Directed Graph Minors and Serial-Parallel Width

Argyrios Deligkas
Leverhulme Research Centre, University of Liverpool, UK
argyrios.deligkas@liverpool.ac.uk

Reshef Meir
Faculty of Industrial Engineering and Management, Technion, Israel
reshefm@ie.technion.ac.il

Abstract

Graph minors are a primary tool in understanding the structure of undirected graphs, with many conceptual and algorithmic implications. We propose new variants of *directed graph minors* and *directed graph embeddings*, by modifying familiar definitions. For the class of 2-terminal directed acyclic graphs (TDAGs) our two definitions coincide, and the class is closed under both operations. The usefulness of our directed minor operations is demonstrated by characterizing all TDAGs with serial-parallel width at most k; a class of networks known to guarantee bounded negative externality in nonatomic routing games. Our characterization implies that a TDAG has serial-parallel width of 1 if and only if it is a directed series-parallel graph. We also study the computational complexity of finding a directed minor and computing the serial-parallel width.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis, Mathematics of computing → Graph theory

Keywords and phrases directed minors, pathwidth

Digital Object Identifier 10.4230/LIPIcs.MFCS.2018.44

Related Version A full version of the paper is available at `https://tinyurl.com/y9hcukyz`.

1 Introduction

Graph theory has been one of the fundamental tools in computer science since its inception and in many computational problems the inputs are in a form of a graph, e.g., analysis of electric circuits and communication networks, and training of neural nets. More important still, numerous problems from various domains are often solved by reducing them to some algorithmic problem on a graph. Some prominent examples include search and path-finding [31]; planning graphs [3]; constraint satisfaction [26]; AND-OR graph [4]; and inference in Bayesian networks [7].

The *structure* of these graphs is often crucial to the modeling of the problem. For instance, the last two examples above use *directed acyclic graphs* (DAGs), which are also used to represent belief structures, influence relations and decision diagrams [15]. Restrictions on the degree, maximum length, or other properties of the underlying graph, can be exploited: problems that are not guaranteed to have a solution in general may behave better on some classes of graphs, and many algorithms are guaranteed to have a lower runtime subject to structural assumptions.

43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018).
Editors: Igor Potapov, Paul Spirakis, and James Worrell; Article No. 44; pp. 44:1–44:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Graph minors

When considering *undirected graphs*, some of the primary tools of structural analysis use *graph embeddings* and *graph minors.* These are substructures whose exclusion from a graph indicates certain "simplicity" properties. Some famous results are the characterization of planar graphs [23], and of graphs with bounded treewidth [32] by excluded minors. In fact, for undirected graphs there is by now a sound theory of graph minors with many applications; see, e.g., [25] for a survey, and [8] for algorithmic implications. The culmination of this theory is the Graph Minor Theorem [40, 33], which states that *any* class of undirected graphs that is closed under the minor operation, can be characterized by a finite set of excluded minors.

Perhaps the most important application of graph minor theory to computer science is its use for developing efficient algorithms on graphs with bounded treewidth and/or other properties [13, 6, 34]. Graph minors were also recently used to characterize classes of graphs induced by planning problems to identify potential effects of time-inconsistent planning [21, 38].

Although the graphs encountered in many theoretical and realistic problems are *directed*, there is no single theory of directed graph minors, and results are far more scarce than in the undirected case. Several papers suggested various definitions of directed minors, embeddings, and subdivisions, and provided various characterization results [17, 20, 14, 18, 19, 22]. However, each such definition uses different graph operations, some of which we explain in detail later on. Certain notions of directed minors are only applicable for subclasses of directed graphs. For example, the definitions in [22] apply only to minors with a certain structure called "crown".

In this paper, we will be interested mainly in directed graphs that are acyclic (DAG), or 2-terminal, or both (TDAG). 2-terminal graphs occur in routing [1], circuit analysis [36] and in many planning problems [21]. Thus understanding the structure of graphs in these classes is an important challenge.

Paper structure and contribution. In the first part of the paper (Section 3) we define new notions of graph embedding and graph minor for general directed graphs.[1] We show that for the class of 2-terminal directed acyclic graphs (TDAGs) these two operations exactly reverse one another. Thus, a TDAG G' is a directed minor of G if and only if it is embedded in G. Also, the class of TDAG is closed under directed minor and directed embedding operations. We thus argue that our definitions provide a sound basis for a theory of directed graph minors, at least for the class of TDAGs.

