Geodetic Graphs: Experiments and New Constructions

Florian Stober   Armin Weiß

Submitted: Mar 11, 2025; Accepted: Apr 20, 2026; Published: Sep 11, 2026

Abstract

In 1962 Ore initiated the study of geodetic graphs. A graph is called geodetic if the shortest path between every pair of vertices is unique. In the subsequent years a wide range of papers has appeared investigating their peculiar properties. Yet, a complete classification of geodetic graphs remains out of reach.

In this work we present an exhaustive search algorithm for enumerating all biconnected geodetic graphs of a given order. Using our program, we find all geodetic graphs with up to 25 vertices and all regular geodetic graphs with up to 32 vertices. This leads to the discovery of a new infinite family of geodetic graphs. Moreover, we present a new result on constructing geodetic subdivisions of geodetic graphs.

Mathematics Subject Classifications: 05C12, 05C30, 05C85, 68R10

1 Introduction

An undirected graph is called geodetic if for any pair of vertices the shortest path between them is unique (if it exists). Research on geodetic graphs began in 1962, when Ore posed the problem of classifying all such graphs [17]. This goal has been achieved for planar geodetic graphs and geodetic graphs of diameter two [23, 25]. Yet, after decades of active research, a full classification of finite geodetic graphs or any characterization by other means has not been attained. Recently, there has been renewed interest in the problem, with several publications attacking the problem from different angles [7, 8, 9] and an application of geodetic graphs in the construction of so-called f-edge fault-tolerant distance sensitivity oracles [1].

The present work complements this by studying small geodetic graphs in the hope of uncovering more of their mysterious properties. Since there are only few known geodetic graphs, the study of small geodetic graphs seems important to us as it might lead to further insights about the structure of geodetic graphs and more general constructions. Indeed, we discovered a new infinite family of geodetic graphs. Moreover, we hope that

University of Stuttgart, Germany (florian.stober@fmi.uni-stuttgart.de, armin.weiss@fmi.uni-stuttgart.de).

having a complete list of small geodetic graphs readily available will be useful for and inspire future research. Since a graph is geodetic if and only if each maximal biconnected subgraph is geodetic, the focus is on biconnected geodetic graphs.

1.1 Contribution

One of our main contributions is a complete list of biconnected geodetic graphs with at most 25 vertices. The scarcity of geodetic graphs is reflected in our results: the list contains only 149 graphs compared to the vast amount of far above 10^{50} biconnected graphs of up to 25 vertices. Most of the geodetic graphs we obtained are subdivisions of smaller geodetic graphs that were already known to be geodetic.

However, our list includes a previously unknown graph, which we denote $H(2,2,2,0)$ and describe in Section 3. We study its relation to existing constructions and, based on that, succeed in discovering a new infinite family of geodetic graphs.

Furthermore, we derive a new theorem on subdivisions of geodetic graphs (Theorem 8). This theorem gives conditions under which the subdivision of a geodetic graph is again geodetic. It generalizes both the construction of subdividing each edge of a geodetic graph by an even number of vertices (see [18]) as well as the construction of subdividing each edge of a cut fulfilling certain properties (see [11] and Theorem 7 below) by allowing a partition into an arbitrary number of connected components. Thus, Theorem 8 is a step towards a full characterization of geodetic subdivisions.

To compute our list of geodetic graphs, we introduce a novel algorithmic approach to the problem of enumerating geodetic graphs. The main idea behind our algorithm is to enumerate all shortest-path trees and then, by a backtracking search, fill in the structure of the graph with the remaining edges. After each step, we perform pruning operations based on several theoretical results in order to check whether the graph can still be extended to a geodetic graph. In addition, we use the graph isomorphism tool `nauty` [15] to identify isomorphic branches in the search space.

Finally, we consider the special case of regular graphs. It has been conjectured that a finite Cayley graph is geodetic if and only if it is either complete or an odd cycle [8, Conjecture 6]. This gives new interest in the special case of regular geodetic graphs. Using our search algorithm, we are able to show that, up to 32 vertices, there are only the previously known regular geodetic graphs: odd cycles, complete graphs, the Petersen graph and one graph with 28 vertices discovered in [4].

1.2 Related Work

So far a characterization of geodetic graphs has been achieved for only two classes of graphs. Stemple and Watkins, both students of Ore, showed that a planar graph is geodetic if and only if each maximal biconnected subgraph is either a single edge, a cycle of odd length, or a geodetic graph homeomorphic to the complete graph K_4 [25] (meaning that it is a subdivision of K_4).

Stemple also obtained a characterization of geodetic graphs of diameter two by describing vertex degrees and number of cliques [23]. Later an equivalent characterization in terms of so-called π-spaces has been obtained [21] and explicit constructions using projective and affine planes were discovered [2].

A related question asks which geodetic graphs are homeomorphic to a given geodetic graph. Starting with a complete graph K_n, labelling its vertices with integers and then subdividing each edge according to the sum of labels of its endpoints, we obtain a homeomorphic geodetic graph [19]. In fact, this construction gives us all geodetic graphs homeomorphic to K_n [24].

The Widespread Petersen graph WP_k is obtained from the Petersen graph: Each edge connecting the inner to the outer pentagon is subdivided k times. The graph WP_k is geodetic [19]; however, there are other geodetic graphs homeomorphic to the Petersen graph [9].

A very simple way of obtaining a geodetic graph from a given one is to subdivide each edge with an even number of vertices [18]. In fact, that result is a special case of a more general construction: from a geodetic graph G and a clique cover of the edge set the authors of [18] construct a geodetic graph G^*. However, in the general case, G^* is not homeomorphic to G. That construction is a special case of geodetic graphs constructible using the operation of pulling subgraphs homeomorphic to complete graphs, presented in [20].

Another family $h(m, n, s)$ of geodetic graphs of arbitrary diameter has been presented by Bosak in [4]. Nevertheless, these graphs are homeomorphic to diameter-three graphs. The only known infinite families of geodetic graphs with larger diameter and without degree-two vertices are of diameter 4 and 5 [5, 22, 7].

A different approach to finding infinite families of geodetic graphs is to search for *all* geodetic graphs up to a certain number of vertices. Very little research has been done in this direction. Still, as part of an undergraduate research project, John Cu computed the biconnected geodetic graphs with up to 11 vertices [6] (see also A337179 and A337178 in OEIS [16]). However, it is difficult to generalize this approach to larger graphs due to the fact that the number of graphs on n vertices grows proportional to $2^{\Theta(n^2)}$.

2 Preliminaries

Throughout this paper we consider only undirected finite simple graphs. We assume the reader is familiar with the basics of graph theory and use notation similar to [3]. In particular, we write uv to denote an edge connecting vertices u and v. We denote the distance (i.e., the length of a shortest path) between vertices u and v by $d(u, v)$ or, to clarify in which graph the distance is measured, by $d_G(u, v)$. A *geodesic* from u to v is a shortest path. A graph G is called *geodetic* if for all $u, v \in V$ there is at most one geodesic from u to v. A graph G is *biconnected* if G is connected and for every $v \in V$ the subgraph induced by $V \setminus \{v\}$ is connected.

Note that a graph is geodetic if and only if every maximal biconnected subgraph (block) is geodetic [25]. We introduce some basic lemmas used in our search algorithm as well as in the constructions below.

Lemma 1 (Theorem 3.3 in [23]). *Let $G = (V, E)$ be a geodetic graph. If four vertices $u, v, w, x \in V$ form a 4-cycle, then they induce a complete subgraph.*

Lemma 2 (Theorem 3.5 in [23]). *Let $G = (V, E)$ be a geodetic graph, $C \subseteq V$ a clique in G, and v adjacent to at least two distinct vertices in C. Then $C \cup \{v\}$ is a clique.*

We denote by $\mathcal{N}^s(v) = \{u \mid d(v, u) = s\}$ the set of vertices at distance s from v. Note that, $\mathcal{N}^1(v)$ is the set of neighbours of v. A consequence of the previous two lemmas is the following lemma, describing the neighbourhood of a vertex in a geodetic graph.

Lemma 3. *Let $G = (V, E)$ be a geodetic graph and $v \in V$ a vertex. Then $\mathcal{N}^1(v)$ can be partitioned into a set of disjoint cliques.*

2.1 Shortest-Path Tree

Lemma 4 (Unique Predecessor Theorem [18]). *A graph $G = (V, E)$ with diameter d is geodetic if and only if for every $v \in V$ each point of $\mathcal{N}^s(v)$ has a unique neighbour in $\mathcal{N}^{s-1}(v)$ for each s with $2 \leqslant s \leqslant d$.*

Theorem 4 leads to the notion of a shortest-path tree: Let G be a connected geodetic graph and $r \in V$. Let $T \subseteq E$ be the set of all edges occurring on some shortest path from r to any other vertex (i.e., we take the union of all shortest paths starting at r). From Theorem 4 it follows that the graph (V, T) is a tree – the *shortest-path tree* rooted at r (we also call simply T the shortest-path tree when there is no risk of confusion). We can assign a *level* to each vertex $v \in V$, which is given by its distance from the root r. Another consequence of Theorem 4 is that, other than T, the edge set of G contains only edges between vertices on the same level of T. A less immediate consequence is as follows.

