The Strongly Regular (45, 12, 3, 3) Graphs

Kris Coolsaet, Jan Degraer
Department of Applied Mathematics and Computer Science
Ghent University
Krijgslaan 281–S9, B–9000 Gent, Belgium
Kris.Coolsaet@ugent.be, Jan.Degraer@ugent.be

Edward Spence
Department of Mathematics
University of Glasgow
Glasgow G12 8QQ, Scotland
ted@maths.gla.ac.uk

Submitted: May 24, 2005; Accepted: Mar 24, 2006; Published: Apr 4, 2006
Mathematics Subject Classifications: 05E30, 05–04

Abstract

Using two backtrack algorithms based on different techniques, designed and implemented independently, we were able to determine up to isomorphism all strongly regular graphs with parameters $v = 45$, $k = 12$, $\lambda = \mu = 3$. It turns out that there are 78 such graphs, having automorphism groups with sizes ranging from 1 to 51840.

1 Introduction

A strongly regular graph with parameters (v, k, λ, μ), denoted by $\mathrm{srg}(v, k, \lambda, \mu)$, is a graph on v vertices that is regular of degree k such that each pair of adjacent vertices has λ common neighbours, and each pair of non-adjacent vertices has μ common neighbours. If A denotes the $v \times v$ adjacency matrix of such a graph then

$$A^2 = kI + \lambda A + \mu(J - I - A), \tag{1}$$

where, as usual, I is the identity matrix and J the all-one matrix, both of size v. From this it is easily seen that A has three distinct eigenvalues, viz., $\theta_0 = k$ of multiplicity $m_0 = 1$ and two others, $\theta_1 > \theta_2$, say, which are the solutions of $x^2 + (\mu - \lambda)x + (\mu - k) = 0$, of multiplicities m_1 and m_2, determined by the equations $1 + m_1 + m_2 = v$ and $k + m_1\theta_1 + m_2\theta_2 = 0$. (Some authors prefer the notations $r = \theta_1$, $s = \theta_2$, $f = m_1$, $g = m_2$.) To avoid trivial examples we assume that $0 < k < v - 1$. Moreover, since the complement of an $\mathrm{srg}(v, k, \lambda, \mu)$ is also an $\mathrm{srg}(v, v-1-k, v-2k+\mu-2, v-2k+\lambda)$, we have the further conditions that $v-2k+\mu-2 \geq 0$ and $v-2k+\lambda \geq 0$. In addition, there is also the condition that $k(k-1-\lambda) = \mu(v-1-k)$, which is obtained from (1) by multiplying both sides by J.

The question of existence, and ultimately, classification, of strongly regular graphs satisfying these necessary conditions has occupied the minds of many over the years. The smallest case for which these conditions are satisfied, but for which no strongly regular graph exists, is given by $(21, 10, 5, 4)$. At present all strongly regular graphs on $v \leq 36$ vertices are known. See, for example [2] in conjunction with [10]. In addition to those whose uniqueness can be established without the use of a computer, there are some sporadic cases where the classification is complete. See [4], [6] and [11]. The smallest set of feasible parameters (v, k, λ, μ) for which the existence of the corresponding strongly regular graph is in doubt is $(65, 32, 15, 16)$. This is a particular example of the case when the two eigenvalues θ_1 and θ_2 are non-integral, the so-called *half case*. They have parameter sets $(4t+1, 2t, t-1, t)$ and have been completely classified for $t \leq 7$. A complete classification for any value of $t > 7$ seems hopeless, unless of course it can be proved that the strongly regular graph does not exist. Thus, apart from the half case, the parameters $(45, 12, 3, 3)$ are the first for which the complete classification has not yet been achieved, and it is these which we consider in this paper.

One example of an $\mathrm{srg}(45, 12, 3, 3)$ is provided by the point graph of the generalized quadrangle $\mathrm{GQ}(4, 2)$, and there are others already known, in fact 58 in total. These were found in [7] by R. Mathon and the third author in a search for such graphs with a particular block structure.