To demonstrate the usefulness of our directed minor theory, we apply it in Section 4 to characterize TDAGs with bounded *parallel width* and *serial-parallel width.* The parallel width of a graph corresponds to the maximal cut separating the source from the target. Serial-parallel width of a graph is a parameter recently introduced in the context of routing games [28], and it is useful for bounding negative externalities. We describe a finite set of graphs (generalized variants of the Braess/Wheatstone network) whose exclusion as directed minors of a TDAG G is necessary and sufficient to determine that G has serial-parallel width lower than k, for any k.

In Section 5 we settle several computational questions arising from our definitions. Some proofs are omitted due to space constraints and are available in the full version of this paper which is attached at the end of the file.

[1] To avoid confusion, we should note that the term *graph embedding* is used in the machine learning literature to describe embedding of graphs in various topological or metric spaces (e.g., [41]), which is a very different problem.

2 Preliminaries

For convenience, we will use the letter H for undirected graphs, and the letter G for directed graphs. We denote a path in graph $\langle V, E\rangle$ by $(v_1, v_2, \dots, v_m)$, where for every $i \leq m-1$, $(v_i, v_{i+1}) \in E$. We use dash to abbreviate the path, e.g. $a - b - c$ is an abbreviation to a path $(a, \dots, b, \dots, c)$; if more than one such path exists, we refer to one of them arbitrarily, unless stated otherwise.

If nodes x, y are on some path p, then p_{xy} denotes the *open* subpath of p between nodes x and y, and $[p_{xy}] = x - p_{xy} - y$ the *closed* subpath that includes the extreme vertices.

▶ **Definition 1** (2-terminal graph [29, 14]). A *2-terminal [directed] graph* $G = \langle V, E, s, t\rangle$ is a [directed] multigraph with no self-loops and two distinguished vertices $s, t \in V$, such that every vertex and edge belong to at least one [directed] simple $s - t$ path.

A *forward-subtree* of a directed 2-terminal graph G is a subset of edges that form a directed tree with a single source. Similarly, a *backward-subtree* of G is a subset of edges that form a directed tree with a single target.

A directed 2-terminal graph with no cycles is referred to as *TDAG* (2-Terminal Directed Acyclic Graph). The vertices of a TDAG can always be sorted in increasing order, called *topological order*, so that all edges, and thus all directed paths, are from v_i to v_j for some $j > i$. In particular, s and t are the first and last vertices, respectively.

▶ **Lemma 2.** *A DAG is a TDAG if and only if it has a unique source and a unique sink.*

3 Directed Graph Minors and Embeddings

In undirected graphs, a graph H' is called a *minor* of H if H' can be obtained from H by a sequence of edge deletions and contractions. As an example of a simple characterization via exclusion of minors, observe that any graph H (not a multigraph) is acyclic if and only if it excludes a triangle as a minor.

3.1 Directed minors.

There are several extensions of the notion of a minor to directed graphs. One that is closest to our needs is the *butterfly minor* [17], see Def. 3 without the underlined part. However, neither the class of 2-terminal graphs nor the class of TDAGs is closed under the butterfly minor operation, since, for example, it may leave an isolated node. We thus modify it by restricting which edges may be deleted (underlined).

▶ **Definition 3** (Directed minor). A graph G' is a *directed minor* (or simply *a d-minor*) of a directed graph G, if G' can be obtained from G by a sequence of the following local operations:

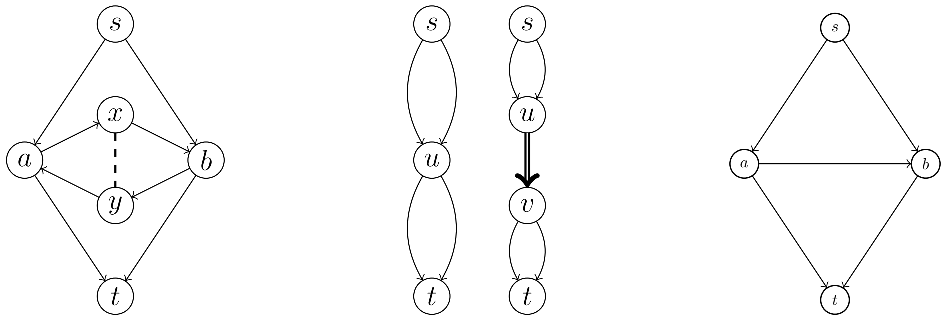
Deletion. Deleting an edge (a, b) where a has outdegree at least 2, and b has indegree at least 2.

Backward contraction. Contracting an edge (a, b) where b has indegree 1.

Forward contraction. Contracting an edge (a, b) where a has outdegree 1.

For example, the edge (a, b) in Fig. 1c may not be contracted, but can be backward-contracted after the edge (s, b) is deleted.

▶ **Lemma 4.** *The class of directed acyclic graphs is closed under d-minor operations.*



(a) **(b)** **(c)** The Braess graph.

