Room Squares

Matthew Henderson

June 24, 2020

Contents

Introduction

Kirkman's Schoolgirl Problem

In 1850 Thomas Penyngton Kirkman, an English mathematician from Bolton, published the following problem in the *Lady's and Gentleman's Diary.*

> *Fifteen young ladies of a school walk out three abreast for seven days in succession: it is required to arrange them daily so that no two shall walk abreast more than once*

In solving this problem Kirkman discovered the following square array, which he observed was a very "curious arrangement".

Table 1: Kirkman's curious arrangement

			hi	kl	mn	op
	il	mo		np	hk	
	no	hl	mp			ik
lp		in	ko	hm		
im		kp			lo	hn

The curiosity of this square is that each of the letters h, i, k, l, m, n, o, p occurs precisely once in every column and row, while in the entire square each of the letters makes a pair with every other letter exactly once. Kirkman was able to employ this square to solve his Schoolgirl Problem. To each pair in the first column he added the element 1, to each pair in the second column 2 and so on. In addition he introduced the missing triple of numbers to each row. (e.g. row one has no elements in any of the first three columns so the numbers 1,2 and 3 would not appear hence he would add the triple 123). The seven rows of unique triples then corresponded to seven days in which the elements, corresponding to schoolgirls, were paired together exactly once throughout the arrangement. Thereby solving the problem.

Table 2: Kirkman's Schoolgirl's Solution

Day 1	123	hi4	kl5	mn6	op7
Day 2	147	il2	mo3	np5	hk6
Day 3	156	no2	hl3	mp4	ik7
Day 4	267	lo2	in3	ko4	hm5
Day 5	245	io2	kp3	lo6	hn7
Day 6	357	ho2	km2	ln4	ip6
Day 7	346	ko2	hp2	io5	lm7

Kirkman was a notable mathematician who is often regarded as the originator of the object in Figure 2, which has subsequently become known as a Room square (after T.G. Room).

Tournaments

Suppose the English Football Association proposed hosting a new type of international tournament to be staged as a one-off event in England. This tournament would involve eight national sides competing in a league that would be staged in various stadia around the country over two weeks. The structure of the tournament would be such that every team played every other team once, with the winner being the team which accumulated most points in the manner of a normal football league (3 points for a win, 1 for a draw).

To know which matches need playing is simple. Suppose the eight invited teams are:

Table 3: Teams

Argentina	England
Brazil	France
Columbia	Germany
Denmark	Holland

If we write matches as alphabetic pairs in the obvious way, (e.g. ab denoting

Argentina versus Brazil). The complete list of matches (the match set, M) is simply all unordered pairs from team set, T:

$$T = \{a, b, c, d, e, f, g, h\}.$$

i.e.

$$M = \{ab, ac, ad, ae, af, ag, ah, bc, bd, be, bf, bg, bh, cd, ce, cf, cg, ch, de, df, dg, dh, ef, eg, eh, fg, fh, g$$

It remains to be decided where and when the matches will be played.

The English F.A., for whatever reason (the financial cost of hosting eight teams, for example), has imposed a time limit of two weeks on the tournament. Realistically the treams can only manage to play on alternate days so it is decided to have, in effect, seven different "rounds" with each team competing once in each round. (Seven being the smallest number of rounds because each team has to play seven others).

For reasons of fairness the F.A. also demands the condition that each team will play once at each stadium. Can such a tournament exist? Suppose the stadia used are the following:

Table 4: Stadium

Wembley
Highbury
Villa Park
Stadium of Light
Stamford Bridge
Old Trafford
St. James Park

Then figure 4 provides a match schedule which is suitable for such a tournament.

Table 5: Fixture List for an International Soccer League

	1	2	3	4	5	6	7
1				ab	cd	ef	gh
2		bd	eg		fh	ah	

	1	2	3	4	5	6	7
3		fg	ad	eh			bc
4	dh		bf	cg	ae		
5	be		ch			bg	af
6	ag	ce		df		bh	
7	cf	ah			bg		de

Looking along the rows, each team plays once in each round. Looking down columns, each stadia hosts each team exactly once. And throughout the tournament as a whole each pair from the original match list appears exactly once, hence every team opposes every other team once. Figure 4 is another Room square of side 7. Alternatively, because the pairs are made from a set containing 8 elements, we say that this is a Room square of order 8.