Lemma 5. *Let G be a geodetic graph with a shortest-path tree T. If there is an edge uv between vertices on the same level, then for no u' in the subtree rooted at u and v' in the subtree rooted at v there is an edge $u'v'$.*

Figure 1 illustrates Theorem 5: If the edge e exists, then the dashed edge f is not allowed.

Proof. First, observe that, if u' and v' are on different levels of T, an edge $u'v'$ would contradict the fact that T is a shortest-path tree of a geodetic graph. Now, let d be the length of the u-u' path in the shortest-path tree. Then d is also the length of the v-v' path in the shortest-path tree. Now, there are two distinct paths from u to v' of length $d + 1$: one path via u' and one path via v. Since the shortest path must be unique, it can have length at most d, but that is not possible without introducing at least one additional edge between two vertices on different levels of the shortest-path tree. □

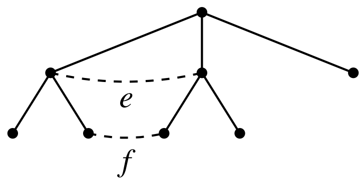


Figure 1: In a geodetic graph we cannot have both the edges e and f.

2.2 Cliques and Independent Sets.

By [10, Corollary 1], every biconnected non-complete geodetic graph containing a clique of size m also contains $K_{1,m}$ (the complete bipartite graph with partitions of size 1 and m) as an induced subgraph. Here, we need a related but slightly different statement, which can be shown using similar ideas as the proof of [10, Corollary 1].

Proposition 6. *Let $G = (V, E)$ be a biconnected geodetic graph containing a clique H. Then G is either complete or every vertex $u \in H$ has a neighbour x_u outside of H such that $\{x_u \mid u \in H\}$ is an independent set of size $|H|$.*

Proof. Assume that G is not complete. If H is not a maximal clique, we can replace H with a maximal clique containing H. Hence, from now on, we may assume that H is maximal.

From here on, our proof has two major steps. First we show that there is a vertex $v_* \in V$ such that there are at least two vertices $x \neq y \in H$ with $d(v_*, x) = d(v_*, y) = d(v_*, H)$ (where $d(v_*, H) = \min\{d(v_*, h) \mid h \in H\}$). The second step is to show that v_* has distance $d(v_*, H)$ to all vertices in H and then conclude the statement of the proposition.

Let v_1 be a vertex that is not in H, but has a neighbour in H. We know such a vertex exists because G is connected, but not complete. We denote by u_1 a vertex in H that is adjacent to v_1. When removing u_1, there must still be a path from v_1 to each of the remaining vertices in H since G is biconnected. We pick one such path and call the vertices on that path $v_1, \ldots, v_k, u_k$, where $u_k \in H$. We may assume $v_i \notin H$ for $1 \leqslant i \leqslant k$ – otherwise we consider the shorter path.

Now we make some definitions. Let $d_i = d(v_i, H)$ be the distance of v_i to the clique. Let $S_i = \{h \in H \mid d(v_i, h) = d_i\}$ be the set of witnesses. If there is an $i \in \{1, \ldots, k\}$ such that $|S_i| \geqslant 2$, then we have found our vertex $v_* = v_i$ that has distance $d(v_*, H)$ to two vertices in H (and we can continue with the next step of the proof).

Now assume that $|S_i| = 1$ for all $1 \leqslant i \leqslant k$. We will show that this leads to a contradiction. Observe that $S_1 = \{u_1\}$ and $S_k = \{u_k\}$, as v_1 and v_k both are at distance one to the clique H. Note that $u_1 \neq u_k$ as by construction u_k is a vertex of $G \setminus \{u_1\}$. Thus, $S_1 \neq S_k$ and there is an index i such that $S_i \neq S_{i+1}$. We write $S_i = \{u_i\}$ and $S_{i+1} = \{u_{i+1}\}$ and distinguish three cases:

1. $d_i = d_{i+1}$. Now there are two paths of length $d_i + 1$ from u_i to v_{i+1}: One using the edge $u_i u_{i+1}$ in the clique and then the path from u_{i+1} to v_{i+1}. The other using the path from u_i to v_i and then the edge $v_i v_{i+1}$. Since shortest paths in geodetic graphs

are unique, there must be a path from u_i to v_{i+1} of length at most d_i. In fact, that path must have length d_i, since $d_i = d_{i+1}$ is the minimum distance of v_{i+1} to any vertex in H. Thus, $u_i \in S_{i+1}$, a contradiction.

2. $d_i = d_{i+1}+1$. There is a path of length d_i from u_{i+1} to v_i: Take the path from u_{i+1} to v_{i+1} of length $d_i - 1$ and then use the edge $v_{i+1}v_i$. Thus, $u_{i+1} \in S_i$, a contradiction.

3. $d_i + 1 = d_{i+1}$. The proof is symmetric to the previous case.

Now for the second step of the proof, we already have a vertex $v_* \notin H$ with distance $d = d(v_*, H)$ to the clique H. This is witnessed by $S = \{u \in H \mid d(v_*, u) = d\}$ with $|S| \geqslant 2$. If $d = 1$, then by Theorem 2 the set $H \cup \{v_*\}$ is a clique, contradicting the maximality of H. Thus, $d \geqslant 2$. Let $u_1, u_2 \in S$ and consider a vertex $z \in H \smallsetminus S$. There are two paths of length $d+1$ from z to v_*: For $i \in \{1, 2\}$ first use the edge zu_i and then the path from u_i to v_* of length d. Thus, there must be a shorter path of length at most d. In fact, the length must be precisely d, because d is the minimum distance of v_* to any vertex in H. Hence, $S = H$. Taking two vertices in H, by Theorem 2 their neighbourhoods only overlap in the other clique vertices. Now, for $u \in H$ let x_u denote the first vertex after u on the unique shortest path from u to v_*. Then $I = \{x_u \mid u \in H\}$ is an independent set satisfying the statement of the proposition. □

3 New Constructions of Geodetic Graphs

3.1 Geodetic Subdivisions

A *cut* (S, T) is a partition of V into two subsets S and T. The corresponding *cut set* is the subset of edges with one vertex in S and the other one in T. The following proposition is an important special case of the construction presented in [11]. Indeed, it covers exactly the case that [11] leads to a subdivision of the original graph. For examples of geodetic graphs, where the proposition can be applied, see Figure 2.

Lemma 7 ([11]). *Let $G = (V, E)$ be a geodetic graph and (S, T) be a cut of G with the following properties:*

1. *S and T are geodetically closed*[1] *(meaning that for all $u, v \in S$ the shortest path from u to v does not leave S).*

2. *Given two edges uv and wx from the cut set with $u, w \in S$, $v, x \in T$, then $d(u, w) + d(v, x)$ is odd.*

Then, for every k, subdividing the edges in the cut set by adding k vertices to each results in a geodetic graph G'.

Here, we generalize this result by allowing to cut the graph into multiple connected components. Note that our theorem also generalizes the construction of subdividing each edge of a geodetic graph with an even number of vertices.

[1] A geodetically closed set is also referred to as convex. Both terms appear in literature.

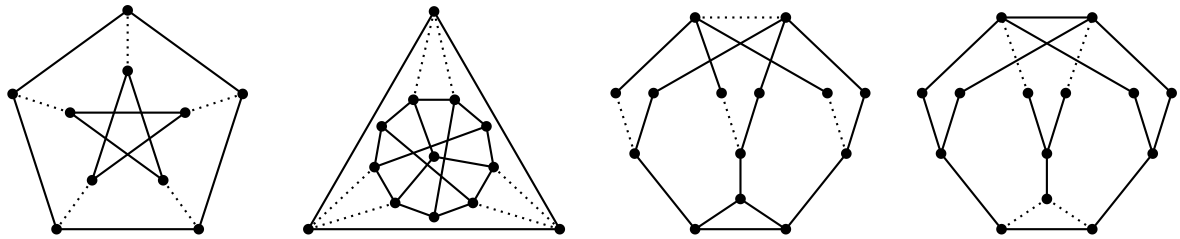


Figure 2: Geodetic graphs with a cut satisfying Theorem 7. Cut edges are dotted.

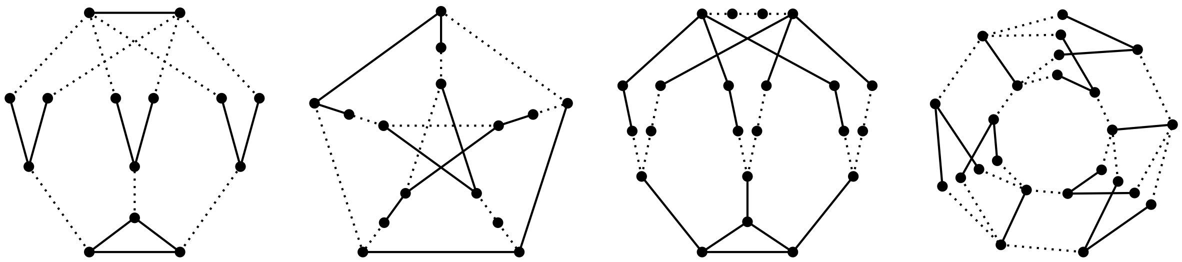


Figure 3: Application of Theorem 8 to different graphs. Edges in C are dotted.