In an exhaustive search, the computer programs (of which more details are given in Sections 2–4) have extended this list to a total of 78 pairwise non-isomorphic graphs, providing a complete classification. These results are presented in Section 5.

Some notation: we write $p \sim q$ to indicate that two vertices p and q of the graph are adjacent ($p \neq q$), and $p \not\simeq q$ when $p \not\sim q$ and $p \neq q$. For a given vertex p of a graph Γ, the neighbourhood graph of that vertex, denoted by $\Gamma(p)$, is the subgraph of Γ induced by all vertices that are adjacent to p.

2 Exhaustive search

The classification of the strongly regular $(45, 12, 3, 3)$ graphs was done by two computer programs that where designed and implemented independently. The central theme of the algorithms used is recursive backtracking with isomorphism rejection.

An $\mathrm{srg}(45, 12, 3, 3)$ is represented on the computer by its adjacency matrix. This is a 45×45 symmetric $(0, 1)$ matrix A with zero diagonal, where rows and columns are indexed by the vertices of the graph. The matrix entry $A_{p,q}$ is 1 if $p \sim q$ and 0 otherwise.

A backtrack algorithm basically generates all possible matrices of this kind in a recursive way, trying each of the two possible values for each of the (upper diagonal) entries, and filters out those that do not satisfy the definition of an $\mathrm{srg}(45, 12, 3, 3)$.

Different techniques are used to do this in an intelligent way and make the program complete its search within reasonable time. Most of them amount to 'pruning' the search tree at points where A is only partially determined, either because it is known that it will never be possible to complete it to a strongly regular graph with the required properties, or because the partially completed matrix can be proved to be isomorphic to an instance which was already encountered (or is sure to be considered later). Also the order in which the various entries are filled in may influence the speed of the algorithm dramatically.

Below we provide some details on the various methods of this kind we have used for this particular problem. Apart from what we describe below we have also made use of *forward checking* and *dynamic variable ordering* techniques similar to those of [3].

3 Constraints

Several properties of strongly regular graphs can be used to prune the search tree. For example, because the graphs to be generated are regular of degree $k = 12$ we may prune the search as soon as a row or column of A contains more than 12 entries equal to 1, or more than $v - k - 1 = 32$ entries that are 0 (excluding the diagonal entry).

Also, each pair of vertices in the resulting graph must have exactly three common neighbours, so the search tree can be pruned as soon as the partially constructed matrix has two vertices p and q that have more than three neighbours in common. In fact, from the parameters of the strongly regular graphs similar constraints can be deduced for numbers of vertices that are adjacent to p but not to q, adjacent to q but not to p and adjacent to neither p nor q. We list the corresponding numbers in the following tables:

when $p \sim q$

	$\sim q$	$\not\sim q$
$\sim p$	3	8
$\not\sim p$	8	24

when $p \not\sim q$

	$\sim q$	$\not\sim q$
$\sim p$	3	9
$\not\sim p$	9	22

Note that these numbers depend on whether p and q are adjacent or not, and this may not be known for a given pair p, q when A is not yet completely filled in. In that case we can still use a weaker constraint: the number is then bounded above by the maximum of the corresponding bounds for $p \sim q$ and $p \not\sim q$. When the adjacency of p and q is finally determined, we can check this constraint again for the correct bound.

The adjacency matrix A of the graph has some interesting *algebraic properties* which can be used to speed up the search. (We refer to [1] for more information on how these properties are derived.) The eigenvalues of A are $\theta_0 = 12$, $\theta_1 = 3$, and $\theta_2 = -3$ with corresponding multiplicities $m_0 = 1$, $m_1 = 20$ and $m_2 = 24$. It is a consequence of the interlacing of eigenvalues (a very good reference for which is [5]) that every principal submatrix P of A has all its eigenvalues except the largest one lying in the interval $[-3, 3]$. Moreover, if P has order n it must have an eigenvalue -3 with multiplicity at least $n - 21$ and an eigenvalue 3 with multiplicity at least $n - 25$.

