On the Complexity of Embedding in Graph Products

Therese Biedl
David R. Cheriton School of Computer Science, University of Waterloo

David Eppstein
Department of Computer Science, University of California, Irvine

Torsten Ueckerdt
Institute of Theoretical Informatics, Karlsruhe Institute of Technology

Abstract

Graph embedding, especially as a subgraph of a grid, is an old topic in VLSI design and graph drawing. In this paper, we investigate related questions concerning the complexity of embedding a graph G in a host graph that is the strong product of a path P with a graph H that satisfies some properties, such as having small treewidth, pathwidth or treedepth. We show that this is NP-complete, even under numerous restrictions on both G and H. In particular, computing the row pathwidth and the row treedepth is NP-hard even for a tree of small pathwidth, while computing the row treewidth is NP-hard even for series-parallel graphs.

Keywords and phrases Row treewidth, Row pathwidth, Row treedepth, Graph product structure, NP-completeness

Digital Object Identifier 10.57717/cgt.v4i2.56

Related Version An extended abstract of this work appeared at the 35th Canadian Conference on Computational Geometry, CCCG 2023, in Montreal, Canada [5].

Funding *Therese Biedl*: Supported by NSERC grant RGPIN-2020-03958.
David Eppstein: Research supported in part by NSF grant CCF-2212129.
Torsten Ueckerdt: Research in part funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 532213793

Acknowledgements This work was initiated at the Workshop on Graph Product Structure Theory (BIRS21w5235) at the Banff International Research Station, November 21–26, 2021. We would like to thank the organizers and all participants of that event.

1 Introduction

Layered treewidth, layered pathwidth, and row treewidth are structural parameters of graphs that have played a central role in recent developments in graph product structure theory. (The original graph product structure theorem was proved by Dujmovíc et al. [9]; see also [22, 8, 11] for improvements and related results.) It is known that testing whether a graph has layered pathwidth ≤ 1 is NP-complete [3]. In this work we ask analogous questions about the computational complexity of the *row treewidth* of a graph G, the minimum possible treewidth of a graph H such that G is a subgraph of the strong product $H \boxtimes P_\infty$ where P_∞ is a 1-way infinite path:

- Is it NP-hard to compute the row treewidth?
- Is it NP-hard to test whether a planar graph has row treewidth 1, its smallest nontrivial value?
- How complicated must G be for these problems to be hard? Are they easier for planar graphs?

© T. Biedl, D. Eppstein, and T. Ueckerdt
licensed under Creative Commons License CC-BY 4.0

Row treewidth can be naturally generalized to other product forms for H. For example, the *row pathwidth* of a graph G is the smallest possible pathwidth of a graph H such that G is a subgraph of $H \boxtimes P_\infty$, and similarly one can define the *row treedepth* or *row simple treewidth* or *row simple pathwidth*. The above questions could be asked for any of these parameters. Let us remark that each of these parameters (row treewidth, row treedepth, etc.) is unbounded for complete graphs, i.e., for every integer $k \geq 1$ there are graphs with row treewidth k and the same holds for the other parameters.

Our problems are special cases of subgraph isomorphism, which is hard even under strong restrictions on both G and the host graph [19]. Our answers are that these problems are indeed hard, even for very simple graphs. It is NP-complete to test whether a tree of bounded pathwidth has row pathwidth one, and the same holds for row simple pathwidth and row treedepth. Row treewidth is trivial for trees, but it is NP-complete to test whether series-parallel graphs of bounded degree and bounded pathwidth have row treewidth one. Under the small set expansion conjecture (a strengthening of the unique games conjecture from computational complexity theory), row treewidth, row pathwidth, layered treewidth, and layered pathwidth are hard to approximate with constant approximation ratio.

These questions have a geometric flavor coming from the grid-like graph products they concern. For example, as we will see, a graph has row simple pathwidth ≤ 1 if and only if it is a subgraph of a strong product of two paths (a grid with diagonals), if and only if it has a unit-distance embedding into the two-dimensional integer lattice with L_∞ distances. We provide a few positive results as well: Testing embeddability in $P_\infty \boxtimes P_\infty$ is polynomial for caterpillars, and testing embeddability in $P_\infty \square P_\infty$ (equivalently, the two-dimensional integer lattice with L_1 distances) is polynomial for planar graphs of bounded treewidth and bounded face size.

1.1 Definitions

A *tree decomposition* of a graph G is a tree T whose vertices (called *bags*) are labeled with subsets of vertices of G. For each vertex v of G, the bags containing v must form a connected subtree of T, and each edge of G must have both endpoints included together in at least one bag. A *path decomposition* is a tree decomposition whose tree T is a path. The *width* of a tree decomposition is the size of the largest bag, minus one. The *treewidth* of G is the smallest width of a tree decomposition of G, and the *pathwidth* is the smallest width of a path decomposition. A tree decomposition is *w-simple* if each set of w vertices belongs to at most two bags. The *simple pathwidth [simple treewidth]* of G is the smallest w such that G has a w-simple path [tree] decomposition of width $\leq w$. The *treedepth* of a graph G is the smallest height of a rooted tree T on the vertices of G such that every edge of G connects an ancestor-descendant pair in T.

For connected graphs with at least one edge, these width parameters have minimum value one. The connected graphs that have treewidth one are exactly the trees. The graphs with treewidth two are the *series-parallel graphs* and their subgraphs. The connected graphs with pathwidth one include not only the path graphs, but *caterpillars*: trees whose non-leaf vertices form a path called the *spine*. (The leaves of a caterpillar are called its *legs*.) The graphs with simple pathwidth one are exactly the paths. The graphs with treedepth one are *stars*, graphs $K_{1,\ell}$ for integer ℓ. To avoid having to specify a specific number of vertices, it is convenient to let $P_\infty = \langle p_0, p_1, \dots \rangle$ denote a *ray*, a one-way infinite path, to let C_∞ denote the caterpillar with infinite-length spine and infinitely many legs at each spine-vertex, and to let S_∞ be a star with infinitely many degree-1 vertices. Thus, the finite graphs of pathwidth one are simply the finite subgraphs of C_∞, the finite graphs of simple pathwidth one are the

finite subgraphs of P_∞, and the finite graphs of treedepth one are the finite subgraphs of S_∞.

The *strong product* of two graphs $G \boxtimes H$ has a vertex (u_i, v_j) for each pair of a vertex u_i in G and a vertex v_j in H, and an edge connecting two pairs (u_i, v_j) and $(u_{i'}, v_{j'})$ when u_i and $u_{i'}$ are either adjacent in G or identical, and v_i and $v_{i'}$ are either adjacent in H or identical. For instance, the strong product of two paths is a *king's graph*, the graph of moves of a chess king on a chessboard whose rows and columns are indexed by the vertices of the paths (see also Figure 1).

A *layering* of a graph G is a partition of the vertices into sets $L_0, L_1, \dots$ (called *layers*) such that for each edge, its two endpoints are either in the same layer or in consecutive layers. A layering can be understood as a representation of a graph G as a subgraph of $P \boxtimes K$ for a path P and a complete graph K; the layers of the layering are the subsets of vertices in this product coming from the same vertex of the path. Layered tree decompositions and path decompositions of a graph consist of a tree or path decomposition of the graph, together with a layering. Their width is the size of the largest intersection of a bag with a layer, minus one. The layered treewidth [10, 21] or layered pathwidth [3] of G is the minimum width of such a decomposition. Instead, the *row treewidth* or *row pathwidth* of G is the minimum treewidth or pathwidth of a graph H for which G is a subgraph of $P \boxtimes H$ for some path P. Intuitively, row treewidth and row pathwidth restrict the notion of layered treewidth and layered pathwidth by requiring each layer to have the same decomposition. These concepts are not equivalent: the layered treewidth of any graph G is at most its row treewidth plus one, but there exist graphs with layered treewidth one and arbitrarily large row treewidth. A similar separation occurs also between layered pathwidth and row pathwidth [7]. We can analogously define layered simple treewidth, layered simple pathwidth, layered treedepth, and row simple treewidth, row simple pathwidth, and row treedepth; to our knowledge these parameters have not been studied previously.

We show that the following problems are NP-complete:

- RowSimplePathwidth: Given a graph G and an integer k, does G have row simple pathwidth at most k? We will show that this is NP-complete even for $k = 1$, where it becomes the question whether G is a subgraph of $P_\infty \boxtimes P_\infty$, i.e., the king's graph, which is why we also call this problem KingGraphEmbedding. See Section 2.
- RowPathwidth: Given a graph G and an integer k, does G have row pathwidth at most k? We will show that this is NP-complete even for $k = 1$, where it becomes the question whether G is a subgraph of $C_\infty \boxtimes P_\infty$. See Section 3.
- RowTreewidth: Given a graph G and an integer k, does G have row treewidth at most k? We will show that this is NP-complete even for $k = 1$, where it becomes the question whether G is a subgraph of $T \boxtimes P_\infty$ for some tree T. See Section 4.
- RowTreedepth: Given a graph G and an integer k, does G have row treedepth at most k? We will show that this is NP-complete even for $k = 1$, where it becomes the question whether G is a subgraph of $S_\infty \boxtimes P_\infty$. See Section 6.

