EURO Journal on Computational Optimization 11 (2023) 100060

Contents lists available at ScienceDirect

EURO Journal on Computational Optimization

journal homepage: www.elsevier.com/locate/ejco

The missing Moore graph as an optimization problem

Derek H. Smith[a,*], Roberto Montemanni[b]

[a] *Computing and Mathematics, University of South Wales, Pontypridd, CF37 1DL, Wales, United Kingdom*

[b] *Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Via Amendola 2, 42122 Reggio Emilia, Italy*

A R T I C L E I N F O

Keywords:
Graph theory
Moore graphs
Diameter
Girth

A B S T R A C T

It has been an open question for 6 decades whether a Moore graph of diameter 2 and degree 57 exists. In this paper the question is posed as an optimization problem and an algorithm is described. The algorithm converges to solutions which are massively short of the number of edges required. This, and other supporting work, tend to suggest that the graph does not exist. The formulation presented is a particularly hard testbed for optimization algorithms. It is left as a challenge to others to develop alternative algorithms that may support the claim, or find solutions with more edges, or even construct the Moore graph.

* Corresponding author.

E-mail addresses: derek.smith@southwales.ac.uk (D.H. Smith), roberto.montemanni@unimore.it (R. Montemanni).

https://doi.org/10.1016/j.ejco.2023.100060

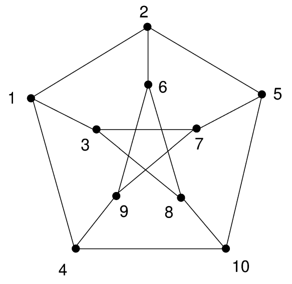

Fig. 1. The Petersen Graph.

1. Introduction

The possible existence of a Moore graph of diameter 2 and degree 57 has been an open problem for more than 60 years. Many have tried to construct such a graph, and others have tried to prove its non-existence. In 1974 Biggs [1] wrote "many claims of its non-existence have been made, but none published", and this situation has continued to the present day.

The Moore bound $M(\Delta, D)$ is an upper bound on the largest possible number of vertices of a graph G with maximum degree Δ and diameter D. For $\Delta > 2$ the Moore bound is

$$M(\Delta, D) = 1 + \Delta \frac{(\Delta - 1)^D - 1}{\Delta - 2}.$$

A graph whose number of vertices equals the Moore bound $M(\Delta, D)$ is called a Moore graph. This paper focuses on regular Moore graphs of diameter 2 and degree k.

Definition 1. A regular Moore graph of diameter 2 and degree k is a simple undirected graph with the maximum number $1 + k^2$ of vertices.

In 1960 Hoffman and Singleton [3] proved that such a Moore graph exists if $k = 2$ (the pentagon), $k = 3$ (the unique Petersen graph with 10 vertices shown in Fig. 1), $k = 7$ (the unique Hoffman-Singleton graph with 50 vertices) or possibly $k = 57$ (the open case). This last case is referred to in [2] as the "missing Moore graph". Further information on Moore graphs and the degree 57 case can be found in [1,2,4].

Let $\Gamma(u)$ denote the neighbourhood of vertices at distance 1 from any vertex u and $\Gamma_2(u)$ denote the set of vertices at distance 2 from vertex u. Construction of a Moore graph will start from a tree with $|\Gamma(u)| = k$ and $|\Gamma_2(u)| = k(k-1)$ as shown in Fig. 2. The edges to be added must all be incident with two vertices of $\Gamma_2(u)$.

The structure of the Moore graph of diameter 2 and degree k is also represented by the intersection matrix: $\begin{pmatrix} 0 & 1 & 0 \\ k & a_1 & c_2 \\ 0 & b_1 & a_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ k & 0 & 1 \\ 0 & k-1 & k-1 \end{pmatrix}$ (see [1]). The integers

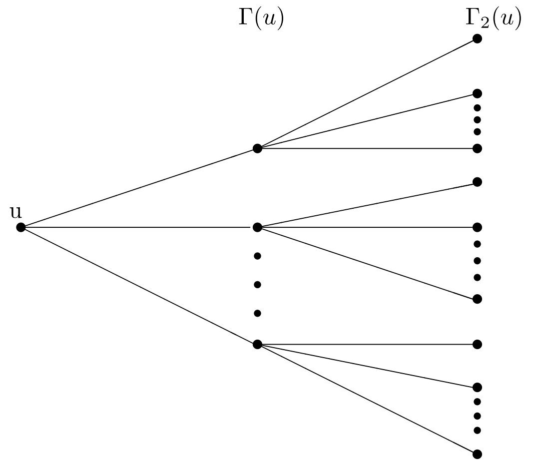

Fig. 2. Starting tree with $1 + k^2$ vertices. The Petersen graph shown in Fig. 1, for example, can be redrawn highlighting such a starting tree independent of which vertex is chosen as root.

