The Eighth International Symposium on Operations Research and Its Applications (ISORA'09)
Zhangjiajie, China, September 20–22, 2009

A Tabu Search Algorithm to Construct BIBDs Using MIP Solvers

Daisuke Yokoya Takeo Yamada

Department of Computer Science, The National Defense Academy, Yokosuka 239-8686, Japan

Abstract BIBD (balanced incomplete block design) is instrumental in design of experiments. This is usually constructed using algebraic tools such as finite projective (or affine) algebra or difference sets. Recently, heuristic algorithms have been tried, including artificial neural networks, simulated annealing, and local search method. In this paper, we present a novel approach to construct BIBDs that makes use of MIP (mixed integer programming) solvers. Based on this, tabu search algorithms is given, and this is compared against the local search method presented by a previous researcher.

Keywords BIBD (balanced incomplete block design); MIP (mixed integer programming problem); Tabu search

1 Introduction

Let V be a set of v elements called *points* and B be a collection (i.e., multi-set) of b non-empty subsets of V called *blocks*. (V,B) is a *balanced incomplete block design* (BIBD [3, 8, 12, 22]), if the followings conditions are satisfied.

1. Each point is contained in exactly r blocks.
2. Each block contains exactly k points.
3. Every pair of distinct points is contained in exactly λ blocks.

With positive integer parameters (v,b,r,k,λ), this is denoted as BIBD(v,b,r,k,λ). BIBD has its origin in statistical design of experiments [6], and recently it is also applied to coding theory [2, 7], failure diagnosis and group testing [10], and scheduling sports leagues [1] among others.

In matrix notation, BIBD is a $v \times b$ binary matrix $X = (x_{il})$ such that

$$\sum_{l=1}^{b} x_{il} = r \ , \ \ i = 1, \cdots, v, \tag{1}$$

$$\sum_{i=1}^{v} x_{il} = k \ , \ \ l = 1, \cdots, b, \tag{2}$$

$$\sum_{l=1}^{b} x_{il} x_{jl} = \lambda \ , \ \ i, j = 1, \cdots, v, \ i < j, \tag{3}$$

$$x_{il} \in \{0,1\} \ , i = 1, \cdots, v, \ l = 1, \cdots, b. \tag{4}$$

Form these relations, the followings are easily shown [12].

Proposition 1. *In* BIBD(v,b,r,k,λ)*, parameters satisfy*

$$vr = bk, \tag{5}$$

$$r(k-1) = \lambda(v-1). \tag{6}$$

Then, b and r are determined by (v,r,λ) as

$$b = \frac{v(v-1)}{k(k-1)}\lambda, \tag{7}$$

$$r = \frac{v-1}{k-1}\lambda. \tag{8}$$

Thus, in literature BIBD(v,b,r,k,λ) is often referred to as (v,k,λ)-BIBD, or alternatively 2-(v,k,λ) *design* [8, 25]. Given a set of integer parameters (v,b,r,k,λ) satisfying (5) and (6), we are concerned with the existence, and if one exists construction, of such a BIBD.

For the limited case of $k = 3$ or $k = 4$, Hanani [13, 14] proved the following.

Theorem 2. *Let v and* λ *be positive integers, and* $k = 3$ *or* $k = 4$*, Then,* BIBD(v,b,r,k,λ) *exists if and only if b and r determined by* (7) *and* (8) *are both integers.*

Triple system is the special case of BIBD with $k = 3$, and *Steiner triple system* is with $k = 3$ and $\lambda = 1$ [8]. These are denoted as TS(v,λ) and STS(v) respectively. For the Steiner triple system, the above result was shown by Kirkman [16] as early as in 1847, together with a construction method for STS(v). Hanani [14] proved similar results for $k = 5$ and $k = 6$ with some minor exceptions. However, except for these neither necessary and sufficient conditions nor construction methods of BIBDs are known in general.

