counting search time. The p-sets and k-sets are stored in an array of unsigned long integers which are dynamically allocated. In this program, we have stored each of the k-sets and p-sets as bit-sets. Thus, we need only one unsigned long to store a k-set or p-set if $n \leq 32$. Suppose, $\{1\ 2\ 3\}$ is a k-set. We store this k-set as $000\ldots0000111$. The bit position will be 1 if the position number represents an element in the k-set or p-set. We will use bitwise operation to perform intersections between a p-set and a k-set and to generate trivially isomorphic elements and blocks. For example, if we perform bitwise AND operation between a k-set and p-set then the number of 1's in the result will represent the value of t, the total number of intersections. As p-set and k-set are declared as unsigned long and as the size of unsigned long is 32 bit, the value of n can be of maximum 32. Both of the 3-block set algorithm and improved 3-block set algorithm use the following functions.

The $initializePsetFlag()$ function initializes the flags for each of the p-set as unrepresented. The parameters of this function are an array of pointers $^*pSetFlag$ (flags) and the number of p sets $nPsetNo$. The $createSet()$ function generates all p-sets and k-sets. The parameters of this function are the total number of elements n, the size of the p or k-set p, an array of pointers to unsigned long integer $^*pSet$ to store the p-sets, total number of p-sets $nTotalSetNo$. The $resetFlag()$ resets the flags of p-sets when the program backtracks to the previous level.

The $cover\_pset()$ function marks the flags of the p-sets represented by a k-set. The parameters passed to this function are the k-set $testKset$ of type unsigned long integer, the list of p-sets $^*pSet$ of type unsigned long integer, the list of flags of p-sets $^*pSetFlag$ of type long integer, the total number of p-sets $nPsetNo$, the total num-

ber of elements of the design n, the number of intersection t, the index of the first unrepresented p-set $startIndex$. This function also invokes a function $match\_block()$ function to find the number intersections between a k-set and a p-set.

In our program, we will generate all possible sets of non-isomorphic three blocks by applying block intersection properties. For each set of the three blocks, we will have to find the rest of the blocks of the design. The program does not backtrack to the third level. Thus, after representing p-sets by the first three blocks, we will create a list of p-sets which are still unrepresented by using $createNewPset()$ function. The parameters passed to this function are all p-sets $^{*}pSet$, flags of p-sets $^{*}pSetFlag$, total number of p-sets $nPsetNo$, an array of pointers $^{*}remainingPSet$ to store the unrepresented p-sets, index of the first unrepresented p-set $uncoverPIndex$. The $Cover\_newPset()$ functions marks the new list of p-sets represented by the fourth, fifth and sixth blocks of the design. Fourth, fifth and sixth blocks are generated by applying trivial isomorphism rejection. The $create\_third\_isolist()$, $create\_forth\_isolist()$ and $create\_fifth\_isolist()$ functions create the list of trivially isomorphic elements at the level 4, 5 and 6 respectively. The $create\_third\_isolist()$ uses the first three blocks $^{*}firstBlock$, $^{*}secondBlock$ and $^{*}thirdBlock$, the $create\_forth\_isolist()$ uses the first four blocks $^{*}firstBlock$, $^{*}secondBlock$, $^{*}thirdBlock$ and $^{*}forthBlock$, and the $create\_fifth\_isolist()$ uses the first five blocks $^{*}firstBlock$, $^{*}secondBlock$,

$^{*}thirdBlock$, $^{*}forthBlock$ and $^{*}fifthBlock$ to generate the corresponding trivially isomorphic elements. $^{*}firstBlock$, $^{*}secondBlock$, $^{*}thirdBlock$, $^{*}forthBlock$ and $^{*}fifthBlock$ are one dimensional array of pointers. As each of the above three functions generate isomorphic elements, global variables, two dimensional array of pointers $^{**}final\_iso\_list$ and one dimensional array of pointers $^{*}iso\_final\_index$ are used to store the isomorphic elements. The $create\_forth\_fifth\_block()$ creates all trivially non-isomorphic blocks at the level 4, 5 and 6 based on the trivially isomor-

phic elements generated.

The function $findUncoverPset()$ uses the flags of p-sets $pSetFlag$ to find the first unrepresented p-set in a list of p-sets.

To optimize the program by applying frequency results, we will maintain a list to count the frequency of each element. The function $initialize\_count()$ initializes the frequency of each element to zero. The parameters passed to this function are the total number of elements in the design n, an one dimensional array of pointers $^*count$ to store the number of occurence of each of the elements of the design. The $update\_count()$ function uses $^*count$ and the added k-set $^*checkBlock$ to update the frequency of each of the elements each time a k-set is added to the design. When the program backtracks from the level X to the level X - 1, the $reset\_count()$ function will reset the frequency of the elements of the block at level X. For a design with X blocks, the $has\_all\_element()$ function checks whether all elements of the design has the frequency at least one at the level X. Otherwise, the program backtracks to the level X - 1.

Finally, we have functions to display the blocks of the designs.

Chapter 4

Results

4.1 Introduction

In this chapter, we will present results in a tabular form. Most of the designs listed are with $10 \leq n \leq 20$. Firstly, we will list the number of non-isomorphic lotto designs for some of the designs with 5 blocks for n, k, p, t less than or equal to 20. Then we will present the number of non-isomorphic lotto designs for some of the designs with 6 blocks. We will also give the list of improved lower bounds in a table. Finally, we will list some of the designs in an encoded form for certain values of the parameters n, k, p, t.

4.2 Total Number of Non-isomorphic designs

The following tables list the number of non-isomorphic lotto designs for some of the designs with 5 or 6 blocks for parameters $n, k, p, t \leq 20$. The first column of the table represents the value of the parameters n, k, p, t of the design, the second column gives the number of non-isomorphic designs and the third column is the total computation

time. The asterisk (*) in the first column represents the designs for which the number of non-isomorphic lotto designs were already found by A.P. Burger et al. [5]. In some of the small cases, where the lotto design is also a covering i.e. $k = t$, these results may already have appeared.

Table for $L(n, k, p, t) = 5$

(n, k, p, t)	$\eta(n, k, p, t)$	Time(sec)
(13, 5, 3, 2,)	2	< 1
(16, 6, 3, 2)	1	32
(18, 7, 3, 2)	9	13 min 43 sec
(10, 3, 4, 2)	2	< 1
(13, 4, 4, 2,)	8	< 1
(14, 4, 4, 2)	1	< 1
(16, 5, 4, 2)	22	23
(17, 5, 4, 2)	2	40
(19, 6, 4, 2)	66	10 min 37 sec
(20, 6, 4, 2)	8	15 min 49 sec
(14, 10, 3, 3)	1	< 1
(11, 6, 4, 3)	3	< 1
(11, 5, 5, 3)	49	< 1
(13, 6, 5, 3)	145	30
(14, 6, 5, 3)	5	2 min 23 sec
(15, 7, 5, 3)	523	9 min 57 sec
(17, 8, 5, 3)	1300	1 hr 4 min 19 sec
(18, 8, 5, 3)	87	5 hr 59 min 53 sec
(19, 9, 5, 3)	3719	15 hr 21 min 10 sec
(20, 9, 5, 3)	305	17 hr 56 min 9 sec
(11, 4, 6, 3)	2	< 1
(16, 6, 6, 3)	7	10 min 35 sec
(19, 7, 6, 3)*	2	16 hr 22 sec
(11, 3, 8, 3)	1	< 1
(14, 4, 8, 3)	4	< 1
(17, 5, 8, 3)	16	3 min 53 sec
(18, 5, 8, 3)	1	9 min 56 sec
(20, 6, 8, 3)	51	3 hr 22 min 40 sec

(n, k, p, t)	$\eta(n, k, p, t)$	Time(sec)
(15, 3, 11, 3)	1	< 1
(19, 4, 11, 3)	1	85
(20, 4, 11, 3)	1	98
(17, 3, 13, 3)	1	< 1
(16, 13, 4, 4)	1	< 1
(17, 14, 4, 4)	1	< 1
(17, 15, 4, 4)	1	< 1
(8, 5, 5, 4)	1	< 1
(11, 7, 5, 4)	2	< 1
(16, 10, 5, 4)*	1	9 min 36 sec
(17, 11, 5, 4)*	11	23 min 7 sec
(19, 12, 5, 4)*	2	5 hr 52 min 56 sec
(11, 6, 6, 4)	47	< 1
(16, 9, 6, 4)	318	29 min 15 sec
(11, 5, 7, 4)	3	< 1
(13, 6, 7, 4)	10	29
(15, 7, 7, 4)*	26	10 min 2 sec
(17, 8, 7, 4)*	67	7 hr 8 min 26 sec
(18, 8, 7, 4)*	1	26 hr 53 sec
(19, 9, 7, 4)*	154	189 hr 17 min 56 sec
(14, 5, 9, 4)	5	32
(19, 7, 9, 4)*	20	21 hr 3 min 18 sec
(14, 4, 11, 4)	1	< 1
(17, 5 , 11, 4)	2	234
(15, 4, 12, 4)	3	< 1
(18, 5, 12, 4)	9	195
(14, 6, 10, 5)	89	30
(16, 7, 10, 5)	385	30 min 16 sec
(18, 8, 10, 5)	1413	15 hr 1 min 41 sec
(20, 9, 10, 5)	4611	95 hr 44 min 24 sec
(16, 6, 11, 5)	1	< 1
(17, 6, 12, 5)	11	4 min 38 sec
(18, 5, 15, 5)	1	5
(19, 5, 16, 5)	4	5
(15, 8, 10, 6)	526	12 min 40 sec
(14, 6, 12, 6)	11	< 1

(n, k, p, t)	$\eta(n, k, p, t)$	Time(sec)
(16, 7, 12, 6)	41	838
(18, 8, 12, 6)	167	11 hr 34 min 23 sec
(20, 9, 12, 6)	534	205 hr 36 min 18 sec
(19, 7, 14, 6)	2	3 hr 56 min 28 sec
(20, 7, 15, 6)	22	5 hr 10 min 42 sec
(12, 10, 8, 7)	2	< 1
(13, 9, 9, 7)	8	< 1
(17, 9, 12, 7)	1534	2 hr 29 min 8 sec
(16, 7, 14, 7)	2	53
(18, 8, 14, 7)	9	1 hr 49 min 41 sec
(20, 9, 14, 7)	33	2 hr 17 min 47 sec
(17, 7, 15, 7)	92	142
(19, 8, 15, 7)	502	3 hr 40 min 55 sec
(15, 12, 9, 8)	1	< 1
(14, 10, 10, 8)	1	< 1
(18, 13, 10, 8)	1	17 min 36 sec
(18, 8, 16, 8)	1	9 min 48 sec
(20, 9, 16, 8)	3	10 min 32 sec
(19, 8, 17, 8)	49	7 min 35 sec
(16, 11, 12, 9)	22	66
(16, 10, 13, 9)	1	3 min 4 sec
(20, 11, 15, 9)	307	133 hr 41 min 2 sec

