1 Introduction

Given a transitive directed reachability graph $G$, we are interested in determining if it can be embedded in a structure $S$. The goal is to find an embedding such that there exists a path between vertex $i$ and vertex $j$ in the structure if and only if there was an edge from $i$ to $j$ in the reachability graph.

This problem is loosely motivated by problems in VLSI chip design. These chips have their wires organized in a grid and must constrain the reachability relationship between points. This is exactly the problem being considered when the target structure is grids.

Some natural structures that will be considered include line, tree, and grid (or lattice) networks. Other interesting properties that could be imposed include bipartiteness of the reachability graph and layering in the structure. Another important property is whether the embedding is allowed to have additional vertices besides the ones in $G$ (referred to as Steiner points).

This paper will give the following results:

- a characterization of what can be embedded into lines both with and without Steiner points
- a characterization of what can be embedded into trees both with and without Steiner points
- a characterization of what can be embedded into complete layered grids without Steiner points
- a characterization of what bipartite reachability graphs can be embedded into complete layered grids with Steiner points
- a comparison of these classes of embeddable reachability graphs
1.1 Related Work

There are some papers that approach very similar problems to the one this paper is considering.

Many of the examples involving grid graphs arise from research done on VLSI chip design. In [1], Dolev et al discuss a method for embedding graphs in a grid. Their method involves using separator theorems. They remove a small number of edges to break the graph into separate parts and then recursively lay out each part. Afterward, the subparts are put back together by embedding the removed edges. The problem with this method is that it allows crossings. Crossings are places where the wires pass through the same point, but since they are on different layers they do not count as intersections. This paper’s goal will be to design graphs without crossings. This is true because the result of a crossing in a graph is the creation of an additional path that could potentially be taken. As a result, two vertices that did not have an edge between them in the reachability graph could possibly now reach one another. The embedding created in this paper will strictly enforce both the non-reachability and the reachability, whereas Dolev et al only care about reachability.

More methods for grid graphs arising from the area of VLSI chip design can be seen in [3]. Here, Bhatt talks about the Thompson grid model, which is a formal model for VLSI graph layout and forces the layout to be a grid graph. There are two main points where Bhatt’s paper differs from this one. First, crossings are again allowed, which runs into all of the problems discussed above. Secondly, it does not allow paths to overlap. This means that for any two distinct paths in the embedding, there cannot be an edge that appears in both paths. Forcing this property would limit the degree of vertices to be 4 or less. To avoid this limitation, this property will not be enforced in this paper.

Other methods for embedding graphs into grid graphs for two very specific classes are explained by Cahit and Kheddouci[2]. The first class denoted $G(\leq 3)$ requires exactly 4 vertices to be of degree 2. It also requires all other vertices to have degree 3. The second class denoted $G(\leq 4)$ also requires exactly 4 vertices to be of degree 2. It also requires all other vertices to have degree 3 or 4. It may be possible to reduce some of the graphs that are being considered into these two classes. However, this paper deals with more general graphs, so the scope to which these methods can be applied are limited.

In [5], there is a discussion about embedding graphs into grids and minimizing the number of bends. A linear time algorithm is given, but the classes of graphs considered is very limited. The maximum vertex degree allowed is 4, which greatly simplifies the process of finding an embedding.

For the tree section of this paper, an article that is relevant is [4]. Bienstock covers the problem of embedding on the leaves of a tree. This is different from the problem considered here because the vertices of the embedding will not be forced to be on leaves, which he does in order to study a different set of problems. Also, the graphs considered by Bienstock are undirected, which is different from the directed graphs here.
2 Preliminaries

There are two inputs for our problem

1. A directed reachability graph \( G(V,E) \)
2. A target structure \( T \)

Definition 1 A Steiner point is a vertex \( s \) in \( S \) that does not correspond to any vertex \( v \) in \( G \).

Definition 2 A tree is a directed graph whose underlying undirected graph is connected and acyclic.

Definition 3 A grid graph is a structure where all vertices are ordered pairs of integers. Furthermore, from a vertex \( (x, y) \) there can only be edges to \( (x - 1, y) \), \( (x + 1, y) \), \( (x, y - 1) \) and \( (x, y + 1) \).

Definition 4 A grid graph is layered if all edges are directed south or east.

Definition 5 A grid graph is complete if for every pair of vertices \( (u, v) \) where the distance between \( u \) and \( v \) is 1, there is either an edge from \( u \) to \( v \) or from \( v \) to \( u \).

Definition 6 \( \Gamma^+(v) \) is the set of vertices \( u \) such that there is an edge from \( v \) to \( u \). \( \Gamma^-(v) \) is the set of vertices \( u \) such that there is an edge from \( u \) to \( v \).

Definition 7 An alternating cycle of length \( 2n \) (also known as an alternating \( 2n \) cycle) is a cycle made of \( 2n \) vertices with successive edges oriented in opposite directions.

3 Embedding into a Line

Definition 8 A directed graph \( H = (U, F) \) is a total ordering if \( U \) can be totally ordered so that \( (i, j) \in F \) if and only if \( i < j \). The minimal and the maximal vertices are called the terminals of this graph.

Definition 9 A saw-tooth graph with two terminals is defined inductively as follows:

1. Any total ordering graph is a saw-tooth graph with the same set of terminals.

2. If \( G_1 \) is a saw-tooth graph and \( G_2 \) is a total-ordering graph, then the graph obtained by identifying one terminal of in(out)-degree 0 in \( G_1 \) with one terminal of in(out)-degree 0 in \( G_2 \) is a saw-tooth graph. The terminals of this graph are the two remaining terminals of \( G_1 \) and \( G_2 \).
3.1 No Steiner points

**Lemma 3.1** The class of saw-tooth graphs is exactly the class of graphs embeddable on a line without Steiner points.

**Proof:** First, notice that an interval where all of the edges are directed in the same direction represents a total ordering. Since the line can be broken into a set of such intervals, it is obvious that a set of total orderings is the only thing that can be embedded in a line. Consider this set of total orderings from left to right. Obviously, the first total ordering by itself is a saw-tooth graph. Each subsequent total ordering shares a terminal with the previous total ordering because that is the point where the edges switched directions. Therefore, adding these total orderings one at a time continues to produce a saw-tooth graph.