Theorem 8. *Let $G = (V, E)$ be a geodetic graph, $C \subseteq E$, and k an integer. The graph G' obtained by subdividing the edges in C by adding k vertices on each is geodetic if the following conditions are met.*

1. *For each pair $s, t \in V$, the number of edges in C traversed by the geodesic from s to t is at most the number of edges from C traversed by any other path from s to t.*

2. *If there are $s_1t_1, s_2t_2 \in C$ with $d_G(s_1, s_2) = d_G(s_1, t_2) = d_G(t_1, s_2) = d_G(t_1, t_2)$, then each of those four geodesics traverses the same number of edges from C.*

3. *If there is a vertex $v \in V$ and an edge $st \in C$ such that the geodesics from v to s and from v to t traverse the same number of edges in C, then k must be even.*

Observe that Condition 1 of Theorem 8 generalizes the requirement that S and T are geodetically closed in Theorem 7. For examples of geodetic graphs where the theorem can be applied, see Figure 3.

Proof. Let $G' = (V', E')$ where $V \subseteq V'$. For $u, v \in V$ the number of edges from C traversed by the geodesic from u to v in G is denoted by $c(u, v)$. We need to show that every pair of vertices in V' has a unique geodesic in G' connecting them. In order to do so, we make a case distinction on whether $u, v \in V$.

Case $\boldsymbol{u, v \in V}$. Firstly, we consider vertices $u, v \in V$. There is a unique shortest path from u to v in G. By Condition 1, $c(u, v)$ is a lower bound for the number of edges from C in any path from u to v. Thus, the path from u to v in G' obtained from the geodesic in G by subdividing the edges in C is again a geodesic.

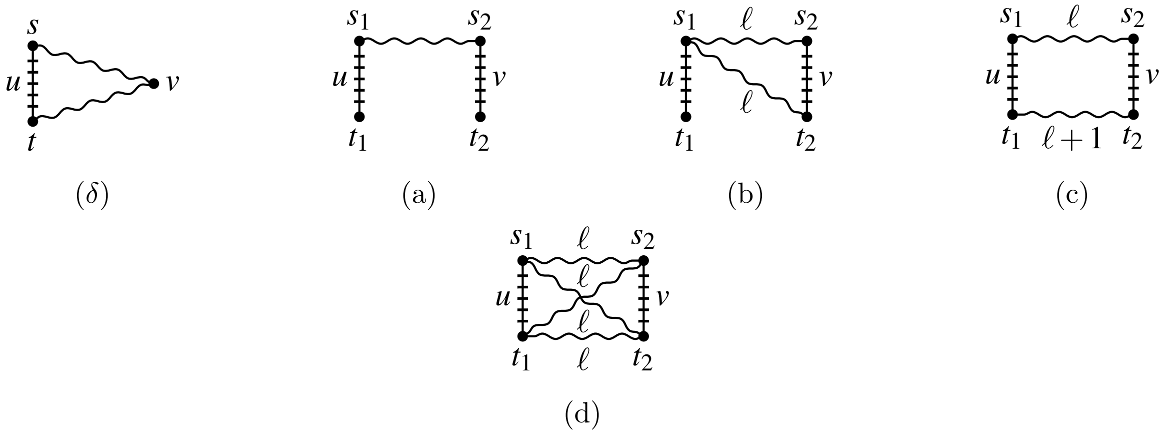


Figure 4: Different situations occurring in the proof of Theorem 8.

Case $u \in V' \setminus V$ and $v \in V$. Secondly, we consider vertices $u \in V' \setminus V$ and $v \in V$. Let $st \in C$ be the edge whose subdivision contains u. This situation is illustrated in Figure 4δ. W.l.o.g. we may assume $d_G(s, v) \leqslant d_G(t, v)$. If $d_G(s, v) < d_G(t, v)$, then the geodesic from t to v in G' traverses u and s. Hence, it contains the geodesic from u to v.

It remains to show that there is a unique geodesic from u to v if $d_G(s, v) = d_G(t, v)$. We have already shown that in G' there are unique geodesics from s to v and from t to v. Any path from u to v must traverse either s or t. If $c(s, v) = c(t, v)$, then $d_{G'}(s, v) = d_{G'}(t, v)$. Additionally, k must be even by Condition 3. Thus, $d_{G'}(u, s) \neq d_{G'}(u, t)$ and there is a unique shortest path.

If $c(s, v) \neq c(t, v)$, then, as a consequence of Condition 1, the difference is exactly 1. Hence, the circuit in G comprising st and the geodesics from s to v and from t to v has odd length and an even number of edges from C. Thus, the corresponding circuit in G' has odd length. As the geodesic from u to v must follow this circuit, it is unique.

Case $u, v \in V' \setminus V$. Thirdly, we consider vertices $u, v \in V' \setminus V$. Let $s_1 t_1 \in C$ be the edge whose subdivision contains u and $s_2 t_2 \in C$ the edge whose subdivision contains v. W.l.o.g. we assume that $\ell = d_G(s_1, s_2)$ is the smallest among $d_G(s_1, s_2)$, $d_G(s_1, t_2)$, $d_G(t_1, s_2)$, and $d_G(t_1, t_2)$. Observe that $\ell \leqslant d_G(t_1, t_2) \leqslant \ell + 2$. In the following, we distinguish four cases based on $d_G(t_1, t_2)$. The case $d_G(t_1, t_2) = \ell + 1$ is split into two further cases.

(a) We begin with the case $d_G(t_1, t_2) = \ell + 2$ as in Figure 4a. Then the unique geodesic from t_1 to t_2 in G' traverses u, s_1, s_2, and v; thus, it contains the unique geodesic from u to v.

(b) Now consider the case where $d_G(t_1, t_2) = \ell + 1$ and the geodesic from t_1 to t_2 traverses s_1, which is illustrated in Figure 4b. (The case where it traverses s_2 is symmetric.) Since we have already shown that the geodesic from v to t_1 in G' is unique, it remains to show that u lies on it. To see this, observe that either s_2 or t_2 must be on the geodesic from v to t_1. Moreover, both the geodesics from s_2 to t_1 and from t_2 to t_1 traverse s_1. Hence, the geodesic from v to t_1 traverses s_1 and, thus, also u.

(c) Next, we consider the case where $d_G(t_1, t_2) = \ell + 1$ but the geodesic from t_1 to t_2 traverses neither s_1 nor s_2 (Figure 4c). Note that this implies $d_G(s_1, t_2) = \ell + 1$. Furthermore, observe that the shortest path from u to v needs to use either the geodesic from s_1 to s_2 or the geodesic from t_1 to t_2. Applying Condition 1 to the geodesic from s_1 to t_2, we obtain that $c(s_1, s_2) \leqslant c(t_1, t_2)$. Moreover, by Condition 1 applied to the geodesic from t_1 to t_2, we have $c(t_1, t_2) \leqslant c(s_1, s_2) + 2$.

If $c(s_1, s_2) = c(t_1, t_2)$, then the circuit in G comprising the edge s_1t_1, the geodesic from t_1 to t_2, the edge t_2s_2, and the geodesic from s_2 to s_1 has odd length and an even number of edges from C. Thus, the corresponding circuit in G' has odd length, and hence, the geodesic from u to v is unique.

If $c(s_1, s_2) + 1 = c(t_1, t_2)$, then that circuit in G still has odd length, but it now uses an odd number of edges from C. However, Condition 3, applied to t_1 and s_2t_2, now restricts k to be even; thus, the corresponding circuit in G' still has odd length, and hence, the geodesic from u to v is unique.

If $c(s_1, s_2) + 2 = c(t_1, t_2)$, then the unique geodesic from u to v traverses s_1 and s_2. This is because $d_{G'}(t_1, t_2) = \ell + 1 + kc(t_1, t_2)$, while the path connecting t_1 to t_2 via s_1 and s_2 (and, thus, u and v) is of length $\ell + 2 + kc(t_1, t_2)$, i.e., only one edge longer.

(d) Lastly, consider the case $d_G(t_1, t_2) = \ell$. As G is geodetic, this implies $d_G(s_1, t_2) = d_G(t_1, s_2) = \ell$. This situation is illustrated in Figure 4d. From Condition 2 it follows that $c(s_1, s_2) = c(s_1, t_2) = c(t_1, s_1) = c(t_1, t_2)$. Therefore, $d_{G'}(s_1, s_2) = d_{G'}(s_1, t_2) = d_{G'}(t_1, s_1) = d_{G'}(t_1, t_2)$. Condition 3 restricts k to be even; hence, we obtain a unique geodesic from u to v that passes through the vertex of $\{s_1, t_1\}$ that is closer to u and of $\{s_2, t_2\}$ the vertex closer to v. □

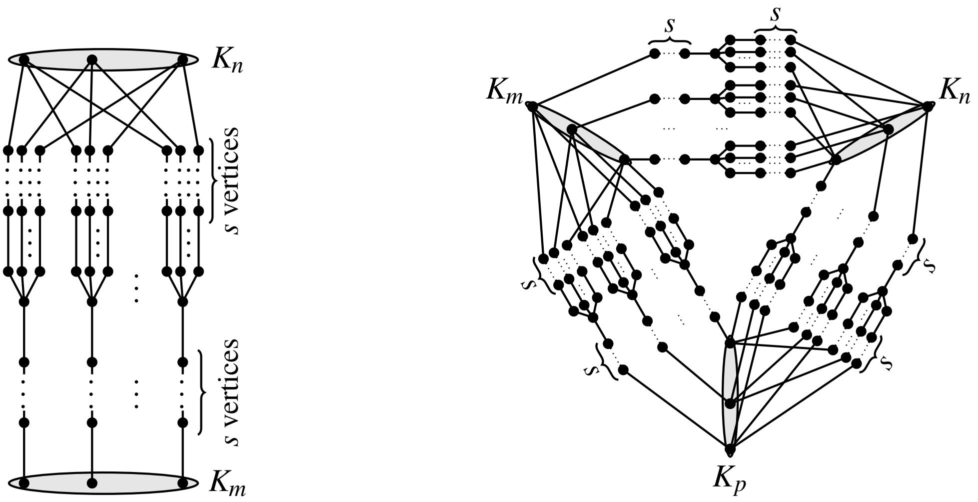
3.2 Joining Three Cliques.