T.G. Room, (1902-86)

In 1955, Thomas Gerald Room, then Professor of Mathematics at the University of Sydney, published a brief note in the Mathematical Gazette entitled *A new type of magic square.*[20] In it he presented another example of a square array with the same properties as Kirkman's. This square, Room explained, had been discovered as "a by-product of another investigation". It was preceded in the note by a particularly efficient statement of the properties of these squares, which have subsequently been known by his name.

> "The problem is to arrange the $n(2n-1)$ symbols rs (which is the same as sr) formed from all pairs of $2n$ digits such that in each row and each column there appear n symbols (and $n-1$ blanks) which among them contain all $2n$ digits."

Room's note went on to explain that while the trivial $n = 1$ Room square exists[1] , the non-existence of those with $n = 2$ (side 2) and $n = 3$ (side 5) is easily proven. Room considered the $n = 2$ proof so straightforward that it was omitted from this note, while for the $n = 3$ case he made reference to a graph-theoretic proof.

Consider the $n = 2$ case, we are required to place all the pairs of 4 digits in a 3x3 array. If we choose to use the set of non-negative integers $\{0, 1, 2, 3\}$,

[1] The Room squre of side1 is just the single array element containing the pair $\{0, 1\}$.

then we need to find somewhere to put each of the pairs $\{01, 02, 03, 12, 13, 23\}$. That we can swap the rows and columns of a Room square without damaging that square's Room-ness is self-evident. Therefore there is no less of generality in assuming that a 3×3 Room square has the pair $\{0, 1\}$ in cell $(1, 1)$.

01

If we hope to make this array a Room square we must place the pair $\{2, 3\}$ in the first row, while to complete the first column we must also place the same pair in either position $(1, 2)$ or $(1, 3)$, but each pair is only allowed to appear once. So there is no Room square of side 3, order 4.

For the $n = 3$, a Room square of side 5, case we consider the following array:

01	23	45	-	-
24	-	-	-	-
35	-	-	-	-

There is no loss in generality in using this array because we can reorder rows and columns to obtain the first row in this form and then the first column must contain either the pairs $\{01, 24, 35\}$ or $\{01, 25, 34\}$ and the latter can be converted into the former by the permutation (45)[2] which leaves the first row unchanged. We now show that completion of this square is impossible. The pairs $\{2, 5\}, \{3, 4\}$ must appear somewhere in the array other than the first three rows or columns. Also they must appear in separate rows/columns to prevent a forced recurrence of $\{0, 1\}$. Suppose we put $\{2, 5\}$ in $(4, 4)$ and $\{3, 4\}$ in $(5, 5)$, then we know that cells $(4, 5)$ and $(5, 4)$ are empty, as the only pair which could legally go in either would be $\{0, 1\}$.

Hence we know that cells $(4, 2), (4, 3), (5, 2), (5, 3)$ each contain pairs. Take cell $(5, 2)$, it could only contain $\{0, 5\}$ or $\{1, 5\}$, and the latter becomes the former under (01), so assume it contains $\{0, 5\}$. We are now forced to fill in the other cells to give the array in Figure 8.

01	23	45	-	-
24	-	-	-	-

[2]This is cycle notation, and stands for the permutation $4 \to 5, 5 \to 4$.

35	-	-	-	-
-	14	03	25	-
-	05	12	-	34

We still need to place the pairs $\{0,2\}$ and $\{0,4\}$, which cannot be done because neither can appear in the second row and they cannot both appear in the third row. Hence there is no Room square of side 5, order 6.