In the reverse direction, it needs to be proved that a saw-tooth graph can be embedded in a line. This will be done by induction on the number of composition operations.

Base case: A single total ordering can be embedded in a line by just placing all the vertices in order and orienting all of the edges in the same direction.

Induction Hypothesis: Assume a saw-tooth graph resulting from \( n \) compositions is embeddable in a line with its terminals at the ends of its embedding.

Given a saw-tooth graph \( G_1 \) resulting from \( n+1 \) compositions, by definition it is composed of a saw-tooth graph \( G_2 \) with terminals \( t1 \) and \( t2 \) made of \( n \) compositions and a total order \( G_3 \) with terminals \( t2 \) and \( t3 \). By the Induction Hypothesis, \( G_2 \) is embeddable in the line with \( t1 \) and \( t2 \) at the ends. \( G_3 \) is also embeddable in a line with \( t2 \) and \( t3 \) at the ends. Since \( G_2 \) and \( G_3 \) share \( t2 \), these two embeddings can be connected at \( t2 \). This means \( G_1 \) is embeddable and its terminals \( t1 \) and \( t3 \) are at the ends. Therefore, the inductive step is proved. \( \square \)

![Figure 1: Graph Embeddable in Line Without Steiner Points](image)

3.2 Steiner Points Allowed

**Lemma 3.2** If Steiner points are allowed, the class of graphs that can be embedded is precisely the disjoint unions of saw-toothed graphs.

**Proof:** There are only 3 types of vertices in an embedding on the line:

- **Type 1**: vertices where both edges are directed away from the vertex
• **Type 2**: vertices where both edges are directed to the vertex

• **Type 3**: vertices where both edges around the vertex are oriented in the same direction

First, notice that adding an extra vertex of Type 3 does not affect whether any vertex can reach any other vertex, so adding Type 3 vertices does not allow the embedding of any additional graphs. If there are two disjoint total orderings that must be placed next to one another, they can be separated by placing a Type 1 vertex (call it $u$) between them. Now, no vertex from either total ordering can reach a vertex in the other total ordering. This is true because none of the vertices can reach $u$, so they certainly cannot reach vertices on the other side of $u$. This represents the only possible use of Type 1 vertices because by definition, it will cause vertices on one side to not be able to reach vertices on the other side. Type 2 vertices serve the same function as Type 1 vertices and thus, do not expand the set of embeddable graphs.

![Figure 2: Graph Embeddable in Line Only With Steiner Points](image)

4 Embedding into a Tree

4.1 No Steiner points

**Definition 10** A tree order is defined inductively as follows:

1. Any total order is by itself a tree order.

2. If $T$ is a tree order and $S$ is a total order, they can be joined to make a larger tree order by identifying exactly one of the end points of $S$ with a vertex in $T$.

**Lemma 4.1** The class of tree orders is exactly the class of graphs embeddable in a tree without Steiner points.
Proof: In the forward direction, the desired fact to prove is that all tree embeddings correspond to tree orders. Take an embedding and consider each edge in the embedding to be a total order on two vertices. A tree order can be created by picking any of the edges to start and then adding the edges one at a time in a manner such that each edge added is connected to the current tree order. When connecting another edge to the existing tree order, exactly one endpoint would already be in the tree order because the embedding is a tree, so it is connected (which means there is always an edge that shares an endpoint with the current tree order) and acyclic (which means both endpoints can not be in the current tree order).

In the reverse direction, the desired fact to prove is that all tree orders can be embedded into trees. A tree order is made of a set of vertices and the embedding will also be made of this same set of vertices. In the embedding, an edge will be drawn from a vertex \( v \) to a vertex \( u \) if \( v \) is the immediate predecessor of \( u \) in one of the total orders of the tree order. The resulting embedding is connected because when another total order is added it must share one of its endpoints. The resulting embedding is also acyclic because only one of the endpoints can be in the tree order and in order to have a cycle, the total order that completes the cycle would need to have both endpoints. Therefore, the resulting embedding is a tree.

An alternative characterization for what graphs are embeddable in trees can be arrived at as follows:

Algorithm for simplifying a graph:

1. Transitively reduce the graph. This is done by removing edges from a vertex \( a \) to a vertex \( c \) if \( \exists \) vertex \( b \) such that there is an edge from \( a \) to \( b \) and an edge from \( b \) to \( c \).

2. Remove all vertices that would be in the middle of a total order. This is done in the following manner: If there is an edge from a vertex \( a \) to a vertex \( b \), an edge from \( b \) to a vertex \( c \), and no other edges involving \( b \), then remove \( b \) and add an edge from \( a \) to \( c \).

Lemma 4.2 A reachability graph can be embedded into a tree without Steiner points if and only if its simplified graph is connected and does not have any undirected cycles.
Proof: For the forward direction, the desired fact to prove is that if the simplified graph is connected and does not have undirected cycles, it can be embedded. Since each total order is represented as an edge, the fact that the simplified graph is connected and doesn’t have undirected cycles means it is already a tree. This embedding can then be augmented back into the original graph by now replacing each edge with the total order it represents. This remains a tree because the way the graph was simplified means that each total ordering is isolated and does not affect the rest of the graph.

For the reverse direction, it needs to be proved that if the simplified graph is not connected or has undirected cycles then it cannot be embedded. Since the only thing that can be embedded in trees are tree orders, it suffices to check whether or not the simplified graph is a tree order. If the simplified graph was not connected, a tree order cannot be constructed because when adding a total order it needs to have exactly 1 endpoint in the already built tree order, but if the graph is not connected, then at some point, none of the remaining total orders can be added. If the simplified graph has an undirected cycle it also cannot be embedded because in order to add the total ordering that closes the cycle, both endpoints would need to be part of the tree order, which is not allowed.

4.2 Steiner points allowed

Lemma 4.3 For the purpose of resolving unembeddable cycles, a Steiner point is only valuable if it has at least 2 incoming edges and at least 2 outgoing edges.