Various approaches have been tried for the construction of BIBDs, including algebraic methods such as *finite projective* (or *affine*) *geometry* and *difference sets* [12, 22], as well as computational methods like *constraint programming* [20], *neural networks* [18] and *meta-heuristic* algorithms [5, 17]. Recently, by an exhaustive computer search over more than 90000 days in total Bilous et al. [4] proved that no $(22,8,4)$-BIBD exists. Similarly, Houghten et al. [15] announced non-existence of $(46,6,1)$-BIBD. On the other hand, Morales [19] discovered six new BIBDs by an elaborate tabu search [11]. Extensive list of BIBDs, as well as unknown BIBDs up to date, are available in the handbook by Colbourn and Dinitz [8].

The purpose of this paper is to present a novel approach for the construction of BIBDs that makes use of *mixed integer programming* (MIP) solvers [9]. In Section 2, we formulate the problem as a non-linear MIP problem, and present a approach to this problem through a repeated solution of linear problems. Based on this, we develop a tabu search algorithm in Section 3. We compare the developed algorithm against the local search method proposed by Prestwich [21] on the same 86 instances given in his paper. Out of 86 instances we solved 78, 23 more than the local search algorithm [21] was able to solve.

2 Mathematical programming approach

Finding a matrix satisfying (1) – (4) is a kind of *constraint satisfaction problem* which can be transformed into the following *nonlinear* 0-1 programming problem [23].

$$P: \quad \text{Maximize} \quad \sum_{i=1}^{v}\Big(\sum_{l=1}^{b} x_{il}\Big)+\sum_{l=1}^{b}\Big(\sum_{i=1}^{v} x_{il}\Big)+\sum_{i=1}^{v-1}\sum_{j=i+1}^{v}\Big(\sum_{l=1}^{b} x_{il}x_{jl}\Big) \tag{9}$$

$$\text{subject to} \quad \sum_{l=1}^{b} x_{il} \le r \ , \ i=1,\cdots,v, \tag{10}$$

$$\sum_{i=1}^{v} x_{il} \le k \ , \ l=1,\cdots,b, \tag{11}$$

$$\sum_{l=1}^{b} x_{il}x_{jl} \le \lambda \ , \ i,j=1,\cdots,v, \ i<j, \tag{12}$$

$$x_{il} \in \{0,1\}, \ i=1,\cdots,v, \ l=1,\cdots,b. \tag{13}$$

Note that equalities (1) – (3) are relaxed to inequalities (10) – (12), and the objective function is simply the sum of the left-hand sides of these inequalities. Then, the following is trivial.

Theorem 3.

1. P is always feasible.
2. For an arbitrary feasible solution $X=(x_{ij})$ to P, let $z(X)$ denote its objective value. Then, we have

$$z(X) \le vr+bk+\frac{v(v-1)}{2}\lambda. \tag{14}$$

3. If (14) is satisfied with equality, (10) – (12) are also satisfied with equality, and thus X gives a BIBD(v,b,r,k,λ).

Therefore, if equality holds in (14) with respect to an optimal solution X^* to P, we are done. Otherwise, if $z(X^*)<vr+bk+v(v-1)\lambda/2$ no BIBD exists. However, since P is a nonlinear 0-1 programming problems, it is difficult to get an optimal solution. Thus, we propose an *incremental* method that determines rows of X one by one.

Now, suppose that the first j rows of X is known as $\overline{X}_j=(\overline{x}_{il})$, $i=1,\cdots,j$. From (1) – (4), this is a binary matrix that satisfies the followings.

$$\sum_{l=1}^{b} \overline{x}_{il} = r \ , \ i=1,\cdots j, \tag{15}$$

$$\sum_{i=1}^{j} \overline{x}_{il} \le k \ , \ l=1,\cdots,b, \tag{16}$$

$$\sum_{l=1}^{b} \overline{x}_{il}\overline{x}_{i'l} = \lambda \ , \ i,i'=1,\cdots,j, \ i<i'. \tag{17}$$