Table for $L(n, k, p, t) = 6$

(n, k, p, t)	$\eta(n, k, p, t)$	Time(sec)
(10, 7, 3, 3)	5	< 1
(12, 8, 3, 3)	1	51
(13, 9, 3, 3)	4	1 min 14 sec
(16, 11, 3, 3)	1	3 hr 6 min 8 sec
(17, 12, 3, 3)	21	3 hr 41 min 1 sec
(6, 3, 4, 3)	1	< 1
(8, 4, 4, 3)	1	< 1
(12, 6, 4, 3)	1	22 min 16 sec
(12, 5, 5, 3)	9	9 min 33 sec
(10, 3, 7, 3)	6	< 1
(13, 4, 7, 3)	17	62
(16, 5, 7, 3)	100	7 hr 13 min 24 sec
(14, 3, 10, 3)	1	< 1
(18, 4, 10, 3)*	4	1 hr 4 min 32 sec
(18, 3, 13, 3)	1	11
(19, 3, 14, 3)	1	14
(20, 3, 15, 3)	1	18
(13, 4, 10, 4)	4	17
(16, 5, 10, 4)	4	2 min 8 sec
(7, 6, 5, 5)	1	< 1
(11, 10, 5, 5)	1	< 1
(14, 12, 5, 5)	1	< 1
(16, 14, 5, 5)	1	< 1
(16, 11, 6, 5)	4	27 hr 26 min 12 sec
(12, 7, 7, 5)	9	30 min 28 sec
(10, 5, 8, 5)	2	9
(16, 5, 13, 5)	1	42 min 28 sec
(11, 7, 8, 6)	12	38
(12, 6, 10, 6)	7	9 min 24 sec
(15, 11, 7, 6)	2	51 min 33 sec
(13, 10, 8, 7)	2	3
(14, 7, 12, 7)	4	2 hr 51 min 3 sec
(14, 9, 11, 8)	7	1 hr 5 min 16 sec
(12, 10, 10, 9)	2	< 1
(15, 13, 10, 9)	2	< 1
(17, 15, 10, 9)	2	< 1
(19, 16, 10, 9)	2	4 min 45 sec
(13, 11, 11, 10)	1	< 1
(14, 12, 11, 10)	1	< 1

In the above tables, we verified $\eta(n, k, p, t)$ for 14 lottery numbers of A.P Burger et al. and generated $\eta(n, k, p, t)$ for 112 new lottery numbers on 5 or 6 blocks for $n, k, p, t \leq 20$.

4.3 New Bounds

The following table lists improved lower bounds for $L(n, k, p, t) \geq 6$. The first column of the table represents the value of the parameters n, k, p, t of the design, the second column gives the value of the previous lower bounds $L(n, k, p, t)$, the third column gives the value of the improved and new lower bounds and the forth column is the total computation time. The asterisk * in the first column represents the designs that A.P. Burger et al. had already improved.

Table of improved bounds

(n, k, p, t)	Previous$L(n, k, p, t)$	New $L(n, k, p, t)$	Time(sec)
(15, 6, 5, 3)	6:8	7:8	33 hr 59 min 9 sec
(16, 6, 5, 3)	6:8	7:8	78 hr 44 min 14 sec
(12, 4, 6, 3)	6:7	7	124
(15, 5, 6, 3)	6:8	7:8	4 hr 13 min 27 sec
(14, 4, 7, 3)	6:8	7:8	6 min 54 sec
(17, 5, 7, 3)	6:7	7	16 hr 59 min 46 sec
(19, 5, 8, 3)	6:7	7	48 min 58 sec
(17, 4, 9, 3)	6:7	7	29 min 18 sec
(12, 7, 5, 4)	6:9	7:9	26 min 18 sec
(14, 7, 6, 4)	6:8	7:8	42 hr 24 min 3 sec
(12, 5, 7, 4)	6:9	7:9	15 min 33 sec
(14, 6, 7, 4)*	6:8	6	23 hr 22 min 32 sec
(16, 7, 7, 4)*	6:7	6	150 hr 3 min 39 sec
(15, 5, 9, 4)	6:8	7:8	39 min 47 sec
(18, 5, 11, 4)*	6:7	7	42 min 17 sec
(16, 11, 6,5)	5:6	6	29 hr 19 min 15 sec
(15, 11, 7, 6)	5:6	6	50 min 52 sec
(19, 11, 14, 9)	5:6	6	23 hr 38 min 36 sec

From the above table, we find 18 improved lower bounds of $L(n, k, p, t)$ for $n, k, p, t \leq 20$.

4.4 Designs in a simplified form

In simplified form a design is represented as follows:

$a_1(b_1), a_2(b_2), a_3(b_3)$

If the design has 5 blocks then we have to convert a_1, a_2 and a_3 into 5 bit binary numbers. The elements 1 to b_1 occur in the blocks corresponding to the 1's in the binary representation of a_1. Similarly, the elements $b_1 + 1$ to $b_1 + b_2$ occur in the blocks corresponding to the 1's in the binary representation of a_2 and so on.

For example, 7(1), 3(1), 5(1), 6(1), 8(3), 16(3) is a design of (10, 3, 4, 2; 5) in simplified form.

Number(a)	Binary	b	Elements
7	00111	1	1
3	00011	1	2
5	00101	1	3
6	00110	1	4
8	01000	3	5-7
16	10000	3	8-10

The binary equivalent of 7 is 00111. Thus, the positions 1, 2 and 3 represent the block, B_1, B_2 and B_3 which contain the element 1 (i.e. the first, second and third blocks have the element 1). Similarly, the first and second blocks have the element 2, the first and third blocks have the element 3, the second and third blocks have the element 4, the fourth block has the elements 5, 6 and 7, the fifth block has the elements 8, 9 and 10.

Hence, we get the following design:

First Block :{1 2 3}

Second Block :{1 2 4}

Third Block :{1 3 4}

Fourth Block :{5 6 7}

Fifth Block :{8 9 10}

Now, we will list the non-isomorphic lotto designs in this simplified form for some of the designs with 5 or 6 blocks. We only list designs without isolated blocks except for a parameter for which only one lotto design exists with isolated blocks. The block where all elements have frequency one is called an *isolated* block. Lotto designs with isolated blocks are easily constructed from smaller lotto designs and so in general we do not list them.

For $L(n, k, p, t) = 5$

$t = 2$

(13 4 4 2; 5)

1. 3(3),5(1),6(1),12(1),20(1),8(3),16(3)
2. 3(2),5(2),6(1),24(1),8(3),16(3)
3. 3(2),5(2),6(1),10(1),20(1),8(3),16(3)

Total number of lotto designs with isolated blocks = 5
Total number of lotto designs = 8

(13 5 3 2; 5)

1. 7(3),9(2),10(2),12(1),20(1),16(4)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 2

(14 4 4 2; 5)

1. 3(2),5(2),6(1),8(4),16(4)

Total number of lotto designs with isolate blocks = 1
Total number of lotto designs = 1

(16 5 4 2; 5)

1. 7(1),3(3),5(1),6(1),12(1),20(1),8(4),16(4)
2. 3(4),5(1),6(1),12(2),20(1),8(3),16(4)
3. 7(1),3(2),5(2),6(1),24(1),8(4),16(4)
4. 7(1),3(2),5(2),6(1),10(1),20(1),8(4),16(4)
5. 7(1),3(2),9(2),10(2),4(4),24(1),16(4)
6. 3(3),13(1),5(1),6(1),20(1),8(4),16(4)
7. 3(3),5(2),6(1),28(1),8(4),16(4)
8. 3(3),5(2),6(1),12(1),24(1),8(3),16(4)
9. 3(3),5(2),6(1),10(1),12(1),20(1),8(3),16(4)
10. 3(3),5(2),6(1),10(1),20(2),8(4),16(3)
11. 3(2),5(2),9(1),6(1),10(1),20(1),8(3),16(4)

Total number of lotto designs with isolated blocks = 11
Total number of lotto designs = 22

(18 7 3 2; 5)

1. 7(1),11(4),17(2),18(2),4(6),24(3)
2. 7(1),11(4),5(2),6(1),18(1),12(3),16(6)
3. 7(1),11(4),5(2),6(1),12(2),24(1),16(6)
4. 7(5),9(2),10(2),12(2),24(1),16(6)
5. 7(4),9(3),10(3),12(1),20(2),16(5)
6. 7(4),9(3),10(2),18(1),12(2),20(1),16(5)

Total number of lotto designs with isolated blocks = 3
Total number of lotto designs = 9

(19 6 4 2; 5)

1. 7(2),3(3),5(1),6(1),12(1),20(1),8(5),16(5)
2. 7(1),3(4),5(1),6(1),12(2),20(1),8(4),16(5)
3. 3(5),5(1),6(1),12(3),20(1),8(3),16(5)
4. 3(5),5(1),6(1),12(2),20(2),8(4),16(4)
5. 7(2),3(2),5(2),6(1),24(1),8(5),16(5)
6. 7(2),3(2),5(2),6(1),10(1),20(1),8(5),16(5)
7. 15(1),3(3),5(2),6(2),20(1),8(5),16(5)
8. 7(1),11(1),3(2),5(2),6(1),20(1),8(5),16(5)
9. 7(1),3(3),13(1),5(1),6(1),20(1),8(5),16(5)
10. 7(1),3(3),5(2),6(1),28(1),8(5),16(5)
11. 7(1),3(3),5(2),6(1),12(1),24(1),8(4),16(5)
12. 7(1),3(3),5(2),6(1),10(1),12(1),20(1),8(4),16(5)
13. 7(1),3(3),5(2),6(1),10(1),20(2),8(5),16(4)
14. 7(1),3(3),9(2),10(2),12(1),4(4),24(1),16(5)
15. 7(1),3(3),9(2),10(2),4(5),24(2),16(4)
16. 3(4),13(1),5(1),6(1),12(1),20(1),8(4),16(5)
17. 3(4),13(1),5(1),6(1),20(2),8(5),16(4)
18. 3(4),5(2),6(1),28(1),12(1),8(4),16(5)
19. 3(4),5(2),6(1),12(2),24(1),8(3),16(5)
20. 3(4),5(2),6(1),12(1),20(1),24(1),8(4),16(4)
21. 3(4),5(2),6(1),10(1),12(2),20(1),8(3),16(5)
22. 3(4),5(2),6(1),10(1),12(1),20(2),8(4),16(4)
23. 3(4),5(2),6(1),10(1),20(3),8(5),16(3)
24. 7(1),3(2),5(2),9(1),6(1),10(1),20(1),8(4),16(5)