In each case, a yes answer is equivalent to the existence of a representation of the given graph as a subgraph of a product of a path with a graph of bounded width or depth. The factors of this product can be assumed to have linear size. If such a representation is given, the width or depth of the factor and the subgraph isomorphism of the given graph with the product can be verified in polynomial time. Therefore, all of these problems are in NP. We focus our proofs of NP-completeness on the nontrivial part of the proofs, showing that these problems are NP-hard.

It is helpful to introduce some notation for the strong product $H \boxtimes P_\infty$. Recall that P_∞ is a ray $\langle p_0, p_1, \dots \rangle$. For any vertex $v \in H$, the *P-extension* is the set of vertices

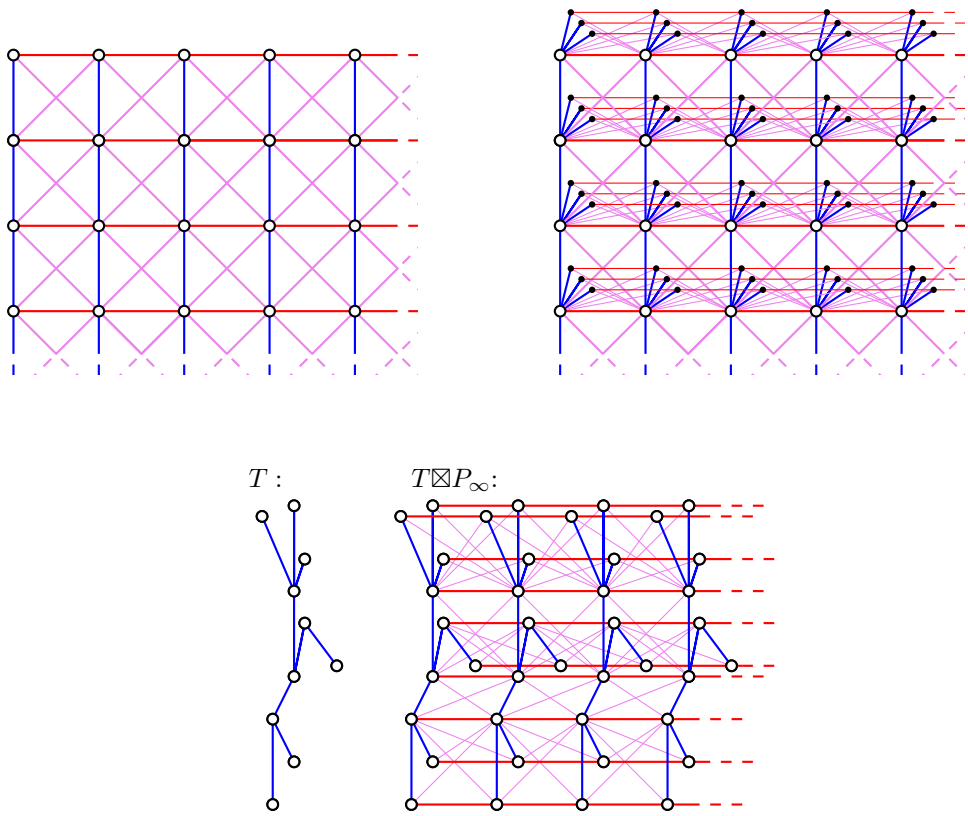

Figure 1 The graphs $P_\infty \boxtimes P_\infty$, $C_\infty \boxtimes P_\infty$, and $T \boxtimes P_\infty$ for a tree T.

$\langle (v, p_0), (v, p_1), \dots \rangle$. For any vertex $(v, p_i) \in H \times P_\infty$, the *$H$-projection* is the vertex v and the *P-projection* is the vertex p_i. These concepts naturally extend to edges and paths. Inspired by the case $H = P_\infty$ (where $H \boxtimes P_\infty$ is the king's graph) we define the following *edge-orientations*: An edge vw of $H \boxtimes P_\infty$ is *horizontal* if v, w have the same H-projection, *vertical* if they have the same P-projection, and *diagonal* otherwise. Since we expand along a path P_∞, every vertex of $H \boxtimes P_\infty$ has only two incident horizontal edges.

We will occasionally also study the *Cartesian product* $H \,\square\, P_\infty$ of two graphs, which is the same as the strong product except diagonals are omitted. In particular, $P_\infty \,\square\, P_\infty$ is the infinite rectangular grid.

2 Grid embeddings

In this section we study KingGraphEmbedding. This problem is closely related to GridEmbedding, the question whether a given graph G is a subgraph of the rectangular grid $P_\infty \,\square\, P_\infty$. GridEmbedding is old and well-studied since at least the 1980s due to its connections to VLSI design. Bhatt and Cosmadakis showed in 1987 [4] that GridEmbedding is NP-complete even for trees of pathwidth 3 (the pathwidth was not studied explicitly by the authors, but can be verified from their construction). Gregori [15] expands their proof to binary trees. Both proofs use a technique later called the "logic engine" by Eades and Whitesides [12]. Recently, Gupta et al. [17] strengthened the result to trees of pathwidth 2.

▶ **Theorem 1** (Gupta et al. [17]). GridEmbedding *is NP-complete even for a tree of pathwidth 2.*

The reductions in [17, 4] by Bhatt and Cosmadakis, and Gupta et al. can be modified to work for KingGraphEmbedding. Even easier is to use the following general-purpose transformation, which shows that any graph G can be modified into a graph G' such that G has an embedding in $P_\infty \,\square\, P_\infty$ if and only if G' has an embedding in $P_\infty \boxtimes P_\infty$.

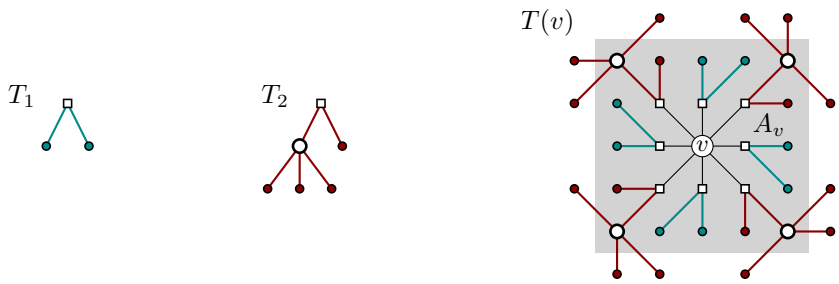

Figure 2 The trees T_1, T_2, and $T(v)$ in Observation 2.

Define T_1 and T_2 to be the trees shown in Figure 2, formed by subdividing one edge of $K_{1,1}$ and $K_{1,4}$ respectively. For a given vertex v, define $T(v)$ to be a tree rooted at v with eight children: four copies each of T_1 and T_2, connected to v at their degree-2 vertices.

▶ **Observation 2.** *Let G be a simple graph. Form G' by replacing each vertex v in G by a new tree $T(v)$, and connecting a degree-4 vertex in $T(u)$ with a degree-4 vertex in $T(v)$ for each edge uv in G. Then $G \subset P_\infty \,\square\, P_\infty$ if and only if $G' \subset P_\infty \boxtimes P_\infty$.*

Proof. The forward direction is obvious: If $G \subset P_\infty \,\square\, P_\infty$, then take the embedding, rotate it by 45° and stretch it such that neighboring grid vertices are $5\sqrt{2}$ units apart. Place this in $P_\infty \boxtimes P_\infty$ and verify that each $T(v)$ can be placed, and for each edge of G the two respective degree-4 can be connected as in Figure 3.