Then, the $(j+1)$th row vector $x=(x_l)$ must satisfy

$$\sum_{l=1}^{b} x_l = r, \tag{18}$$

$$x_l \le k - \sum_{i=1}^{j} \overline{x}_{il}\ ,\ l=1,\cdots,b, \tag{19}$$

$$\sum_{l=1}^{b} \overline{x}_{il} x_l = \lambda\ ,\ i=1,\cdots,j. \tag{20}$$

If such an x is found, we can augment X_j to a $(j+1)\times b$ matrix

$$\overline{X}_{j+1} = \begin{pmatrix} \overline{X}_j \\ x \end{pmatrix}. \tag{21}$$

To obtain a binary vector x satisfying (18) – (20), we formulate the following optimization problem.

$$P_j(X_j): \qquad \text{Maximize} \quad \sum_{l=1}^{b} x_l + \sum_{l=1}^{b} \Big(\sum_{i=1}^{j} \overline{x}_{il} \Big) x_l \tag{22}$$

$$\text{subject to} \quad \sum_{l=1}^{b} x_l \le r, \tag{23}$$

$$x_l \le k - \sum_{i=1}^{j} \overline{x}_{il}\ ,\ l=1,\cdots,b, \tag{24}$$

$$\sum_{l=1}^{b} \overline{x}_{il} x_l \le \lambda\ ,\ i=1,\cdots,j, \tag{25}$$

$$x_l \in \{0,1\},\ l=1,\cdots b. \tag{26}$$

Contrary to nonlinear P, $P_j(X_j)$ is a *linear* 0-1 programming problem which can be solved (in many cases) using free or commercial MIP solvers. Let the optimal objective value to $P_j(X_j)$ be $z_j^*(X_j)$. Then, since the objective function (22) is the sum of the left-hand sides of (23) and (25), any optimal solution to $P_j(X_j)$ gives a binary x satisfying (18) – (20) if and only if

$$z_j^*(X_j) = r + j\lambda. \tag{27}$$

Without loss of generality the first and second rows of X_j can be assumed to be

$$X_2 = \begin{pmatrix} \overbrace{11\cdots11\cdots1}^{r} 0\cdots0\ 0\cdots0 \\ \underbrace{11\cdots1}_{\lambda} 0\cdots0 \underbrace{1\cdots1}_{r-\lambda} 0\cdots0 \end{pmatrix}, \tag{28}$$

and a BIBD is obtained if we can augment this through (21) to a $v\times b$ matrix X_v. On the other hand, if

$$z_j^*(X_j) < r + j\lambda \tag{29}$$

no BIBD exists as an extension of X_j. In this case, X_j may include a 'bad' row vector which should not be included in BIBD. Measures to cope with this problem will be discussed in the next section.

Example 1.
For BIBD$(12,22,11,6,5)$*, except for trivial inequalities and 0-1 conditions,* $P_2(X_2)$ *is*

$$\begin{aligned} &\text{Maximize} \quad (3333322222\ 2222222111\ 11)x \\ &\text{subject to} \quad \begin{pmatrix} 1111111111\ 1111111111\ 11 \\ 1111111111\ 1000000000\ 00 \\ 1111100000\ 0111111000\ 00 \end{pmatrix} x \leq \begin{pmatrix} 11 \\ 5 \\ 5 \end{pmatrix}. \end{aligned}$$

Solving this we have $x^* = (0111100010\ 0000100111\ 11)$ *with* $z_2^*(X_2) = 21 = r + \lambda j$*, and*

$$X_3 = \begin{pmatrix} 1111111111\ 1000000000\ 00 \\ 1111100000\ 0111111000\ 00 \\ 0111100010\ 0000100111\ 11 \end{pmatrix}.$$

Continuing this for $j = 3, \cdots, 8$*, we obtain*

$$X_8 = \begin{pmatrix} 1111111111\ 1000000000\ 00 \\ 1111100000\ 0111111000\ 00 \\ 0111100010\ 0000100111\ 11 \\ 0001100101\ 1011001100\ 11 \\ 0001111001\ 0110010111\ 00 \\ 1100010101\ 0100101110\ 01 \\ 1100001011\ 0001011101\ 10 \\ 1001001100\ 1001110011\ 01 \end{pmatrix}. \tag{30}$$

We stop here since $z_8^*(X_8) = 50 < r + \lambda j = 51$.