Hence, during the backtrack search, whenever a partially completed matrix is available that fully defines the entries of a principal submatrix P, we compute the eigenvalues of P and check whether they satisfy the above interlacing property. If not, we may prune the search at that point. In practice this turns out to be computationally expensive, and therefore we do not check every principal submatrix, but typically only the top left submatrices, and not at every point in the search process, and only whenever we have completed a new column of A.

For each eigenvalue θ_i we may define a corresponding *minimal idempotent* matrix E_i. E_0 is the all-1 matrix J and E_1, E_2 can be computed as follows:

$$\begin{aligned} E_1 &= \frac{(A-\theta_0\, I)(A-\theta_2\, I)}{(\theta_1-\theta_0)(\theta_1-\theta_2)} = \frac{1}{90}(A-12\, I)(A-3\, I), \\ E_2 &= \frac{(A-\theta_0\, I)(A-\theta_1\, I)}{(\theta_2-\theta_0)(\theta_2-\theta_1)} = -\frac{1}{72}(A-12\, I)(A+3\, I) \end{aligned}$$

and then

$$E_1 = \frac{4}{9}\, I + \frac{1}{9} A - \frac{1}{18}(J-A-I), \qquad E_2 = \frac{8}{15}\, I - \frac{2}{15} A + \frac{1}{30}(J-A-I).$$

In other words, E_1 is the 45×45 matrix with values $4/9$ on the diagonal, $1/9$ on positions that correspond to adjacent points and $-1/18$ elsewhere, while E_2 has values $8/15$ on the diagonal, $-2/15$ on 'adjacent' positions and $1/30$ everywhere else.

Two important properties of these minimal idempotents E_i are the following: they have *rank* m_i and they are *positive semidefinite*. It follows that any principal submatrix of E_i must have rank $\leq m_i$ and must be positive semidefinite. We may use this to prune the search tree in much the same way as the interlacing property was used above.

In fact, both methods tend to prune at approximately the same points in the search process and there seemed to be no point in applying them both at the same time. We have therefore used one method in one program and the other method in the other.

4 Extending subgraphs

4.1 Subgraph induced by $\Gamma(x) \cup \Gamma(y)$

The generation of the graphs proceeds in several stages.

In a preliminary stage we consider the plausible neighbourhood graphs $\Gamma(x)$ of a vertex x in an srg$(45, 12, 3, 3)$. It is clear that such a neighbourhood graph has 12 vertices and is regular of degree 3. There exist (up to isomorphism) 94 regular graphs of order 12 and degree 3 (they can easily be generated using `geng` [8]). Of these, 21 can immediately be discarded as plausible neighbourhood graphs because they contain at least one pair of vertices that have more than two common neighbours (which together with x violate the $\lambda = \mu = 3$ constraint).

In a further stage, the 73 matrices $N_1, \ldots N_{73}$ obtained in this way are extended to (symmetric) principal submatrices M of order 21 which correspond to possible subgraphs induced by $\Gamma(x) \cup \Gamma(y)$, where $y \in \Gamma(x)$. Without loss of generality we may reorder the rows and columns of M in such a way that M, with vertices $v_1, v_2, \ldots, v_{21}$, say, takes the form as depicted in Figure 1. Here $x = v_1$, $y = v_2$, $\Gamma(x)$ has vertices $v_2, v_3, \ldots, v_{13}$ and $\Gamma(y)$ has vertices $v_1, v_3, v_4, v_5, v_{14}, \ldots, v_{21}$. Thus, assuming $\Gamma(x)$ and $\Gamma(y)$ have been found, the only part of M that is unknown corresponds to the adjacencies between vertices $v_6, v_7, \ldots, v_{13}$ and $v_{14}, v_{15}, \ldots, v_{21}$, denoted by X in Figure 1.