For the other direction, assume that G' has an embedding in $P_\infty \boxtimes P_\infty$. Observe that for any vertex v in G, the set $S_v = \{w \in V(G') \colon \operatorname{dist}(v,w) \le 2\}$ has size $|S_v| = 1 + 8 + 16 = 25$, and occupies a 5×5 square area A_v in $P_\infty \boxtimes P_\infty$. For any edge $e = uv$ in G, the corresponding 5-path u-s_1-s_2-s_3-s_4-v must be embedded along five diagonals of $P_\infty \boxtimes P_\infty$ with the same slope. This holds as s_2 has four neighbors outside S_u (s_3 and three vertices of $T(u) \setminus S_u$) and thus must be on a corner of A_u; and symmetrically s_3 lies on a corner of A_v. Finally (s_2, s_3) must be diagonal (and have the same slope), otherwise there would not be six vertices of $P_\infty \boxtimes P_\infty$ that are outside $S_u \cup S_v$ but adjacent to s_2 or s_3. ◀

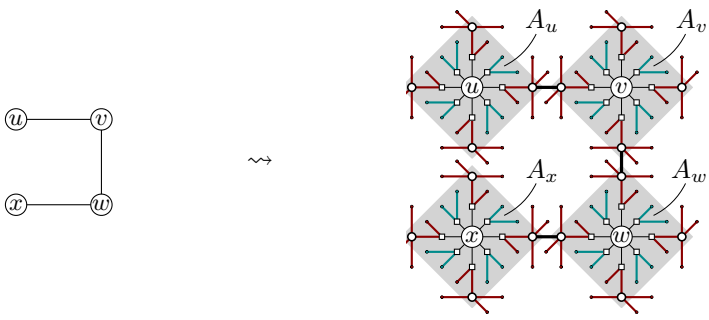

Figure 3 If $G \subset P_\infty \,\square\, P_\infty$, then $G' \subset P_\infty \boxtimes P_\infty$. (We show a 45° rotation of $P_\infty \boxtimes P_\infty$.)

The transformation in Observation 2 clearly maintains a tree and increase the size by only a constant factor. Replacing each vertex v by a tree $T(v)$ of radius 3 increases the pathwidth by at most 3, so applying the transformation to the tree of Gupta et al. [17] gives the following.

▶ **Corollary 3.** KingGraphEmbedding *is NP-complete, even for a tree of pathwidth at most* 5.

In fact, one can easily adapt the reduction of Gupta et al. [17] to show NP-completeness of KingGraphEmbedding even for a tree of pathwidth 2. We defer a sketch of a proof of this to Section 3 where similar techniques are used.

2.1 Caterpillars

On the other hand, for pathwidth 1 (i.e., caterpillars), we can solve KingGraphEmbedding in linear time.

▶ **Theorem 4.** *For any caterpillar G the following are equivalent.*
(1) $G \subset P_\infty \boxtimes P_\infty$
(2) G can be embedded in $P_\infty \boxtimes P_\infty$ such that all spine edges are diagonal
(3) For every subpath Q of the spine of G we have $\sum_{v \in V(Q)} \deg(v) \leq 6|V(Q)| + 2$.

Proof. (1) $\Longrightarrow$ (3): Assume that G is a caterpillar that is a subgraph of $H = P_\infty \boxtimes P_\infty$. Let Q be any fixed subpath of the spine of G. Clearly, for any vertex $v \in P_\infty \boxtimes P_\infty$ the neighborhood $N_H(v)$ of v in graph H satisfies $|N_H(v)| \leq 8$ and for any edge $uv \in P_\infty \boxtimes P_\infty$ we have $|N_H(u) \cap N_H(v)| \geq 2$. As caterpillar G contains no triangles, for any two adjacent vertices x, y in G we have $N_G(x) \cap N_G(y) = \emptyset$. Hence

$$\sum_{v \in V(Q)} \deg(v) \leq 8|V(Q)| - 2|E(Q)| = 6|V(Q)| + 2.$$

(3) $\Longrightarrow$ (2): Assume that $G = (V, E)$ is a caterpillar, say with spine $\langle v_1, \ldots, v_k \rangle$. The vertices of $P_\infty \boxtimes P_\infty$ naturally correspond to $\mathbb{N} \times \mathbb{N}$, where (p_i, p'_j) is mapped to (i, j). We embed the spine of G along the main diagonal, i.e., place v_i at (i, i) for $i = 1, \ldots, k$. Then, we proceed along the spine from v_1 to v_k, always placing the next leg at v_i at the positions (x, y) adjacent to (i, i) with $x + y$ as small as possible. Let us say that v_i is *free* if two leaves at v_i are embedded successfully at $(i - 1, i)$ and $(i, i - 1)$, respectively. In particular, the first vertex with degree at least 4 is always free. (We assume that there exists such a vertex, otherwise G clearly can be embedded.)

Assume that this embedding procedure fails to find a suitable position for a leaf at v_j for some $j \in [k]$. Let $i \leq j$ be the largest index such that v_i is free, and $Q = \langle v_i, \ldots, v_j \rangle$ be the subpath of the spine of G from v_i to v_j. Observe that $\deg_G(v_i), \deg_G(v_j) \geq 4$ and further that for $A = \{(i, i), \ldots, (j, j)\}$, there is a vertex in $V(Q) \cup N_G(Q)$ on each of the $5(j - i) + 9$ points in $A \cup N(A)$ in $P_\infty \boxtimes P_\infty$. With $|V(Q) \cup N_G(Q)| = \sum_{v \in V(Q)} \deg(v) - (j - i) + 1$, it follows that

$$\begin{aligned} & |V(Q) \cup N_G(Q)| > |A \cup N(A)| \\ \Leftrightarrow & \sum_{v \in V(Q)} \deg(v) - (j - i) + 1 > 5(j - i) + 9 \\ \Leftrightarrow & \sum_{v \in V(Q)} \deg(v) > 6(j - i) + 8 = 6(j - i + 1) + 2 = 6|V(Q)| + 2, \end{aligned}$$

which implies that G does not satisfy (3).

(2) $\Longrightarrow$ (1): This is immediate, as (2) is a restriction of (1). ◀

▶ **Corollary 5.** KingGraphEmbedding *can be solved in linear time for n-vertex caterpillars.*

Proof. Let G be a caterpillar with spine $\langle v_1, \ldots, v_k \rangle$, $k \leq n$. Using (3) in Theorem 4, G admits *no* embedding into $P_\infty \boxtimes P_\infty$ if and only if the sequence $(\deg(v_i) - 6)_{i \in [k]}$ has a contiguous subsequence whose sum is at least 3. Finding such a subsequence is the MaximumSubarray problem and can be solved in time $O(n)$ [16]. An embedding itself, rather than the merely the fact that the embedding exists, can be found in time $O(n)$ by following the proof strategy for (3) $\Longrightarrow$ (2), above. ◀

3 Row pathwidth

Now we consider row pathwidth. We show that testing whether the row pathwidth of a graph is 1 is NP-complete. This is the same as asking whether the given graph G is a subgraph of $C_\infty \boxtimes P_\infty$. We also consider the related problem of embedding in $C_\infty \,\square\, P_\infty$. Both problems are easily shown NP-complete using another observation concerning how graph transformations affect embeddability.

▶ **Observation 6.** *Let G be a simple graph, and for $k \in \{4, 6\}$ let G'_k be the result of adding (at any original vertex v of G) $\max\{0, k - \deg(v)\}$ leaves that are adjacent to v. Then*

- *$G \subset P_\infty \,\square\, P_\infty$ if and only if $G'_4 \subset C_\infty \,\square\, P_\infty$.*
- *$G \subset P_\infty \boxtimes P_\infty$ if and only if $G'_6 \subset C_\infty \boxtimes P_\infty$.*

Proof. The forward direction is obvious: If G is such a subgraph, then take the embedding of G in the grid, and use the P-extensions of k legs at each spine-vertex of C_∞ to place the added leaves at each vertex v.

For the other direction, observe that all vertices on P-extensions of legs of C_∞ have degree at most 3 in $C_\infty \,\square\, P_\infty$, and degree at most 5 in $C_\infty \boxtimes P_\infty$. We constructed G'_k (for $k \in \{4, 6\}$) such that the vertices of G have degree k, so they must be placed on the P-extension of a spine-vertex. If we set π to be the spine of C_∞, therefore G is embedded in $\pi \,\square\, P_\infty$ (respectively $\pi \boxtimes P_\infty$) as desired. ◀

▶ **Theorem 7.** *It is NP-complete to test whether a tree is a subgraph of $C_\infty \boxtimes P_\infty$. It is also NP-complete to test whether a tree is a subgraph of $C_\infty \,\square\, P_\infty$. Both results hold even for trees with constant maximum degree and pathwidth.*

Proof. By Corollary 3, testing whether $G \subset P_\infty \boxtimes P_\infty$ is NP-complete, even for a tree with constant pathwidth. Convert G into G' using Observation 6 with $k = 6$. This preserves a tree, increases the size only by a constant factor, increases the pathwidth by at most 1, and the maximum degree is 8. Also $G \subset P_\infty \boxtimes P_\infty$ if and only if $G' \subset C_\infty \boxtimes P_\infty$, which proves the first claim. The second claim is similar, using Theorem 1 and Observation 6 with $k = 4$. ◀

▶ **Corollary 8.** RowPathwidth *is NP-complete, even for trees of bounded degree and pathwidth, and even if we only want to know whether the row pathwidth is 1.*

4 Row treewidth

We now prove that computing row treewidth NP-hard. More precisely, it is NP-complete to test whether the row treewidth of a graph is 1. This is equivalent to testing whether the given graph G is a subgraph of $T \boxtimes P_\infty$ for some tree T. We show that this is NP-complete even when G is a series-parallel graph. We remind the reader of the following equivalent definitions of series-parallel graphs: (a) Connected graphs without a K_4-minor, (b) connected graphs of treewidth at most 2, (c) graphs obtained from an edge by attaching leaves and duplicating or subdividing edges, (d) connected graphs for which all 3-connected components contain at most three vertices.