3 Tabu Search

If $z_j^*(X_j) < r + j\lambda$ our method stops, and to restart we have to get rid of a bad row from the current X. Here we make a guess of such a bad row vector as follows. When we solve $P_j(X_j)$ and obtain an optimal solution $x^* = (x_l^*)$ with $z_j^*(X_j) = r + j\lambda$, we move forward by adding x^* to X_j. On the other hand, if $z_j^*(X_j) < r + j\lambda$ we have either

$$\sum_{l=1}^{b} \overline{x}_{il}\, x_l^* < \lambda \tag{31}$$

for some row i, or

$$\sum_{l=1}^{b} x_l^* < r. \tag{32}$$

In the case of (31), we consider x_i as a bad row vector and eliminate it from X_j. If no such row exists, we have (32). Then, we pick up a row in X_j at random, and remove it from

X_j. In either of these cases, after eliminating such a row vector from X_j we restart with a reduced matrix with $j-1$ rows.

To prevent a removed vector from returning to X_j in the subsequent few steps, we introduce a *tabu list* $\mathscr{T}$ to keep the eliminated vectors for a certain period of steps. The size of $\mathscr{T}$ is specified by a parameter TL (tabu length), and vectors in the tabu list are forbidden to be an optimal solution to $P_j(X_j)$. To this end, let $\mathscr{T} = \{x^{(1)}, x^{(2)}, \cdots, x^{(p)}\}$ be the current tabu list, and consider the following 0-1 programming problem.

$$\begin{aligned} P_j(X_j, \mathscr{T}): \quad & \text{Maximize} \ \ (22) \\ & \text{subject to} \ \ (23) - (26), \\ & \qquad x^{(s)} \cdot x \le r-1, \quad s = 1, \cdots, p. \end{aligned} \tag{33}$$

Let $z_j^*(X_j, \mathscr{T})$ be the optimal objective value to this problem. Due to (33) no vectors in the tabu list satisfy (23) with equality. Then, if $z_j^*(X_j, \mathscr{T}) = r + \lambda j$ holds, the optimal $x_j^*(X_j, \mathscr{T})$ is not tabooed, and thus we can expand X_j by appending this vector to X_j. Otherwise, we choose a row vector as stated before and eliminate it from X_j. The tabu search algorithm [11], with the tabu length explicitly shown as TABU_BIBD(TL), is as follows.

Algorithm TABU_BIBD

Step 1. Let $j := 2$, X_2 be given as (28), and $\mathscr{T} := \emptyset$.

Step 2. If $j = v$, then X_j is a BIBD. Output this and stop.

Step 3. Solve an IP problem $P_j(X_j, \mathscr{T})$ĄC If $z_j^*(X_j, \mathscr{T}) = r + j\lambda$, go to Step 4. Otherwise go to Step 5.

Step 4. Append the solution to X_j and augment it to X_{j+1}. Go back to Step 2ĄD

Step 5. If there exists a row i satisfying (31), go to Step 6. Otherwise pick up a row vector from X_j at random and go to Step 6.

Step 6. Eliminate the selected vector from X_j, let $j \leftarrow j-1$, and add the eliminated vector to the tabu list $\mathscr{T}$. (If $|\mathscr{T}| > TL$, the oldest vector is removed from $\mathscr{T}$)

Step 7. If terminating condition is satisfied, then print “Tabu search failed," and stop. Otherwise go back to Step 2.