25. 3(3),29(1),5(2),6(1),8(5),16(5)
26. 3(3),13(1),21(1),5(1),6(1),8(5),16(5)
27. 3(3),13(1),5(2),22(1),6(1),8(5),16(5)
28. 3(3),13(1),5(2),6(1),24(1),8(4),16(5)
29. 3(3),5(3),6(1),24(2),8(4),16(4)
30. 3(3),13(1),5(2),6(1),10(1),20(1),8(4),16(5)
31. 3(3),13(1),5(2),6(1),18(1),12(1),8(4),16(5)
32. 3(3),13(1),5(2),6(1),18(1),20(1),8(5),16(4)
33. 3(3),5(3),14(1),6(1),10(1),20(1),8(4),16(5)
34. 3(3),5(3),14(1),6(1),18(1),20(1),8(5),16(4)
35. 3(3),5(3),6(1),26(1),12(1),8(4),16(5)
36. 3(3),5(3),6(1),10(1),12(1),24(1),8(3),16(5)
37. 3(3),5(3),6(1),10(1),20(1),24(1),8(4),16(4)
38. 3(3),5(3),6(1),10(2),12(1),20(1),8(3),16(5)
39. 3(3),5(3),6(1),10(2),20(2),8(4),16(4)
40. 3(3),5(3),6(1),10(1),18(1),12(1),20(1),8(4),16(4)
41. 3(3),5(2),9(1),6(1),10(1),12(1),20(1),8(3),16(5)
42. 3(3),5(2),9(1),6(1),10(1),20(2),8(4),16(4)
43. 3(3),5(2),9(1),6(1),18(1),12(2),8(3),16(5)
44. 3(3),5(2),9(1),6(1),18(1),12(1),20(1),8(4),16(4)

Total number of designs with isolated blocks = 22
Total number of lotto designs = 66

(20 6 4 2; 5)

1. 3(4),5(2),6(1),12(1),20(1),8(5),16(5)
2. 3(3),5(3),6(1),24(1),8(5),16(5)
3. 3(3),5(3),6(1),10(1),20(1),8(5),16(5)

Total number of lotto designs with isolated blocks = 5
Total number of lotto designs = 8

$t = \mathbf{3}$

(11 3 8 3; 5)

1. 3(2),5(1),6(1),4(1),8(3),16(3)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 1

(11 5 5 3; 5)

1. 15(1),7(2),11(1),13(1),14(1),24(1),16(4)
2. 7(3),11(1),13(1),14(1),24(2),16(3)
3. 7(3),11(1),17(1),18(1),12(2),24(2),16(1)
4. 7(2),11(2),13(1),14(1),28(1),16(4)
5. 7(2),11(2),13(1),14(1),20(1),24(1),16(3)
6. 7(2),3(2),13(1),14(1),20(1),24(3),16(1)
7. 7(2),3(2),13(1),10(1),20(2),24(3)
8. 7(2),11(2),17(1),18(1),12(2),20(1),24(1),16(1)
9. 7(2),11(1),19(1),9(1),10(1),20(3),24(1),8(1)
10. 15(1),3(3),13(1),14(1),20(2),24(2),16(1)
11. 15(1),3(3),13(1),6(2),20(2),24(3)
12. 7(1),11(1),3(2),13(1),14(1),20(2),24(2),16(1)
13. 7(1),11(1),3(2),13(1),6(1),20(2),24(3)
14. 7(1),3(3),13(1),14(1),28(1),20(1),24(2),16(1)
15. 7(1),3(3),13(1),14(1),20(2),24(3)
16. 7(1),3(3),13(1),6(1),28(1),20(1),24(3)
17. 7(1),3(3),13(1),10(1),28(1),20(2),24(2)

18. 3(4),13(1),14(1),28(2),20(1),24(1),16(1)
19. 3(4),13(1),14(1),28(1),20(2),24(2)
20. 3(4),13(1),6(1),28(2),20(1),24(2)
21. 7(3),9(2),10(1),18(1),20(2),24(2)
22. 7(3),9(1),17(1),10(1),18(1),12(1),20(1),24(2)
23. 7(2),11(1),13(1),17(1),10(2),20(2),24(1),16(1)
24. 7(2),11(1),5(1),17(1),18(2),12(2),24(2)
25. 7(2),11(1),9(2),18(2),12(1),20(2),24(1)
26. 7(2),11(1),9(1),17(1),10(1),18(1),12(1),20(2),24(1)
27. 7(2),11(1),17(2),10(1),18(1),12(2),20(1),24(1)
28. 7(1),3(2),13(1),21(1),10(2),20(2),24(2)
29. 15(1),3(2),13(1),17(1),6(2),10(1),20(2),24(2)
30. 7(1),11(1),3(1),13(1),17(1),6(1),10(1),20(2),24(2)
31. 7(1),11(1),19(1),5(1),9(1),6(1),10(1),20(2),24(2)
32. 7(1),3(2),13(1),9(1),14(1),18(1),20(2),24(2)
33. 7(1),3(2),13(1),25(1),6(1),10(1),20(2),24(2)
34. 7(1),3(2),13(1),17(1),6(1),10(1),28(1),20(1),24(2)
35. 7(1),3(2),25(2),10(1),18(1),12(2),20(2)
36. 3(3),13(2),14(1),22(1),20(1),24(2),16(1)
37. 3(3),13(2),22(1),6(1),20(1),24(3)
38. 3(3),13(1),21(1),14(1),6(1),20(1),24(3)
39. 3(3),13(2),14(1),18(1),20(2),24(2)
40. 3(3),29(1),13(1),6(1),10(1),20(2),24(2)
41. 3(3),13(2),22(1),10(1),20(2),24(2)
42. 3(3),13(2),6(1),18(1),28(1),20(1),24(2)

43. 3(3),13(1),21(1),14(1),10(1),20(2),24(2)
44. 3(3),13(1),5(1),14(1),18(1),28(1),20(1),24(2)
45. 3(3),13(1),21(1),6(1),10(1),28(1),20(1),24(2)
46. 3(3),13(1),17(1),6(1),10(1),28(2),20(1),24(1)
47. 7(2),25(1),9(1),17(1),10(2),18(1),12(1),20(2)
48. 7(1),11(1),5(1),25(1),17(1),6(1),18(2),12(2),24(1)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 49

(11 6 4 3; 5)

1. 7(3),11(2),17(1),18(1),28(3),24(1)
2. 7(2),11(2),3(1),17(1),18(1),28(4)
3. 7(3),11(1),25(2),26(2),12(1),20(2)

Total number of lotto designs with isolated blocks = none
Total number of lotto designs = 3

(14 6 5 3; 5)

1. 3(4),13(2),6(1),20(2),24(4)
2. 3(4),13(2),6(1),10(1),20(3),24(3)
3. 3(3),13(2),17(1),6(1),10(1),20(2),24(3)
4. 7(2),9(2),17(2),10(2),18(2),12(2),20(2)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 5

(14 10 3 3; 5)

1. 15(4), 23(4),25(2),26(2),28(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(16 6 6 3; 5)

1. 7(3),3(1),9(2),10(2),12(2),20(1),16(5)
2. 7(3),3(1),9(2),10(1),18(1),12(3),16(5)
3. 7(4),9(1),17(1),18(2),20(2),8(5),16(1)

Total number of lotto designs with isolated blocks = 4
Total number of lotto designs = 7

(17 5 8 3; 5)

1. 3(4),5(1),6(1),12(1),20(1),4(1),8(4),16(4)
2. 3(3),5(2),6(1),4(1),24(1),8(4),16(4)
3. 3(3),5(2),6(1),10(1),20(1),4(1),8(4),16(4)
4. 3(3),5(2),6(1),2(1),12(1),20(1),8(4),16(4)
5. 3(2),5(2),9(1),6(1),18(1),4(1),8(4),16(4)

Total number of lotto designs with isolated blocks = 11
Total number of lotto designs = 16

(18 5 8 3; 5)

1. 3(3), 5(2),6(1),4(1),8(5),16(5)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 1

(19 4 11 3; 5)

1. 3(1), 1(3), 2(3),4(4),8(4),16(4)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 1

(20 4 11 3; 5)

1. 1(4),2(4),4(4),8(4),16(4)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 1

(20 6 8 3; 5)

1. 7(1),3(4),5(1),6(1),12(1),20(1),4(1),8(5),16(5)
2. 7(1),3(3),5(2),6(1),4(1),24(1),8(5),16(5)
3. 7(1),3(3),5(2),6(1),10(1),20(1),4(1),8(5),16(5)
4. 7(1),3(3),5(2),6(1),2(1),12(1),20(1),8(5),16(5)
5. 7(1),3(3),9(2),10(2),4(5),24(1),8(1),16(5)
6. 3(4),13(1),5(1),6(1),20(1),4(1),8(5),16(5)
7. 3(4),5(2),6(1),28(1),4(1),8(5),16(5)
8. 3(4),5(2),6(1),12(1),20(1),8(5),16(5)
9. 3(4),5(2),6(1),12(1),4(1),24(1),8(4),16(5)
10. 3(4),5(2),6(1),10(1),20(2),4(1),8(5),16(4)
11. 3(4),5(1),9(1),10(2),12(1),4(4),24(1),8(1),16(5)
12. 7(1),3(2),5(2),9(1),6(1),18(1),4(1),8(5),16(5)
13. 3(3),5(3),6(1),24(1),8(5),16(5)
14. 3(3),13(1),5(2),6(1),18(1),4(1),8(5),16(5)
15. 3(3),13(1),5(2),6(1),2(1),20(1),8(5),16(5)
16. 3(3),5(3),14(1),6(1),18(1),4(1),8(5),16(5)
17. 3(3),5(3),6(1),26(1),4(1),8(5),16(5)
18. 3(3),5(3),6(1),10(1),20(1),8(5),16(5)
19. 3(3),5(3),6(1),10(1),4(1),24(1),8(4),16(5)
20. 3(3),5(3),6(1),10(2),20(1),4(1),8(4),16(5)
21. 3(3),5(2),9(1),6(1),10(1),20(1),4(1),8(4),16(5)
22. 3(3),5(2),9(1),6(1),18(1),12(1),4(1),8(4),16(5)
23. 3(3),5(2),9(1),6(1),2(1),12(1),20(1),8(4),16(5)
24. 3(3),5(2),9(1),6(1),2(1),20(2),8(5),16(4)

25. 3(3),5(1),9(1),17(1),10(3),12(1),4(4),8(1),16(5)