▶ **Theorem 9.** *It is NP-complete to test whether a graph G is a subgraph of $T \boxtimes P_\infty$ for some tree T, even for a series-parallel graph G.*

The reduction is from NAE-3SAT. The problem NAE-3SAT asks for a given collection of clauses, each consisting of three literals (positive or negated Boolean variables), whether there exists a truth assignment such that each clause has at least one true and at least one false value, i.e., there are Not All Equal. The NAE-3SAT problem is known to be NP-complete [20]. Our reduction from NAE-3SAT here uses the *logic engine* by Eades and Whitesides [12], which maps instances of NAE-3SAT to geometric gadgets called *frames*, *armatures* and *flags*, in such a way that the entire construct has an embedding in the plane if and only if the NAE-3SAT instance has a solution. The frame here has only one embedding (up to symmetry), while the armatures have some flexibility (and express whether a variable is true or false), and the flags are used to enforce that every clause has at least one true and at least one false variable. In our case we ask for a given graph G for an embedding of G in another graph (namely $T \boxtimes P_\infty$), and therefore the gadgets are small subgraphs.

We first show NP-completeness of a closely related problem. Assume that with a graph G, we are also given labels 'hor' and 'ver' on some of its edges. We say that an embedding of G in $T \boxtimes P_\infty$ is *orientation-constrained* if the edges marked 'hor/ver' are horizontal and vertical, respectively. (Recall that horizontal/vertical means that the two endpoints of the edge have the same T-projection/P-projection.)

▶ **Lemma 10.** *Consider the following problem: 'Given a graph G with labels hor/ver on some edges, does it have an orientation-constrained embedding in $T \boxtimes P_\infty$ for some tree T?' This is NP-complete, even for a series-parallel bipartite graph G.*

Proof. Let I be an instance of NAE-3SAT with n variables and m clauses. We construct G and at the same time discuss possible orientation-constrained embeddings of G in the grid, i.e., in $P_\infty \boxtimes P_\infty$, see also Figure 4. (Since we restrict *all* edges to be horizontal or vertical, it does not matter whether the grid includes the diagonals or not.) Start with the *frame* (orange squares in the figure) which consists of three paths connecting two vertices t, b; the *middle path* has $H := m + 2n + 1$ vertical edges, while the two *outer paths* have $2n$ horizontal, H vertical, and then another $2n$ horizontal edges each. An orientation-constrained embedding of the frame in the grid is unique up to symmetry. The middle m rows of this embedding are called the *clause-rows* and marked with one clause each.

Next we add the *armature of* x_i (light/dark cyan circles) for each variable x_i. This consists of two paths that attach at the vertices of the middle path at distance i from t and b. Each path consist of $2n+1-2i$ horizontal edges at both ends with $H-2i$ vertical edges inbetween. The paths are assigned to literals x_i and $\overline{x_i}$. An orientation-constrained embedding of frames and armatures in the grid is unique up to symmetry and up to horizontally flipping each armature; in particular the row of each clause is unchanged over all such embeddings.

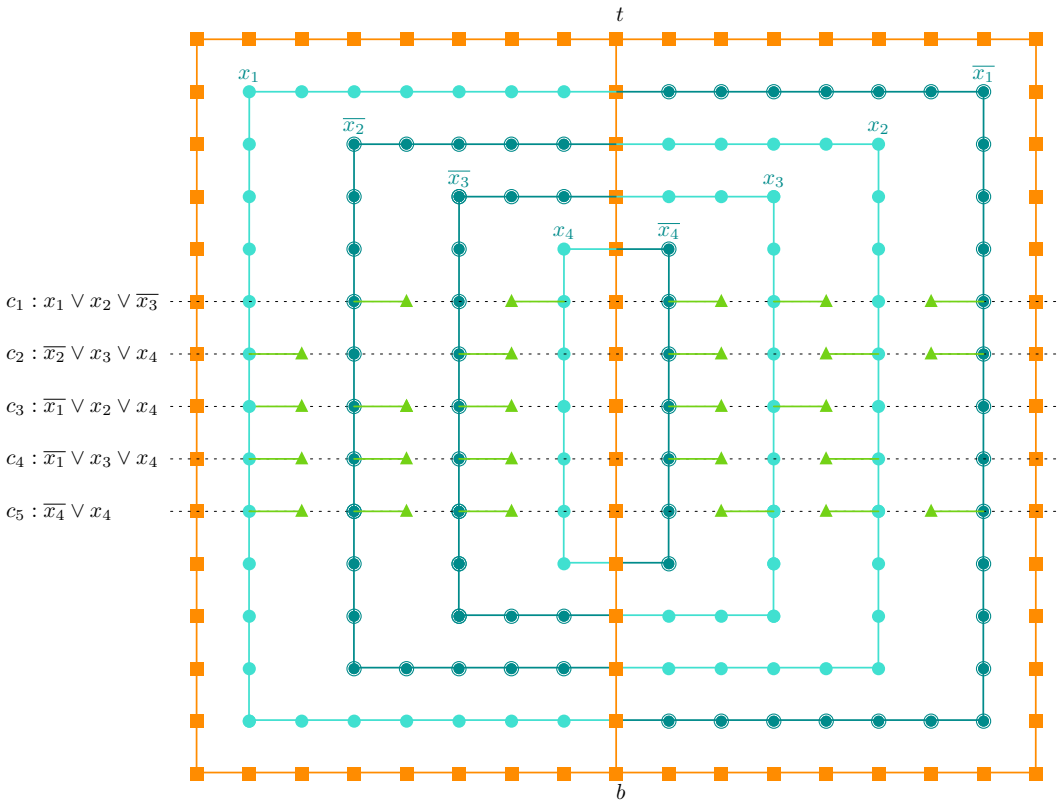

Figure 4 The reduction for row treewidth if we can fix the orientation of edges.

Finally we attach *flags* (green triangles) at the intersections of armatures and clause-rows. Namely, at the vertex where the armature of literal ℓ_i intersects the row of c_j, we attach a leaf (via a horizontal edge) if and only if ℓ_i does *not* occur in c_j. For each flag we have the choice of whether to place it to the right or to the left of its attachment vertex, as long as this spot has not been used by a different flag already. Graph G is clearly series-parallel, because we can reduce it to an edge by deleting leaves and multiple edges and contracting degree-2 vertices. Moreover, observe that G has polynomial size in terms of the NAE-3SAT instance I.

If I has a solution, then flip the armatures such that the left paths correspond to the literals of the solution. For each clause c_j there exists at least one true literal, hence there are at most $n-1$ flags in the row of c_j and left of the middle path; we can arrange them as to fit within the gaps. There also exists at least one false literal, hence at most $n-1$ flags in the row of c_j to the right of the middle path. So we can find an orientation-constrained embedding of G in the grid. Vice versa, if we have such an embedding, then taking the literals that are left of the middle path gives a solution to I because for each clause c_j there must be at most $n-1$ flags on each side of the middle path, so at least one literal is true and at least one literal is false.

So I has a solution if and only if G has an orientation-constrained embedding in the grid. To finish the NP-completeness, we must argue that any orientation-constrained embedding of G in $T \boxtimes P_\infty$ for some tree T actually must reside within a grid. To see this, let π be the path in T that corresponds to the T-projection of one outer path of the frame. Since the edge-orientations on the outer path are fixed, π has length H and connects the T-projections

t', b' of t and b, so t', b' have distance H in T. We claim that the embedding of G actually resides within $\pi \boxtimes P_\infty$, i.e., for any vertex v of G the T-projection v' of v is on π. To show this, observe that we can find a path from t to v by walking through the frame, then (perhaps) an armature and then (perhaps) along a flag, and always only go downward. Similarly find a path from v to b that only goes downward. The combined walk σ_v from t to b via v uses exactly H non-horizontal edges. The T-projection σ'_v of σ_v connects t' to b' and has length H, which by uniqueness of paths in trees implies that $\sigma'_v = \pi$ contains v'. ◀

To prove Theorem 9, we take the construction of Lemma 10, but add more vertices and edges to obtain a graph G for which edge-orientations are forced in any embedding of G in $T \boxtimes P_\infty$.