Example 2.
In applying the tabu search algorithm to BIBD(16,16,6,6,2), the tabu search produced a correct BIBD in 0.02 seconds as follows. Initially, the tabu list is initialized as $\mathscr{T} = \emptyset$. *At*

$j = 8$ *we obtain*

$$X_8 = \begin{pmatrix} 1111110000\ 000000 \\ 1100001111\ 000000 \\ 1100000000\ 001111 \\ 0100010100\ 110010 \\ 0010010110\ 000101 \\ 1001000010\ 110100 \\ 0001011010\ 001010 \\ 0001100101\ 000110 \end{pmatrix}.$$

Here we have an optimal solution $x^* = (001000100\ 010011)$ *with* $z_8^*(X_8) = 21 < r + \lambda j = 22$. *The first row conflicts against* x^*, *since* $x_1 \cdot x^* = 1 < \lambda = 2$. *We remove* x_1 *from* X_8 *and solve* $P_7(X_7', \mathscr{T})$ *with the reduced matrix* X_7'. *Again we have a conflicting row* x_4 *and eliminating this obtain a matrix* X_6' *with 6 rows. After this, 10 forward moves are repeated and at this point we obtain a BIBD(16,16,6,6,2) as*

$$\begin{pmatrix} 1100001111\ 000000 \\ 1100000000\ 001111 \\ 0100010100\ 110010 \\ 1001000010\ 110100 \\ 0001011010\ 001010 \\ 0001100101\ 000110 \\ 0101101000\ 100001 \\ 0110110010\ 000100 \\ 0111000001\ 011000 \\ 1011010100\ 000001 \\ 1000110001\ 101000 \\ 0000100110\ 011001 \\ 0000011001\ 010101 \\ 0010001100\ 101100 \\ 0010000011\ 100011 \\ 1010101000\ 010010 \end{pmatrix}.$$

Prestwich [21] proposed a local search algorithm, and tried 86 instances with v and b satisfying $vb \leq 1000$. Tables 1 and 2 compare his result against our tabu search method with MIP solvers. In these tables CPU time in seconds is shown for each method. The column of 'Prestwich' is the result of his method on a DEC Alphaserver 100A 5/300 computer (300MHz), while 'TABU' is by TABU_BIBD(10) on a faster DELL Precision 670 machine (3.8GHz $\times$ 2).

From these tables we see that the tabu search method of this paper solves all the instances that were solved by Prestwich. The tabu search algorithm solved some additional instances that Prestwich's algorithm were not able to solve. Table 3 gives a summary of the numbers of instances out of 86 solved by Prestwich's and our method. Tabu search solved approximately 20 more instances than the Prestwich's, irrespective of tabu length.

Table 1: Result of computation for Prestwich's 86 instances (Part 1).

v	b	r	k	λ	vb	Prestwich	TABU
8	14	7	4	3	112	0.00	0.00
11	11	5	5	2	121	0.04	0.01
10	15	6	4	2	150	0.04	0.02
9	18	8	4	3	162	0.05	0.00
13	13	4	4	1	169	0.01	0.00
10	18	9	5	4	180	0.12	0.01
8	28	14	4	6	224	0.06	0.00
15	15	7	7	3	225	0.61	0.01
11	22	10	5	4	242	1.20	0.01
16	16	6	6	2	256	0.54	0.02
12	22	11	6	5	264	1.23	0.02
10	30	12	4	4	300	0.12	0.01
16	20	5	4	1	320	0.16	0.01
9	36	16	4	6	324	0.21	0.00
8	42	21	4	9	336	0.25	0.00
13	26	8	4	2	338	0.50	0.06
13	26	12	6	5	338	7.93	0.08
10	36	18	5	8	360	1.08	0.03
19	19	9	9	4	361	9.73	0.05
11	33	15	5	6	363	1.79	0.01
14	26	13	7	6	364	—	0.88
16	24	9	6	3	384	9.80	0.05
12	33	11	4	3	396	0.33	0.22
21	21	5	5	1	441	0.22	0.02
8	56	28	4	12	448	0.12	0.01
10	45	18	4	6	450	0.08	0.01
15	30	14	7	6	450	—	0.95
16	30	15	8	7	480	—	0.44
11	44	20	5	8	484	1.13	0.01
9	54	24	4	9	486	0.21	0.00
13	39	12	4	3	507	1.22	0.02
13	39	15	5	5	507	11.20	0.01
16	32	12	6	4	512	—	3.57
15	35	14	6	5	525	—	0.05
12	44	22	6	10	528	27.90	0.03
23	23	11	11	5	529	—	0.07
10	54	27	5	12	540	0.98	0.02
8	70	35	4	15	560	0.13	0.00
17	34	16	8	7	578	—	7.37
10	60	24	4	8	600	0.75	0.01