Total number of designs with isolated blocks = 26
Total number of lotto designs = 51

$t = \mathbf{4}$

(11 5 7 4; 5)

1. 7(1),3(3),13(1),14(1),20(2),24(3)
2. 3(4),13(1),14(1),28(1),20(2),24(2)
3. 3(3),13(2),14(1),18(1),20(2),24(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 3

(11 6 6 4; 5)

1. 15(5),17(1),18(1),20(1),24(1),16(2)
2. 15(4),7(1),17(1),18(1),20(1),24(2),16(1)
3. 15(3),7(2),17(1),18(1),20(1),24(3)
4. 15(4),3(1),17(1),18(1),20(2),24(2)
5. 15(3),7(1),11(1),17(1),18(1),20(2),24(2)
6. 15(1),7(3),11(1),17(1),18(1),28(2),24(2)
7. 7(3),11(2),21(1),18(1),28(2),24(2)
8. 15(1),7(2),11(2),17(1),18(1),28(3),16(1)
9. 15(1),7(2),11(2),17(1),18(1),28(2),20(1),24(1)
10. 15(1),7(2),11(1),3(1),17(1),18(1),28(3),24(1)
11. 7(3),27(1),11(1),25(1),26(1),12(1),20(2),24(1)
12. 7(3),11(2),25(1),18(1),28(3),16(1)
13. 7(3),11(2),25(1),18(1),28(2),20(1),24(1)
14. 7(3),11(2),17(1),18(1),28(3),24(1)
15. 15(1),7(1),11(1),19(2),13(1),14(1),20(2),24(2)

16. 7(2),11(2),3(1),21(1),18(1),28(3),24(1)
17. 7(2),27(1),11(2),25(1),26(1),12(1),20(3)
18. 7(2),11(2),3(1),17(1),18(1),28(4)
19. 15(1),19(4),13(1),14(1),12(2),20(1),24(1)
20. 7(1),11(4),21(1),18(1),28(2),20(2)
21. 7(1),11(3),3(1),21(1),18(1),28(3),20(1)
22. 15(1),7(2),11(1),13(1),17(1),26(2),20(2),24(1)
23. 15(1),7(2),11(1),9(1),17(1),18(2),28(3)
24. 7(3),11(1),25(2),26(2),12(1),20(2)
25. 7(2),11(2),21(2),14(1),18(1),28(1),24(2)
26. 7(2),11(2),13(1),21(1),18(2),28(2),24(1)
27. 7(2),11(2),21(2),26(2),12(1),20(1),24(1)
28. 7(2),11(2),21(1),25(1),14(1),18(1),28(2),16(1)
29. 7(2),11(2),21(1),25(1),14(1),18(1),28(1),20(1),24(1)
30. 7(2),11(2),21(1),25(1),22(1),26(1),12(1),20(1),24(1)
31. 7(2),11(2),21(1),17(1),14(1),18(1),28(2),24(1)
32. 7(2),11(2),21(1),25(1),18(2),28(2),12(1)
33. 7(1),11(1),19(2),13(1),21(1),14(2),20(1),24(2)
34. 7(1),11(3),21(2),14(1),18(1),28(2),16(1)
35. 7(1),11(3),21(2),14(1),18(1),28(1),20(1),24(1)
36. 7(1),11(3),21(2),22(1),26(1),12(1),20(1),24(1)
37. 7(1),11(3),21(2),6(1),18(1),28(2),24(1)
38. 7(1),11(2),3(1),21(2),14(1),18(1),28(2),24(1)
39. 7(1),11(3),13(1),21(1),18(2),28(2),20(1)
40. 7(1),11(3),21(2),18(2),28(2),12(1)

41. 7(1),11(3),21(1),5(1),18(2),28(3)
42. 7(1),11(2),19(1),13(2),26(2),20(3)
43. 7(1),11(3),13(1),17(1),22(1),26(1),28(1),20(2)
44. 7(1),11(3),21(1),17(1),6(1),18(1),28(3)
45. 7(1),11(2),3(1),13(1),17(1),22(1),26(1),28(2),20(1)
46. 7(2),11(1),21(1),25(2),14(1),10(1),18(1),28(1),20(1)
47. 7(2),11(1),21(1),25(2),22(1),26(1),18(1),12(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 47

(11 7 5 4; 5)

1. 15(4),23(1),19(1),21(1),22(1),24(3)
2. 15(4),19(2),21(1),22(1),28(1),24(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(13 6 7 4; 5)

1. 7(2),3(3),13(1),14(1),20(2),24(4)
2. 15(1),3(4),13(1),14(1),20(3),24(3)
3. 7(1),11(1),3(3),13(1),14(1),20(3),24(3)
4. 7(1),3(4),13(1),14(1),28(1),20(2),24(3)
5. 3(5),13(1),14(1),28(2),20(2),24(2)
6. 7(2),11(2),17(2),18(2),12(3),20(1),24(1)
7. 7(1),11(2),19(1),5(2),6(1),20(1),24(4)
8. 7(1),3(3),13(2),14(1),18(1),20(2),24(3)
9. 7(1),11(3),17(2),18(2),12(3),20(2)
10. 3(4),13(1),21(1),22(2),12(2),24(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 10

(14 4 11 4; 5)

1. 3(2),5(2),6(1),8(4),16(4)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 1

(15 7 7 4; 5)

1. 7(3),3(3),13(1),14(1),20(2),24(5)
2. 15(1),7(1),3(4),13(1),14(1),20(3),24(4)
3. 7(2),11(1),3(3),13(1),14(1),20(3),24(4)
4. 7(2),3(4),13(1),14(1),28(1),20(2),24(4)
5. 15(1),3(5),13(1),14(1),28(1),20(3),24(3)
6. 7(1),11(1),3(4),13(1),14(1),28(1),20(3),24(3)
7. 7(1),3(5),13(1),14(1),28(2),20(2),24(3)
8. 3(6),13(1),14(1),28(3),20(2),24(2)
9. 7(4),11(1),17(2),18(2),12(3),24(3)
10. 7(3),11(2),17(2),18(2),12(3),20(1),24(2)
11. 7(3),11(1),19(1),9(2),10(2),20(4),24(2)
12. 7(2),11(2),19(1),5(2),6(1),20(1),24(5)
13. 7(2),3(3),13(2),14(1),18(1),20(2),24(4)
14. 7(1),11(2),19(2),5(2),6(1),12(1),20(1),24(4)
15. 7(1),3(4),13(2),14(1),22(1),20(2),24(4)
16. 15(1),3(4),13(2),14(1),18(1),20(3),24(3)
17. 7(1),11(1),3(3),13(2),14(1),18(1),20(3),24(3)
18. 7(1),3(4),13(2),14(1),26(1),20(3),24(3)
19. 7(1),3(4),13(2),14(1),18(1),28(1),20(2),24(3)

20. 7(1),3(4),25(2),26(2),12(3),20(3)
21. 3(5),29(1),13(1),14(2),20(3),24(3)
22. 3(5),13(2),14(1),22(1),28(1),20(2),24(3)
23. 7(3),11(1),13(1),17(2),10(3),20(3),24(2)
24. 7(3),11(1),25(1),17(2),18(3),12(4),24(1)
25. 7(1),3(3),13(2),17(1),14(2),18(1),20(2),24(3)
26. 3(4),13(2),21(1),14(2),18(1),20(2),24(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 26

(16 13 4 4; 5)

1. 31(1),15(3),23(3),27(3),29(3),30(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(17 8 7 4; 5)

1. 7(4),3(3),13(1),14(1),20(2),24(6)
2. 15(1),7(2),3(4),13(1),14(1),20(3),24(5)
3. 7(3),3(4),13(1),14(1),28(1),20(2),24(5)
4. 15(2),3(5),13(1),14(1),20(4),24(4)
5. 15(1),7(1),11(1),3(4),13(1),14(1),20(4),24(4)
6. 15(1),7(1),3(5),13(1),14(1),28(1),20(3),24(4)
7. 7(2),11(2),3(3),13(1),14(1),20(4),24(4)
8. 7(2),3(5),13(1),14(1),28(2),20(2),24(4)
9. 15(1),3(6),13(1),14(1),28(2),20(3),24(3)
10. 7(1),11(3),3(3),13(1),14(1),20(5),24(3)
11. 7(1),11(2),3(4),13(1),14(1),28(1),20(4),24(3)
12. 7(1),11(1),3(5),13(1),14(1),28(2),20(3),24(3)

13. 7(1),3(6),13(1),14(1),28(3),20(2),24(3)
14. 3(7),13(1),14(1),28(4),20(2),24(2)
15. 7(5),11(1),17(2),18(2),12(3),24(4)
16. 7(4),11(2),17(2),18(2),12(3),20(1),24(3)
17. 7(4),11(1),19(1),9(2),10(2),20(4),24(3)
18. 7(3),3(3),13(2),14(1),18(1),20(2),24(5)
19. 7(3),11(2),19(1),5(2),6(1),20(1),24(6)
20. 7(3),11(3),17(2),18(2),12(3),20(2),24(2)
21. 7(3),11(2),19(1),9(2),10(2),20(5),24(2)
22. 7(3),11(2),19(1),17(2),18(2),12(4),20(1),24(2)
23. 7(2),11(2),19(2),5(2),6(1),12(1),20(1),24(5)
24. 7(2),3(4),13(2),14(1),22(1),20(2),24(5)
25. 15(1),7(1),3(4),13(2),14(1),18(1),20(3),24(4)
26. 7(2),11(1),3(3),13(2),14(1),18(1),20(3),24(4)
27. 7(2),3(4),13(2),14(1),26(1),20(3),24(4),
28. 7(2),3(4),13(2),14(1),18(1),28(1),20(2),24(4),
29. 29. 15(1),3(5),13(2),14(1),22(1),20(3),24(4)
30. 15(1),3(5),21(2),22(2),12(3),24(4)
31. 7(1),11(1),3(4),13(2),14(1),22(1),20(3),24(4)
32. 7(1),11(1),3(4),21(2),22(2),12(3),24(4)
33. 7(1),3(5),29(1),13(1),14(2),20(3),24(4)
34. 7(1),3(5),13(2),14(1),22(1),28(1),20(2),24(4)
35. 15(1),3(5),13(2),14(1),18(1),28(1),20(3),24(3)
36. 7(1),11(1),3(4),13(2),14(1),18(1),28(1),20(3),24(3)
37. 7(1),3(5),13(2),14(1),26(1),28(1),20(3),24(3)