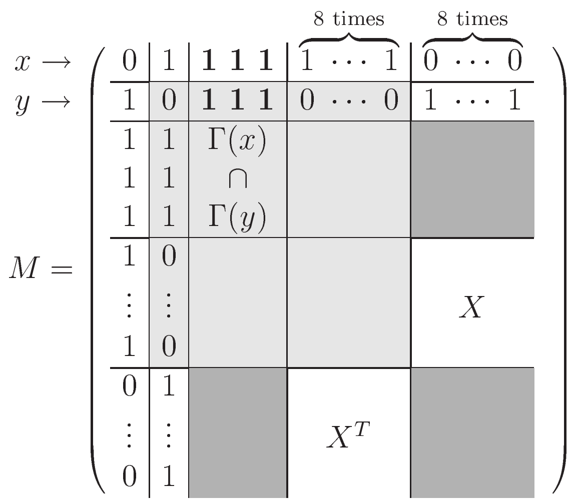

Figure 1: General form of the principal submatrix induced by $\Gamma(x) \cup \Gamma(y)$.

We copy each of the matrices N_i in turn into the part of M which corresponds to the subgraph induced by $\Gamma(x)$ (cf. light grey part of Figure 1). In each case we then generate all possibilities for the adjacencies between vertices $v_3, v_4, \ldots, v_{13}$ and vertices $v_{14}, v_{15}, \ldots, v_{21}$ and the mutual adjacencies between vertices $v_{14}, v_{15}, \ldots, v_{21}$ (cf. X and dark grey part of Figure 1). As mentioned earlier, two different methods were used to achieve this. Both methods employ a recursive algorithm that prunes the search using the constraints described in Section 3. We give a brief description of each.

The *first method* recursively fills the upper diagonal matrix entries of columns 14 to 21, column by column, starting with column 14. This method involves a pruning criterion based on the maximum clique size of the strongly regular graph Γ. From $\lambda = 3$ it follows that Γ can have cliques of size at most 5, (and certainly must have one of size 3) and hence that neighbourhood graphs in Γ can have cliques of size at most 4. We list the number of graphs N_i with a given clique size in Table 1.

Maximum clique size	4	3	2
Number of graphs	5	50	18

Table 1: Maximum clique size of plausible neighbourhood graphs.

Since M is to be embedded in the adjacency matrix of an $\mathrm{srg}(45, 12, 3, 3)$, if we assume that that adjacency matrix has maximum clique size s $(3 \leq s \leq 5)$, we may also assume the same to be true for M. Moreover, we may assume that both x and y belong to a clique of maximum size. Thus both $\Gamma(x)$ and $\Gamma(y)$ will have maximum clique size $s - 1$. In the extension process to the matrix of order 21 we prune the search tree whenever we meet a clique which has a size larger than s.

Searching for such small cliques is not very expensive in time, because we already need to keep track of the number of common neighbours of any two points in order to check some of the constraints listed in Section 3. As in [4], we use a *look-back* strategy to further improve the clique search. Suppose that the choice of 1 for the matrix entry $(M)_{p,q}$ completes a clique of size $d+1$ (with d the size of a maximum clique), which means that we are able to prune the search at this point. Then this choice remains forbidden, so that $M_{p,q} = 0$, until a backtrack occurs to the second last choice of entry that contributed to the same clique.

We also make use of another isomorphism rejection technique: we fix an ordering of the list $N_1, \ldots, N_{73}$ in such a way that the maximum clique size decreases while we traverse the list from front to back. Then, whenever we have fully generated the submatrix $\Gamma(y)$ we determine using McKay's program `nauty` [9], the index of the unique graph in this list which is isomorphic to $\Gamma(y)$. If this index is smaller than the index of $\Gamma(x)$, we discard the result. Indeed, the matrix we obtain by interchanging x and y in this result will also be generated during the search and yields an isomorphic subgraph.

A final pruning criterion that avoids the construction of too many isomorphic submatrices M, is that we require for each $p, q \in \Gamma(y) \setminus \{\Gamma(x) \cup \{x\}\}$ with $p \leq q$ that the tuple $(M_{p,1}, \ldots, M_{p,21})$ is *lexicographically* smaller then or equal to the tuple $(M_{q,1}, \ldots, M_{q,21})$. This first method produced approximately $35,000,000$ pairwise non-isomorphic candidates M.