Table 2: Result of computation for Prestwich's 86 instances (Part 2).

v	b	r	k	λ	vb	Prestwich	TABU
11	55	20	4	6	605	0.80	0.01
11	55	25	5	10	605	4.45	0.06
18	34	17	9	8	612	—	30.91
25	25	9	9	3	625	—	0.24
15	42	14	5	4	630	42.90	0.05
21	30	10	7	3	630	—	61.42
16	40	10	4	2	640	4.22	0.08
16	40	15	6	5	640	—	0.84
9	72	32	4	12	648	0.54	0.01
15	45	21	7	9	675	—	0.06
13	52	16	4	4	676	3.76	0.01
13	52	24	6	10	676	49.70	0.02
10	72	36	5	16	720	1.33	0.01
19	38	18	9	8	722	—	—
11	66	30	5	12	726	1.52	0.01
22	33	12	8	4	726[†]	—	—
14	52	26	7	12	728	—	0.07
27	27	13	13	6	729	—	0.54
21	35	15	9	6	735	—	—
10	75	30	4	10	750	0.92	0.02
25	30	6	5	1	750	14.30	0.03
20	38	19	10	9	760	—	—
16	48	15	5	4	768	43.20	0.15
16	48	18	6	6	768	—	0.37
12	66	22	4	6	792	1.38	0.02
12	66	33	6	15	792	12.80	0.03
9	90	40	4	15	810	0.69	0.02
13	65	20	4	5	845	1.40	0.01
11	77	35	5	14	847	5.11	0.02
21	42	10	5	2	882	—	798.86
21	42	12	6	3	882	—	—
21	42	20	10	9	882	—	—
16	56	21	6	7	896	—	0.11
10	90	36	4	12	900	0.99	0.01
15	60	28	7	12	900	—	0.05
18	51	17	6	5	918	—	1.34
22	42	21	11	10	924	—	—
15	63	21	5	6	945	30.70	0.20
16	60	15	4	3	960	6.31	0.02
16	60	30	8	14	960	—	1.12
31	31	6	6	1	961	2.04	0.04
31	31	10	10	3	961	—	1.70
31	31	15	15	7	961	—	0.10
11	88	40	5	16	968	4.07	0.04
22	44	14	7	4	968	—	—
25	40	16	10	6	1000	—	—

[†] No solution exists. See [4].

Table 3: Summary of computation.

Methods	Solved	Unsolved
Prestwitch	55	31
TABU_BIBD(5)	77	9
TABU_BIBD(10)	77	9
TABU_BIBD(20)	78	8

4 Conclusion

We have presented tabu search algorithm to find BIBDs that make use of MIP solvers. The developed method solved more instances than the previous algorithm based on local search method. However, with this algorithm we were unable to find new BIBDs. To solve problems with $v \geq 30$, some new ideas are required, and thus make the algorithm more efficient.

References

[1] I. Anderson, *Combinatorial Designs and Tournaments*, Oxford University Press, 2006.

[2] T. Beth, D. Jungnickel and H. Lenz, *Design theory, Vol. 2*, Cambridge University Press, 1999.