c_i, a_i, b_i are defined as follows: let u and v be two vertices for which $d(u, v) = i$, then the number of vertices w such that $d(w, v) = 1$ and $d(w, u) = i - 1, i, i + 1$ respectively is defined to be c_i, a_i, b_i respectively. For a regular Moore graph of degree k and diameter 2 the matrix must be 3×3 and the column sums are k. Note also that a_1 must be 0 and c_2 must be 1 or the number of vertices would be strictly less than $1 + k^2$. Thus the intersection matrix shows that each vertex of $\Gamma(u)$ is adjacent to u and to $k - 1$ vertices of $\Gamma_2(u)$, and also that each vertex of $\Gamma_2(u)$ is adjacent to 1 vertex of $\Gamma(u)$ and $k - 1$ vertices of $\Gamma_2(u)$. In fact the eigenvalues of the intersection matrix are the distinct eigenvalues of the adjacency matrix A [1]. These are k and $(-1 \pm \sqrt{(4k-3)})/2$ (λ_1 and λ_2 say).

As $(A^2 - kI)_{ij}$ gives the number of paths of length 2 from vertex i to vertex j it can be seen that

$$A^2 + A - (k-1)I = J$$

so

$$(A^2 + A - (k-1)I)(A - kI) = J(A - kI) = 0$$

and the minimal polynomial of A is

$$(x^2 + x - (k-1))(x - k)$$

giving an alternative derivation of the eigenvalues, not needing properties of the intersection matrix.

As readers with expertise in optimization may be unfamiliar with ideas in algebraic graph theory, the proof of existence of these Moore graphs [3,4] will be indicated here. As A is a $(1+k^2)\times(1+k^2)$ matrix, the multiplicities m_1 and m_2 of λ_1 and λ_2 satisfy:

$$m_1 + m_2 = k^2 \tag{1}$$

As the sum of the eigenvalues of a matrix equals the trace (the sum of the elements on the main diagonal) and A has all diagonal elements zero, the multiplicities m_1 and m_2 of λ_1 and λ_2 satisfy:

$$0 = trace(A) = k + m_1\lambda_1 + m_2\lambda_2 \tag{2}$$

If λ_1 and λ_2 are irrational then $m_1 = m_2 = k^2/2$ so Equation (2) gives $k = 2$. If λ_1 and λ_2 are rational then let $4k - 3 = m^2$, where m must be an integer. Solving Equations (1) and (2) using $k = (m^2 + 3)/4$ gives the equation:

$$m^5 + m^4 + 6m^3 - 2m^2 + (9 - 32m_1) - 15 = 0 \tag{3}$$

From the constant term -15 it can be seen that the integer m must be a divisor of 15, so m =1, 3, 5,or 15, giving $k = 1$, 3, 7, or 57. Ignoring the trivial case of $k = 1$ gives the required result.

2. Near Moore graphs and a different formulation

Miller and Sirán [4] describe extensive work to find graphs smaller than, but close to, the Moore bound (here $1 + k^2$). In this work a quite different approach is adopted. Recalling that the girth g of a graph is the length of a smallest cycle, it can be seen from the intersection matrix that a regular graph of degree k and girth 5 with $1 + k^2$ vertices is necessarily a Moore graph of diameter 2. These degree and girth conditions are the properties of the Moore graph used in the algorithm presented here. Consider non-regular graphs of girth 5 with maximum degree k, $|E|$ edges and $1 + k^2$ vertices. Define the *deficit* of such a graph as $k(k^2 + 1)/2 - |E|$. The challenge is to minimize the deficit. If some algorithm reduces the deficit to zero for $k = 57$ the missing Moore graph will have been constructed.

3. Algorithmic approaches

Before development of the main algorithm described below, both a genetic algorithm and a tabu search approach were considered. However, any check for small cycles when adding edges is demanding, unless it fails early. Thus maintaining multiple candidate solutions in a genetic algorithm would be expensive. Similarly, worsening moves in a tabu search algorithm would be expensive. It should also be noted that cycling is very unlikely given the nature of the search space.

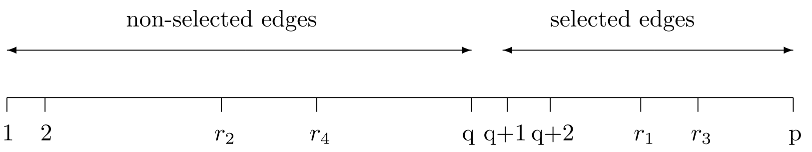

Fig. 3. Layout of ordered list of potential edges.