38. 7(1),3(5),13(2),14(1),18(1),28(2),20(2),24(3)
39. 7(1),3(5),25(2),26(2),28(1),12(3),20(3)
40. 3(6),29(1),13(1),14(2),28(1),20(3),24(3)
41. 3(6),13(2),14(1),22(1),28(2),20(2),24(3)
42. 7(3),27(1),11(1),17(3),18(3),12(5),24(1)
43. 7(3),11(2),17(3),18(3),28(1),12(4),24(1)
44. 7(3),11(1),19(1),25(1),9(2),10(3),20(5),24(1)
45. 7(2),11(3),5(3),14(1),18(2),20(2),24(4)
46. 7(2),3(3),13(2),17(1),14(2),18(1),20(2),24(4)
47. 7(2),11(3),13(1),17(2),18(3),12(3),20(2),24(1)
48. 7(2),11(3),21(1),17(2),18(3),12(4),20(1),24(1)
49. 7(2),11(2),19(1),13(1),17(2),18(3),12(4),20(1),24(1)
50. 7(2),11(2),19(1),21(1),17(2),18(3),12(5),24(1)
51. 15(1),7(1),11(3),17(3),18(3),12(4),20(2)
52. 7(2),11(3),25(1),17(2),18(3),12(4),20(2)
53. 7(2),27(1),11(2),17(3),18(3),12(5),20(1)
54. 7(1),11(3),19(1),5(3),6(1),26(1),20(2),24(4)
55. 7(1),3(4),13(3),14(1),22(1),18(1),20(2),24(4)
56. 7(1),3(4),13(2),21(1),14(2),18(1),20(2),24(4)
57. 7(1),11(4),5(3),14(1),18(2),20(3),24(3)
58. 7(1),11(1),3(3),13(3),14(1),18(2),20(3),24(3)
59. 7(1),3(4),13(3),14(1),26(1),18(1),20(3),24(3)
60. 15(1),3(4),13(2),17(1),14(2),18(1),20(3),24(3)
61. 7(1),11(1),3(3),13(2),17(1),14(2),18(1),20(3),24(3)
62. 7(1),3(4),13(2),25(1),14(2),18(1),20(3),24(3)

63. 7(1),11(4),21(1),17(2),18(3),12(4),20(2)
64. 3(5),13(3),14(1),22(2),20(2),24(4)
65. 3(5),13(2),21(1),14(2),22(1),20(2),24(4)
66. 3(5),13(3),14(1),22(1),26(1),20(3),24(3)
67. 3(5),13(2),21(1),14(2),26(1),20(3),24(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 67

(17 11 5 4; 5)

1. 15(6),23(3),19(1),21(1),22(1),24(5)
2. 15(6),23(2),19(2),21(1),22(1),28(1),24(4)
3. 15(1),7(3),27(6),13(1),14(1),28(2),20(3)
4. 7(4),27(6),13(1),14(1),28(3),20(2)
5. 31(1),15(6),19(2),21(2),22(2),24(4)
6. 15(6),23(1),27(1),19(1),21(2),22(2),24(4)
7. 15(6),23(1),19(2),21(2),22(1),26(1),28(1),24(3)
8. 15(6),23(1),19(2),25(2),26(2),28(1),20(3)
9. 15(6),19(3),29(1),21(1),22(2),28(1),24(3)
10. 15(6),19(3),21(2),22(1),26(1),28(2),24(2)
11. 15(6),19(2),21(2),25(1),22(2),26(1),28(1),24(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 11

(17 14 4 4; 5)

1. 31(2),15(3),23(3),27(3),29(3),30(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(17 15 4 4; 5)

1. 31(7),15(2),23(2),27(2),29(2),30(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(18 8 7 4; 5)

1. 3(6),13(2),14(2),20(4),24(4)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(19 7 9 4; 5)

1. 7(5),9(1),17(1),10(2),12(2),8(2),16(6)
2. 7(4),3(1),9(2),10(2),12(2),20(1),8(1),16(6)
3. 7(4),3(1),9(2),10(1),18(1),12(3),8(1),16(6)
4. 7(3),3(2),9(2),10(2),12(3),20(1),16(6)
5. 7(3),3(2),9(2),10(1),18(1),12(4),16(6)
6. 7(4),9(3),10(3),12(1),20(1),4(1),16(6)
7. 7(4),9(3),10(2),18(1),12(2),4(1),16(6)
8. 7(3),3(1),5(1),9(2),10(3),12(2),20(1),16(6)
9. 7(3),3(1),5(1),9(1),17(1),10(3),12(3),16(6)

Total number of lotto designs with isolated blocks = 11
Total number of lotto designs = 20

(19 12 5 4; 5)

1. 15(7),23(1),19(2),21(2),22(2),24(5)
2. 15(7),19(3),21(2),22(2),28(1),24(4)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(20 4 16 4; 5)

1. 1(4),2(4),4(4),8(4),16(4)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 1

$t = \mathbf{5}$

(14 6 10 5; 5)

1. 7(3),11(2),13(1),14(1),12(1),24(1),16(5)
2. 7(3),11(2),17(1),18(1),12(3),24(1),16(3)
3. 7(1),11(2),19(2),5(1),6(1),4(3),24(4)
4. 7(1),3(4),13(1),14(1),12(1),20(2),24(3),16(1)
5. 7(1),3(4),13(1),14(1),20(3),24(3),8(1)
6. 7(1),3(4),13(1),14(1),20(2),4(1),24(4)
7. 7(1),3(4),13(1),6(1),12(1),20(2),24(4)
8. 7(1),3(4),13(1),10(1),12(1),20(3),24(3)
9. 3(5),13(1),14(1),28(1),12(1),20(2),24(2),16(1)
10. 3(5),13(1),14(1),28(1),20(3),24(2),8(1)
11. 3(5),13(1),14(1),12(1),20(3),24(3)
12. 3(5),13(1),6(1),28(1),12(1),20(2),24(3)
13. 7(2),11(2),5(1),17(1),18(2),12(3),24(1),16(2)
14. 7(2),11(2),17(2),18(2),12(3),20(1),8(1),16(1)
15. 7(1),11(2),3(1),13(2),14(2),20(1),16(5)
16. 7(1),3(3),13(2),14(2),20(1),24(2),16(3)
17. 7(1),3(3),13(2),14(1),6(1),20(1),24(3),16(2)
18. 7(1),3(3),13(2),6(1),20(1),24(4),16(1)
19. 7(1),3(3),13(1),5(1),14(1),6(1),20(1),24(4),16(1)
20. 7(1),3(3),13(1),21(1),6(1),12(1),24(4),16(1)

21. 7(1),11(2),19(1),5(2),6(1),18(1),4(2),24(4)
22. 7(1),3(3),13(2),14(1),10(1),20(2),24(2),16(2)
23. 7(1),3(3),13(2),14(1),2(1),20(2),24(3),16(1)
24. 7(1),3(3),13(2),6(1),10(1),20(2),24(3),16(1)
25. 7(1),3(3),13(2),6(1),2(1),20(2),24(4)
26. 7(1),3(3),13(1),5(1),14(1),10(1),20(2),24(3),16(1)
27. 7(1),3(3),13(1),5(1),14(1),2(1),20(2),24(4)
28. 7(1),3(3),13(1),5(1),6(1),10(1),20(2),24(4)
29. 7(1),3(3),5(2),14(1),10(1),20(2),24(4)
30. 7(1),3(3),13(2),10(2),20(3),24(2),16(1)
31. 7(1),3(3),13(2),10(1),2(1),20(3),24(3)
32. 7(1),3(3),13(1),5(1),10(2),20(3),24(3)
33. 7(1),3(3),13(1),9(1),14(1),10(1),20(3),24(2),16(1)
34. 7(1),3(3),13(1),9(1),14(1),2(1),20(3),24(3)
35. 7(1),3(3),13(1),9(1),6(1),10(1),20(3),24(3)
36. 7(1),3(3),13(1),17(1),6(1),10(1),12(1),20(2),24(3)
37. 7(1),11(3),5(1),17(1),18(2),12(3),20(1),16(2)
38. 7(1),3(3),13(1),9(1),10(2),20(4),24(2)
39. 7(1),3(3),13(1),17(1),10(2),12(1),20(3),24(2)
40. 7(1),11(3),17(2),18(2),12(3),20(1),4(1),16(1)
41. 7(1),11(2),19(1),17(2),18(2),12(4),4(1),16(1)
42. 7(1),11(2),3(1),17(2),18(2),12(4),20(1),16(1)
43. 3(4),13(2),14(2),20(2),24(2),16(2)
44. 3(4),13(2),14(1),6(1),20(2),24(3),16(1)
45. 3(4),13(2),6(1),20(2),24(4)

46. 3(4),13(1),5(1),14(1),6(1),28(1),20(1),24(3),16(1)
47. 3(4),13(1),5(1),14(1),6(1),20(2),24(4)
48. 3(4),13(2),14(1),18(1),20(3),24(2),8(1)
49. 3(4),13(2),14(1),2(1),20(3),24(3)
50. 3(4),13(2),6(1),10(1),28(1),20(2),24(2),16(1)
51. 3(4),13(2),6(1),10(1),20(3),24(3)
52. 3(4),13(2),6(1),18(1),12(1),20(2),24(3)
53. 3(4),13(1),5(1),14(1),10(1),28(1),20(2),24(2),16(1)
54. 3(4),13(1),5(1),14(1),10(1),20(3),24(3)
55. 3(4),13(1),21(1),6(1),10(1),12(1),20(2),24(3)
56. 3(4),13(1),5(1),6(1),10(1),28(1),20(2),24(3)
57. 3(4),13(1),17(1),6(1),10(1),28(1),12(1),20(2),24(2)
58. 3(4),5(1),25(1),26(1),10(1),12(2),20(3),24(1)
59. 7(3),9(3),10(1),18(2),20(3),24(1),8(1)
60. 7(3),9(1),17(2),10(2),18(1),12(1),20(2),24(1),8(1)
61. 7(3),9(1),17(2),10(1),18(2),12(3),24(1),16(1)
62. 7(2),11(1),9(1),17(2),10(2),18(1),12(2),20(2),16(1)
63. 7(2),11(1),9(1),17(2),10(1),18(2),12(2),20(2),8(1)
64. 7(2),3(1),9(2),17(1),10(1),18(2),12(2),20(2),24(1)
65. 7(1),3(2),13(1),17(2),6(1),10(2),12(1),20(2),24(2)
66. 3(3),13(1),5(2),14(1),10(1),18(1),20(2),24(3)
67. 3(3),13(1),21(1),5(1),6(1),10(2),20(2),24(3)
68. 3(3),13(1),21(2),6(1),10(1),2(1),12(1),20(1),24(3)
69. 3(3),13(2),5(1),10(2),18(1),20(3),24(2)
70. 3(3),13(2),17(1),14(1),6(1),10(1),20(2),24(2),16(1)