Proof: If there were only 1 vertex on either side, the Steiner point can be deleted and all the vertices that would have connected to the Steiner point can connect to the vertex itself. If there were 0 vertices on either side, the Steiner point can not help to resolve unembeddable cycles because no additional reachability has been added.
Definition 11 The Steiner embedded alternating 4 cycle has 5 vertices: a, b, c, d, and e. a and b can reach e and e can reach c and d.

Lemma 4.4 When Steiner points are allowed, an alternating 4 cycle can be embedded.

Proof: The alternating 4 cycle is embedded in a tree in the form of the Steiner embedded alternating 4 cycle where e is a Steiner point.

Lemma 4.5 The Steiner embedded alternating 4 cycle is the only way to embed the alternating 4 cycle.

Proof: It was proved in the previous section that the alternating 4 cycle can’t be embedded without Steiner points.

There are not enough vertices to justify 2 or more Steiner points. This is true because if the 2 Steiner points do not connect to each other, there cannot be both a common in-vertex and a common out-vertex because then an undirected cycle would have been created and the embedding would no longer be a tree. As a result, at least 6 vertices are needed. If there is an edge between Steiner points, there would need to be 3 incoming edges and 3 outgoing edges. One vertex cannot appear on both sides of the Steiner points because then an undirected cycle would have been created. Therefore, it is not possible to have 2 or more Steiner points because there would need to be 6 vertices to make them useful.

Also, the way described above was the only possible way the sole Steiner point could have been connected because there needs to be 2 vertices on each side.

Lemma 4.6 The non-alternating 4 cycle (when A reaches B and C and both B and C reach D) remains unembeddable in trees.
Proof: First, in the previous section it was proved that this cycle cannot be embedded without Steiner points. If there is only 1 Steiner point, it cannot be useful because there is no place that it could be inserted such that there would be 2 incoming edges and 2 outgoing edges. If there are 2 Steiner points, by the same logic as in the proof of the previous lemma, there are not enough vertices to justify it. As a result, this type of 4 cycle cannot be embedded.

Lemma 4.7 The alternating 4 cycle cannot be embedded if any of the sides are paths instead of edges.

Proof: To see this, look at the four cycle, but without loss of generality, put another vertex F between A and C. If the 4 points A, B, C, and D are replaced with the 5 points A, B, C, D, and E in the Steiner structure described (which must be done because that was the only way to embed the alternating 4 cycle), then there is an unembeddable 4 cycle (A, E, F, and C) as described in the previous lemma. Therefore, this structure (A, B, C, D, and F) cannot be embedded.

Lemma 4.8 An alternating cycle with more than 4 vertices cannot be embedded even with Steiner points.

Proof: If there is 1 Steiner point to add, there is no place that it could be added usefully because all the places in this graph have only 1 source for 2 sinks or 1 sink for 2 sources and a useful Steiner point needs 2 sinks and 2 sources. If there are 2 Steiner points and there is not an edge between them, then there is the same problem as in the case with 1 Steiner point. If there is some edge from s to t where both are Steiner points, there would be at least 3 sinks flowing out of this structure of Steiner points. Therefore, any source vertex r such that there is an edge from r to s would need to reach 3 sinks, but all of the sources only reach 2. As a result, there is no use for Steiner points for resolving these alternating cycles and since it was proved that alternating cycles cannot be embedded without Steiner points, they cannot be embedded.
Algorithm for simplifying the graph:

1. Go through the given reachability graph and replace all alternating 4 cycles with the Steiner embedded alternating 4 cycle.

2. Simplify the graph using the algorithm described in the previous case

Lemma 4.9 If the simplified graph has any undirected cycles, it cannot be embedded.

Proof: All of the alternating 4 cycles were removed on the first pass, so any remaining 4 cycles resulted from a reduction which was proved to be not embeddable in Lemma 4.7. It was also proved that all alternating cycles are unembeddable. Any other undirected cycle $U$ can be reduced to an alternating cycle by joining together adjacent edges in the same direction into 1 edge. $U$ must be harder to embed than the alternating cycle $V$ it reduces to because if $U$ is embeddable, $V$ can be embedded by adding as Steiner points the vertices that would turn $V$ into $U$. However, since $V$ is unembeddable, $U$ must also be unembeddable.

Lemma 4.10 A reachability graph can be embedded into a tree if and only if there are no undirected cycles in the simplified graph.

Proof: The previous lemma proved that a graph is unembeddable if there are any undirected cycles. Therefore, all that needs to be proved is that a graph is embeddable if its simplified graph doesn’t have any undirected cycles.

First, connectivity does not matter when Steiner points are allowed because Steiner points can be added to turn an unconnected graph into a connected one. This is done in the following manner. For two unconnected subgraphs $T$ and $U$ that need to be connected, pick any vertex $y$ from $T$ and any vertex $w$ from $U$. Then create a Steiner point $s$ and draw edges from $s$ to $y$ and $s$ to $w$. This now connects these two graphs, but no additional reachability is added because the vertices are directed away from $s$. It is possible to continuously repeat this procedure until the tiling is connected.
In order for the simplified graph case to not have any undirected cycles, it means that the only cycles in the original graph were alternating 4 cycles, which can be embedded with Steiner points.

Combining the above 2 facts with the proof Lemma 4.2 means that a graph is embeddable if its simplified graph doesn’t have any undirected cycles.

5 Grids

5.1 Complete Layered Grid, No Steiner points

Lemma 5.1 Without Steiner points, the only thing that could be embedded is a p by q complete grid, where pq is the number of vertices in the original reachability graph.

Proof: This is trivially true because the vertices must form a rectangle because without Steiner points there cannot be any empty slots. Also, in this case, the final grid must be complete, so anything that wasn’t initially complete can’t be embedded.