3.1. The main algorithm

It is clear that the algorithm can start from a tree as in Fig. 2, where the root vertex u has degree $k = 57$ and each vertex of $\Gamma(u)$ is also adjacent to $k-1 = 56$ leaf vertices in $\Gamma_2(u)$ with all leaf vertices distinct. Also, without loss of generality, it was possible to add a further $(k-1)^2+1$ edges as follows. After labelling the leaf vertices of the tree from $k+2$ to k^2+1, these edges were $(i, i+j(k-1)), i \in \{k+2, \ldots, 2k\}, j \in \{1, \ldots, k-1\}$. The algorithm then took the remaining edges incident with two leaf vertices, but ignored edges that would immediately increase the maximum degree to $k+1$ or create a cycle of length < 5. There remained $p = 4743199$ *potential edges* that could be added in order to reduce the deficit. These were arranged in an ordered list. In any current solution, the potential edges that are in the solution are referred to as the *selected edges*, the potential edges not in the solution are *non-selected edges*. At the general step, when the current solution has $p-q$ selected edges and q non-selected edges the list was always re-ordered with the non-selected edges first (otherwise in the order in which they appeared) and the selected edges second (again in the order in which they appeared).

The objective function to be reduced by the algorithm is the deficit of the current solution. At the first iteration, and at subsequent iterations after a permutation to be defined below, a new solution is created greedily as follows. Non-selected edges are added starting from the last in the list down to the first, provided they do not cause the maximum degree to exceed k and do not create a cycle of length 3 or 4. This last check accounts for around 94% of the run time. If the deficit does not increase the new order of potential edges (re-ordered as above) and deficit is accepted. Otherwise the previous order and deficit is retained for the next iteration.

Initially the permutation consisted of a single transposition of potential edges. With the relevant parameters this algorithm was able to find the unique Hoffman-Singleton graph of degree 7 in a matter of seconds. The degree 57 case is much more demanding. As progress slowed the single transposition was replaced by permutations of the list consisting of a number t of disjoint directed cycles, as illustrated for $t = 2$ in Fig. 3.

Here there are $t = 2$ disjoint directed cycles defined by 4 distinct random numbers r_1, r_2, r_3, r_4, with r_1, r_3 selected edges and r_2, r_4 non-selected edges. The permutation was defined by $(1, r_2, q+1, r_1)(2, r_4, q+2, r_3)$. The generalisation for $t > 2$ distinct directed cycles should be clear, with the i'th directed cycle being $(i, r_{2i}, q+i, r_{2i-1})$. Then $|selected|$ reduces by t, $|non-selected|$ increases by t and the effect of the permutation is to change t of the selected edges to non-selected edges and to reorder the list of non-selected edges. The rationale behind the choice of permutation is to remove edges randomly from the current solution and give priority to the insertion of non-selected edges over the re-insertion of removed edges. Eventually it was found that t selected randomly in the range $200 \leq t \leq 400$ was a good choice. If t was larger than 400 the solutions tended to become much worse.

3.2. Restarting and parallelism

An advantage of the algorithm is that it can be restarted and continued from a current solution. Every few hundred iterations the current minimum deficit and the order of potential edges that gave it can be written to a file, and used for restarting. This also allowed a parallel version. Up to 8 processes were run with access to a single common file. Every few hundred iterations the file was examined. If the current minimum deficit of the process was smaller than that in the file, the deficit and the order of potential edges that gave it were written to the file. If the current minimum deficit of the process was greater than that in the file, the deficit in the file and the order of potential edges recorded in the file were used by the process in subsequent iterations.

3.3. A useful check

The implementation of the main algorithm allows the output of a symmetric (3250×3250) $(0,1)$-matrix that should be the adjacency matrix of a valid solution. This allows a useful check. If a solution with deficit 0 has been found, the row sums of this matrix should be 57 and $A^2 + A - (k-1)I = J$ should be satisfied as described in Section 1. For a solution with deficit > 0 the relevant check is that the row sums should be $\leq$57 and $(A^2 + A - (k-1)I)_{ij} \leq 1 \; \forall\{i,j\}$.