71. 3(3),13(2),17(1),14(1),6(1),2(1),20(2),24(3)
72. 3(3),13(2),1(1),14(1),6(1),18(1),20(2),24(3)
73. 3(3),13(2),17(1),6(1),10(1),28(1),20(1),24(2),16(1)
74. 3(3),13(2),17(1),6(1),10(1),20(2),24(3)
75. 3(3),13(2),17(1),6(1),18(1),12(1),20(1),24(3)
76. 3(3),13(1),5(1),9(1),14(1),6(1),18(1),20(2),24(3)
77. 3(3),13(1),5(1),17(1),14(2),10(1),20(2),24(2),16(1)
78. 3(3),5(2),9(1),14(2),26(1),20(2),24(2),16(1)
79. 3(3),13(2),17(1),6(1),10(1),18(1),12(1),20(2),24(2)
80. 3(3),13(2),17(1),6(1),10(1),2(1),28(1),20(2),24(2)
81. 3(3),13(1),5(1),9(1),14(1),18(2),12(1),20(2),24(2)
82. 3(3),13(1),21(1),9(1),6(1),18(2),12(2),20(1),24(2)
83. 3(3),13(1),5(1),17(1),6(1),10(2),28(1),20(2),24(2)
84. 3(3),5(2),25(1),10(2),18(1),28(1),12(1),20(2),24(1)
85. 3(3),5(1),9(2),14(1),26(2),20(3),4(1),24(1)
86. 3(3),5(1),25(2),26(1),10(2),12(1),20(3),4(1)
87. 7(2),9(2),17(2),10(2),18(2),12(2),20(2)

Total number of lotto designs with isolated blocks = 2
Total number of lotto designs = 89

$t = \mathbf{6}$

(14 6 12 6; 5)

1. 3(5),13(1),14(1),12(1),20(3),24(3)
2. 3(4),13(2),14(2),20(2),24(2),16(2)
3. 3(4),13(2),14(1),6(1),20(2),24(3),16(1)
4. 3(4),13(2),6(1),20(2),24(4)

5. 3(4),13(1),5(1),14(1),6(1),20(2),24(4)
6. 3(4),13(2),14(1),2(1),20(3),24(3)
7. 3(4),13(2),6(1),10(1),20(3),24(3)
8. 3(4),13(1),5(1),14(1),10(1),20(3),24(3)
9. 3(3),13(2),17(1),6(1),10(1),20(2),24(3)
10. 7(2),9(2),17(2),10(2),18(2),12(2),20(2)

Total number of lotto designs with isolated block = 1
Total number of lotto designs = 11

(16 7 12 6; 5)

1. 7(1),3(5),13(1),14(1),12(1),20(3),24(4)
2. 3(6),13(1),14(1),28(1),12(1),20(3),24(3)
3. 7(2),11(3),13(2),14(2),20(1),16(6)
4. 7(2),11(3),17(2),18(2),12(4),20(1),16(2)
5. 7(2),11(2),19(1),17(2),18(2),12(5),16(2)
6. 7(1),3(4),13(2),14(2),20(2),24(3),16(2)
7. 7(1),3(4),13(2),14(1),6(1),20(2),24(4),16(1)
8. 7(1),3(4),13(2),6(1),20(2),24(5)
9. 7(1),3(4),13(1),5(1),14(1),6(1),20(2),24(5)
10. 7(1),3(4),13(1),21(1),6(1),12(1),20(1),24(5)
11. 7(1),3(4),13(2),14(1),10(1),20(3),24(3),16(1)
12. 7(1),3(4),13(2),14(1),2(1),20(3),24(4)
13. 7(1),3(4),13(2),6(1),10(1),20(3),24(4)
14. 7(1),3(4),13(1),5(1),14(1),10(1),20(3),24(4)
15. 7(1),3(4),13(2),10(2),20(4),24(3)
16. 7(1),3(4),13(1),9(1),14(1),10(1),20(4),24(3)

17. 3(5),13(2),14(2),20(3),24(3),16(1)
18. 3(5),13(2),14(1),6(1),28(1),20(2),24(3),16(1)
19. 3(5),13(2),14(1),6(1),20(3),24(4)
20. 3(5),13(2),14(1),18(1),12(1),20(3),24(3)
21. 3(5),13(2),14(1),2(1),28(1),20(3),24(3)
22. 3(5),13(2),6(1),10(1),28(1),20(3),24(3)
23. 3(5),13(1),5(1),14(1),10(1),28(1),20(3),24(3)
24. 7(1),3(3),13(2),17(1),6(1),10(1),20(2),24(4)
25. 7(1),3(3),13(2),17(1),6(1),10(2),20(3),24(3)
26. 7(1),3(3),13(2),17(1),10(3),20(4),24(2)
27. 7(1),11(3),5(1),17(2),18(3),12(4),20(1),16(1)
28. 7(1),11(3),17(3),18(3),12(4),20(1),4(1)
29. 3(4),13(3),6(1),18(1),20(2),24(4)
30. 3(4),13(2),5(1),14(2),18(1),20(2),24(3),16(1)
31. 3(4),13(2),5(1),14(1),6(1),18(1),20(2),24(4)
32. 3(4),13(2),21(1),6(1),10(1),20(2),24(4)
33. 3(4),13(3),6(1),10(1),18(1),20(3),24(3)
34. 3(4),13(2),5(1),14(1),10(1),18(1),20(3),24(3)
35. 3(4),13(2),17(1),14(2),2(1),20(3),24(3)
36. 3(4),13(2),17(1),14(1),6(1),10(1),20(3),24(3)
37. 7(3),9(3),17(1),10(2),18(2),12(1),20(3),24(1)
38. 7(3),9(2),17(2),10(2),18(2),12(2),20(2),24(1)
39. 3(3),13(2),21(1),17(1),6(1),10(2),20(2),24(3)
40. 3(3),13(2),17(2),6(1),10(2),28(1),20(2),24(2)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 41

$t = \mathbf{7}$

(12, 10, 8, 7; 5)

1. 31(3),15(2),23(2),27(2),29(1),30(1),28(1)
2. 31(2),15(2),23(2),27(2),29(2),30(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(13, 9, 9, 7; 5)

1. 15(4),23(4),25(1),26(1),28(1),24(2)
2. 15(4),23(3),19(1),25(1),26(1),28(2),24(1)
3. 15(4),23(2),19(2),25(1),26(1),28(3)
4. 15(3),23(3),11(1),19(1),25(1),26(1),28(3)
5. 15(4),23(3),25(2),26(2),28(1),20(1)
6. 15(3),23(3),7(1),25(2),26(2),28(2)
7. 15(4),23(2),19(1),21(1),25(1),26(2),28(2)
8. 15(3),23(3),11(1),21(1),25(1),26(2),28(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 8

(16 7 14 7; 5)

1. 3(5),13(2),14(2),20(3),24(3),16(1)
2. 3(5),13(2),14(1),6(1),20(3),24(4)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(18 8 14 7; 5)

1. 7(1),3(5),13(2),14(2),20(3),24(4),16(1)

2. 7(1),3(5),13(2),14(1),6(1),20(3),24(5)
3. 7(1),3(5),13(2),14(1),10(1),20(4),24(4)
4. 3(6),13(2),14(2),28(1),20(3),24(3),16(1)
5. 3(6),13(2),14(2),20(4),24(4)
6. 3(6),13(2),14(1),6(1),28(1),20(3),24(4)
7. 7(2),11(3),17(3),18(3),12(5),20(1),16(1)
8. 3(5),13(3),14(1),6(1),18(1),20(3),24(4)
9. 3(5),13(2),5(1),14(2),18(1),20(3),24(4)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 9

(20 9 14 7; 5)

1. 7(2),3(5),13(2),14(2),20(3),24(5),16(1)
2. 7(2),3(5),13(2),14(1),6(1),20(3),24(6)
3. 7(2),3(5),13(2),14(1),10(1),20(4),24(5)
4. 15(1),3(6),13(2),14(2),20(4),24(4),16(1)
5. 15(1),3(6),13(2),14(1),6(2),20(4),24(5)
6. 7(1),11(1),3(5),13(2),14(2),20(4),24(4),16(1)
7. 7(1),11(1),3(5),13(2),14(1),6(1),20(4),24(5)
8. 7(1),3(6),13(2),14(2),28(1),20(3),24(4),16(1)
9. 7(1),3(6),13(2),14(2),20(4),24(5)
10. 7(1),3(6),13(2),14(1),6(1),28(1),20(3),24(5)
11. 7(1),3(6),13(2),14(1),10(1),28(1),20(4),24(4)
12. 3(7),13(2),14(2),28(2),20(3),24(3),16(1),
13. 3(7),13(2),14(2),28(1),20(4),24(4)
14. 3(7),13(2),14(1),6(1),28(2),20(3),24(4)

15. 7(3),11(3),17(3),18(3),12(5),20(1),24(1),16(1)
16. 7(3),11(2),19(1),17(3),18(3),12(6),24(1),16(1)
17. 7(2),11(4),17(3),18(3),12(5),20(2),16(1)
18. 7(1),3(5),13(3),14(2),18(1),20(3),24(4),16(1)
19. 7(1),3(5),13(3),14(1),6(1),18(1),20(3),24(5)
20. 7(1),3(5),13(2),5(1),14(2),18(1),20(3),24(5)
21. 7(1),3(5),13(3),14(1),10(1),18(1),20(4),24(4)
22. 7(1),3(5),13(2),9(1),14(2),18(1),20(4),24(4)
23. 3(6),13(3),14(1),22(1),6(1),20(3),24(5)
24. 3(6),13(2),21(1),14(3),20(3),24(4),16(1),
25. 3(6),13(2),21(1),14(2),6(1),20(3),24(5)
26. 3(6),13(3),14(2),18(1),20(4),24(4)
27. 3(6),13(3),14(1),22(1),10(1),20(4),24(4)
28. 3(6),13(3),14(1),6(1),18(1),28(1),20(3),24(4)
29. 3(6),13(2),21(1),14(2),10(1),20(4),24(4)
30. 3(6),13(2),5(1),14(2),18(1),28(1),20(3),24(4)
31. 7(2),11(3),5(1),17(3),18(4),12(5),20(1),24(1)
32. 7(2),11(3),9(1),17(3),18(4),12(5),20(2)
33. 7(1),11(4),5(1),17(3),18(4),12(5),20(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 33

$t = \mathbf{8}$

(14 10 10 8; 5)

1. 15(4),23(4),25(2),26(2),28(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(15 12 9 8; 5)

1. 15(3),23(3),27(3),29(3),30(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(18 8 16 8; 5)