The real significance of Room's note was that mathematicians soon took on the task of determining the spectra of Room squares (those values of n for which Room squares exist). Research which cumulated 19 years later in the complete statement of the existence of Room squares, made by W.D. Wallis, that:

> *"Room squares exist for all odd positive integer sides except 3 and 5"* [24]

Proving this statement, which was suspected to be true from an early stage, turned out to be protracted and difficult.

The most significant breakthrough came in 1968 when Stanton and Mullin introduced the starter-adder method for constructing Room squares. This method reduces the problem of constructing Room squares to the problem of finding a certain type of initial row from which a Room square can be developed straightforwardly.

In this work much emphasis will be placed upon the proof of the existence of Room squares.

The Galois Field

Throughout this work much use will be made of a particular *finite field*, known as the Galois field, denoted by $GF(p^n)$. Whenever p^n is a prime (i.e. $n = 1$) the Galois field is precisely the integers under modulo p arithmetic, denoted Z_p. The Galois field has a number of important properties which are used in many of the proofs that follow, we introduce some of these now.

- Every Galois field (every finite field in fact) has a *primitive element*. An element, x say, is primitive in $GF(q)$ if $x^0, x^1, x^2, ..., x^{q-1}$ are all the non-zero members of $GF(q)$.

Example

$x = 2$ is a primitive element in $GF(11)$ because,

$$x^0 = 1 \quad x^1 = 2 \quad x^2 = 4 \quad x^3 = 8 \quad x^4 = 5$$

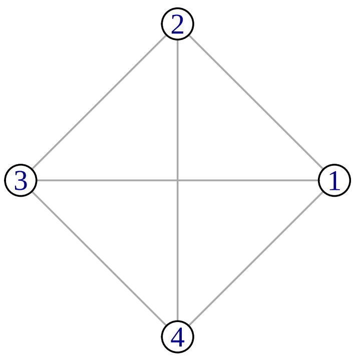
$$x^5 = 10 \quad x^6 = 9 \quad x^7 = 7 \quad x^8 = 3 \quad x^9 = 6$$

- It can be shown [3] that $x^{q-1} = 1$ is always true for any $GF(q)$ where q is odd, and $x^i \neq 1$ for any $1 \leq i \leq q - 1$
- $x^{q-1} = 1$ implies that $(x^{\frac{1}{2}(q-1)} - 1)(x^{\frac{1}{2}(q-1)} + 1) = 0$, therefore either $x^{\frac{1}{2}(q-1)} = 1$ or $x^{\frac{1}{2}(q-1)} = -1$. Clearly because of the previous remark, only the latter can be true.
- If b is a non-zero residue modulo p, then b is a quadratic residue (or square) if $x^2 \equiv b (\text{mod } p)$ has solutions, otherwise b is a quadratic non-residue (or non-square). So the non-zero squares are precisely the even powers of the primitive element, while the non-zero non-squares are the odd powers.
- There are precisely $\frac{1}{2}(p-1)$ squares mod p, and $\frac{1}{2}(p-1)$ non-squares.
- -1 is a square if $q \equiv 1 (\text{mod } 4)$, but not a square for $q \equiv 3 (\text{mod } 4)$
  - $q \equiv 1 (\text{mod } 4)$, then if x^i is a square so is $-x^i$.
  - $q \equiv 3 (\text{mod } 4)$, then x^i is a square $-x^i$ is a non-square.

A graph-theoretic approach to constructing Room squares

Graph factorisations

A graph $G(V, E)$ consists of two sets. The first V, is called the vertex-set, while the other E consists of unordered pairs of V and is called the edge set. Usually graphs are represented with diagrams where the members of V are drawn as points and the members of E as lines connecting points. Adjacency for two vertices means being connected by an edge. The **complete graph** K_n is the graph on n vertices in which all distinct vertices are adjacent.



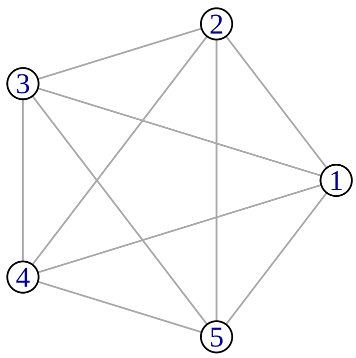


Figure 1: K_4 and K_5

A **one-factor** f_i is a set of edges in which each vertex appears exactly once.