Algorithm for detecting whether a reachability graph can be embedded:

1. Transitivity reduce the graph.
2. Identify all vertices of degree 0 (there should be only 1, if not this cannot be embedded) or degree 1. Together, these should form two total orders. If this is the first pass, define p and q by the lengths of the total orders. If this is not the first pass, the required lengths of the total orders will be passed in and if the total orders are not of the required lengths, return that this is not embeddable.
3. Recurse on the graph minus the vertices identified in step 1 with required total order lengths 1 less than the current required total order lengths
4. Connect the two total orders identified in step 1 as the leftmost column and topmost row and ensure that all reachability constraints with regard to these vertices are satisfied (Note: if p = q, this could require switching which total order is put vertically and which is put horizontally).

Lemma 5.2 A reachability graph $G$ is embeddable in a complete layered grid if and only if the algorithm above can embed it.

Proof: In the forward direction, it needs to be proved that if the reachability graph is embeddable, the algorithm returns embeddable. If it is embeddable, it can be drawn as a complete p by q grid without any empty slots. By the definition of what is being looking for, in step 1 there will only be 1 vertex of in degree 0 and the topmost row and leftmost column will be found as the two total orders. When the algorithm recurses, it will continue to find only 1 vertex of in degree 0 and identify the topmost row and leftmost column, and both will
be of the correct length. Since the rectangle is complete, when the algorithm moves into step 3, all of the expected reachability edges will be there and thus, the algorithm will not fail in this step. Therefore, the algorithm doesn’t fail in any step and will return embeddable.

In the reverse direction, it needs to be proved that if the algorithm returns embeddable then the reachability graph actually is embeddable. This is true trivially because the algorithm only returns true if it found a possible embedding, which of course implies that the reachability graph is embeddable.

5.2 Complete Layered Grid, Steiner points allowed

5.2.1 Input is Bipartite

First, note that since the grid is layered, if the graph to be embedded has a directed cycle then it cannot be embedded. This is true because it is impossible for a layered grid to have a directed cycle.

Definition 12 An antichain in a reachability graph $G(V, E)$ is a subset $U \subseteq V$ such that for all $u, v \in U$, $u$ cannot reach $v$ in $G$.

Lemma 5.3 In the embedding of an antichain, there is an induced total order by looking at the vertices going from south to north.

Proof: Since the vertices form an antichain, for every pair of vertices one must be strictly east and strictly north of the other. Since the grid is complete, if this were not true, one of the vertices would be able to reach the other, which is a contradiction since this is an antichain.

Definition 13 For two vertices $a$ and $b$ in an antichain, $a < b$ in an embedding if $b$ is north of $a$.

To aid us in creating an embedding of an antichain, we will introduce a notion called between. This will help us capture when $a < b < c$.

Path Condition: For three vertices $a$, $b$, and $c$ in an antichain, if there is an undirected path from $b$ to $c$ where none of the vertices on the path are in $\Gamma^+(a) \cup \Gamma^-(a)$, $a$ is not between $b$ and $c$.

Proof: Without loss of generality, say $b < c$. The following proof by induction on the number of intermediate vertices of the path will show that it is impossible to have $b < a < c$.

Base case 1 (number of intermediate vertices = 1): This means there is some vertex $d$ that can either be reached from both $b$ and $c$ or reaches both $b$ and $c$. Without loss of generality, say $d$ can be reached from both $b$ and $c$. This means $d$ is east of $c$ and south of $b$. If we were to have $b < a < c$, then $d$ would be east and south of $a$ and thus reachable from $a$. However, this is a contradiction because $a$ is not able to reach any vertex on the path.
Base case 2 (number of intermediate vertices = 2): Let us call the two vertices on the path \( s \) and \( t \). Without loss of generality, say \( s \) reaches \( c \). Since there must be an edge either from \( t \) to \( s \) or \( s \) to \( t \), if \( t \) reaches \( b \), then this can be reduced to a path with only 1 intermediate vertex and this is covered in the previous case. Therefore, there must be an edge from \( b \) to \( t \). If there is an edge from \( t \) to \( s \), then there is a directed path from \( b \) to \( c \), which contradicts the fact that they are in an antichain together. Therefore, there is an edge from \( s \) to \( t \).

Let us assume \( b < a < c \). \( a \) must be east of \( t \) because \( a \) cannot reach \( t \) and \( t \) is south of \( a \) (since \( b \) is south of \( a \) and \( t \) is south of \( b \)). This means \( a \) is east of \( s \) (since \( t \) is east of \( s \)). Also, \( c \) is south of \( s \) (since \( s \) reaches \( c \)) and \( a \) is south of \( c \) (since \( a < c \)). Therefore, \( a \) is both south and east of \( s \), which means it is reachable from \( s \). This is a contradiction because none of the vertices on the path are supposed to be able to reach \( a \).

**Induction Hypothesis:** Assume it is impossible to have \( b < a < c \) if the number of intermediate vertices is \( \leq n \).

**Inductive Step:** Assume there is a path \( P \) between \( b \) and \( c \) that has \( n + 1 \) intermediate vertices. Take any middle point in the path and call it \( f \).

If \( f \) is not in an antichain with \( b \) and \( c \), say without loss of generality that there is an edge from \( f \) to \( b \). A shorter path from \( b \) to \( c \) can be constructed by making the first vertex \( f \) and following \( P \) from \( f \) to \( c \). This removes at least one vertex since \( f \) was not an endpoint. Furthermore, since all the vertices used in the new path were also in \( P \), none of them are in \( \Gamma^+(a) \cup \Gamma^-(a) \). By the induction hypothesis, it is impossible to have \( b < a < c \).

If \( f \) is in antichain with \( b \) and \( c \), it is possible to break \( P \) into two parts such that there is a path from \( b \) to \( f \) and a path from \( f \) to \( c \) where all of the vertices are not in \( \Gamma^+(a) \cup \Gamma^-(a) \). It is obvious that both paths have length \( \leq n \). Also, by definition, there is no edge from \( f \) to \( a \) and no edge from \( a \) to \( f \), so \( f \) is in an antichain with \( a \) also. Therefore, by the Induction Hypothesis, it is impossible to have \( b < a < f \) or \( f < a < c \). This means it is impossible to have \( b < a < c \). \( \square \)

![Diagram](image_url)

Figure 8: Example of Path Condition
Private Vertex Condition: For three vertices $a$, $b$, and $c$ in an antichain, if $\exists$ vertex $d \in \Gamma^+(a) \cap \Gamma^+(b) \cap \Gamma^+(c)$ and $\exists$ vertex $e \in \overline{\Gamma^-(d)} \cap \Gamma^+(a) \cap \Gamma^+(b) \cap \Gamma^+(c)$, then $a$ is not between $b$ and $c$.