Proof of Theorem 9. Assume that we are given an instance I of NAE-3SAT. We may assume that one clause of I is $x_n \vee \overline{x_n}$, for if there is no such clause, then we can add it without affecting the solvability of I. Now let G_0 be the graph constructed for instance I as in the proof of Lemma 10. As before, G_0 has a unique orientation-constrained embedded in the grid up to horizontal flipping of armatures and flags, so the y-coordinates of vertices are fixed. We call the vertices and edges of G_0 *original*.

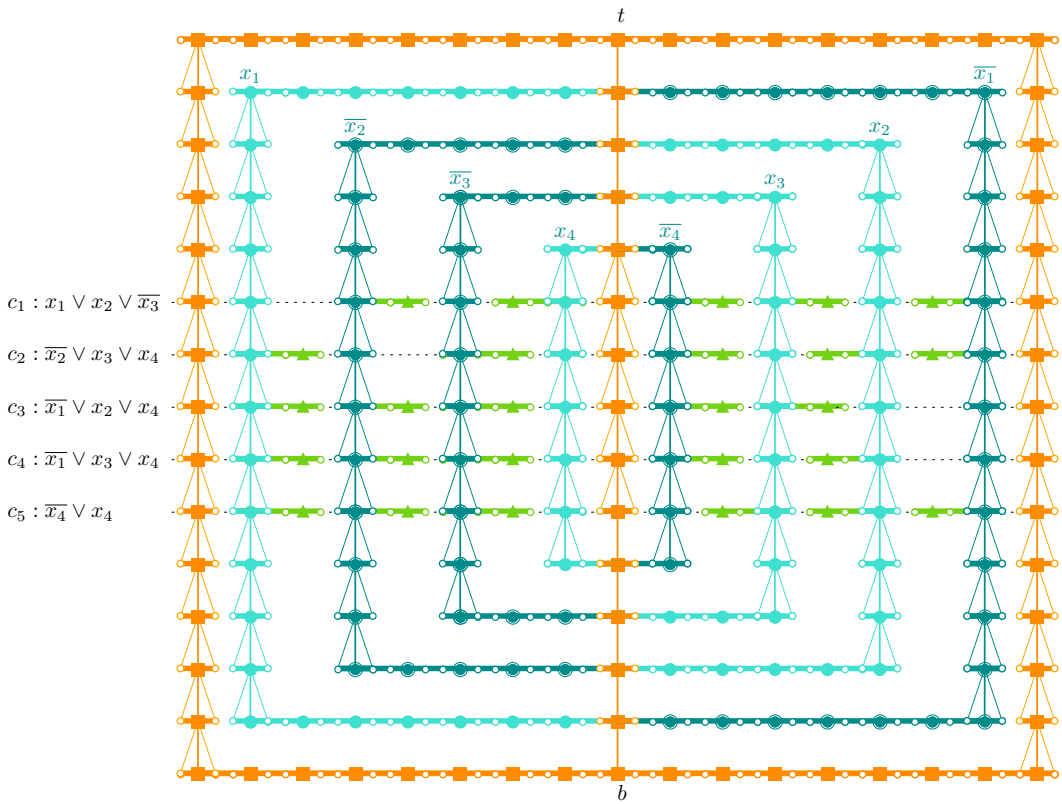

Figure 5 The reduction for row-treewidth. Bold edges indicate an attached $K_{2,5}$. Original vertices are solid.

As our next step, we "triple the width". Roughly speaking, we insert a new column before and after every column that we had in the drawing of G_0. Formally (and explained on the graph, rather than the drawing), subdivide every horizontal edge twice, and at any vertex v incident to k horizontal edges, attach $2 - k$ leaves. All new edges are again required to

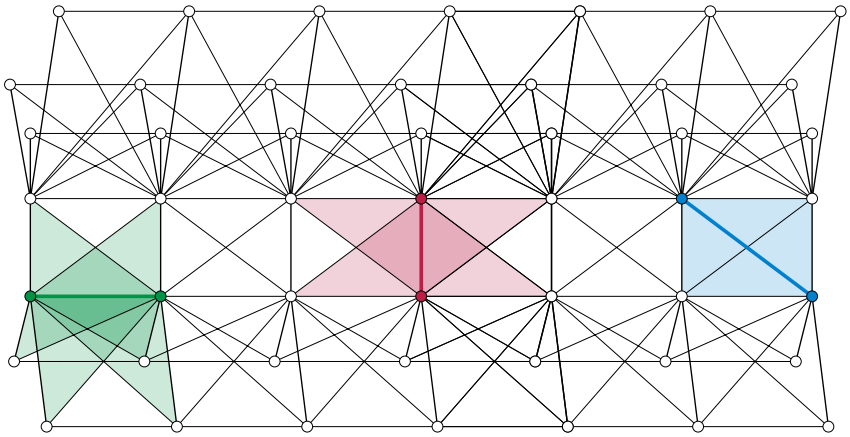

Figure 6 Illustration for Observation 11. In the strong product of a tree and a path, the two endpoints of a horizontal edge (left) can have five or more common neighbors, but the endpoints of a vertical edge (center) can have at most four common neighbors, and the endpoints of a diagonal edge (right) can have at most two common neighbors.

be horizontal. See Figure 5, ignoring bold lines and diagonal edges for now. The resulting graph G_1 likewise has an orientation-constrained embedding in the grid if and only if the NAE-3SAT instance has a solution. It also clearly is series-parallel since it is obtained from G_0 by subdividing edges and attaching leaves. Also, G_1 is only a constant factor larger than G_0.

Next we obtain G_2 by *adding an arrow-head* at some vertical edges vw. Assuming v is below w, this means adding the edges (v_ℓ, w) and (v_r, w), where v_ℓ, v_r are the two neighbors of v adjacent to it via horizontal edges. We add arrow-heads at all vertical edges of G_1 except at those vertical edges where the lower endpoint is incident to an armature or outer path. The graph stays series-parallel since each arrow-head $\{v_\ell, v, v_r, w\}$ contains a cutting pair that separates it (and perhaps an attached path of a flag) from the rest of the graph, so adding the edges of the arrow-heads does not affect whether there are non-trivial 3-connected components. Again, G_2 is only a constant factor larger than G_1.

For the final modification we need a simple but crucial observation, which one proves by inspecting the neighborhood of two adjacent vertices in $T \boxtimes P_\infty$ for all possible orientations of the edge between them (Figure 6).

▶ **Observation 11.** *Let G be a graph embedded in $T \boxtimes P_\infty$ for some tree T. If uv is an edge of G for which u, v have at least five common neighbors, then uv must be horizontal.*

Thus, we turn G_2 into G by *adding a $K_{2,5}$* at every 'hor' edge uv of G_2, i.e., adding five new vertices that are adjacent to both u and v. This keeps the graph series-parallel and force uv to be horizontal in any embedding of G in $T \boxtimes P_\infty$. To avoid clutter we do not show $K_{2,5}$ in Figure 5, but indicate it with a bold edge.

This ends the description of our construction. Observe that the size of the final graph G is still polynomial in terms of the original NAE-3SAT instance I. It should be straightforward to see that a solution to the NAE-3SAT instance I implies that G can be embedded in $C_\infty \boxtimes P_\infty$. Namely, we can embed G_0 in $\pi \boxtimes P_\infty$ where π is the spine of C_∞, subdivide each edge of P_∞ twice to embed G_1, realize the arrow-heads along diagonals, and finally use 5 legs at each vertex of π to embed the attached $K_{2,5}$'s on their P-extensions. Vice versa, assume that G is embedded in $T \boxtimes P_\infty$ for some tree T. We know that all bold edges must be

horizontal. We also claim that if vw was a 'ver' edge of G_1 that received an arrow-head, then the orientation of vw in the embedding is vertical. To see this, assume that the arrow-head was $\{v_\ell, v, v_r, w\}$, with v_ℓ, v_r connected to v via horizontal edges. Then vw belongs to two triangles $\{v_\ell, v, w\}$ and $\{v_r, v, w\}$, and the two horizontal edges (v_ℓ, v) and (v_r, v) of these triangles share endpoint v. No two such triangles exist at a diagonal edge, and vw cannot be horizontal since the two horizontal edges at v are vv_ℓ and vv_r. So vw is vertical.

We claim that the embedding of G_2 in $T \boxtimes P_\infty$ actually resides within $\pi \boxtimes P_\infty$ for some path π, i.e., in a grid. This is argued almost exactly as in Lemma 10. Let π be the T-projection of one outer path of G_0; since the orientations of the edges on the outer path is fixed π has length H. For any vertex v of G_2, we can find a walk σ_v from t to b via v that uses exactly H non-horizontal edges (they may now be diagonal). As before this implies that the T-projection of v is also in π, so the embedding of G_2 is within $\pi \boxtimes P_\infty$, i.e., the king's graph. As before, we can hence associate vertices of G_2 with points in $\mathbb{N} \times \mathbb{N}$, and speak of rows and columns of this embedding.