Figure 1 The graph in Fig. 1a is a directed 2-terminal graph (solid edges only). Adding the dashed edge (x, y), regardless of its direction, results in a non-2-terminal graph. Fig. 1b: The graph G' on the left is d-embedded in G on the right, as we can forward-split u into (u, v) (u retains all incoming edges, and v retains at least one outgoing edge). However, there is no edge we can add or subdivide to get G from G' so G' is not h-embedded in G. The Braess graph G_B is on Fig. 1c. Examples.

3.2 Graph Embeddings

There are various definitions of graph embeddings and subdivisions [12, 29, 14], which can be summarised together as follows.

▶ **Definition 5** (Homeomorphic embedding)**.** A [directed] graph G' is *h-embedded* in (or a topological minor of) G, if G (or a graph isomorphic to G) can be derived from G' by a sequence of the following operations:

Addition. The addition of a new edge joining two existing vertices.

Subdivision. Replacement of an edge (a, b) by two edges (a, x) and (x, b).

Terminal extension. (only for 2-terminal graphs) Addition of a new edge e joining s or t with a new vertex, which becomes the new source or target.

For an undirected graph H', every h-embedding operation maintains various properties like being a 2-terminal graph. However, for a 2-terminal directed graph G', an h-embedding operation may not maintain this property (see Fig. 1a). Also, this set of operations is not rich enough for our needs. Thus, we propose a new definition for directed embeddings.

▶ **Definition 6** (Directed embedding)**.** A directed graph G' is *d-embedded* in a directed graph G if G' is isomorphic to G or to a graph derived from G by a sequence of the following operations:

Addition. Addition of a new edge (a, b), such that there is no path $b - a$.

Forward split. Replacement of node $a \neq t$ with outdegree greater than zero, by two nodes a_1 and a_2 and an edge (a_1, a_2), where a_1 retains all incoming edges, and a_2 retains at least one outgoing edge.

Backward split. Replacement of node $a \neq s$ with indegree greater than zero, by two nodes a_1 and a_2 and an edge (a_1, a_2), where a_2 retains all outgoing edges, and a_1 retains at least one incoming edge.

It is not hard to see that a subdivision of an edge (directed or undirected) can be replicated by splitting one of its end nodes, and a terminal extension can be replicated by splitting the terminal (backward split of s or forward split of t). We thus allow the operations of **edge subdivision** and **terminal extension** as valid d-embedding operations as well.

▶ **Lemma 7.** *The classes of 2-terminal directed graphs and directed acyclic graphs are closed under d-embedding.*

In particular, if G' is a TDAG and G' is d-embedded in G, then G is a TDAG.

For a 2-terminal directed graph G, the graph G' is a *valid subgraph* of G if it is a subgraph of G and is also 2-terminal. While the next lemma may seem trivial, note that it does not hold for general 2-terminal directed graphs, since a single edge is not d-embedded in any cyclic graph.

▶ **Lemma 8.** *Let G be a TDAG. If G' is a valid subgraph of G, then G' is d-embedded in G.*

We will need the following lemma later on, but it is useful to know regardless. An immediate corollary is that embedding steps only add paths and increase the connectivity of a graph.

▶ **Lemma 9.** *If G, G' differ by a single split step of vertex a into (a, b), then there is a one to one mapping between paths in G' and paths in G.*

3.3 Relations among graph operations

The way we defined them, d-minors are more restrictive than butterfly minors, whereas d-embeddings are more permissive than h-embeddings when restricting attention to acyclic graphs; see Fig. 1b. However, d-embeddings are not infinitely richer than h-embeddings. A vertex is called a *hub* if it has both an indegree and an outdegree larger than one.

▶ **Proposition 10.** *Let $G' = \langle V', E' \rangle$ and let $J \subseteq V'$ be the hubs of G'. There is a set $\mathcal{G}$ of at most $2^{|J| \times |V'|^2}$ graphs, such that for any $G = \langle V, E \rangle$, graph G' is d-embedded in G if and only if some graph in $\mathcal{G}$ is h-embedded in G. Each such graph has at most $|V|(1 + |J|)$ vertices.*

For the class of TDAGs, the concepts of directed-minor and directed-embedding turn out to be equivalent.

▶ **Theorem 11.** *Let G and G' be TDAGs. G' is d-embedded in G if and only if G' is a d-minor of G.*

Intuitively, addition and deletion operations cancel one another, as do split and contraction operations. This equivalence does not hold for general directed graphs, as added edges may not qualify for deletion (e.g. if we add an edge (a, b) where a has only incoming edges), and vice versa (if we remove an edge that is part of a cycle).