Proof: Without loss of generality, say $b < c$. Suppose $b < a < c$. By definition, the vertex $d$ is reachable from $b$ and $c$, so it must be south of $b$ and east of $c$. Also, the vertex $e$ is reachable from $a$, so it must be south and east of $a$, which makes it east of $b$ and south of $c$ (since $b < a$ means $a$ is east of $b$ and $a < c$ means $a$ is south of $c$). Since $e$ is not reachable from $b$ or $c$, it must be north of $b$ and west of $c$. This makes $d$ reachable from $e$, which is a contradiction. \[\square\]

Definition 14 For an antichain containing $a$, $b$, and $c$, $b$ is between vertices $a$ and $c$ if:

1. $a$ is not between $b$ and $c$ by the Path or Private Vertex Conditions
2. $c$ is not between $b$ and $a$ by the Path or Private Vertex Conditions

Lemma 5.4 If $b$ is between $a$ and $c$, $a < b < c$ or $c < b < a$.

Proof: Without loss of generality, say $a < c$. Since $a$ is not between, by the definition of the Path and Private Vertex Conditions, it is impossible to have $b < a < c$. Similarly, it is impossible to have $a < c < b$. Therefore, it must be true that $a < b < c$. \[\square\]

Corollary 1 If $b$ is between $a$ and $c$ and $a$ is between $b$ and $c$, then the graph cannot be embedded.

Proof: Let us suppose there is an embedding for $a$, $b$, and $c$. Without loss of generality, say $a < c$. Now, since $b$ is between $a$ and $c$, $a < b < c$. However, since $a$ is also between $b$ and
\( c, b < a < c \). Since it is impossible to have both \( a < b \) and \( b < a \), this is a contradiction and no embedding can exist.

**Lemma 5.5** For any antichain, there must be at least 1 vertex that is not between any other 2 vertices in the antichain or else there is no possible embedding.

**Proof:** Let us suppose there is no such vertex and there is an embedding. In the antichain, there must be a least vertex \( v \) according to the ordering, which means that for any other vertex \( u \) in the antichain, \( v < u \). However, by the assumption there must be vertices \( a \) and \( c \) with \( v \) between \( a \) and \( c \). From the previous lemma, one of \( a \) or \( c \) must be \( < v \), which is a contradiction to \( v \) being the least vertex.

**Corollary 2** Any alternating cycle of length \( \geq 6 \) cannot be embedded.

**Proof:** The set of all source vertices in the cycle is an antichain by definition. Since length is at least 6, no 2 source vertices share both their destination vertices. Therefore, for any source vertex \( b \), one of its destination vertices will be shared with another source vertex \( a \) and the other destination vertex will be shared with a source vertex \( c \) \( (\neq a) \). This means \( b \) is between \( a \) and \( c \) by the Path Condition on both sides. Therefore, by the previous lemma, this cannot be embedded.

**Algorithm for creating an ordering of an antichain**

At any stage of the algorithm, the vertices of the antichain are partitioned into a totally ordered collection of groups. Initially, all vertices are in the same group.

1. Pick any vertex \( R \) that is not between any other vertices in this antichain. If there is no such vertex, fail.
2. Set \( R \) as the next lowest vertex in the ordering.
3. Refine the existing groups as follows: Break each group into subgroups based on which vertices in \( \Gamma^+ (R) \) they can reach.
4. If we only have 1 group at this point and none of the previously placed vertices share out vertices with this group, return to step 1.
5. Order the groups first based on any previous ordering and then by which of the vertices that group can reach that \( R \) can also reach (with the group with the most vertices being ordered first). Now, form a descending sequence of the vertices that each group can reach that \( R \) can also reach. If the sequence is not monotonically decreasing sequence with respect to subset, then fail.
6. From the first group (call it \( G \)), if there is only 1 vertex, go to step 7. If not, go to step 8. Call the group of vertices that \( R \) and \( G \) can both reach \( H \).
7. Set the only vertex in \( G \) as \( R \) and return to step 2.
8. For every vertex T in G, find the set of vertices in $\Gamma^+(T)$ that cannot reach any vertices in H. Organize these in an ascending sequence, and if the resulting sequence is not monotonically increasing, fail. Among the vertices for which the group constructed is minimal, choose any vertex that is not between any two unplaced vertices. Call this chosen vertex R and return to step 2.

**Lemma 5.6** In Step 1 of the previous algorithm, it is always possible to find a vertex that is not between any other vertices in this antichain.

**Proof:** All remaining unplaced vertices form an antichain. If it is not possible to find a vertex not between any other two vertices, it would be a violation of Lemma 5.5.

**Lemma 5.7** For any already placed vertex A that shares common out vertices with G (the current lowest group), there cannot be two vertices B and C $\in G$ where

1. $\exists$ a vertex $D \in \Gamma^+(B) \cap \Gamma^+(A) \cap \Gamma^+(C)$ such that $\exists$ a vertex $E \in \Gamma^-(D) \cap \Gamma^+(B) \cap \Gamma^+(A) \cap \Gamma^+(C)$ AND

2. $\exists$ a vertex $F \in \Gamma^+(B) \cap \Gamma^+(A) \cap \Gamma^+(C)$ such that $\exists$ a vertex $G \in \Gamma^-(F) \cap \Gamma^+(C) \cap \Gamma^+(A) \cap \Gamma^+(B)$

**Proof:** First, note that A is between B and C by the Private Vertex Condition on both sides.

Since B and C are currently in the same group, they must always have been in the same group. Therefore, this can be broken into two cases.