Before describing the family $H(m, n, p, s)$ of geodetic graphs, we look at the family $h(m, n, s)$ described in [4]. It will be the starting point for our own construction. The parameters m, n, and s are integers with $m, n \geqslant 2$ and $s \geqslant 0$. The graph $h(m, n, s)$ is constructed from a K_m, a K_n, and m copies of a star with n leaves. The subgraphs are joined by paths of length $s + 1$ as can be seen in Figure 5a.

The graph $H(m, n, p, s)$ is obtained by taking an $h(m, n, s)$, an $h(n, p, s)$, and an $h(p, m, s)$. The subgraphs are joined together by identifying the cliques. For instance, each vertex of the K_n in $h(m, n, s)$ is identified with the corresponding vertex of the K_n in $h(n, p, s)$. Figure 5b illustrates the construction.

Proposition 9. *The graph $H(m, n, p, s)$ is geodetic for all $m, n, p \geqslant 2$ and $s \geqslant 0$.*

Proof. Take two vertices in the subgraph $h(m, n, s)$ and a path joining them. If the path contains vertices not part of the subgraph $h(m, n, s)$, then it is not a shortest path: If a path segment leaves and enters the subgraph through the same clique, then the segment can be replaced with a clique edge. If it leaves through the K_m and enters through the K_n (or vice versa), then the segment has length at least $4s+6$ and can be replaced by the



(a) $h(m, n, s)$ (b) $H(m, n, p, s)$

Figure 5: The family $h(m, n, s)$ [4] and the newly discovered family $H(m, n, p, s)$.

shortest path inside the subgraph $h(m, n, s)$, which has length $2s + 3$. Since $h(m, n, s)$ is geodetic, shortest paths between two vertices in the subgraph are unique. By symmetry this also applies to the subgraphs $h(n, p, s)$ and $h(p, m, s)$.

Now consider two vertices u and v not in the same subgraph and assume that there are two distinct paths of length d from u to v. We have to show that then the shortest path has length less than d: The two paths form a cycle of even length. The smallest even cycle not contained in one of the subgraphs has diameter $3s+5$ (i.e., length $6s+10$). This is because it either must traverse one of the subgraphs entirely from left to right and back and one of the subgraphs until the middle or it must go around the complete "cycle" once (see Figure 5b). Thus, $d \geqslant 3s + 5$. However, the diameter of $H(m, n, p, s)$ is $3s + 4$; thus, the shortest path from u to v has length at most $3s + 4$ and is unique. □

4 Generating Geodetic Graphs Using `nauty`

John Cu computed the biconnected geodetic graphs with up to 11 vertices as part of his undergraduate research project [6]. In this section, we reproduce his results by a computer search using `nauty`, a popular graph isomorphism library by B. McKay [15]. Here we rely on `nauty`'s `geng` utility: we generate all biconnected graphs using Theorem 1 for pruning and then check for each one whether it is geodetic. We have written a simple function to check whether a graph is geodetic shown in Listing 1 in the appendix. That function can be baked right into `nauty`'s `geng` tool by compiling it with `-DPRUNE=check_geodetic`.

The `PRUNE` function is not only called for the final graph, but also for intermediate graphs: Graphs are constructed by adding one vertex after another. Each time a vertex is added, the prune function is called with the intermediate graph, which is an induced subgraph of the final graph. We take advantage of this by checking whether each 4-cycle

Table 1: Statistics on the `nauty` based Algorithm.

Vertices	**Intermediate Graphs**	**Final Graphs**	**Geo.**
3	2	1	1
4	5	3	1
5	13	7	2
6	105	30	1
7	290	132	3
8	984	844	1
9	4346	6652	3
10	26768	64627	4
11	216599	747768	3
12	2156780	10226920	1
13	26681137	163847610	9
14	395136409	3057525463	2
15	6869665268	66140495796	4
16	139198227567	1652830274122	8

in an intermediate graph induces a complete subgraph as required by Theorem 1.

We run `geng` with the `-C` argument to only generate biconnected graphs. Spreading the workload to 24 threads, we were able to enumerate all biconnected geodetic graphs with up to 16 vertices in a couple of days. We have gathered some statistics in Table 1: in the second and third column we show the respective numbers of intermediate and final graphs the prune function has been called with. In the last column we show the number of biconnected geodetic graphs. For $n = 16$, there were approximately $0.14 \cdot 10^{12}$ intermediate and $1.65 \cdot 10^{12}$ final graphs for which the prune function was called. These numbers suggest that this approach is infeasible for larger values of n.

As with any proof relying on exhaustive computer search, there is a possibility that we might have missed some geodetic graph due to a coding or hardware error. Using `nauty` gives us confidence in our results. We wrote only 40 lines of code ourselves, which is still feasible to check for correctness. Additionally, we rely on the correctness of the `nauty` package.

5 A Custom Search Algorithm

To efficiently enumerate geodetic graphs with more than 16 vertices, we need a different approach than in the previous section. As before, we focus on biconnected geodetic graphs as every geodetic graph is composed of biconnected geodetic blocks. Instead of enumerating all graphs and checking whether they are geodetic, we start with a tree. The idea is that the tree could be a shortest-path tree of a geodetic graph rooted at some vertex of maximum degree. If it is, we can obtain that graph by adding more edges to the tree. There are far fewer trees than graphs – in fact, there are few enough trees that enumerating them is feasible up to above 30 vertices. We use the algorithm

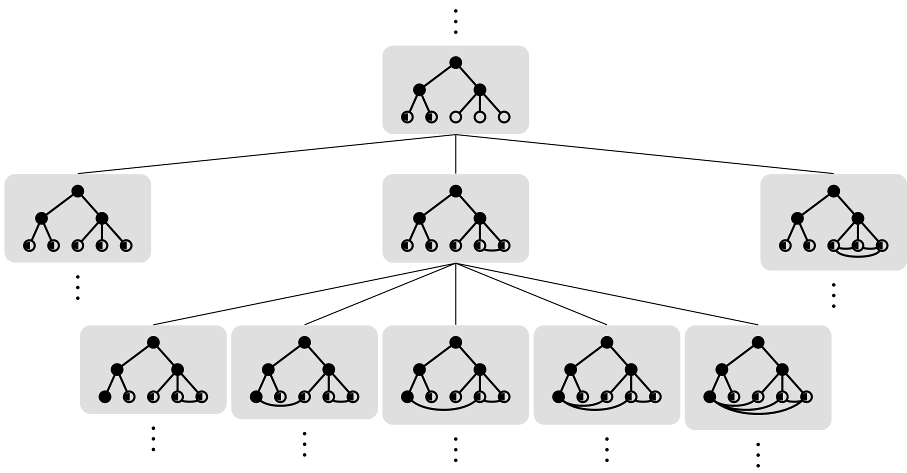


Figure 6: Excerpt of the search space. In the first depicted step sibling edges for the three vertices on the bottom right are generated. Note that the two other ways to insert a single sibling edge yield graphs isomorphic to the middle one. In the second step non-sibling neighbours of the vertex on the bottom left are generated. A vertex in the completed state is represented by a filled circle, semi-complete by a half-full circle.

presented in [13] to generate all rooted trees. We have implemented our algorithm in Rust taking advantage of parallelization by distributing the trees to different threads. We store graphs as adjacency matrices using a bit vector (in our case each line is stored in a single 32-bit integer). In particular, we use the same format as `nauty`, so that we can perform isomorphism checks efficiently without converting the format. The source code and the list of biconnected geodetic graphs with up to 25 vertices are available on `osf.io`: https://osf.io/9nxzm/?view_only=7c57ac8fe5f648aea7e9fb02a4308a3e.

In the following, we describe our backtracking algorithm for extending a tree to a geodetic graph.

5.1 Overview

To extend a tree to a geodetic graph, we implement a backtracking search to insert additional edges. We assume that we are given a total order on the vertices with the property that a vertex is smaller than any of its children and children of the same parent are consecutive. We call this order the *level-order*. The search traverses the vertices of the tree in that order. By Theorem 5, all non shortest-path tree edges must be between vertices on the same level, so it is sufficient to consider those. Among the edges between vertices on the same level, we distinguish sibling edges and non-sibling edges. A *sibling edge* is an edge between two vertices that share the same parent. We call vertices sharing the same parent *siblings*. As a consequence of this distinction, the search alternates between two steps, each being applied to an entire level of the tree: partitioning sibling vertices into cliques and generating non-sibling edges. Figure 6 illustrates the two steps and Figure 7 illustrates an entire run of the algorithm on a single tree. At any point of time, each vertex has one of the following three states – with the stated invariants for the subsequent recursive steps.

unprocessed	Additional incident edges may still be added.
semi-complete	Only non-sibling edges can still be added, incident sibling edges will not change anymore.
complete	No additional incident edges can be added.