The *second method* of generating possible 21×21 submatrices differed essentially in only one way. Instead of using maximum cliques to order the matrices $N_1, N_2, \ldots, N_{73}$, they were assumed to be in standard form and ordered lexicographically. The *standard form* of the adjacency matrix A of a graph, denoted by $St(A)$, is defined to be the greatest binary integer obtained by permuting the rows and corresponding columns of A, and concatenating the rows in the upper diagonal part of the resulting matrix.

Given $\Gamma(x) = N_i$, for some i, the first stage in the construction of M involved finding all pairwise non-isomorphic matrices M with $X = 0$ and $St(\Gamma(y)) \geq N_i$, and for which the inner product of any two distinct rows was at most 3.

When it came to filling in the entries of X in the matrix M of Figure 1, these were generated column by column, starting with column 14. Whenever column s $(14 \leq s \leq 21)$ was completed satisfactorily according to the constraints outlined in Section 2, the corresponding $s \times s$ principal submatrices of E_1 and E_2 were tested to ensure that not only were they positive semidefinite, but also that all their eigenvalues were bounded above by 1.

In the end this method produced approximately $21,000,000$ pairwise non-isomorphic candidates M.

4.2 Extension to the full matrix A

In the last stage, having produced a list of non-isomorphic candidate submatrices M, we attempt recursively to extend each of them to a full adjacency matrix A. Again we apply all the constraints from Section 3.

Most of the time the extension fails rather quickly. The application in this last stage of forward checking and dynamic variable ordering techniques similar to those of [3] certainly contributes to this behaviour.

Since most of the time the extension fails rather quickly and checking which of the N_i is isomorphic to $\Gamma(z)$ for a given $z \in \Gamma(x)$ is a rather costly operation, it turns out to be no longer necessary in this last stage to compare neighbourhood graphs of other points of $\Gamma(x)$ with the list $N_1, \ldots, N_{73}$. We still prune on maximum clique sizes, because that check is fairly quick.

As in the first stage we we require for each $p, q \notin \Gamma(y) \setminus \{\Gamma(x) \cup \{x\}\}$ with $p \leq q$ that the row $(A_{p,1}, \ldots, A_{p,45})$ is lexicographically smaller or equal than the row $(A_{q,1}, \ldots, A_{q,45})$.

Finally we use `nauty` [9] to filter out isomorphic matrices A among the results obtained so that we retain exactly one representative from each isomorphism class.

Maximum clique size 5		Maximum clique size 4	
$\lvert$Aut (srg)$\rvert$	number of srgs	$\lvert$Aut (srg)$\rvert$	number of srgs
51840	1	162	1
1152	1	54	1
216	1	18	5
72	2	9	1
48	3	6	16
36	1	3	7
18	1	2	15
12	3	1	8
10	1		
8	3		
4	5		
2	2		
Total	24	Total	54

Table 2: Automorphism group size for each srg$(45, 12, 3, 3)$.

5 Results

As stated in the introduction, both computer programs independently proved that there are up to isomorphism exactly 78 different strongly regular graphs with parameters

$(v, k, \lambda, \mu) = (45, 12, 3, 3)$, providing a complete classification. A file containing all of these graphs can be obtained from [12].

Of these 78 graphs, 24 graphs contain a clique of size 5. There are 54 graphs which do not contain a clique of size 5 but do have a clique of size 4. No srg$(45, 12, 3, 3)$ exists which does not contain a clique of size 4.

In Table 2 we list these graphs and the sizes of their automorphism groups. The column labeled "number of srgs" gives the number of isomorphism classes of srg$(45, 12, 3, 3)$s whose automorphism group has the size indicated in the column labeled "$|\mathrm{Aut}\,(\mathrm{srg})|$".