Since the orientations of edges on the outer paths are fixed, the drawing of the outer paths is fixed up to symmetry and spans $12n + 3$ columns (including the space for the arrowheads). The rest of G_2 must lie inside the outer-paths, so in particular the row of clause $x_n \vee \overline{x_n}$ (which we call the *spacer-row*) has $12n + 3$ points that could host vertices. But there are three paths of the frame, $2n$ armature-paths and $2(n-1)$ flags in this row, meaning $4n + 1$ original vertices use the spacer-row. Since we tripled the width, all $12n + 3$ possible points in the spacer-row are used in the embedding of G_2. Furthermore, the vertices in the spacer-row come as triplets connected by horizontal edges, with the middle vertex the original vertex. Up to a translation therefore all original vertices in the space-row have x-coordinate divisible by 3. This forces any original vertex w to have x-coordinate divisible by 3 as well, because we can get from w to an original vertex v in the spacer-row using only edges that must be vertical (due to an arrow-head) or horizontal (due to a $K_{2,5}$), and the horizontal parts have length divisible by 3. In consequence, all edges on the middle path of the frame must be vertical, even those that do not have an arrowhead on them. (This argument would be simplified if we added arrow-heads everywhere, but then the graph would not be series-parallel.) With this, the embedding of G implies an embedding of G_1 in $\pi \boxtimes P_\infty$ that is orientation-constrained, and we can hence extract a solution to the NAE-3SAT instance as Lemma 10. This finishes the proof. ◀

The graph in our construction in the above proof of Theorem 9 has maximum degree 16 and pathwidth $O(1)$, so computing the row treewidth remains NP-hard even under these restrictions.

▶ **Corollary 12.** RowTreewidth *is NP-complete, even for series-parallel graphs of bounded degree and pathwidth, even if we only want to know whether the row treewidth is 1.*

Two variations: We now give two variations of this NP-hardness proof. First, an almost identical construction shows that testing whether $G \subset T \square P_\infty$ for some tree T is also NP-complete. Namely, use the same construction (G_0 to G' to G), except omit the diagonal edges and replace '$K_{2,5}$' by 'three paths of length 2'. This forces all horizontal edges to have the desired orientation in any embedding of G in $T \square P_\infty$. We can argue as above that in this case G lies within $\pi \square P_\infty$ for a path π. Therefore any vertical edge uv must have the desired orientation, because both u and v have two incident horizontal edges. So this gives an embedding of G_0 in the grid that respects the given orientation, hence a solution to NAE-3SAT.

Second, as mentioned earlier we can modify the reduction by Gupta et al. [17] to show NP-completeness of KingGraphEmbedding even for a tree of pathwidth 2. The reduction from [17] is again from NAE-3SAT, illustrated in Figure 7, and again uses the logic engine. The *frame* (orange squares) consists of a *main path* of length $2n + 8$, with two pairs of paths of length $2m + 1$ attached at the fourth and the fourth-to-last vertex; additionally attach degree-1 vertices at these paths in such a way that makes the embedding unique up to rotation and renaming of vertices. Without loss of generality we may assume that the main path is hence drawn along the x-axis, and we reserve the rows with y-coordinates $2j - 1$ and $-(2j - 1)$ for clause c_j for $j = 1, \ldots, m$. For each variable x_i, define the *armature* (light/dark cyan circles) to be a pair of paths of length $2m + 1$, attached at the $2i$th vertex of the middle part of the main path; the paths are assigned to literals x_i and $\overline{x_i}$. The armatures have an attached degree-1 vertex at every non-clause row and an attached degree-1 vertex (a *flag*, a green triangle) in a clause row it and only if the literal does not occur in the clause. One can argue that this tree (which has pathwidth 2) can be embedded in a grid if and only if the NAE-3SAT instance has a solution, see [17].

To convert this into an instance of KingGraphEmbedding, we convert some of the non-leaf vertices to *high-degree vertices* by adding degree-1 neighbors. Specifically, we first add degree-1 vertices to the spine such that for each subpath Q of the paths the degree-sum is $6|V(Q)| + 2|$. Since these paths plus their neighbors form caterpillars, one can therefore show (using the techniques from Theorem 4) that the frame has a unique embedding (up to rotation and renaming of vertices). Additionally we add degree-1 vertices at every vertex of the armatures that is not incident to a flag, see also Figure 7. The choice of the degrees at high-degree vertices is such that a solution to NAE-3SAT gives a king's graph embedding of the resulting tree of pathwidth 2, and no grid-point adjacent to a high-degree vertex remains free. Vice versa, since the frame has a unique embedding, one can argue that flags and must be in the clause-rows since otherwise the high-degree vertices could not use disjoint sets of grid-points for their neighbors. Since the armatures also use at least one vertex per variable in each clause-row, we can therefore read a solution to NAE-3SAT from the positioning of the two paths of each armature, and this is a solution to due the flags. Since our tree was obtained from a tree of pathwidth 2 by adding degree-1 vertices at non-leaves, KingGraphEmbedding is NP-complete even for trees of pathwidth 2.

5 Inapproximability

The best approximation ratio known for a polynomial-time approximation algorithm for the treewidth is $O(\sqrt{\log w})$, where w is the treewidth [13]. No polynomial-time constant-factor approximation algorithms are known, and they are unlikely to exist, since such algorithms would violate a standard assumption in computational complexity theory, the small set expansion conjecture [23]. As we now show, the same hardness results apply to the approximation of row treewidth and row pathwidth:

▶ **Theorem 13.** *If there exists an approximation algorithm for row treewidth, row pathwidth, layered treewidth, or layered pathwidth with approximation ratio ρ, then there exists an approximation algorithm for treewidth or pathwidth (respectively) with approximation ratio at most 3ρ. As a consequence, the small set expansion conjecture implies that ρ cannot be $O(1)$.*

Proof. Let G be a graph for which we wish to approximate the treewidth or pathwidth, let w be its treewidth or pathwidth, and form graph G^+ with treewidth or pathwidth $w + 1$ by adding a universal vertex to G. (A universal vertex is one that is connected by an edge to all

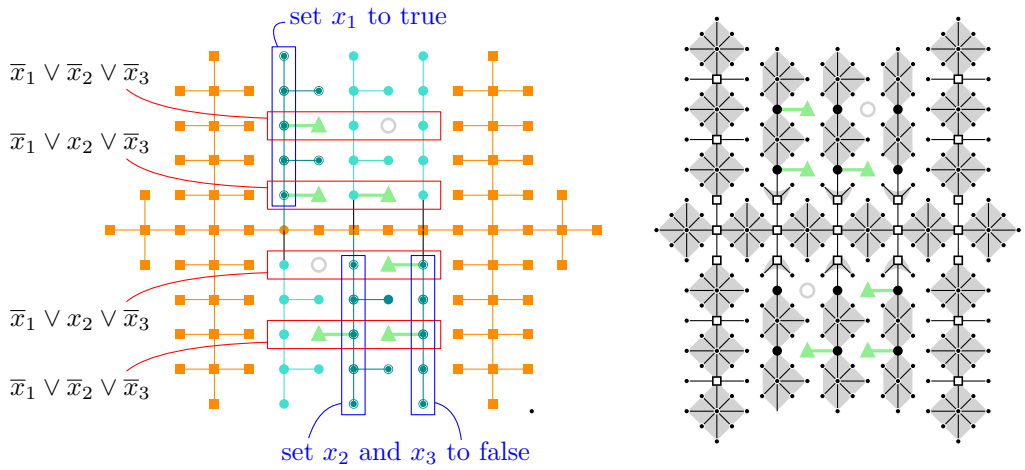

Figure 7 Left: The tree of pathwidth 2 of Gupta et al. [17] for the NAE-SAT instance $\varphi = (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$ in its GridEmbedding for solution $\{x_1 = \text{true}, x_2 = \text{false}, x_3 = \text{false}\}$. Right: A corresponding tree of pathwidth 2 in its corresponding king's graph embedding. ($P_\infty \boxtimes P_\infty$ is rotated 45°.) High-degree vertices and their neighborhoods are shaded.

other vertices of G.) The universal vertex forces every layering of G^+ to use at most three layers. Graph G^+ has a trivial layering with one layer and row treewidth or row pathwidth $w + 1$. Any other layering has row treewidth, row pathwidth, layered treewidth, or layered pathwidth at least $(w + 1)/3$, because it gives a tree decomposition for G^+ with bags that are the unions of bags in three layers. Therefore, any approximation for the row treewidth, row pathwidth, layered treewidth, or layered pathwidth of G^+ gives an approximation for the treewidth or pathwidth of G^+, and therefore of G, with approximation ratio increased by at most a factor of three. ◀

Note that the constructed graph G^+ is not necessarily planar. In fact, for planar graphs there are $O(1)$-approximation algorithms for the treewidth [18].