**Case 1 (A was in an earlier group than B and C when chosen):** This means $\exists$ vertex $H \in \Gamma^+(A) \cap \Gamma^+(B) \cap \Gamma^+(C)$. This is true because that is the vertex that separates A into a different group. If H cannot reach D or cannot reach E, B would be between A and C by the Private Vertex Condition on both sides, which is a contradiction of Corollary 1. If H can reach both D and E, $\exists$ vertex Z chosen before A that reaches H and therefore reaches both D and E. Therefore, it is possible to recurse and repeat this analysis on Z. Eventually, it must not be possible to get into this situation because only a finite number of vertices have been placed and recursing only goes to vertices that were chosen earlier.

**Case 2 (A was in the same group as B and C when chosen):** First, it is impossible to have chosen A in step 1, since A is between B and C and in step 1, the algorithm may only choose vertices that are not between any other pair of vertices. This means that A could not have been the first vertex chosen.

Now, suppose A was chosen in Step 8. Look at the vertex chosen before A and call it Z. The group with A, B, and C must be the lowest because A was the next vertex chosen and if Z does not share any vertices with this group, then the algorithm would have gone back to step 1. Therefore, there is some vertex H that all 4 can reach. In order for A to be chosen before B and C, there must be some vertex I that B can reach that neither A or Z can and some vertex J that C can reach that neither A or Z can. I and J cannot be the same vertex.
because then B and C would share a vertex that A doesn’t, which would make it impossible for A to be between B and C by the Path Condition. Furthermore, by definition both I and J cannot reach H, which means it is possible to recurse and repeat this analysis on Z.

If in Step 8, the algorithm cannot form a monotonically increasing sequence of the groups of private vertices, then it is in exactly the situation described by the above lemma. Therefore, failing is valid because there is no possible embedding in that situation.

**Lemma 5.8** In Step 5 of the previous algorithm, a sequence S of groups is produced. It is also possible to make a map M that takes a group and returns \( \Gamma^+(W) \cap \Gamma^+(R) \) where W is any vertex in the group (because this map would return the same value for any vertex in the group) and R is defined in previous steps. The sequence T that is formed by mapping the sequence S using M is a monotonically decreasing sequence with respect to subset.

**Proof:** Suppose this was not true. There is a first time the violation occurs. This means that there are vertices D and E with E in a higher group before Step 3 and \( \exists \) a vertex F in \( \Gamma^+(E) \cap \Gamma^+(R) \cap \Gamma^+(D) \). Let us designate as A the earlier vertex that separated D and E. This means there is a vertex C in \( \Gamma^+(A) \cap \Gamma^+(D) \cap \Gamma^+(E) \). F cannot reach C because E can reach F but cannot reach C. Since this is the first violation, R must be able to reach C as well since D is chosen after C.

**Case 1 (R and D were in the same group before Step 3):** In order for R to be chosen before D, D must be able to reach vertices that A and R cannot. Since F exists, this would be a violation of lemma 5.7.

**Case 2 (R and D were in different groups before Step 3):** This means there is some previously placed vertex B that separated R into an earlier group than D. This means \( \exists \) vertex J in \( \Gamma^+(B) \cap \Gamma^+(R) \cap \Gamma^+(D) \). If B is A or was placed before A, then J is reachable from A also because this is the first time a violation is occurring. If B was placed after A, then B must be able to reach C also because this is the first time a violation is occurring. Therefore, by similar analysis to that just performed on R, B must have been in an earlier group than D when chosen. By continuously repeating this analysis, eventually the analysis gets to A or a vertex placed before A since only a finite number of vertices have been placed. It is possible to create a path from A to R of the vertices this analysis considered and the vertices that forced them into lower groups. Therefore, A is between R and D by the Path Condition for D and the Private Vertex Condition for R.

Knowing this, all that is left to be proved is that it is impossible for A to have been chosen before R. A could not have been chosen in step 1 because it is between R and D. If A was in a lower group than R, then it reaches a vertex that R and D both cannot reach. Since C is reachable from all 3, A can not be between R and D by the Private Vertex Condition, which is a contradiction. If A and R are in the same group, there must be some vertex K placed before A and a vertex I in \( \Gamma^+(K) \cap \Gamma^+(A) \cap \Gamma^+(R) \). By the Path Condition on both sides, R is between K and E (path to K through I and path to E through F) and R is between D and E (path to D through C and path to E through F).

If K cannot reach C, R is also between K and D (path to K through I and path to D through C). This is impossible because say without loss of generality that \( K < R < E \). This
also means that \( K < R < D \). However, this means that \( R < E \) and \( R < D \), which is a contradiction of \( R \) being between \( E \) and \( D \).

If \( K \) can reach \( C \), it is possible to recurse and repeat this analysis on \( K \) because \( K \) is also between \( R \) and \( D \). Since there are only a finite number of already placed vertices, this analysis must eventually no longer be able to reach this situation. Therefore, \( A \) could not have been chosen before \( R \), which is a contradiction.

Lemma 5.9 If the input is a bipartite graph, the above algorithm can be used to determine whether or not there is an embedding.

Proof: Since the input graph is bipartite, it can be divided into two antichains: the source vertices and the destination vertices. Run the above algorithm on the source vertices. If the algorithm returns fail, then an embedding does not exist. Otherwise, the algorithm returns an ordering of the source vertices. It is now possible to find an ordering of the destination vertices by doing the following:

1. Sort the destination vertices on the lowest source vertex that reaches them and organize the resulting groups in ascending order

2. Refine the ordering by sorting on the highest source vertex that reaches each destination vertex and organize in ascending order

Let us consider the ways that an embedding would not exist. If an embedding did not exist, then there must be some violation of the ordering. This can occur by finding a pair of destination vertices \((u, v)\) with \( u < v \), such that either \( v \) is forced to be south or west of \( u \) or \( u \) is forced to be north or east of \( v \). Therefore, the following two cases are exhaustive.

Case 1 (there is some source vertex \( x \) such that there is an edge from \( x \) to \( u \), there is no edge from \( x \) to \( v \), and \( x \) is higher in the ordering than all vertices in \( \Gamma^{-}(v) \)):

If \( u \) and \( v \) are in the same group after the first sort, by definition the second sort should make \( v < u \), which is a contradiction.