1. 3(6),13(2),14(2),20(4),24(4)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(18 13 10 8; 5)

1. 15(5),23(1),27(5),21(2),22(2),28(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(19 8 17 8; 5)

1. 3(7),13(1),14(1),12(2),20(4),24(4)
2. 7(3),11(3),13(2),14(2),12(1),16(8)
3. 7(3),11(3),17(2),18(2),12(5),16(4)
4. 3(6),13(2),14(2),12(1),20(3),24(3),16(2)
5. 3(6),13(2),14(2),20(4),24(4),0(1)
6. 3(6),13(2),14(2),20(4),24(3),8(1),16(1)
7. 3(6),13(2),14(1),6(1),12(1),20(3),24(4),16(1)
8. 3(6),13(2),14(1),6(1),20(4),24(4),8(1)
9. 3(6),13(2),14(1),6(1),20(3),4(1),24(5)
10. 3(6),13(2),6(1),12(1),20(3),24(5)
11. 3(6),13(1),5(1),14(1),6(1),12(1),20(3),24(5)

12. 3(6),13(1),21(1),6(1),12(2),20(2),24(5)
13. 3(6),13(2),14(1),2(1),12(1),20(4),24(4)
14. 3(6),13(2),6(1),10(1),12(1),20(4),24(4)
15. 3(6),13(1),5(1),14(1),10(1),12(1),20(4),24(4)
16. 7(3),11(2),17(3),18(3),12(5),8(1),16(2)
17. 7(2),11(3),5(1),17(2),18(3),12(5),16(3)
18. 7(2),11(3),17(3),18(3),12(5),4(1),16(2)
19. 3(5),13(3),14(1),6(1),20(2),24(4),16(2)
20. 3(5),13(3),6(1),20(2),24(5),16(1)
21. 3(5),13(2),5(1),14(2),6(1),20(2),24(4),16(2)
22. 3(5),13(2),5(1),14(1),6(1),20(2),24(5),16(1)
23. 3(5),13(2),21(1),6(1),12(1),20(1),24(5),16(1)
24. 3(5),13(2),21(1),6(1),20(2),24(5),8(1)
25. 3(5),13(3),14(1),6(1),10(1),20(3),24(3),16(2)
26. 26. 3(5),13(3),14(1),6(1),2(1),20(3),24(4),16(1)
27. 27. 3(5),13(2),5(1),14(2),2(1),20(3),24(4),16(1)
28. 3(5),13(2),5(1),14(1),6(1),10(1),20(3),24(4),16(1)
29. 3(5),13(2),5(1),6(1),10(1),20(3),24(5)
30. 3(5),13(1),5(2),14(2),10(1),20(3),24(4),16(1)
31. 3(5),13(1),5(2),14(1),6(1),10(1),20(3),24(5)
32. 3(5),5(3),14(2),10(1),20(3),24(5)
33. 3(5),13(3),14(1),2(2),20(4),24(4)
34. 3(5),13(3),6(1),10(1),2(1),20(4),24(4)
35. 3(5),13(2),5(1),14(1),10(1),2(1),20(4),24(4)
36. 3(5),13(2),5(1),6(1),10(2),20(4),24(4)

37. 3(5),13(1),5(2),14(1),10(2),20(4),24(4)
38. 3(5),13(2),1(1),14(2),2(1),20(4),24(4)
39. 3(5),13(2),1(1),14(1),6(1),10(1),20(4),24(4)
40. 3(5),13(2),17(1),6(1),10(1),12(1),20(3),24(4)
41. 3(5),13(1),5(1),9(1),14(1),6(1),10(1),20(4),24(4)
42. 3(5),13(2),17(1),6(1),10(2),12(1),20(4),24(3)
43. 3(4),13(3),17(1),6(1),10(1),20(2),24(4),16(1)
44. 3(4),13(3),17(1),6(1),10(2),20(3),24(3),16(1)
45. 3(4),13(2),5(1),17(1),6(1),10(2),20(3),24(4)
46. 3(4),13(2),5(1),17(1),6(1),10(3),20(4),24(3)
47. 3(4),13(2),17(2),6(1),10(2),12(1),20(3),24(3)
48. 7(3),9(3),17(2),10(3),18(2),12(2),20(3),16(1)
49. 3(3),13(2),5(1),17(2),6(1),10(3),20(3),24(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 49

(20 9 16 8; 5)

1. 7(1),3(6),13(2),14(2),20(4),24(5)
2. 3(7),13(2),14(2),28(1),20(4),24(4)
3. 3(6),13(3),14(2),18(1),20(4),24(4)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 3

$t = \mathbf{9}$

(16 10 13 9; 5)

1. 7(2),27(6),13(2),14(2),20(4)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(16, 11, 12, 9; 5)

1. 15(5),23(5),25(1),26(1),28(1),24(3)
2. 15(4),7(1),27(5),21(1),22(1),28(2),20(2)
3. 15(4),7(1),27(4),19(1),21(1),22(1),28(3),20(1)
4. 15(4),7(1),27(3),19(2),21(1),22(1),28(4)
5. 15(3),7(2),27(5),21(1),22(1),28(3),20(1)
6. 15(2),7(3),27(5),21(1),22(1),28(4)
7. 15(5),23(4),25(2),26(2),28(2),16(1)
8. 15(5),23(4),25(2),26(2),28(1),20(1),24(1)
9. 15(4),23(4),7(1),25(2),26(2),28(2),24(1)
10. 15(4),23(4),3(1),25(2),26(2),28(3)
11. 15(4),7(1),27(4),21(2),22(2),28(2),24(1)
12. 15(4),7(1),27(4),21(2),22(1),26(1),28(2),20(1)
13. 15(4),7(1),27(4),21(2),22(1),18(1),28(3)
14. 15(4),7(1),27(3),19(1),21(2),22(1),26(1),28(3)
15. 15(3),7(2),27(4),21(2),22(1),26(1),28(3)
16. 15(3),7(1),27(5),13(1),21(1),22(2),28(2),20(1)
17. 15(3),7(1),27(5),21(2),22(2),28(2),12(1)
18. 15(3),7(1),27(5),21(2),22(1),6(4),28(3)
19. 15(3),7(1),27(4),11(1),21(2),22(2),28(3)
20. 15(2),7(2),27(5),13(1),21(1),22(2),28(3)
21. 15(5),23(2),19(1),21(1),25(2),22(1),26(2),28(2)
22. 15(4),23(3),11(1),21(1),25(2),22(1),26(2),28(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 22

For $L(n, k, p, t) = 6$

$t = 3$

(6, 3, 4, 3; 6)

1. 7(1),11(1),49(1),50(1),28(1),44(1)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(8, 4, 4, 3; 6)

1. 7(2),11(1),49(1),50(1),28(1),44(1),56(1)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(10, 3, 7, 3; 6)

1. 3(2),5(1),6(1),4(1),24(1),40(1),8(1),48(2)

Total number of lotto designs with isolated blocks = 4
Total number of lotto designs = 5

(10, 7, 3, 3; 6)

1. 31(3),47(3),49(1),50(1),52(1),56(1)
2. 31(1),15(1),23(1),39(2),59(1),57(1),58(1),60(2)
3. 15(2),55(3),59(1),57(1),58(1),28(1),44(1)
4. 15(2),55(3),27(1),57(1),42(1),60(2)
5. 31(3),39(1),43(1),45(1),49(1),54(1),58(1),60(1)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 5

(12, 5, 5, 3; 6)

1. 7(4),25(1),26(1),28(1),40(2),48(2),32(1)
2. 7(4),25(1),26(1),12(1),40(2),48(3)
3. 7(3),11(1),49(1),50(1),20(2),24(1),40(3)

4. 7(3),11(1),49(1),50(1),20(1),36(1),24(2),40(2)
5. 15(2),19(2),33(1),34(1),20(3),40(3)
6. 7(2),11(2),49(1),50(1),20(3),40(3)
7. 7(2),11(2),49(1),50(1),20(2),36(1),24(1),40(2)
8. 3(3),29(2),14(1),22(1),36(1),40(2),48(2)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 9

(12, 6, 4, 3; 6)

1. 7(2),11(2),49(2),50(2),28(2),44(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(12, 8, 3, 3; 6)

1. 15(2),23(2),39(2),57(2),58(2),60(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(13, 4, 7, 3; 6)

1. 3(3),13(1),14(1),20(2),24(1),40(1),16(1),32(3)
2. 3(3),13(1),6(1),20(2),24(2),40(1),32(3)
3. 3(3),13(1),6(1),20(1),36(1),24(3),32(3)
4. 3(2),5(2),14(1),18(1),20(1),24(2),40(1),32(3)
5. 3(2),13(1),17(1),6(1),10(1),20(1),36(1),24(2),32(3)

Total number of lotto designs with isolated blocks = 12
Total number of lotto designs = 17

(13, 9, 3, 3; 6)

1. 31(4),39(2),43(2),49(1),50(1),60(3)
2. 15(2),23(2),39(2),59(2),57(1),58(1),60(3)

3. 63(1),15(2),23(2),39(2),57(2),58(2),60(2)

4. 31(1),47(1),15(1),23(2),39(2),57(2),58(2),60(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 4

(14, 3, 10, 3; 6)

1. 3(2),5(1),6(1),4(1),8(3),16(3),32(3)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 1

(16, 11, 3, 3; 6)

1. 31(1),15(2),55(1),27(2),43(3),53(2),54(2),60(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

$t = 4$

(13, 4, 10, 4; 6)

1. 3(3),5(1),6(1),4(2),24(2),40(2),48(2)

2. 3(2),5(2),6(2),24(2),40(2),48(2),0(1)

3. 3(2),5(2),6(2),24(2),40(2),48(1),16(1),32(1)

Total number of lotto designs with isolated blocks = 1
Total number of lotto designs = 4

(14, 6, 7, 4; 6)

1. 7(5),25(1),26(1),28(1),40(3),48(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(16, 5, 10, 4; 6)

1. 3(4),13(1),14(1),20(3),24(2),40(1),32(4)

Total number of lotto designs with isolated blocks = 3
Total number of lotto designs = 4

$t = \mathbf{5}$

(7, 6, 5, 5; 6)

1. 63(1),31(1),47(1),55(1),59(1),61(1),62(1)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(10, 5, 8, 5; 6)

1. 7(2),11(1),25(1),49(1),50(2),28(1),44(2)
2. 7(1),11(1),21(1),41(1),49(1),38(1),26(1),50(1),28(1),44(1)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(11, 10, 5, 5; 6)

1. 63(5),31(1),47(1),55(1),59(1),61(1),62(1)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(12, 7, 7, 5; 6)