Example 2.1.1

Two possible one-factors of K_4 are:

$$f_1 = \{12, 34\}, f_2 = \{13, 24\}$$

Figure 10 Two one-factors of K_4

A **one-factorisation** of the complete graph is a set of one-factors in which all possible edges (i.e. all unordered pairs from the edge-set) appear exactly once.

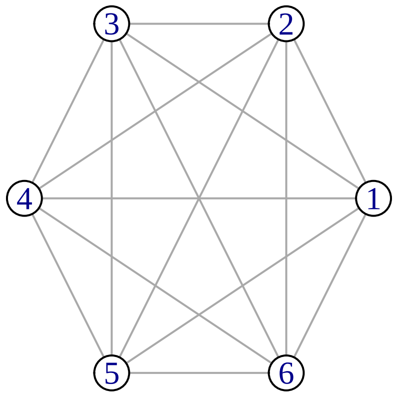


Figure 2: K_6

Example 2.1.2

Here $G = K_6$ the complete graph on 6 vertices with

$$V = \{1, 2, 3, 4, 5, 6\}$$

$$E = \{12, 13, 14, 15, 16, 23, 24, 25, 26, 34, 35, 36, 45, 46, 56\}$$

The one-factors are

$f_1 = \{12, 35, 46\}$ $\quad f_2 = \{14, 23, 56\}$ $\quad f_3 = \{16, 25, 34\}$ $\quad f_4 = \{13, 26, 45\}$ $\quad f_5 = \{15, 24, 3$

because $f_1 \cup f_2 \cup f_3 \cup f_4 \cup f_5 = E$ $F = \{f_1, f_2, f_3, f_4, f_5\}$ is a one-factorisation of G shown in Figure 11.

```
E(g)[c(1, 11, 14)]$color <- "red"
E(g)[c(3, 6, 15)]$color <- "blue"
E(g)[c(5, 8, 10)]$color <- "green"
E(g)[c(2, 9, 13)]$color <- "yellow"
E(g)[c(4, 7, 12)]$color <- "black"

plot(g, layout = layout_in_circle(g), vertex.color = "white", vertex.size = 
```

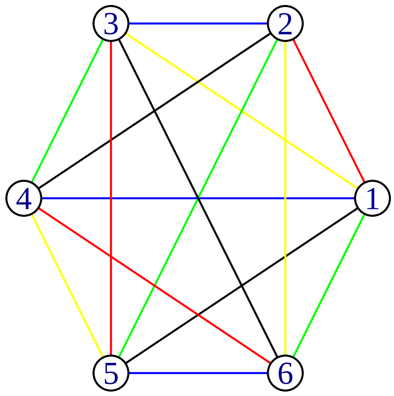


Figure 3: One-factorisation of K_6

Two one factors f and l are said to be ***orthogonal*** if $f \cap l$ contains at most one edge. Two one-factorisations F and L are orthogonal if every one-factor in F is orthogonal to every one-factor in L.

Once again consider the square array in Figure 2. If the individual elements within the array constituted the vertex set of a graph (call it R) and the pairs within each box of the array were edges, we know that each row is a one-factor and each column is a one-factor (because each member of R occurs precisely once in each row and once in each column). Further more, because all edges from the edge-set of the complete graph (i.e. all unordered pairs from R) appear once within the array, we know that the rows together form a one-factorisation and the columns form another, different, one-factorisation of K_8. Also, because any row factor intersects any column factor in only one pair (edge), all the row factors are orthogonal to all the column factors and hence the two one-factorisations are orthogonal. We have demonstrated the following theorem, given in [8] and proven in [19].

Theorem 2.1.1

The existence of a Room square of side n is equivalent to the existence of two orthogonal one-factorisations of the complete graph K_{n+1}.

An example is given in figure 12 based on the Room square in Figure 2.

Figure 12 Two orthogonal one-factorisations of K_8 based on Kirkman's square of 1850.