Therefore, \( u \) is in a lower group after the first sort. Let us denote the lowest source vertex in \( \Gamma^{-}(u) \) as \( y \). \( y \) must be below the lowest source vertex in \( \Gamma^{-}(v) \). After the algorithm chooses \( y \), \( x \) will be in a lower group than all of the vertices in \( \Gamma^{-}(v) \cap \Gamma^{-}(u) \). Furthermore, \( x \) cannot be in a higher group than the vertices in \( \Gamma^{-}(v) \cap \Gamma^{-}(u) \) and since these vertices can reach \( v \) and \( x \) cannot, \( x \) would have to be chosen before them. Therefore, it is impossible to have \( x \) higher than the vertices in \( \Gamma^{-}(v) \), which is a contradiction.

Case 2 (there is some source vertex \( x \) such that there is an edge from \( x \) to \( v \), there is no edge from \( x \) to \( u \), and \( x \) is lower in the ordering than all vertices in \( \Gamma^{-}(u) \)):

By definition, the first sort will make \( v < u \), which is a contradiction.

Therefore, an embedding must exist.
5.2.2 Input is not Bipartite

When the input graph is not bipartite, the algorithm used in the previous section will need to be extended. This is true because when ordering a particular antichain, the algorithm may get to a point where it has multiple choices for which vertex to choose next. In the ordering of just one antichain, any of these choices can be picked and it is still possible to make an ordering. However, there may have been some edges between the first antichain and the third (or higher) antichain that prevents the first antichain from seeing all the same between relationship as the second antichain. Therefore, when ordering the second antichain, the algorithm may find that an embedding was only possible if a certain choice had been made. As a result, there must be a mechanism to allow the necessary information to be passed upwards.

As a start, there will need to be an additional notion added to the between definition. B is between A and C if all of the following are true:

1. $\exists$ a triple of vertices (D,E,F) with E between D and F
2. $E \in \Gamma^+(B) \cap \Gamma^+(C) \cap \Gamma^+(A)$
3. $(D \in \Gamma^+(A) \cap \Gamma^+(C) \text{ and } F \in \Gamma^+(C) \cap \Gamma^+(A)) \text{ OR } (F \in \Gamma^+(A) \cap \Gamma^+(C) \text{ and } D \in \Gamma^+(C) \cap \Gamma^+(A))$

**Lemma 5.10** If B is between A and C by this new notion, then either $A < B < C$ or $C < B < A$ in the ordering of the antichain.

**Proof:** Assume this lemma is true for D, E, and F. Without loss of generality, say $D < E < F$. Also, without loss of generality, say $D \in \Gamma^+(A) \cap \Gamma^+(C)$ and $F \in \Gamma^+(C) \cap \Gamma^+(A)$. By definition, E must be south and east of B. If $B < A$, then A must be east of E because A cannot reach E and E is south of A (since $B < A$). However, this means A is east of D (since $D < E$) and thus, A cannot reach D, which is a contradiction. Therefore, $A < B$. By similar analysis, $B < C$. Therefore, $A < B < C$. \qed

**Conjecture 1** This additional definition captures all "inherited" notions of between from one antichain to another.

**Proposed Algorithm for creating an embedding**

1. Break the graph into groups of vertices on the length of the longest path to a in-degree 0 vertex

2. Create an ordering for the lowest group not yet ordered using the above algorithm (with a slight modification in that in steps 1 and 8 where it is possible to choose multiple vertices, the algorithm chooses vertices reachable from the lowest possible vertex in all previous orderings). Check that this ordering is consistent with all previous orderings (ie if $u < v$ in any previous ordering, for any vertex w that is reachable from u but not v and x that is reachable from v and not u, this ordering must have $w < x$.) If not consistent, fail.
3. Repeat step 2 until all vertices have been ordered

**Conjecture 2** The above algorithm finds an embedding if one exists and fails otherwise.

### 5.3 Non-layered Grids

There are currently two unembeddable reachability graphs.

The first is a bipartite graph with 10 source vertices and 5 destination vertices where each of the source vertices can reach a distinct pair of destination vertices.

**Lemma 5.11** In any possible embedding of the above reachability graph, no paths originating from distinct vertices and ending at distinct vertices cross.

**Proof:** Assume that a path from $x$ to $y$ crosses a path from $u$ to $v$ (with $x \neq u$ and $y \neq v$). Therefore, both $x$ and $u$ can reach both $y$ and $v$, which is impossible because the source vertices reach distinct pairs of destination vertices.

Suppose an embedding existed. This graph can be transformed into a planar embedding by making all of the edges undirected and separating paths that come from the same source vertex or go to the same destination vertex. This transformed graph would be a planar embedding of $K_5$, which cannot exist.

The second unembeddable reachability graph is also a bipartite graph with 9 source vertices and 6 destination vertices that can be broken into 2 groups of 3. Each of the source vertices can reach exactly 1 vertex from the first group and exactly 1 vertex from the second group and no 2 source vertices can reach the same pair.

Suppose an embedding existed. Following the same method as with the first structure would yield a planar embedding of $K_{3,3}$ which cannot exist.

### 6 Conclusion

This paper has given characterizations of embeddable reachability graphs for lines (with and without Steiner points), trees (with and without Steiner points), complete layered grids without Steiner points, and complete layered grids with Steiner points on bipartite inputs. From these characterizations, it is possible to conclude the following relationships between the classes of embeddable graphs.

$S$ without Steiner points $\subset S$ with Steiner points where $S$ can be trees, lines, or complete grids

It is obvious that any graph that can be embedded without Steiner points can be embedded with Steiner points by using 0 Steiner points.

The following graph $G$ is embeddable for all 3 structures with Steiner points, but not embeddable without Steiner points. There are 4 vertices ($a$, $b$, $c$, and $d$) and 2 edges ($a$ to $b$ and $c$ to $d$).
Figure 10: $G$

lines without Steiner points $\subset$ trees without Steiner points

This is obvious because saw-tooth graphs are simply a special case of tree orders where terminals can be in at most two total orders.