The two steps, which in Section 5.2 below are described in more detail, are as follows:

Step 1: Partition sibling vertices into cliques. This step is motivated by Theorem 3, which states that the neighbours of each vertex in a geodetic graph are partitioned into disjoint cliques. The algorithm recursively goes through all sets of siblings on the level. For a given set of siblings, each partition is checked by adding the edges from the partition and continuing recursively with the next set of siblings. After fixing a partition, the vertices go from the unprocessed to the semi-complete state.

Step 2: Generate non-sibling edges. Once the first step is complete for all vertices on a level, the algorithm recursively traverses the vertices of that same level to generate the non-sibling edges. During this step, we use several pruning techniques to avoid choices which cannot lead to geodetic graphs as detailed in Section 5.3. For each vertex, the algorithm recurses on all combinations of possible incident edges on the same level. That is, for each combination, add the edges, mark the vertex as completed and recursively continue the search with the next vertex in level-order. At that stage only non-sibling edges that connect to vertices not yet in the complete state are considered possible. Note that, since the first step has been completed for the entire level, every vertex on the level is either in the semi-complete or complete state. Moreover, the vertices in the complete state precede the vertices in the semi-complete state.

5.2 Detailed Description and Pseudocode

Step 1. The generation of sibling-edges is realized by the function GenCliques(T, E, v), where T is the tree, E the edge set of the graph being generated, and v is the vertex whose children are to be partitioned into cliques. The function Children(v) refers to the children of v in the tree T, NextOnLevel(v) is the next vertex on the same level as v, and NextLevel(v) is the set of vertices on the level below v. The algorithm is initiated by calling GenCliques(T, E, r) where r is the root of T and E the set of edges of T. We give the pseudocode of the function:

function GenCliques(T, E, v)
  for each partition P of Children(v) **do**
    $E' \leftarrow E \cup \bigcup\{\binom{C}{2} \mid C \in P\}$
    if exists NextOnLevel(v) **then**
      GenCliques(T, E', NextOnLevel(v))
    else if NextLevel(v) $\neq \emptyset$ **then**
      GenNeighbour(T, E', First(NextLevel(v)))

8: **else if** E' is geodetic and biconnected **then**
9: Print E'

In the third line the edge set is extended by adding cliques according to the partition of the children of v. If v is not the last vertex on the level, then, in line 5, the function recursively calls itself for the next vertex. If v is the last vertex on the level, but not the last vertex in the tree, then the function GenNeighbour (Step 2) is called, which realizes the generation of non-sibling edges. If v is the last vertex in the tree, then the recursion is terminated and we check whether the generated tree is geodetic and biconnected.

Step 2. We give the pseudocode of the GenNeighbour function:

1: **function** GenNeighbour(T, E, v)
2: $K \leftarrow \{u \in V \mid \text{level}(u) = \text{level}(v) \wedge u > v \wedge \text{par}(u) \neq \text{par}(v)\}$
3: **for** each $U \subseteq K$ **do**
4: $E' \leftarrow E \cup \{vu \mid u \in U\}$
5: Prune(T, E', v)
6: **if** exists NextOnLevel(v) **then**
7: GenNeighbour(T, E', NextOnLevel(v))
8: **else**
9: GenCliques(T, E', First(Level(v)))

Like in Step 1, the vertices are traversed in level-order, which in the above pseudocode is denoted by $u > v$. Therefore, a call to GenNeighbour(T, E, v) only generates edges to vertices that have not been traversed yet. The parent of a vertex u is denoted by par(u). In line 2, the set K of semi-complete vertices that do not share the parent with v is computed. The Prune function checks whether adding the subset $U \subseteq K$ of edges still can lead to the discovery of new geodetic graphs and prevents the recursive calls in line 7 and 9 (i.e., resumes the for loop with the next U) if that is not the case.

5.3 Pruning Techniques

We use the following pruning techniques to reduce the search space.

Maximum Vertex Degree. Recall that we assume the root of the tree to be a vertex of maximum degree in the geodetic graph. Thus, the degree of the root is an upper bound for the degree of any other vertex. To take advantage of this, when generating sibling edges, we immediately discard partitions that violate the maximum degree of some vertex. When generating non-sibling edges, we use this to limit the number of edges added to a vertex in a recursion step and we do not add edges to another vertex that already has the maximum degree.

Non-Clique Neighbour. By Theorem 6, every vertex in a clique has a neighbour that is not part of that clique. Every vertex except the root forms a clique with its parent and potentially some sibling vertices. Thus, it needs a non-clique neighbour. When generating the non-sibling edges of a leaf vertex that does not yet have any non-sibling edges, we make sure to add at least one such edge.

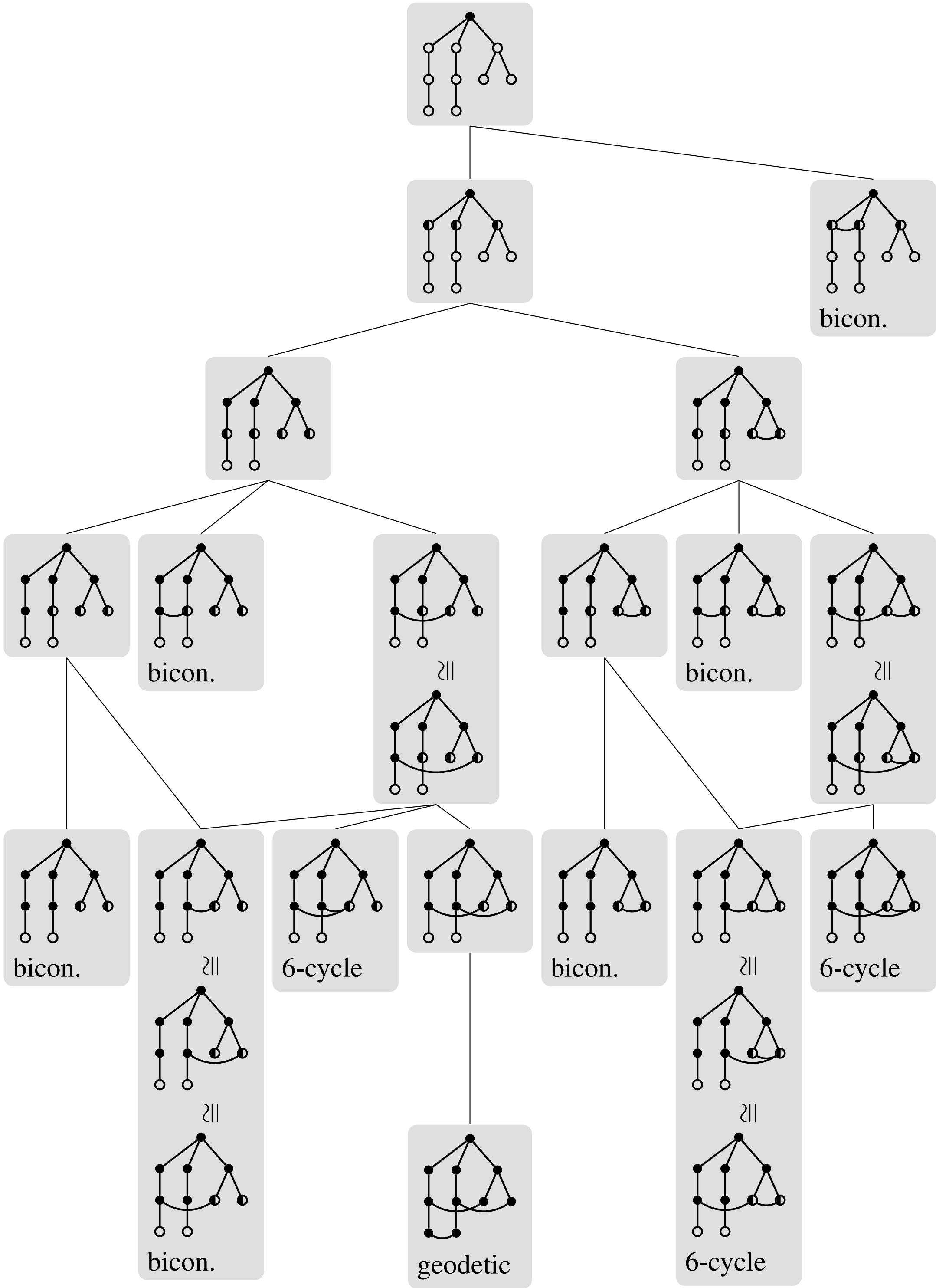


Figure 7: Search space of the backtracking search for a single tree. We found one geodetic graph, a subdivision of K_4. For each unfinished graph without children, the pruning heuristic due to which it is discarded is displayed under the graph.

Forbidden Edges. Theorem 5 states that, whenever there is an edge between two vertices on the same level, there cannot be certain edges on the levels below. To take advantage of this, we maintain a set of forbidden edges in our implementation. The forbidden edges are not considered when generating the non-sibling edges.