Hill-Climbing Algorithm for Room Squares

The idea behind Hill-climbing algorithms is to suppose there exists a **neighbourhood** of feasible solutions to some problem **instance**. With each **feasible** solution there is an associated **cost** (or profit) and finding an optimal solution becomes a matter of finding the solution with minimum cost (or maximum profit).

A hill-climbing algorithm non-deterministically selects a solution from the neighbourhood system such that the cost is less than that of some initial solution until its procedure fails, hence finding the locally optimal solution.

An algorithm for One-Factorisations

Consider how to find a one-factorisation of the complete graph. Here the problem instance is simply the even integer n and vertex set V.
Recall:

- A one-factor of K_n is a set of $n/2$ edges (hence n is even) which partition V.
- A one-factorisation of K_n is a set of $n-1$ one-factors which partitions the edge set of K_n.

Suppose we choose to represent a one-factorisation by a set of $\frac{n}{2}(n-1) = (n^2-n)/2$ pairs each of the form $(f_i, \{x,y\})$, where $x \neq y, i = 1...n-1$, and the following two conditions hold.

1. Every $\{x,y\}$ occurs in a unique pair $(f_i, \{x,y\})$.
2. For every one-factor f_i and every vertex x, there is a unique pair of the form $(f_i, \{x,y\})$.

where f_is are one-factors.

Then we consider a feasible solution to be a partial one-factorisation, again represented by pairs having the same form but this time,

1. Every $\{x,y\}$ occurs in at most one pair $f_i, \{x,y\})$.
2. For every one-factor f_i and every vertex x there is at most one pair of the form $(f_i, \{x,y\})$.

Where the f_is are **partial one-factors**.

Which enables a definition for the cost of a feasible solution F to be given by:

$$c(F) = (n^2-n)/2 - |F|$$

So that F is a one-factorisation if and only if $c(F) = 0$, i.e. $|F| = (n^2-n)/2$

Now suppose that we can implement some procedure X, say, which either reduces the cost or leaves it unaffected (i.e. it never increases the cost) then the following "hill-climbing" algorithm, provided it terminates, will find a one-factorisation.

While $c(F) \neq 0$
Do X

A procedure such as X is called a heuristic. The following two heuristics (due to Dinitz & Stinson [9]) when used together are suitable for finding a one-factorisation.

Let F be a partial one-factorisation of K_n:

Heuristic H_1 [8]

1. Choose any vertex x such that x does not occur in every partial one-factor of F (such a vertex is said to be a **live point**).
2. Choose any partial one-factor f_i such that x does not occur in f_i.
3. Choose any $y \neq x$ such that there is no partial one-factor f_j for which $(f_j, \{x, y\}) \in F$ (we say that x and y *do not occur together*).
4. **if** y does not occur in f_i, **then**
5. Replace F with $F \cup \{(f_i, \{x, y\})\}$.
6. **Else** there is a pair in F of the form $(f_i, \{z, y\}) \quad (z \neq x)$
7. Replace F with $F \cup \{(f_i, \{x, y\})\} \backslash \{(f_i, \{z, y\})\}$.

Heutistic H_2 [8]

1. Choose any partial one-factor f_i which does not occur in exactly $n/2$ pairs in F (such a partial one-factor is said to be ***live***).
2. Choose any x and y such that x and y do not occur together in f_i.
3. **if** x and y do not occur together, **then**
4. Replace F with $F \cup \{(f_i, \{x, y\})\}$.
5. **Else** there is a pair in F of the form $(f_j, \{x, y\}) \quad (j \neq i)$
6. Replace F with $F \cup \{(f_i, \{x, y\})\} \backslash \{(f_j, \{x, y\})\}$.

Example

Suppose we are in the process of trying to find a one-factorisation for K_6, and have generated a partial one-factorisation represented by the set F.