6 Row treedepth

Recall that S_∞ (the infinite *star*) is the tree that consists of one *center* that is adjacent to all other vertices (the *leaves*), with no restriction on its number of leaves.

▶ **Theorem 14.** *It is NP-complete to test whether a tree is a subgraph of $S_\infty \boxtimes P_\infty$, even for a tree of pathwidth 2.*

Proof. We use a reduction from 3-Partition. The input is a multi-set $A = \{a_1, \ldots, a_{3n}\}$ of positive integers. The goal is to split these $3n$ integers into n groups of three that all sum to the same integer $B = \frac{1}{n}\sum_{i=1}^{3n} a_i$. This problem is strongly NP-complete [14]: it remains NP-complete even if A is encoded in unary. We may assume that all input numbers are multiples of 8 (otherwise multiply them all by 8; this does not affect NP-completeness). We describe the construction of our tree T and at the same time argue what any embedding Γ of T in $S_\infty \boxtimes P_\infty$ must look like. In $S_\infty \boxtimes P_\infty$, we call the P-extension of the center c the *center row*; as in Observation 6 we use a degree argument to force many vertices of T to be in the center row. Finding enough space in this row to hold all of them is the crucial idea for our reduction.

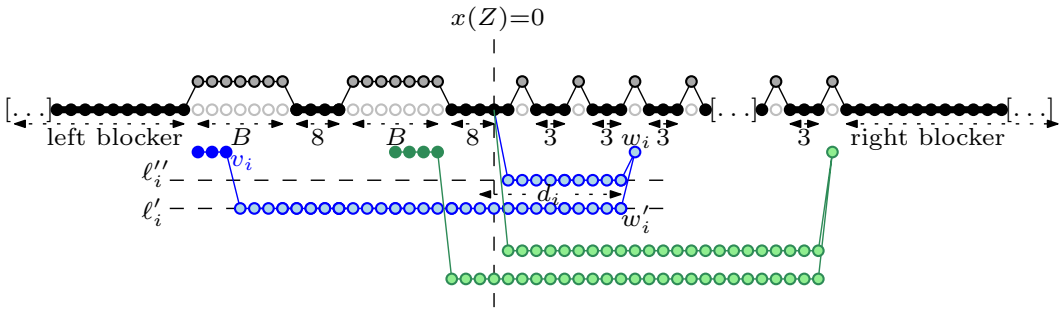

Figure 8 NP-completeness of embedding in $S_\infty \boxtimes P_\infty$. The figure is not to scale. Filled dots represent c-vertices (hence have six leaves attached). We only show two paddles, one green and one blue.

Tree T consists of a *frame* as well as a *paddle* for each a_i, $i = 1, \ldots, 3n$, see also Figure 8. The frame is a very long path, with most vertices on the path having six leaves attached. (These leaves are not shown in our picture.) The vertices with attached leaves are called *c-vertices* and are forced to be on the central row since all other vertices of $S_\infty \boxtimes P_\infty$ have degree at most 5. All other vertices of the frame are called *ℓ-vertices* because they could be on a *leaf-row* (the P-extension of a leaf of S_∞). The specific spacing along the path is as follows:

- Begin with $n(B+8)$ c-vertices (the *left blocker*). Since c-vertices must be on the central row, and no two central-row vertices are adjacent unless they are consecutive, this path (and similarly any path of c-vertices used below) occupies a consecutive set of vertices on the central row.
- Continue with B ℓ-vertices, followed by 8 c-vertices. The ℓ-vertices could be on a leaf-row, hence keep up to B vertices of the central row unused. We call this a *group-gap*.
- We create n consecutive group-gaps (we have $n = 2$ in Figure 8).
- The last vertex Z of the last group-gap is called the *anchor*; the paddles (defined below) will attach at Z.
- Starting at Z, we alternate between three c-vertices and one ℓ-vertex that together define one *fold-gap* (it permits to omit one vertex from the center row). There are $\frac{1}{8}n(B+8)$ fold-gaps.
- Finally we finish with $n(B+8)$ c-vertices (the *right blocker*).

Note that the left and right blocker are so long that no sub-path of ℓ-vertices could extend beyond them; in particular this forces all c-vertices that are not in the blockers to be between them in the central row.

Now for each $a_i \in A$, we define the *a_i-paddle*. This starts at anchor Z, continues with a path (the *handle*) that has $n(B+8)-1$ ℓ-vertices, and culminates at the *blade*, which consists of a_i c-vertices. The handle is not long enough to extend beyond the blockers, so the c-vertices of the blade must be at $a_i \geq 2$ consecutive central-row vertices between the blockers. Since each fold-gap leaves at most one central-row vertex free, the blade must hence occupy central-row vertices left free by a group-gap. There are at most nB such central-row vertices in Γ, and they come in blocks of at most B consecutive central-row vertices each. By $\sum_{i=1}^{3n} a_i = nB$, it follows that in any realization Γ the group-gaps leave exactly n blocks of exactly B central-row vertices each, and the blades exactly fill these gaps, hence giving the desired partition of A.

The resulting tree has $O(nB)$ vertices in the frame: $O(nB)$ each for the left and right blockers, $O(B)$ each for n group-gaps, and $O(1)$ each for $O(nB)$ fold-gaps. Each of the n paddles also has $O(nB)$ vertices. Therefore, the whole tree has $O(n^2B)$ vertices, polynomial in the (unary) representation of the given 3-Partition instance, and this reduction can easily be carried out in polynomial time.

We must still argue that if there is a solution to 3-Partition, then we can embed T in $S_\infty \boxtimes P_\infty$, and for this, need the fold-gaps and $\Theta(n)$ leaves at the star. Embed first the frame as in Figure 8, so all gaps leave the maximal possible number of central-row vertices free. (We also use 6 leaf-rows, not shown here, to embed the leaves attached at c-vertices.) We treat the center row as if it were the x-axis with Z at the origin; this defines an x-coordinate $x(\cdot)$ for all embedded vertices with $x(Z) = 0$. Embed the blades of $a_1, \ldots, a_{3n}$ in the group-gaps according to the solution to 3-Partition. For $i = 1, \ldots, 3n$, let v_i be the rightmost central-row vertex of the blade of a_i. To place the handle, we use two further leaf-rows, say ℓ'_i and ℓ''_i. We go from v_i diagonally rightward to ℓ'_i, then rightward for $|x(v_i)| - 1$ edges to reach x-coordinate -1. Hence we could now go to the anchor diagonally, but the handle is longer than this. Therefore we continue rightward for another $d_i := \frac{1}{2}(n(B+8) - |x(v_i)|)$ edges along ℓ'_i. Recall that each a_i (and hence also B) is divisible by 8. Since there are 8 c-vertices at each group-gap, and all group-gaps are completely filled by paddles, x-coordinate $x(v_i)$ is also divisible by 8. Thus d_i is divisible by 4, and the vertex w'_i that we reach is one unit left of the central-row vertex w_i of some fold-gap. Go diagonally from w'_i to w_i, and from there diagonally back to $x(w'_i)$ on the other leaf-row ℓ''_i. Then we go leftward along leaf-row ℓ''_i to x-coordinate 1 and then diagonally to Z. In total we have used $|x(v_i)| - 1 + 2d_i = n(B+8) - 1$ vertices, which is exactly the length of the handle. Observe that vertex w_i cannot have been used by a different paddle (say the a_j-paddle) because $v_j \neq v_i$ are distinct central-row vertices, and their x-coordinates determine the fold-gap to be used.

Thus a solution to 3-Partition gives an embedding of G in $S_\infty \boxtimes P_\infty$ and vice versa and the problem is NP-complete. Clearly we constructed a tree T; and if we removed the path π that defined the frame then all components of $T \setminus \pi$ are either singleton-vertices (at c-vertices of the frame) or caterpillars (at the paddles). Therefore T has pathwidth 2. ◀

The same result also holds for the problem of embedding in $S_\infty \square P_\infty$. We use almost exactly the same tree T, except at each gap of the frame the path of ℓ-vertices is longer by two vertices and the handles have four more vertices. Details are left to the reader.

Our constructed trees have pathwidth 2. For a tree T of pathwidth 1, the answer to 'is $T \subset S_\infty \boxtimes P_\infty$' is trivial because the answer is always 'Yes': Such a tree is a subgraph of C_∞, and C_∞ can be embedded in $S_\infty \boxtimes P_\infty$ by placing the spine on the center row.