Isomorphism Tests. The tree and the generated graphs often have large automorphism groups. Generating the same graph multiple times has a significant impact on the performance. In our implementation we use two strategies to avoid generating certain isomorphic graphs in the first place: First, when generating sibling edges, we only consider non-isomorphic partitions (with respect to isomorphism classes of the corresponding subtrees). Second, when generating non-sibling edges, we implemented explicit isomorphism checks using `nauty`. We added those checks after each recursive generation step. These are realized using a cache of fixed size, which is the only part of the algorithm that requires a substantial amount of memory. For each step of the generation process we use a separate cache each containing 5000000 entries. In total one thread uses up to 40 GB.

Biconnectivity. As we are only interested in biconnected graphs, one pruning step is to test whether the final graph can still be made biconnected. To do so, we take the set of edges that has already been generated and add all the edges that could still be generated (excluding those in the forbidden edge set) and test whether the resulting graph is biconnected. As this test is rather computationally expensive, we omit it for the last couple of vertices as well as for vertices where no edges can be generated.

4-cycles. By Theorem 1, every 4-cycle needs to be a clique. From that we derive the following restrictions on non shortest-path tree edges.

1. Let u, v_1, v_2 be vertices on the same level such that v_1 and v_2 have the same parent but u has a different parent. Then it is not allowed to have both of the edges uv_1 and uv_2. One of them is allowed, though. Depending on the order in which we visit the vertices u, v_1, v_2, we either implement this as part of the pruning or via the forbidden edge set.

2. Let u, v, w, x be vertices on the same level. If (u, v, w, x) is a 4-cycle, then the four vertices need to be a clique.

6-cycles. We also make use of the following fact due to Stemple.

Lemma 10 (Theorem 3.4 of [23]). *If a geodetic graph G contains a cycle of length six, then one of the following three cases applies:*

1. *The vertices of the 6-cycle induce a complete subgraph.*

2. *The vertices can be labelled cyclically* $1, 2, \ldots, 6$ *such that there is an edge between* 1 *and* 3 *and a path from* 2 *to* 5 *of length two whose intermediate vertex is not on the 6-cycle.*

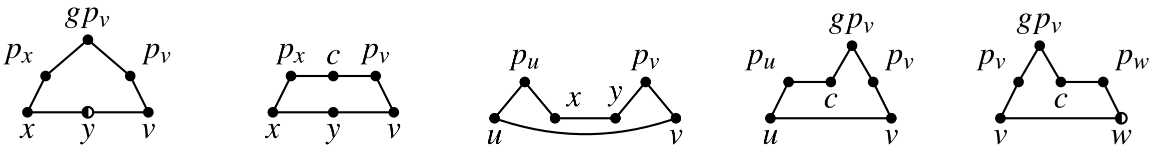


Figure 8: 6-cycles considered for pruning.

Table 2: Number of geodetic graphs.

Vertices	Total	Connected	Bicon.
1	1	1	–
2	2	1	1
3	4	2	1
4	9	4	1
5	21	10	2
6	52	23	1
7	138	66	3
8	384	185	1
9	1146	586	3
10	3563	1880	4
11	11599	6360	3
12	39013	21975	1
13	135359	78230	9

Vertices	Total	Connected	Bicon.
14	480489	283087	2
15	1741210	1043329	4
16	6413809	3895505	8
17	23966313	14726263	6
18	90633268	56234210	5
19	346364720	216719056	13
20	1335756420	841857211	3
21	5192999602	3293753840	13
22	20333304508	12969219563	19
23			11
24			3
25			33

3. Labelling as above, there are three paths of length two connecting 1 *to* 4*,* 2 *to* 5*, and* 3 *to* 6 *whose intermediate vertices are all different and not on the 6-cycle.*

We check whether the conditions of Theorem 10 are satisfied for the 6-cycles shown in Figure 8. Vertices drawn above others are on a higher level in the tree. The vertex v, is the vertex whose edges were finished before calling the prune function. We decided on those patterns because we can both detect them and check the conditions of the lemma efficiently. In particular, we can easily look up parent vertices in the tree. Thanks to the adjacency matrix representation of the graphs (using 32-bit integers), for two vertices we can enumerate all distance two paths between them with a simple binary And. The patterns we consider have at most four vertices on the same level, ruling out the first case of Theorem 10.

Instead of explicitly checking the conditions of Theorem 10, we use the following observation. In cases 2 and 3 of Theorem 10, there is a unique path of length two between any pair of opposing vertices on the cycle. So for each 6-cycle corresponding to one of the patterns in Figure 8, we test whether those unique paths exist.

6 Experimental Results

Using our algorithm, we enumerated all biconnected geodetic graphs with n vertices for $n \leqslant 25$. With 24 threads, this took 16 days of computation time, of which almost 15 days were spent for $n = 25$. Thus, we are able to confirm the results Cu and Elder obtained

Table 3: Comparison of Running Times.

Vertices	13	14	15	16	17	18	19	20	21	22	23	24	25
`nauty`-based algorithm	8.9s	3.4m	93m	56h									
Custom search algorithm	10ms	29ms	87ms	0.5s	1.0s	2.3s	11s	58s	5m	41m	4.8h	34h	350h
Custom without Stemple Pruning	90ms	0.45s	8.1s	94s	14m	50m	5.9h						
Custom without Biconnectivity	7ms	24ms	0.1s	0.46s	1.6s	2.8s	11s	63s	6.7m	47m	6h		
Custom without Clique Neighbour	14ms	51ms	0.2s	1.1s	2.9s	5.7s	26s	115s	13m	82m	9.7h		

for $n \leqslant 11$ [6]. Table 2 shows the number of geodetic graphs with n vertices (we did not compute the numbers of connected geodetic graphs for $n \geqslant 23$ because there are too many of them).

In Table 3 we compare the running times of the `nauty`-based algorithm to our custom search algorithm described in Section 5. We ran both algorithms on a machine with a Ryzen 9 5900X (12 cores/ 24 threads) processor and 128 GB of RAM. Additionally, we collected running times with some of the pruning techniques and optimizations disabled in order to demonstrate their effectiveness. In particular, we disabled the pruning based on 6-cycles (Stemple Pruning), the check whether it is still possible to obtain a biconnected graph from an intermediate graph (Biconnectivity), and the optimization enforcing that each clique vertex has a neighbour outside of the clique (Clique Neighbour).

A list of all the biconnected graphs discovered by our computer search is given in Table 4. There, the graphs $G_c(p)$, $G_d(p)$, and $G_e(p)$ denote the constructions given in [2] in Sections 2c–2e and p is the order of the projective/ affine plane the construction is based on. Unless noted otherwise, subdivision refers to Theorem 7.

The columns describe properties of the graphs: here, r denotes the radius (i.e., $\min_{u \in V} \max_{v \in V} d(u, v)$), d the diameter and δ the minimum vertex degree. The "re." and "ha." columns state whether the graph is regular (resp. hamiltonian). By Aut we denote the automorphism group of the graph using the following notation: C_n is the cyclic group of order n, D_{2n} the dihedral group of order $2n$, and S_n the symmetric group on n elements. A semidirect product is denoted by $\rtimes$.

As expected, our search found all the known geodetic biconnected graphs with up to 25 vertices – that is complete graphs and the graphs obtained using the constructions from [2, 4, 18]. Most of the graphs found by our search are subdivisions of other geodetic graphs (in particular, complete graphs). Moreover, as foreseen by [19, 24], there are certain values n for which we found many such subdivisions and other values n that do not admit any such subdivision such as $n = 12, 14$, or 24 (moreover, for $n = 18, 20$ there are also no subdivisions of complete graphs but there are subdivisions of other small geodetic graphs). The most exciting discovery is the graph $H(2, 2, 2, 0)$ which to the best of our knowledge has not appeared in the literature before nor can it be constructed using one of the previously known constructions. It allowed us to derive a new family of geodetic graphs presented in Section 3.2. Finally, note here that, except the complete graphs, the Petersen graph and four graphs described in [2], all graphs have minimum vertex degree two. Moreover, the only regular graphs in the table are the Petersen graph, cycles and complete graphs.

7 Regular Geodetic Graphs.

Very few regular geodetic graphs are known. There are odd cycles and complete graphs, which are trivially geodetic. Beyond those, there are the strongly regular Moore graphs of diameter two. These are the Petersen graph, the Hoffman-Singleton graph, and possibly some graphs of order 3250 [12]. In addition, a cubic (regular of degree three) block of order 28 is depicted in [4] as an example for a construction presented in that paper. The same block is shown in Figure 9 on the right. To our knowledge, those are the only known regular geodetic blocks.

We used our algorithm to confirm that among the regular graphs with at most 32 vertices there are no geodetic graphs other than the ones mentioned above.

We observe that the cubic graph in [4] is actually part of an infinite family of regular geodetic graphs. In fact, it is the flag graph of the Fano plane. Flag graphs of certain incidence structures are known to be geodetic [5, 7]. When the incidence structure is a projective plane, then they are also regular.

Observation 11. *For each prime power k there is a $(k+1)$-regular geodetic graph with k^3+3k^2+3k+2 vertices.*

We briefly describe the construction, and refer the reader to [7] for more details. Take a finite projective plane of order k (meaning that it consists of k^2+k+1 points). For each line in the plane, create a $(k+1)$-vertex complete subgraph. Each of these vertices corresponds to one point on the line. Furthermore, for each point in the plane, create a separate vertex. Finally, for each line and each point on that line add an edge between the vertex of the complete subgraph corresponding to that line and point and the separate vertex corresponding to the same point.