The following graph $G_1$ can be embedded in trees without Steiner points, but not lines without Steiner points. There are 4 vertices ($a$, $b$, $c$, and $d$) and 3 edges ($a$ to $b$, $c$ to $b$, and $d$ to $b$).

Figure 11: $G_1$

lines with Steiner points $\subset$ trees with Steiner points

This is also obvious because disjoint unions of saw-tooth graphs are again a special case of disjoint unions of tree orders. Disjoint unions of tree orders are a subset of the embeddable graphs in trees with Steiner points.

$G_1$ can be embedded in trees with Steiner points, but not in lines with Steiner points.

no subset relationships between lines without Steiner points and complete
layered grids without Steiner points

The following graph $G_2$ can be embedded in lines without Steiner points, but not in complete layered grids without Steiner points. There are 3 vertices ($a$, $b$, and $c$) and 2 edges ($a$ to $b$ and $c$ to $b$).

![Figure 12: $G_2$](image12.png)

The following graph $G_3$ can be embedded in complete layered grids without Steiner points, but not in lines with or without Steiner points. There are 4 vertices ($a$, $b$, $c$, and $d$) and 4 edges ($a$ to $b$, $a$ to $c$, $b$ to $d$, and $c$ to $d$).

![Figure 13: $G_3$](image13.png)

lines with Steiner points $\subset$ complete layered grids with Steiner points

Proof by induction that saw-tooth graphs are embeddable in complete layered grids:

Base case: A total order is obviously embeddable in a complete layered grid by placing all vertices in a line going either horizontally or vertically.

Induction Hypothesis: Assume a saw-tooth graph resulting from $n$ compositions is embeddable in a complete layered grid with its terminals at the ends of its embedding.
Given a saw-tooth graph $H_1$ resulting from $n + 1$ compositions, by definition it is composed of a saw-tooth graph $H_2$ with terminals $t_1$ and $t_2$ made of $n$ compositions and a total order $H_3$ with terminals $t_2$ and $t_3$. By the Induction Hypothesis, $H_2$ is embeddable in the complete layered grid with $t_1$ and $t_2$ at the ends. $H_3$ is also embeddable in a complete layered grid with $t_2$ and $t_3$ at the ends. Since $H_2$ and $H_3$ share $t_2$, these two embeddings can be connected at $t_2$ with $H_3$ oriented horizontally if $t_2$ is its max vertex and oriented vertically otherwise. This means $H_1$ is embeddable and its terminals $t_1$ and $t_3$ are at the ends. Therefore, the inductive step is proved.

$G_3$ can be embedded in complete layered grids with Steiner points, but not in lines with Steiner points.

**no subset relationships between trees with or without Steiner points and complete layered grids with without Steiner points**

The following graph $G_4$ can be embedded in trees without Steiner points, but not in complete layered grids with or without Steiner points. There are 7 vertices ($a$, $b$, $c$, $d$, $e$, $f$, and $g$) and 6 edges ($a$ to $d$, $b$ to $d$, $c$ to $d$, $a$ to $e$, $b$ to $f$, $c$ to $g$).

![Diagram](image)

Figure 14: $G_4$

$G_3$ can be embedded in complete layered grids without Steiner points, but not in trees with or without Steiner points.

Due to time constraints, this paper was not able to consider all of the cases that I originally hoped it would. The case of complete layered grids with Steiner points turned out to be much more difficult than anticipated. My faculty advisors and I were not able to predict that this seemingly constrained structure would require so many lemmas and such complex definitions and algorithms. As a result, there was not enough time to do any significant work on partial layered grids or non-layered grids. However, this problem is still of great interest to us. The methods used in the complete layered grid case and the section that presents some unembeddable structures will serve as a good starting point for any future research into this area.
References


This paper was published in Advances in Computing Research, which only accepts original work. At the time of publication, the authors were all faculty members at either MIT or Stanford. The paper considers the problem of embedding graphs in grids, which is motivated by VLSI chip design. The main result of the paper is to use separator theorems to layout a graph in relatively good area. It does this by removing a small number of edges to break the graph into parts and then recursively lays out each of the parts. Reading about this method has been very informative for me and given me some ideas about how to approach problems, but I have not been able to apply it. This is true because it allows crossings or places where wires cross through the same point. I cannot allow crossings because they potentially create additional reachability among vertices.


This conference paper was published by the IEEE Computer Society. It considers the problem of giving a planar embedding of graphs that have maximum degree of 4. In particular, it considers 2 very specific classes of graphs called G(≤3) and G(≤4). I haven’t really been able to apply this paper that much, but I am hoping to possibly reuse some of the methods or be able to reduce graphs to the classes considered by this resource.


This scholarly paper was submitted to the MIT Department of Electrical Engineering and Computer Science to fulfill the requirements of a PhD degree. This paper is rather old in age, but it remains very relevant and was listed as a resource in many of the other papers I looked at. It covers in great detail the problem of laying out a VLSI chip design, which is very similar to my problem of embedding a graph into a grid. Many of the methods have been interesting for me to read about and have given me some ideas about how to approach problems. However, I have not been able to apply any of them directly because these methods allow paths to cross but not overlap, while my problem calls for allowing paths to overlap but not cross.


This journal article was published in the Journal of Combinatorial Theory, which is peer reviewed and only accepts original research. It was written by Dan Bienstock while he was at Bell Communications Research. This paper considers the problem of embedding graphs into trees. This is similar to the problem I am considering, but there are 2 rather large differences: it only considers undirected graphs and it forces all vertices to be at
leaves of the tree. This resource was interesting to read and got me to consider some ideas, but was ultimately not that influential to my project.


This journal article was published by the IEEE Circuits and Systems Society. It was peer recommended by a guest editor of the journal and supported by multiple research programs. In the article, the authors consider the problem of embedding graphs into grids. They are able to show very strong results regarding the running time, number of bends used, and area of the embedding. However, the graphs that are considered have maximum degree 4, which greatly constrains the problem and limits the applicability to my paper.