$F = \{(f_1, \{4, 6\}), (f_1, \{3, 5\}), (f_2, \{5, 6\}), (f_3\{1, 6\}), (f_3\{3, 4\}), (f_4, \{2, 3\}), (f_4, \{4, 5\})\}$

Now apply H_1:

1. Choose $x = 2$. Live, because it doesn't appear in f_1, f_2, f_3 or f_5.
2. Of these four partial one factors, choose f_1.
3. 2 only occurs together with 3 (in f_4), so pick $y = 5$.
4. 5 already appears in f_1 so $\{z, y\} = \{3, 5\}$. So replace F by $F \cup \{(f_1, \{2, 5\}) \backslash (f_1, \{3, 5\})\}$

So we have extracted one edge from the one-factorisation and replaced it with another edge, leaving the cost unchanged. If in 3. we had picked 1 then according to the heuristic we should replace F with $F \cup (f_1, \{2, 1\})$, increasing $|F|$ by one, and so decreasing the cost by the same. Because the cost cannot increase H_1 is a suitable heuristic for use in a hill-climbing algorithm.

Now apply H_2 to the new one-factorisation $F_1 = F \cup (f_1, \{2, 1\})$

1. We can pick any of f_2, f_3, f_4, f_5, because all are live. Choose f_2.
2. Choose $x = 2, y = 3$, because neither appear in f_2.
3. 2 and 3 occur together in f_4. So replace F_1 with $F_1 \cup \{(f_2, \{2, 3\}) \backslash (f_4, \{2, 3\})\}$

Again the cost remains unchanged by this procedure, and if in 2. we had chosen $x = 1, y = 4$ instead then we would have replaced F_1 with $F_1 \cup \{(f_2, \{1, 4\})\}$ decreasing the cost by one. As with H_1, the cost cannot increase, which makes H_2 a suitable heuristic. The hill-climbing algorithm for constructing one-factorisations which was first given in [9] has a very simple form.

1. **While** $c(F) \neq 0$, **do**
2. choose $r = 1$ or $r = 2$ with equal probability
3. perform H_r

An Algorithm for Room Squares

To generate a Room square all that remains is to produce another one-factorisation G, say, which is orthogonal to F. This will inevitably require slight modifications to be made to H_1 and H_2. Now if an array R is constructed in which the rows are labelled with the one-factors of $F(f_1, f_2, ..., f_{n-1})$, and the columns are labelled with the partial one-factors of $G(g_1, g_2, ..., g_{n-1})$. Then R will be a Room square if the (f_i, g_j) cell contains $\{x, y\}$, if and only if $(f_i, \{x, y\}) \in F$ and $(g_j, \{x, y\}) \in G$ and is empty otherwise.

Again these two heuristics are due to Dinitz & Stinson and originally presented in [9]. Although a necessary correction has been made as will become apparent.

OH_1

1. Choose any live point x.
2. Choose any partial one-factor g_i such that x does not occur in g_i.
3. Choose any $y \neq x$ such that x and y do not occur together in G.
4. Let f_j be the one-factor of F which contains the edge $\{x, y\}$.
5. **if** $R(f_j, g_i)$ is not empty **then**
6. OH_1 fails.
7. **Else if** y does not occur in g_i, **then**
8. Replace G by $G \cup (g_i, \{x, y\})$.
9. Define $R(f_j, g_i) = \{x, y\}$.
10. **Else** there is a pair in G of the form $(g_i, \{z, y\}) \quad z \neq x$.
11. Replace G by $G \cup (g_i, \{x, y\}) \backslash (g_i, \{z, y\})$.
12. Define $R(f_k, g_i)$, to be empty^i, where $(f_k, \{z, y\}) \in F$.