It may be of interest to the reader to know the extent of the computational power and the computer languages that were used in the investigation. As far as the first two authors are concerned, the language used was Java and the computation done on a batch of 24 Linux-based AMD Athlon MP 1800 and 2600 PCs, while the third author had access to more modest equipment. He used a twin processor, 2.2 GHz Linux-based workstation with gcc's C compiler. An exact record of the time taken was not kept, but the whole investigation took weeks, rather than days.

6 Elimination of errors

The computer programs need to perform a lot of floating point computations, which by their nature only yield approximate results. It is important to be certain that these numerical errors are sufficiently small as not to invalidate our main conclusions. We have used two different methods to ensure this. For the first two authors this was needed when checking that certain matrices were positive semidefinite, while for the third author this was done when investigating the eigenvalues themselves.

In the first case a candidate matrix A is eliminated if for one of the associated matrices E_i there can be found a real vector $\mathbf{x}$ such that $\mathbf{x}E_i\mathbf{x}^T < 0$. Such a vector $\mathbf{x}$ is first computed using techniques from numerical algebra. Afterwards, to catch accumulations of rounding errors, the expression $\mathbf{x}E_i\mathbf{x}^T$ is re-evaluated and compared to $-\epsilon$ for a suitably small $\epsilon > 0$. (Rounding errors on the value of $\mathbf{x}E_i\mathbf{x}^T$ can be estimated very accurately.) If $\mathbf{x}$ fails this test, then A is accepted. Of course this may mean that we sometimes falsely assume that a matrix is positive semidefinite when it is not. This, however, is not a problem.

In the second case an iterative procedure was used to determine eigenvalues to within a chosen error and with an upper bound to the number of iterations allowed. A candidate matrix A whose eigenvalues were not determined within the maximum number of iterations was accepted even though its eigenvalues may well have been outside the required range.

Acknowledgment The authors are grateful to the referee for his comments. They helped greatly towards an improvement in the presentation of the results.

References

[1] A. E. Brouwer, A. M. Cohen and A. Neumaier, *Distance-Regular Graphs*, Ergeb. Math. Grenzgeb. (3) **18**, Springer-Verlag, Berlin (1989).

[2] A. E. Brouwer, *Strongly Regular Graphs*, in The CRC Handbook of Combinatorial Designs, (C.J. Colbourn, J.H. Dinitz; eds), CRC Press, New York, (1996), 667–685,

[3] J. Degraer, K. Coolsaet, *Classification of three-class association schemes using backtracking with dynamic variable ordering*, Discrete Mathematics, **300(1–3)** (2005), 71–81.

[4] J. Degraer, K. Coolsaet, *Classification of some strongly regular subgraphs of the McLaughlin graph*, to appear in Discrete Mathematics.

[5] W. H. Haemers, *Eigenvalue Techniques in Design and Graph Theory*, Mathematical Centre Tracts **121**, Mathematisch Centrum, Amsterdam, (1980).

[6] W. H. Haemers, E. Spence, *The Pseudo-Geometric Graphs for Generalized Quadrangles of order* $(3,t)$, European J. Combin., **22(6)** (2001), 839–845.

[7] R. Mathon, E. Spence, *On* 2-$(45,12,3)$ *designs*, Journal of Combinatorial Designs, **4(3)** (1996), 155–177.

[8] B. D. McKay, *Practical Graph Isomorphism*, Congressus Numerantium, **30** (1981), 45–87.

[9] B. D. McKay, *Nauty user's guide (version 1.5)*, Technical Report TR-CS-90-02, Computer Science Department, Australian National University, (1990).

[10] B. D. McKay, E. Spence, *The Classification of Regular Two-Graphs on* 36 *and* 38 *Vertices*, Australas. J. Combin., **24** (2001), 293–300.

[11] E. Spence, *The Strongly Regular* $(40,12,2,4)$ *Graphs*, Elec. Journ. Combin. **7(1)**, (2000).

[12] E. Spence, *Strongly Regular Graphs*, URL: `http://www.maths.gla.ac.uk/~es/srgraphs.html`, (2004).