7 Outlook

In this paper, we proved that computing graph parameters such as the row pathwidth and row treewidth are NP-hard to compute, even under strong restrictions on the input graph. In fact, most of these restrictions rule out hopes for fixed-parameter tractability (or at least the possibility of finding polynomial-time algorithms in special situations). We do state here a few possibilities of situations where finding an embedding may be polynomial, but this mostly remains for future work:

- Given a graph with bounded radius, is it possible to solve RowTreewidth or Row-Pathwidth in polynomial time? In all our hardness constructions, the graph had radius $\Theta(n)$. Bounded radius forces any layering to use a bounded number of rows, so if the

row treewidth or row pathwidth is also bounded, then the treewidth or pathwidth of the original graph must also be bounded, but it is not obvious how to take advantage of this in an algorithm.
GridEmbedding *is* polynomial for graphs of bounded radius, because a graph can be embedded in a grid only if it has bounded maximum degree, and together with bounded radius this would imply bounded size, hence a constant-time algorithm.
- For the results for GridEmbedding ([4] and graph G_0 in the proof of Theorem 9) we very much needed the ability to change the embedding of the graph, so that we could flip armatures and flags. What is the status if the embedding is fixed? In particular, is testing whether a tree can be embedded in a grid NP-complete if the embedding of the tree is fixed, possibly similar to the results in [2, 1]?

One could also ask for results for planar graphs with a fixed embedding where faces are small (e.g., triangulations). In all our constructions, some faces have degree $\Theta(n)$. Can we solve any of the problems (but especially KingGraphEmbedding) for triangulated planar graphs? This remains open, but we can make some progress if additionally also the treewidth is small.

▶ **Theorem 15.** *Let G be a planar graph with treewidth t and a planar drawing Γ where all faces have degree at most Δ. Then we can test whether G can be embedded in the grid (in a way that respects embedding Γ) in time $O^*(n^{3(t+1)\Delta})$, i.e., in polynomial time if $t \cdot \Delta \in O(1)$.*

Proof. In 2013, the first author and Vatshelle [6] studied the PointSetEmbedding problem, where we are given a set of points S and a planar graph G, and we ask whether G has a planar straight-line drawing where all vertices are placed at points of S. They showed that if G has treewidth at most t and face-degree at most Δ, then PointSetEmbedding can be solved in $O^*(|S|^{1.5(t+1)\Delta})$ time. Their approach is to use a so-called carving decomposition of the dual graph, which results in a hierarchical decomposition of G into ever smaller subgraphs H (ending at one face) for which the *boundary* (the vertices of H that may have neighbors outside H) has small size. The main idea to solve PointSetEmbedding is then to do dynamic programming in this carving decomposition, and the parameter for the dynamic program is all possible embeddings of the boundary of H in the given point set S.

To adapt this algorithm to our situation, we need two changes. First, we fix the point set S to be the points of an $n \times n$-grid. (Clearly no bigger grid can be required.) In particular, we have $|S| = n^2$. Second, when considering possible embeddings of the boundary of H, we *only* consider such embeddings where this boundary is drawn along edges of the grid with diagonals. With this restriction, the same dynamic program will test whether a grid embedding exists in the desired time. ◀

Sadly this approach does not seem to generalize to our other embedding problems.

References

1. Hugo A. Akitaya, Maarten Löffler, and Irene Parada. How to fit a tree in a box. In *Proceedings of the 26th International Symposium on Graph Drawing and Network Visualization (GD 2018)*, LNCS, pages 361–367. Springer, Cham, 2018. doi:10.1007/978-3-030-04414-5_26.
2. Hugo A. Akitaya, Maarten Löffler, and Irene Parada. How to fit a tree in a box. *Graphs and Combinatorics*, 38(5):155, 2022. doi:10.1007/s00373-022-02558-z.
3. Michael J. Bannister, William E. Devanny, Vida Dujmović, David Eppstein, and David R. Wood. Track layouts, layered path decompositions, and leveled planarity. *Algorithmica*, 81(4):1561–1583, 2019. doi:10.1007/s00453-018-0487-5.

4 Sandeep Bhatt and Stavros Cosmadakis. The complexity of minimizing wire lengths in VLSI layouts. *Information Processing Letters*, 25(4):263–267, 1987. doi:10.1016/0020-0190(87)90173-6.

5 Therese Biedl, David Eppstein, and Torsten Ueckerdt. On the complexity of embedding in graph products. In *Proc. 35th Canadian Conference on Computational Geometry (CCCG '23)*, pages 77–88, 2023. URL: https://wadscccg2023.encs.concordia.ca/assets/pdf/CCCG_2023_proc.pdf.

6 Therese Biedl and Martin Vatshelle. The point-set embeddability problem for plane graphs. *Int. J. Comput. Geom. Appl.*, 23(4-5):357–396, 2013. doi:10.1142/S0218195913600091.

7 Prosenjit Bose, Vida Dujmović, Mehrnoosh Javarsineh, Pat Morin, and David R. Wood. Separating layered treewidth and row treewidth. *Discrete Mathematics & Theoretical Computer Science*, 24(1):P18:1–P18:10, 2022. doi:10.46298/dmtcs.7458.

8 Prosenjit Bose, Pat Morin, and Saeed Odak. An optimal algorithm for product structure in planar graphs. In Artur Czumaj and Qin Xin, editors, *18th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2022*, volume 227 of *LIPIcs*, pages 19:1–19:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.SWAT.2022.19.

9 Vida Dujmovic, Gwenaël Joret, Piotr Micek, Pat Morin, Torsten Ueckerdt, and David R. Wood. Planar graphs have bounded queue-number. *J. ACM*, 67(4):22:1–22:38, 2020. doi:10.1145/3385731.

10 Vida Dujmović, Pat Morin, and David R. Wood. Layered separators in minor-closed graph classes with applications. *J. Combinatorial Theory, Ser. B*, 127:111–147, 2017. doi:10.1016/j.jctb.2017.05.006.

11 Zdeněk Dvořák, Tony Huynh, Gwenael Joret, Chun-Hung Liu, and David R. Wood. Notes on graph product structure theory. In Jan de Gier, Cheryl E. Praeger, and Terence Tao, editors, *2019-20 MATRIX Annals*, pages 513–533. Springer International Publishing, Cham, 2021. doi:10.1007/978-3-030-62497-2_32.

12 Peter Eades and Sue Whitesides. The logic engine and the realization problem for nearest neighbor graphs. *Theoretical Computer Science*, 169(1):23–37, 1996. doi:10.1016/S0304-3975(97)84223-5.

13 Uriel Feige, Mohammadtaghi Hajiaghayi, and James R. Lee. Improved approximation algorithms for minimum weight vertex separators. *SIAM J. Comput.*, 38(2):629–657, 2008. doi:10.1137/05064299X.

14 M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

15 Angelo Gregori. Unit-length embedding of binary trees on a square grid. *Information Processing Letters*, 31(4):167–173, 1989. doi:10.1016/0020-0190(89)90118-X.

16 David Gries. A note on a standard strategy for developing loop invariants and loops. *Science of Computer Programming*, 2(3):207–214, 1982. doi:10.1016/0167-6423(83)90015-1.

17 Siddharth Gupta, Guy Sa'ar, and Meirav Zehavi. Grid recognition: Classical and parameterized computational perspectives. *J. Comput. Syst. Sci.*, 136:17–62, 2023. doi:10.1016/J.JCSS.2023.02.008.

18 Frank Kammer and Torsten Tholey. Approximate tree decompositions of planar graphs in linear time. *Theor. Comput. Sci.*, 645:60–90, 2016. doi:10.1016/j.tcs.2016.06.040.

19 Jirí Matousek and Robin Thomas. On the complexity of finding iso- and other morphisms for partial k-trees. *Discret. Math.*, 108(1-3):343–364, 1992. doi:10.1016/0012-365X(92)90687-B.

20 Thomas J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, STOC '78, page 216–226, New York, NY, USA, 1978. Association for Computing Machinery. doi:10.1145/800133.804350.

21 Farhad Shahrokhi. New representation results for planar graphs. In *Proc. 29th European Workshop on Computational Geometry (EuroCG '13)*, pages 177–180, 2013. arXiv:1502.06175.

22 Torsten Ueckerdt, David R. Wood, and Wendy Yi. An improved planar graph product structure theorem. *Electron. J. Comb.*, 29(2), 2022. doi:10.37236/10614.

23 Yu Wu, Per Austrin, Toniann Pitassi, and David Liu. Inapproximability of treewidth, one-shot pebbling, and related layout problems. *J. Artificial Intelligence Research*, 49:569–600, 2014. doi:10.1613/jair.4030.