3.4. Results

The longest run of the main algorithm extended over around 18 months, with several improvements to the implementation to speed the algorithm in the early part of this time. For most of this time the parallel version of Section 3.2 was used, with 8 processes running in parallel. The computer used was an Intel(R) Core(TM) i7-4770 CPU with 4 cores, and 8 logical processors running at 3.40 GHz and with 8Gb RAM. A small number t of disjoint directed cycles was used for the first months, but as the solution improved the choice of a randomly selected t in the range $200 \leq t \leq 400$ seemed most effective. Improvements to the deficit were fast initially, but slowed as the deficit reduced, with the last improvement taking 29 days. It appeared that there were fewer and fewer permutations of t disjoint directed cycles giving an improvement as the algorithm proceeded. The final deficit obtained was 41482. This value is so large that a deficit of 0 may be unachievable.

The deficit begins at around 48000 and (with the latest version of the algorithm) will reduce to around 43000 in under a day. The reduction from 43000 to 41482 over time is

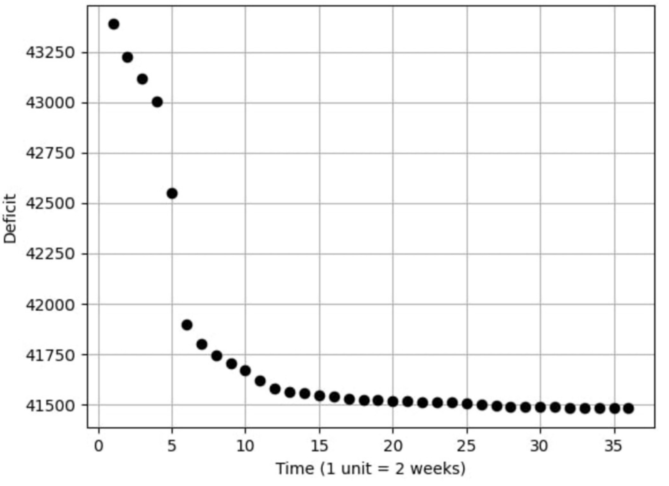

Fig. 4. Reduction of deficit with time. One unit of time is approximately two weeks.

illustrated in Fig. 4. Initially some improvements to the implementation are apparent. Then it can be seen that progress slows markedly, with the algorithm apparently converging to 41482 (or perhaps a value slightly smaller than 41482 given an even longer run). The important question arises of whether 41482 is (or is close to) a local minimum or a global minimum. The run described above started from the order of potential edges in the order they were generated. To provide some evidence of whether there may be much better solutions in a distant part of the search space a further 8 processes were started, each beginning from a different random order of potential edges. It should be noted that these were independent processes, not the parallel version described in Section 3.2. In consequence progress may be even slower. After 85 days of computation the deficits of the 8 processes are 42364, 42347, 42375, 42349, 42377, 42378, 42379, 42390, and progress has slowed in a similar way to the original run. This suggests that they may all converge to a similar value to the original run.

3.5. An experiment with subsolutions

An experiment was applied to the final two solutions of the main run, with deficits 41483 (51142 edges) and 41482 (51143 edges) respectively. The permutation described in Section 3.1 was applied to the initial solution for several hours, for each value of $t \in \{1000, 2000, \ldots, 15000\}$. The initial aim was to determine the solution (with t fewer selected edges) that gave the maximum number e of non-selected edges that could be added individually to the reduced solution. For this solution the selected edges were fixed and a slightly modified version of the main algorithm was used to recover a good solution (or possibly improve the starting solution). The results are shown in Table 1, including the timings for the second part of the algorithm.

Table 1
Results of subsolution experiments.

Initial deficit	t	e	final deficit	time to final deficit (seconds)	total run time (seconds)
41483	1000	1034	41483	27	26820
41483	2000	2113	41483	105	23400
41483	3000	3281	41483	200	21180
41483	4000	4599	41483	410	21000
41483	5000	6144	41483	790	35460
41483	6000	8069	41483	1170	29940
41483	7000	10603	41483	860	20340
41483	8000	14042	41483	1500	12000
41483	9000	18713	41483	2230	36960
41483	10000	24843	41483	2340	22620
41483	11000	33556	41483	4560	31200
41483	12000	45163	41483	6060	41280
41483	13000	60877	41483	25200	32400
41483	14000	80283	41483	28800	50400
41483	15000	107102	43767	154800	154800
41482	1000	1035	41482	45	30600
41482	2000	2110	41482	180	22860
41482	3000	3270	41482	240	16680
41482	4000	4573	41482	240	37200
41482	5000	6148	41482	360	19200
41482	6000	8074	41482	600	30780
41482	7000	10539	41482	720	31320
41482	8000	14102	41482	1200	29340
41482	9000	18594	41482	1740	31920
41482	10000	24943	41482	1620	20100
41482	11000	33562	41482	2820	10800
41482	12000	45362	41482	2700	30600
41482	13000	61021	41482	5400	68400
41482	14000	81040	41482	24000	35220
41482	15000	106680	44710	194400	194400