1. 15(3),23(1),51(2),37(1),38(1),28(1),56(3)
2. 15(3),51(3),53(1),54(1),28(2),40(2)
3. 15(3),51(3),21(1),22(1),44(2),56(2)
4. 15(2),23(1),43(1),51(2),37(1),38(1),28(2),56(2)
5. 15(1),23(1),43(2),51(2),37(1),38(1),28(3),56(1)
6. 7(1),59(5),13(1),14(1),20(2),36(2)
7. 7(1),59(1),27(2),43(2),13(1),14(1),52(4)
8. 7(1),27(3),43(2),45(1),14(1),52(4)
9. 15(2),23(2),27(1),41(1),49(1),34(2),60(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 9

(14, 12, 5, 5; 6)

1. 63(2),31(2),47(2),55(2),59(2),61(2),62(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(16, 5, 13, 5; 6)

1. 3(3),5(2),6(2),4(1),24(3),40(2),48(2),32(1)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(16, 11, 6, 5; 6)

1. 15(3),23(2),59(5),37(1),38(1),60(3),52(1)
2. 15(3),23(2),59(4),51(1),37(1),38(1),60(4)
3. 15(2),23(2),7(1),59(5),37(1),38(1),60(4)
4. 15(3),23(1),59(5),53(2),54(2),28(1),44(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 4

(16, 14, 5, 5; 6)

1. 63(4),31(2),47(2),55(2),59(2),61(2),62(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

$t = \mathbf{6}$

(11, 7, 8, 6; 6)

1. 15(2),23(2),39(2),57(1),58(1),60(1),56(2)
2. 31(4),39(1),43(1),45(1),46(1),48(3)
3. 31(4),39(1),43(1),49(1),50(1),44(2),48(1)

4. 15(2),23(2),39(1),35(1),57(1),58(1),60(2),56(1)
5. 31(4),35(2),45(1),38(1),52(1),56(2)
6. 31(3),15(1),35(2),53(1),54(1),44(1),56(2)
7. 15(2),23(2),35(2),57(1),58(1),60(3)
8. 15(1),23(1),39(2),11(1),19(1),57(1),58(1),60(3)
9. 31(4),39(1),41(1),49(1),42(1),50(1),44(1),52(1)
10. 15(2),23(2),39(1),57(2),58(1),50(1),60(1),44(1)
11. 15(2),23(2),39(1),57(2),58(1),34(1),60(2)
12. 15(2),23(2),39(1),57(2),42(1),50(1),60(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 12

(12, 6, 10, 6; 6)

1. 7(2),11(2),49(2),50(2),28(2),44(2)
2. 7(3),25(3),42(3),52(3)
3. 7(3),25(2),41(1),26(1),50(2),44(2),52(1)
4. 7(3),25(2),41(1),42(1),50(2),28(1),44(1),52(1)
5. 7(3),25(2),41(1),50(3),28(1),44(2)
6. 7(3),25(1),41(1),49(1),26(1),42(1),50(1),28(1),44(1),52(1)
7. 7(2),11(1),21(1),41(1),49(1),26(1),50(2),28(1),44(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 7

(15, 11, 7, 6; 6)

1. 31(4),47(4),55(1),51(1),53(1),54(1),56(3)
2. 31(4),47(4),51(2),53(1),54(1),60(1),56(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

$t = \mathbf{7}$

(13, 10, 8, 7; 6)

1. 31(3),47(3),55(3),57(1),58(1),60(1),56(1)
2. 31(3),47(3),55(2),51(1),57(1),58(1),60(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(14, 7, 12, 7; 6)

1. 7(2),11(2),21(1),25(1),49(1),50(3),28(1),44(3)
2. 7(2),11(2),21(1),41(1),49(1),50(3),28(2),44(2)
3. 7(2),11(2),21(1),49(2),26(1),50(2),28(1),44(3)
4. 7(1),11(3),21(2),49(1),50(3),28(1),44(3)
5. 7(1),11(3),21(1),37(1),49(1),50(3),28(2),44(2)
6. 7(1),11(3),21(1),49(2),22(1),50(2),28(1),44(3)
7. 7(1),11(3),21(1),49(2),38(1),50(2),28(2),44(2)
8. 7(2),11(1),21(1),41(2),49(1),38(1),26(1),50(2),28(2),44(1)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 8

$t = \mathbf{8}$

(14, 9, 11, 8; 6)

1. 7(3),59(5),29(1),30(1),44(2),52(2)
2. 15(3),23(2),39(2),57(2),58(2),60(2),48(1)
3. 15(2),23(2),39(3),57(2),58(2),60(1),28(1),56(1)
4. 15(2),23(2),39(2),7(1),57(2),58(2),60(2),56(1)
5. 15(2),7(1),27(2),43(2),53(2),54(2),60(2),56(1)
6. 15(2),7(1),27(2),43(2),53(2),54(1),50(1),60(3)

7. 15(1),7(1),27(3),43(2),53(2),54(1),38(1),60(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 7

t = 9

(12, 10, 10, 9; 6)

1. 63(1),31(2),47(2),55(2),59(2),61(1),62(1),60(1)
2. 31(2),47(2),55(2),59(2),61(2),62(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(15, 13, 10, 9; 6)

1. 63(4),31(2),47(2),55(2),59(2),61(1),62(1),60(1)
2. 63(3),31(2),47(2),55(2),59(2),61(2),62(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(17, 15, 10, 9; 6)

1. 63(6),31(2),47(2),55(2),59(2),61(1),62(1),60(1)
2. 63(5),31(2),47(2),55(2),59(2),61(2),62(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

(19 16 10 9; 6)

1. 63(2),31(3),47(3),55(3),59(3),61(2),62(2),60(1)
2. 63(1),31(3),47(3),55(3),59(3),61(3),62(3)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 2

t **= 10**

(13, 11, 11, 10; 6)

1. 63(1),31(2),47(2),55(2),59(2),61(2),62(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

(14, 12, 11, 10; 6)

1. 63(2),31(2),47(2),55(2),59(2),61(2),62(2)

Total number of lotto designs with isolated blocks = 0
Total number of lotto designs = 1

Chapter 5

Conclusion

5.1 Summary

Lotto designs are relatively new in the field of combinatorics. Several researchers have worked on (n, k, p, t) lotto designs to compute upper and lower bounds of $L(n, k, p, t)$. In this thesis, we develop an improved algorithm to compute the number of non-isomorphic optimal lotto designs on 5 or 6 blocks for $n, k, p, t \leq 20$ or if this is impossible to determine, we compute if there is a lotto design on 6 blocks for these parameters. Our algorithm uses block intersection properties to generate all sets of non-isomorphic three blocks. We then generate the rest of the blocks using an exhaustive backtracking algorithm. We prune the search tree by using trivial isomorphism rejection in the algorithm. We further optimize our algorithm by using the fact that there exists a minimal lotto design in which every element occurs at least once. The algorithm may still generate isomorphic lotto designs. So we use Kocay's program to get all non-isomorphic optimal lotto designs for a particular set of parameters n, k, p and t. We find that this improved algorithm is substantially faster than a straightforward backtracking algorithm. This algorithm also finds results that A.P.

Burger et al. could not do. With this algorithm, we verified $\eta(n, k, p, t)$ for 14 lottery numbers of A.P Burger et al. and generated $\eta(n, k, p, t)$ for 112 new lottery numbers and improved lower bounds of $L(n, k, p, t)$ for 18 lottery numbers.

5.2 Future Work

Most of the backtracking algorithms developed for lotto designs are sequential. These algorithms require much search time to traverse the state space tree or search tree. The search tree grows exponentially for larger values of the parameters n, k, p, t. Hence, the program that implements these sequential backtracking algorithms may not compute the upper bounds of $L(n, k, p, t)$ in a reasonable time. By parallelizing the backtracking algorithms, it may be possible to compute the upper bounds of $L(n, k, p, t)$ for slightly larger value of n, k, p and t than is possible for sequential algorithms.

Bibliography

[1] J. A. Bate. *A Generalized Covering Problem*. PhD thesis, University of Manitoba, 1978.

[2] J. A. Bate and G. H. J. van Rees. Lotto designs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 288:15–39, 1998.

[3] Riccardo Bertolo, Iliya Bluskov, and Heikki Hämäläinen. Upper bounds on the general covering number $C_\lambda(v, k, t, m)$. *Journal of Combinatorial Designs*, 12(5):362–380, 2004.

[4] A.E Brouwer and M. Voorhoeve. Turán theory and the lotto problem. *Mathematical Centrum Tracts*, 106:99–105, 1995.

[5] A.P. Burger, W.R. Grundlingh, and J.H. van Vuuren. The lottery problem. Ars Combinatoria (submitted).

[6] A.P. Burger, W.R. Grundlingh, and J.H. van Vuuren. On the optimality of belic's lottery design listings. Journal of Combinatorial Math and Combinatorial Computing (submitted).

[7] Colbourn C.J. *The CRC Handbook of Combinatorial Designs*, chapter Winning the lottery, pages 578–584. CRC Press, 1996.

[8] D. de Caen. Extension of a theorem of moon and moser on complete subgraphs. *Ars Combin.*, 16:5–10, 1983.

[9] Z. Füredi, G. Székely, and Z. Zubor. On the lottery problem. *Journal of Combinatorial Designs*, 4(1):5–10, 1996.

[10] Dan Gordon. La Jolla covering repository. Available from http://www.ccrwest.org, October 2004.

[11] Daniel M. Gordon, Greg Kuperberg, and Oren Patashnik. New constructions for covering designs. *Journal of Combinatorial Designs*, 3:269–284, 1995.

[12] H. Hanani, D. Ornstein, and V.T. Sós. On the lottery problem. *Magyar Tud. Acad. Mat. Kutató*, pages 155–158, 1964.

[13] W.L. Kocay. Abstract data types and graph isomorphism. *J. of Comb. Info. and S.S.*, 9(4):247–259, 1984.

[14] P. C. Li. *Some Results on Lotto Design.* PhD thesis, University of Manitoba, 1999.

[15] P. C. Li. Lotto tables. Available from www.cs.umanitoba.ca/∼ lipakc/lottotables.html, December 2004.

[16] P. C. Li and G. H. J. van Rees. Lotto design tables. *Journal of Combinatorial Designs*, 10(5):335–359, 2002.

[17] P. C. Li and G. H. J. van Rees. New results on lotto designs. Preprint, March 2004.

[18] K.J. Nurmela and P.R.J. Östergard. Upper bounds for covering designs by simmulated annealing. *Congr. Numer.*, 96:93–111, 1993.

[19] Belic R. Lottery systems and toto systems to win wheel game. Available from http://www.xs4all.nl/∼rbelic/, October 2003.