The resulting graph is regular of degree $k+1$, which can be seen as follows: Each vertex in a clique corresponding to a line of the plane has k neighbours within the clique; moreover, it is connected to the separate vertex corresponding to the same point; hence, it has degree $k+1$. Furthermore, every point in a finite projective plane of order k lies on $k+1$ lines; hence, each vertex corresponding to a point is adjacent to $k+1$ clique vertices.

Figure 9 demonstrates the construction with the Fano plane. The resulting graph is exactly the cubic graph on 28 vertices in [4]. Note that in [4] the graph is also presented as an example for a larger family, however that construction does not rely on projective planes. Instead, it uses a clique edge cover of a complete graph. In general, this leads to graphs that are not regular. Here, the finite projective plane of order k comes to play: it yields a covering of the K_{k^2+k+1} by k^2+k+1 cliques of size $k+1$.

Table 4: List of biconnected geodetic graphs.

n	**Graph**	r	d	δ	re.	ha.	Aut
n	K_n	1	1	$n-1$	y	y[a]	S_n
	C_n $(n = 2k+1)$	k	k	2	y	y	D_{2n}
7	Subdiv. of K_4	2	2	2			S_3
9	Subdiv. of K_5	2	2	2			S_4
10	2 Subdiv. of K_4						
	Petersen Graph	2	2	3	y		S_5
11	Subdiv. of K_6	2	2	2			S_5
13	3 Subdiv. of K_4						
	2 Subdiv. of K_5						
	Subdiv. of K_7	2	2	2			S_6
	$G_d(2) = G_e(3)$	2	2	3		y	S_4
14	$h(3,2,0)$	3	3	2			D_{12}
15	Subdiv. of K_8	2	2	2			S_7
	WP_3	3	3	2			$C_5 \rtimes C_4$
16	5 Subdiv. of K_4						
	2 Subdiv. of K_6						
17	3 Subdiv. of K_5						
	Subdiv. of K_9	2	2	2			S_8
18	2 Subdiv. of $h(3,2,0)$						
	$h(4,2,0)$	3	3	2			$C_2 \times S_4$
	$h(3,3,0)$	3	3	2			$S_3 \times S_3$
19	6 Subdiv. of K_4						
	2 Subdiv. of K_7						
	Subdiv. of K_{10}	2	2	2			S_9
	Subdiv. of $G_d(2)$	3	3	2			S_4
	K_5^*	4	5	2			D_8
20	Subdiv. of Petersen Graph[b]	4	4	2			D_8
	WP_4	4	4	2			$C_5 \rtimes C_4$
21	5 subdiv. of K_5						
	3 Subdiv. of K_6						
	Subdiv. of K_{11}	2	2	2			S_{10}
	$G_c(3)$	2	2	4		y	$C_3^2 \rtimes \mathrm{GL}(2,3)$
	$G_e(4)$	2	2	4		y	S_5
22	9 Subdiv. of K_4						
	2 Subdiv. of K_8						
	5 Subdiv. of $h(3,2,0)$						
	$h(3,4,0)$	3	3	2			$S_3 \times S_4$
	$h(5,2,0)$	3	3	2			$C_2 \times S_5$
23	Subdiv. of K_{12}	2	2	2			S_{11}
	2 Subdiv. of $h(4,2,0)$						
	2 Subdiv. of $h(3,3,0)$						
	Subdiv. of K_5^*	5	6	2			C_2
	$h(4,3,0)$	3	3	2			$S_3 \times S_4$
	$h(3,2,1)$	5	5	2			D_{12}
	Pulling[c] of $K_5(2,1,1,1,1)$	5	5	2			D_8
24	$H(2,2,2,0)$	4	4	2			$C_2 \times A_4$
	K_6^* [18][d]	4	5	2			S_4
25	11 Subdiv. of K_4						
	7 Subdiv. of K_5						
	3 Subdiv. of K_7						
	2 Subdiv. of K_9						
	Subdiv. of K_{13}	2	2	2			S_{12}
	2 Subdiv. of Petersen Graph[b]						
	Subdiv. of $G_d(2)$	4	4	2			S_4
	WP_5	5	5	2			$C_5 \rtimes C_4$
	K_6^* [18][d]	4	5	2			D_{12}
	Pulling[c] of $K_6(1,1,1,1,1,0)$	3	5	2			D_8
	$G_d(3)$	2	2	4		y	$C_3^2 \rtimes \mathrm{GL}(2,3)$

[a] K_n is hamiltonian for $n \geqslant 3$

[b] Two subdivisions of the Petersen graph, one with 20 and one with 25 vertices, are not explained by Theorem 8.

[c] The pulling operation has been introduced by Plesnik [20]. See Section A.3.

[d] The construction also depends on a so-called $\mathcal{O}$-cover of K_6 – therefore, different graphs can arise as K_6^*.

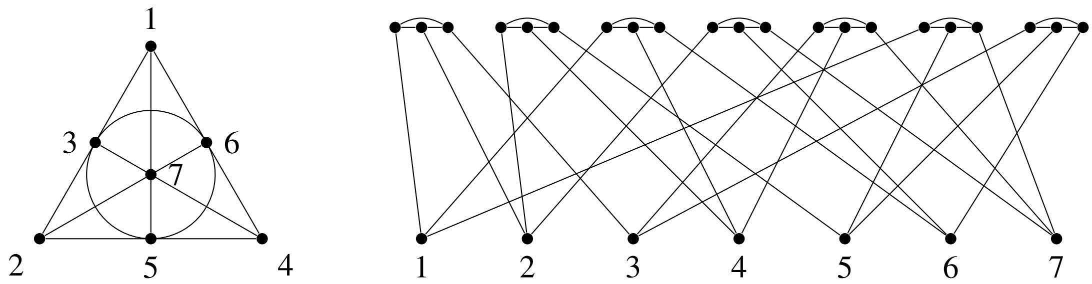


Figure 9: The Fano plane (left) and the regular geodetic graph constructed from it (right)

8 Conclusion and Open Questions

We implemented an exhaustive computer search in order to find all geodetic graphs with up to 25 vertices. The results of the computer search lead us to the discovery of a new family of geodetic graphs. Moreover, we used our algorithm to find all regular geodetic graphs with up to 32 vertices. Furthermore, we discovered a new result on geodetic subdivisions of geodetic graphs. A full classification of geodetic graphs remains the aim for future research. We conclude this work by mentioning a few questions for further research in the area.

Growth Rate of Biconnected Geodetic Graphs. Let $g(n)$ denote the number of biconnected geodetic graphs on n vertices. Looking at the numbers for $g(n)$ with $n \leqslant 25$ in Table 2, gives rise to a natural question about the growth rate of $g(n)$: it seems to grow slowly, but there are points (for $n = 20, 24$) where there are almost no biconnected geodetic graphs. Yet, for certain values of n there is a super-polynomial lower bound:

Observation 12. *$g(k^2) \geqslant e^{2\sqrt{k}}/14$.*

Proof. We only count geodetic subdivisions of K_k with k^2 vertices. By [19, 24] (see Section 1), every such subdivision is described by a multi-set of k non-negative integers $\ell_1, \ldots, \ell_k$ summing up to k. The number of such multi-sets is the partition number $P(k)$, which is bounded from below by $e^{2\sqrt{k}}/14$ [14, Corollary 3.1]. □

Let us highlight two questions on the growth rate of $g(n)$:

- Is there a constant c such that there are infinitely many n with $g(n) \leqslant c$?
- How much can the lower bound from Theorem 12 be improved? In particular, is there a constant c such that $g(n) \leqslant 2^{c\sqrt[4]{n}}$ for all n?

Subdivisions. In our exhaustive search we found several geodetic graphs arising as subdivisions of other geodetic graphs. Most of them fall in one of two categories using either Theorem 8 or the construction from [19] starting with a complete graph and assigning numbers to the vertices. Apart from those constructions, there are cases of geodetic subdivisions that do not fall in any of these categories: There are two geodetic subdivision of the Petersen graph, one with 25 vertices and one with 20 vertices, which have been studied previously in [9]. Thus, the question for characterizing all geodetic subdivisions of a given geodetic graph remains an open problem. Even more generally, which (arbitrary) graphs have a geodetic subdivision?

Extending the Exhaustive Search. Using the exhaustive search for geodetic graphs with more vertices is highly non-trivial. Given the running times for up to 25 vertices, it seems reasonable that $n = 26$ could be finished within a few months. Larger values for n seem only possible using some new heuristics for the pruning method. One possible such pruning heuristic would be to use a lemma similar to Theorem 10 on how an 8-cycle (instead of a 6-cycle) can be embedded in a geodetic graph. There are two challenges to overcome for such a result: first, to find a theoretical statement in the spirit of Theorem 10, and, second, to find a way to implement those cases efficiently.

Regular Geodetic Graphs. It is reasonable to assume that the graphs we presented in Section 7 are not the only regular geodetic graphs. It would be interesting to know how many k-regular geodetic graphs there are for each k. We know that there are infinitely many 2-regular geodetic graphs (all odd cycles). For larger k it is not even known whether there are finitely or infinitely many k-regular geodetic graphs.