[3] N.L. Biggs, E.K. Lloyd and R.J. Wilson, "The history of combinatorics," R.L. Graham, M. Grötschel and L. Lovász (Editors), *Handbook of Combinatorics, Vol. 2*, MIT Press, 1996, pp. 2163-2198.

[4] R. Bilous, C. W. H. Lam, L. H. Thiel, B. P. C Li, G. H. J. van Rees, S. P. Radziszowski ,W. H. Holzmann and H. Kharaghani, "There is no 2-(22, 8, 4) block design," *Journal of Combinatorial Designs*, vol. 15, pp. 262-267, 2006.

[5] P. Bofill, R. Guimera and C. Torras, "Comparison of simulated annealing and mean field annealing as applied to the generation of block designs," *Neural Networks*, Vol. 16, pp. 1421-1428, 2003.

[6] G. E. Box, W. G. Hunter, J. S. Hunter and W. G. Hunter, *Statistics for Experiments: Design, Innovation, and Discovering, 2nd Ed.*, John Wiley & Sons, 2005.

[7] A. E. Brouwer, "Block designs," R.L. Graham, M. Grötschel and L. Lovász (Editors) *Handbook of Combinatorics, Vol. 1*, MIT Press, 1996, pp. 693-745.

[8] C. J. Colbourn, J. H. Dinitz, *Handbook of Combinatorial Designs, 2nd Ed.*, Chapman & Hall/CRC, 2007.

[9] ILOG CPLEX 10.0, http://www.ilog.com/products/cplex/, 2006.

[10] D. Z. Du, F. K. Hwang, *Combinatorial Group Testing and its Applications, 2nd Ed.*, World Scientific Publ, 2000.

[11] F. Glover, M. Laguna, *Tabu Search*, Kluwer, 1997.

[12] M. Hall, Jr., *Combinatorial Theory, 2nd Ed.*, John Wiley & Sons, 1998.

[13] H. Hanani, "The existence and construction of balanced incomplete block designs", *The Annals of Mathematical Statistics*, Vol. 32, pp. 361-386, 1961.

[14] H. Hanani, "Balanced incomplete block designs and related designs", *Discrete Mathematics*, Vol. 11, pp. 255-369, 1975.

[15] S. K. Houghten, L. H. Thiel, J. Janssen and C. W. H. Lamet, "There is no (46, 6, 1) block design", *Journal of Combinatorial Designs*, Vol. 9, pp. 60-71, 2001.

[16] T. Kirkman, “On a problem in combinations", *Cambridge and Dublin Mathematical Journal*, Vol. 2, pp. 191-204, 1847.

[17] D. L. Kreher, D. R. Stinson, *Combinatorial Algorithms: Generation, Enumeration and Search*, CRC Press, 1999.

[18] T. Kurokawa, Y. Takefuji, “Neural network parallel computing for BIBD problems", *IEEE Trans on Circuits and Systems, II*, Vol. 39, pp. 243-247, 1992.

[19] L. B. Morales, “Constructing difference families through an optimization approach: six new BIBDs", *Journal of Combinatorial Designs*, Vol. 8, pp. 221-309, 2000.

[20] S. Prestwich, “Balanced incomplete block design as satisfiability", *Irish Conference on AI and Cognitive Science*, Vol. 12, pp. 189-198, 2001.

[21] S. Prestwich, “A local search algorithm for balanced incomplete block designs", *Lecture Notes in Computer Science*, Vol. 2627, pp.132-143, Springer, 2003.

[22] V. N. Sachkov, *Combinatorial Methods in Discrete Mathematics*, Cambridge University Press, 1977.

[23] A. Schrijver, *Theory of Linear and Integer Programming*, John Wiley & Sons, 1998.

[24] R. Sedegewick, *Algorithms in C, 3rd Ed.*, Addison-Wesley, 1998.

[25] D. R. Stinson, *Combinatorial Designs, Construction and Analysis*, Springer, 2004.