The results are consistent between the two starting solutions. For $t \leq 14000$ a solution with the same deficit as the starting solution is found quickly, although in general it is not the same solution as the starting solution. No improvements were found. For larger t the problem of recovering a good solution appears to become very difficult. Two conclusions can be drawn. Firstly, the main algorithm appears to be very effective for the subproblems addressed. Secondly, it appears that if a solution has more than 73% of the edges of a good solution, then the minimum deficit achievable is essentially determined.

3.6. A heuristic

Prior to the experiments described in Section 3.5 a promising heuristic was investigated. Given a solution of deficit 41483 or 41482 a fixed permutation as described in Section 3.1 with $t \leq 3200$ was applied. Then each non-selected edge was considered in turn. After temporarily adding it to the set of selected edges the number of edges that could be added individually at the next step was counted. For one edge that gave the maximum number, the addition to the set of selected edges was confirmed and the iteration was repeated. Other edges were removed from the set of selected edges after consideration.

The heuristic appeared promising, often recovering the starting deficit, but never finding an improvement. For larger values of t run times were excessive. The experiment in Section 3.5 explains why no improvement was found for practical values of t. If the heuristic could be applied from the beginning (with $|selected| = 0$) it might be very promising. However, even the selection of a single edge requires $(p - |selected|)(p - |selected| - 1)$ checks for cycles of length 3 or 4, and so is impractical. A variation of this heuristic starting with $|selected| = 0$ is practical. Take a starting tree with maximum degree 57, root u, $\Gamma(u) = 57$ and $\Gamma_2(u) = 3192$. Note that of the 56 potential edges joining a leaf of one branch to the leaves of a different branch, only one can be selected in a solution or a cycle of length 4 is created. Taking these sets of 56 edges in order, and using the criterion of the heuristic to select the best, this variation only takes around 30 hours. The variation is not effective though, giving a deficit of 44410.

4. Discussion

When considering whether the local minimum of, or close to, 41482 provides strong evidence for the non-existence of the Moore graph it is useful to consider whether improved solutions would be rare. Assume that the Moore graph of degree 57 exists (with 92625 edges, so the deficit is 0). It is possible that it would be unique. However, there will be many solutions with deficit < 41482 simply by taking subsets of between 51144 and 92624 edges. Furthermore, for each such solution except those with deficit very close to 41482 it is possible to add edges as follows. Add an edge not in the Moore graph, which may lead to a breaking of the degree condition or the girth ≥ 5 condition. It requires at most 58 edges to be removed to restore the conditions, so the deficit increases by at most 57. This can be repeated until the deficit becomes too close to 41482. Thus solutions with deficit < 41482 will not be rare. It should also be noted that even when the algorithm described here struggles to find an improved solution, it finds it very easy to obtain different solutions with unchanged deficit. This must be a result of the existence of more complex rearrangements of edges in incomplete solutions. Even if a solution with deficit 0 is well hidden in the search space, it does not seem plausible that the multiplicity of solution with deficit less than 41482 described above could all be well hidden.

5. The challenge

Using the new formulation, the algorithm presented here appears to work well and suggests non-existence of the Moore graph. There are many other optimization algorithms that could be applied that may support this suggestion. Alternatively, they may find massively reduced deficits or maybe construct the Moore graph. Even if the Moore graph does not exist, the task of finding the minimum possible deficit is an interesting and challenging optimization problem. It would be particularly useful if highly parallel algorithms were developed by those with access to suitable computer resources.

6. Conclusions

A formulation has been presented and an algorithm developed that attempts to find the Moore graph of diameter 2 and degree 57. The main algorithm and related algorithmic work tend to support the view that the Moore graph does not exist. A challenge to other authors to support or counter this claim is proposed.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Norman Biggs, Algebraic Graph Theory, Cambridge Tracts in Mathematics, vol. 67, Cambridge University Press, Cambridge, UK, 1974.
[2] Cristina Dalfó, A survey on the missing Moore graph, Linear Algebra Appl. 569 (2019) 1–14.
[3] A.J. Hoffman, R.R. Singleton, On Moore graphs with diameters 2 and 3, IBM J. Res. Dev. 4 (5) (1960) 497–504.
[4] Mirka Miller, Jozef Sirán, Moore graphs and beyond: a survey of the degree/diameter problem, Electron. J. Comb. DS14-May (2013).