References

[1] Bilò, D., Chechik, S., Choudhary, K., Cohen, S., Friedrich, T., Krogmann, S., Schirneck, M.: Approximate distance sensitivity oracles in subquadratic space. In: Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023. pp. 1396–1409. ACM (2023). doi:10.1145/3564246.3585251

[2] Blokhuis, A., Brouwer, A.: Geodetic graphs of diameter two. Geometriae Dedicata **25**, 527–533 (1988)

[3] Bollobás, B.: Modern graph theory, vol. 184. Springer Science & Business Media (2013). doi:10.1007/978-1-4612-0619-4

[4] Bosák, J.: Geodetic graphs. In: Combinatorics (Proc. Colloq. Keszthely 1976), North Holland, Amsterdam. pp. 151–172 (1978)

[5] Bridgland, M.F.: Geodetic graphs and convexity. Ph.D. thesis, Louisiana State University and Agricultural & Mechanical College (1983)

[6] Cu, J., Elder, M.: Length of embedded circuits in geodetic graphs, https://vrs.amsi.org.au/wp-content/uploads/sites/84/2021/01/cu_john_vrs-report.pdf

[7] Etgar, A., Linial, N.: On the connectivity and diameter of geodetic graphs. European J. Combin. **116**, Paper No. 103886, 10 (2024). doi:10.1016/j.ejc.2023.103886

[8] Federici, B.: Interactions between large-scale invariants in infinite graphs. Ph.D. thesis, University of Warwick (2017), https://wrap.warwick.ac.uk/108882/

[9] Frasser, C., Vostrov, G.: Geodetic graphs homeomorphic to a given geodetic graph. International Journal of Graph Theory and its Applications **3**, 13–44 (12 2020)

[10] Gorovoy, D., Zmiaikou, D.: On graphs with unique geoodesics and antipodes. Discrete Math. **347**(4), 113864 (2024). doi:10.1016/j.disc.2023.113864

[11] Híc, P.: A construction of geodetic blocks. Mathematica Slovaca **35**(3), 251–261 (1985)

[12] Hoffman, A.J., Singleton, R.R.: On Moore graphs with diameters 2 and 3. IBM Journal of Research and Development **4**(5), 497–504 (1960). doi:10.1147/rd.45.0497

[13] Li, G., Ruskey, F.: The advantages of forward thinking in generating rooted and free trees. In: SODA. vol. 1999, pp. 939–940. Citeseer (1999)

[14] Maróti, A.: On elementary lower bounds for the partition function. Integers: Electronic J. Comb. Number Theory **3**, A10 (2003)

[15] McKay, B.D., Piperno, A.: Practical graph isomorphism, II. Journal of Symbolic Computation **60**, 94–112 (2014). doi:10.1016/j.jsc.2013.09.003

[16] OEIS Foundation Inc.: The on-line encyclopedia of integer sequences. Published electronically at https://oeis.org (2023)

[17] Ore, O.: Theory of graphs. In: Colloquium Publications. American Mathematical Society (1962)

[18] Parthasarathy, K.R., Srinivasan, N.: Some general constructions of geodetic blocks. Journal of Combinatorial Theory, Series B **33**(2), 121–136 (1982)

[19] Plesník, J.: Two constructions of geodetic graphs. Mathematica Slovaca **27**(1), 65–71 (1977)

[20] Plesník, J.: A construction of geodetic graphs based on pulling subgraphs homeomorphic to complete graphs. Journal of Combinatorial Theory, Series B **36**(3), 284–297 (1984)

[21] Scapellato, R.: Geodetic graphs of diameter two and some related structures. Journal of Combinatorial Theory, Series B **41**(2), 218–229 (1986)

[22] Srinivasan, N., Opatrný, J., Alagar, V.S.: Construction of geodetic and bigeodetic blocks of connectivity $k \geqslant 3$ and their relation to block designs. Ars Combin. **24**, 101–114 (1987)

[23] Stemple, J.G.: Geodetic graphs of diameter two. Journal of Combinatorial Theory, Series B **17**(3), 266–280 (1974)

[24] Stemple, J.G.: Geodetic graphs homeomorphic to a complete graph. Annals of the New York Academy of Sciences **319**(1), 512–517 (1979)

[25] Stemple, J.G., Watkins, M.E.: On planar geodetic graphs. Journal of Combinatorial Theory **4**(2), 101–117 (1968)

A Appendix

A.1 Prune Function for nauty

In Listing 1 one can see the prune function we used with `nauty`. The `PRUNE` function is not only called for the final graph, but also for intermediate graphs: Graphs are constructed by adding one vertex after another. Each time a vertex is added, the prune function is called with the intermediate graph, which is an induced subgraph of the final graph. We take advantage of this by checking whether each 4-cycle in an intermediate graph induces a complete subgraph as required by Theorem 1.

The `nauty` library represents graphs as adjacency matrices. `MAXN` is the maximum number of nodes of the graph supported by `nauty`. Each line of the adjacency matrix is stored as a bit vector (in our case in a single 32-bit integer). The type aliases `graph` and `set` refer to the same integer type. Our code assumes that a line of the adjacency matrix is exactly one `set`, thus, `MAXM == 1`. A graph is represented using an array of `set`. The constant `bit[i]` is a `set` with the i-th most significant bit set. For more details we refer the reader to the documentation of `nauty`.

```
#include "gtools.h"

_Static_assert(MAXM == 1, "MAXM␣==␣1");
_Static_assert(MAXN >= 1, "MAXN␣>=␣1");

int check_geodetic(graph *g, int n, int maxn) {
  int rounds = n - 1;
  if (n != maxn) {
    /* Intermediate graph: one iteration. */
    rounds = 1;
  }
  /* vertices reachable in round steps */
  graph p[MAXN * MAXM];
  for (int k = 0; k <= n * MAXM; k++) {
    p[k] = g[k];
  }
  for (int r = 1; r <= rounds; r++) {
    for (int i = 0; i < n; i++) {
      /* compute vertices at distance r+1 from i */
      set reach = p[i] & ~bit[i];
      set not_reach = ~(p[i] | bit[i]);
      for (int j = 0; j < n; j++) {
        if (reach & bit[j]) {
          // reach via j
          set reach_new = g[j] & not_reach;
          if (p[i] & reach_new) {
            return 1;
          }
          p[i] |= reach_new;
        }
      }
    }
  }
  return 0;
}
```

Listing 1: Function to check whether a graph is geodetic, using `nauty`.

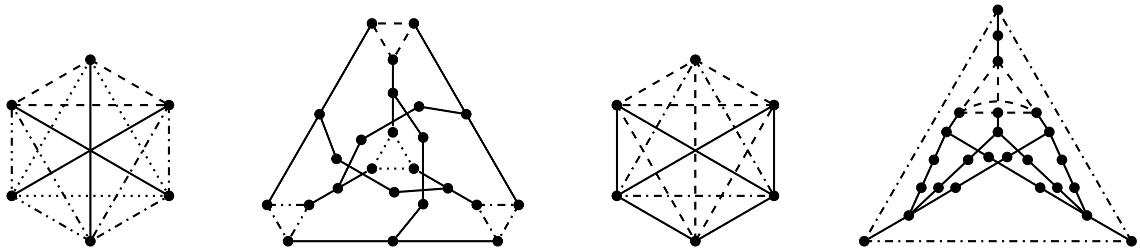


Figure 10: The star operation applied to the K_6 in two different ways.

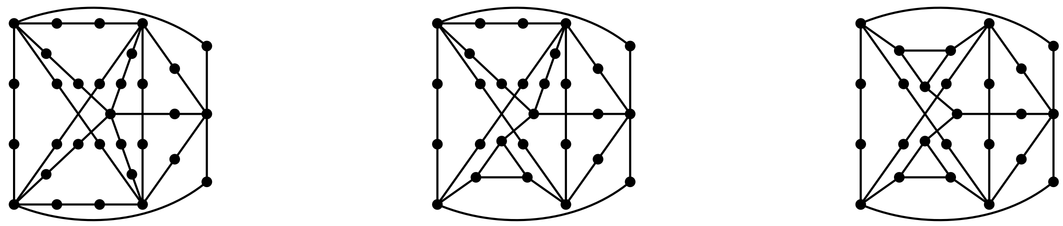


Figure 11: The pulling operation. Geodetic subdivision of K_6 (left), result of the first pulling operation (middle), graph after the second pulling operation (right).

Let A be the adjacency matrix of the graph. The idea of the algorithm is to compute powers A^r of the adjacency matrix, for $1 \leqslant r \leqslant n$. However, we only store whether an entry is 0. Thus, we can use the same type of bitvector, that is used to store the graph. If an entry (i, j) is 0 in some A^r, but not zero in A^{r+1}, and there is more than one path of length $r + 1$, then the algorithm will return 1 in line 27, indicating that the graph is not geodetic. For intermediate graphs, only A^2 is computed, that is we only check for 4-cycles.

A.2 Star Operation

Figure 10 shows the result of applying the star operation as defined in [18] to a K_6. To obtain the graph on the left with 24 vertices, start with an $\mathcal{O}$-cover of K_6 with 4 triangles and 4 single edges. The graph on the right with 25 vertices is obtained starting with an $\mathcal{O}$-cover of K_6 with one K_4, one triangle and 6 single edges.

A.3 Pulling of Geodetic Subgraphs

Plesník introduced the operation of pulling a geodetic subgraph homeomorphic to a complete graph as a method for constructing a geodetic graph from another [20]. Applying this operation twice to a geodetic subdivision of K_6 obtained from the labelling $(1, 1, 1, 1, 1, 0)$, we obtain a geodetic graph with 25 vertices. This is visualized in Figure 11.