OH_2

1. Choose any live partial one-factor g_i.
2. Choose any x and $y \neq x$ such that x and y do not occur together in g_i.
3. Let f_j be the one-factor of F which contains the edge $\{x, y\}$.
4. **if** $R(f_j, g_i)$ is not empty **then**
5. OH_2 fails.
6. **Else if** x and y do not occur together, **then**
7. Replace G by $G \cup (g_i, \{x, y\})$.
8. Define $R(f_j, g_i) = \{x, y\}$.
9. **Else** there is a pair in G of the form $(g_k, \{x, y\}) \quad (k \neq i)$
10. Replace G by $G \cup (g_i, \{x, y\}) \backslash (g_k, \{x, y\})$
11. Define $R(f_j, g_i) = \{x, y\}$
12. Define $R(f_j, g_k)$ to be empty

Example 2.2.1

Suppose the factorisation F from the earlier example has been completed and is represented by the set:

$$ M={

$(f_1\{1,2\}), (f_1\{3,5\}), (f_1\{4,6\}), (f_2\{1,4\}), (f_2\{2,3\}), (f_2\{5,6\}), (f_3\{1,6\})$
$(f_3\{2,5\}), (f_3\{3,4\}), (f_4\{1,3\}), (f_4\{2,6\}), (f_4\{4,5\}), (f_5\{1,5\}), (f_5\{3,6\})$

}

Notice that this is precisely the one-factorisation of K_6 given on page 9.

Now suppose we have established the following one-factors in G:

$$G = \{(g_1, \{1,4\}), (g_2, \{1,6\}), (g_3, \{3,6\}), (g_5\{5,6\}), (g_5\{1,2\})\}$$

At this state R looks like

$$R=

	g_1	g_2	g_3	g_4	g_5
f_1					$1,2$
f_2	$1,4$				$5,6$
f_3		$1,6$			
f_4					
f_5			$3,6$		

$$

Figure 13

Now apply OH_1:

1. Choose $x = 5$, suitably live.
2. Choose g_3, in which 5 does not occur.
3. 5 does not occur together with 2 in G, so we are free to choose $y = 2$.
4. In F, $\{2.5\} \in f_3$.
5. f_3, g_3 is empty in R, also $y = 2 \notin g_3$.
6. Replace G with $G \cup (g_3, \{5,2\})$.
7. Define $R(f_3, g_3) = \{5,2\}$.

This decreases the cost by one, alternatively we might have chosen, at stage 3. $y = 3$, in that case.

4. $\{3,5\} \in f_1$.
5. f_1, g_3 is empty in R, also $y \in g_3$, occurring in the pair $(g_3, \{3,6\}), z = 6$.
6. Replace G with $G \cup (g_3, \{3,5\}) \backslash (g_3, \{3,6\})$.
7. Define $R(f_1, g_3) = \{3,5\}$.
8. Define $R(f_5, g_3)$ to be empty.

Which leaves the cost unaffected. Suppose now that R is the array after this second version of the application of OH_1:

$$R=

	g_1	g_2	g_3	g_4	g_5
f_1			$3,5$		$1,2$
f_2	$1,4$				$5,6$
f_3		$1,6$			
f_4					
f_5					

$$

Figure 14

Now if we apply OH_2:

1. Choose g_4, a live partial one-factor.
2. Choose $x = 1, y = 2$, neither of which occur in g_4.
3. $(f_1, \{1, 2\}) \in F$.
4. f_1, g_4 is empty in R, also x and y do occur together, $(g_5, \{1, 2\}) \in G$.
5. Replace G with $G \cup (g_4, \{1, 2\}) \backslash (g_5, \{1, 2\})$
6. Define $R(f_1, g_4) = \{1, 2\}$
7. Define $R(f_1, g_5)$ to be empty.

This procedure leaves the cost unaffected and if instead we had chosen at 2. $x = 3, y = 4$, then would have been required to replace G with $G \cup (g_4, \{3, 4\})$, and put $\{3, 4\}$ in cell (f_3, g_4) of R, an action which reduces the cost by one. However, we know that two orthogonal one-factorisations of K_6 are equivalent to a Room square of side 5, which has been shown not to exist. Hence it would be futile to continue with this method in this particular case. Nevertheless the example shows how the heuristics work.

There is no guarantee of success with repeated use of these heuristics, although Dinitz & Stinson are quick to point out that the algorithm involving H_1 and H_2 has never[3] failed to produce the desired one-factorisation. If we hope to use the OH_1 and OH_2 in a similar algorithm then the possibility of failure becomes a real possibility. Two possibilities exist, either both heuristics fail or successive use of them leads to an infinite loop. In order to avoid both we introduce a ***threshold*** function, which simply arrests the progress of the algorithm after a certain number of iterations of the heuristics. Dinitz & Stinson found after experimentation that the following function is suitable.

$$T(n) = 100n$$

[3]In over ten-million attempts, they claim

Then the hill-climbing algorithm for finding a Room square is as follows [8]:

1. Use the previous hill climbing algorithm to construct F, a one-factorisation of K_n.
2. Number of iterations initialised to be 0
3. While (number of iterations $< T(n)$) and $c(G) \neq 0$, do
4. Choose $r = 1$ or $r = 2$ at random with equal probability
5. Perform OH_r
6. Increment number of iterations

The Room Square Generator

Dinitz and Stinson choose to implement the above algorithm in Pascal, and ran in on an Amdahl 5850 workstation. It was very successful, finding many Room squares with sides ranging from 11 to 101. For each successful trial they had 9 or 10 failures (the program being stopped by the threshold function) and timings ranged from 0.09 seconds for an 11x11 Room square, to 7.3 on average for the 25 different 101x101 Room squares they found.

I chose to implement the hill-climbing algorithms in Visual Basic 6.0 on a Pentium III-450/Win 98 Desktop. Needless to say, it was slightly less successful – exhibiting a similar probability of success but unfortunately becoming very slow for Room squares bigger than 21. It found square of side 21 after an all-night search, but after 48 hours looking for one of 23x23 I decided to call the search off.

Despite the failures at higher order, the Room square generator was very successful in finding smaller squares. It found 7x7 Room squares in as little as 4 seconds, and even 15x15 squares only took a few minutes.

Annotated code for the Room square generator can be found along with some of the larger squares in Appendix I and below is a screen shot of the application having successfully located a 9x9 Room square in a little over one minute after 507 iterations of the heuristics OH_1 and OH_2. The uppermost panel represents some of the one-factorisation generated by the algorithm involving H_1 and H_2, while the second panel shows part of the orthogonal one-factorisation generated by OH_1 and OH_2. The lower panel is a Room square of side 9.

Figure 15 Screenshot of the Room Square Generator

Proving the existence of Room squares

Balanced Room squares

Closing Remarks

The results we have established in the previous chapter regarding the existence of balanced Room squares represent by no means the complete story. Du and Hwang [10] have established the existence of $SSBS$ for all prime powers

$$q = 2^\alpha t + 1, \alpha \geq 2, t \geq 3, t$$

odd. Further, Anderson has shown that consequently the construction due originally to Hwang, Kang and Yu but corrected in [2], allows us to state the existence of the corresponding $BRS(2q+2)$ in one particular case.

By far the most significant remaining result which has not been included in the previous chapter is due to B.A. Anderson who proved that $BRS(2^n)$ exist for all odd $n \geq 3$. His construction was based upon the theory of finite geometry, an area which has also contributed constructions for Room squares (the non-balanced kind). Other similar geometrical constructions have been used to establish the existence of $BRS(2^n)$, for $4 \leq n \leq 18$, n even. The two smallest values of $n \equiv 0 (\text{mod } 4)$ for which the existence of a $BRS(n)$ remains in doubt are 36 and 92. The first of these, along with many others, would be established by the doubling construction if a $SSBS$ could be found in Z_{17}. This remains one of the most significant open problems for BRS, namely to establish the existence of $SSBS$ in Z_n when n is a Fermat prime.

The link between graph theory and Room squares that was touched upon in the second chapter has opened many avenues of research. Possibly the most interesting of which is the existence of perfect Room squares. A one-factorisation of K_n is said to be perfect if the union of two of its one-factors is a hamiltonian cycle of K_n. A perfect Room square is one of side n in which both row and column factorisations of K_{n+1} are perfect. Very little seems to be known about perfect one-factorisations. Individual examples of perfect Room squares of side 11 have been constructed but no infinite classes have yet been found.