Proof. By induction, it is sufficient to show this for G', G that differ by a single d-embedding or d-minor operation. "⇒" There are 3 cases, depending on the embedding operation:

1. The addition of edge (a, b) to G' can be reversed by deleting the same edge from G. Note that $b \neq s$ as otherwise there is a path in G' from $b = s$ to a, and similarly $a \neq t$. Thus, a has outdegree at least 1 in G' and at least 2 in G. Similarly, b has indegree at least 2 in G, and thus deleting the edge (a, b) is a valid d-minor step.
2. Suppose that a vertex a in G' is split to $\{a, b\}$ with a forward split. Then, since a retains all incoming edges, b has a single incoming edge (a, b) in G. Thus, we can contract the edge (a, b) in G using backward contraction.
3. Similarly, a backward split can be reversed with a forward contraction.

"⇐" There are 3 cases, depending on the d-minor operation:

1. If the edge (a, b) is deleted from G, then since G is acyclic there is no path $b - a$. Thus adding (a, b) to G' is a valid d-embedding step.

2. Suppose that the edge (a, b) in G is backward-contracted to some vertex x in G'. This means that b has a single incoming edge. Thus all edges incoming to the pair $\{a, b\}$ are leading to a. Let $R(a)$ and $R(b)$ be the out-neighbors of a and b in G, respectively. Then by forward-splitting node x in G' and split the outgoing edges of x according to $R(a)$ and $R(b)$, we get the graph G^i.
3. Similarly, forward contraction can be reversed with backward split. ◀

4 Serial-Parallel Width

A *cut* in a 2-terminal graph $G = \langle V, E, s, t\rangle$ is a set of edges $C \subseteq E$ such that there is no $s - t$ path in $E \setminus C$. C is *minimal* if there is no cut $C' \subsetneq C$.

A set of edges $S \subseteq E$ is *parallel* if there is some $C \subseteq E$ s.t. $S \subseteq C$, and C is a minimal cut;S is *serial* if there is a simple directed $s - t$ path p that contains S.

▶ **Definition 12** (Parallel Width). The *parallel width* of a directed 2-terminal graph, $PW(G)$, is the size of the largest parallel set $S \subseteq E$.

▶ **Definition 13** (Serial-Parallel Width [28]). The *serial-parallel width* of a directed 2-terminal graph, $SPW(G)$, is the size of the largest set $S \subseteq E$ that is both serial and parallel.

Intuitively, the parallel width is the size of a maximum $s-t$ cut. For example, the width of an electric circuit coincides with the parallel width of its underlying TDAG [5]. A serial-parallel width of k means that there are at least k source-target paths, and some additional path that edge-intersects all of them. It was shown in [28] that in nonatomic routing games with diverse players, the negative externality is bounded by the serial-parallel width of the underlying network.

▶ **Example 14.** Consider the Braess graph in Fig. 1c. The minimal $s - t$ cuts are: $\{sa, sb\}, \{at, bt\}$, $\{sa, bt\}$, and $\{sb, ab, at\}$. The set $\{sa, bt\}$ is both parallel and serial, which means $SPW(G_B) \geq 2$. The set $\{sa, at\}$ is serial but not parallel; and $\{sa, sb, ab\}$ is neither. In fact, the *only* parallel set of size greater than 2 is $\{sb, ab, at\}$, which is not serial, thus $SPW(G_B) < 3$. We conclude that the serial-parallel width of the Braess graph is 2.

In contrast, both graphs in Fig. 1b have $PW(G) = 2$ and $SPW(G) = 1$.

For any 2-terminal graph G, we have $1 \leq SPW(G) \leq |V| - 1$. The lower bound is since any single edge is both parallel and serial, and the upper bound since there is no simple path of length $|V|$ or more.

▶ **Definition 15.** For any $k \geq 2$, we define the *k-serial-parallel graph* $G_{SP(k)}$ as follows. $G = \langle V, E, s, t\rangle$, where $V = \{s, t, a_2, \ldots, a_k, b_1, \ldots, b_{k-1}\}$, and $E = \bigcup_{i=2}^{k-1}\{(s, a_i), (a_i, b_i), (b_i, t), (b_i, a_{i+1})\} \cup \{(s, b_1), (a_k, t)\}$ (see Fig.2). Furthermore, $G_{P(k)}$ is a TDAG that contains k internally disjoint $s - t$ paths.

▶ **Definition 16.** A graph G is a *variant of* $G_{SP(k)}$ if we replace the edges $\{(s, a_i)\}_{i=2}^{k}$ with an arbitrary forward-subtree that respects the lexicographic order $s, a_2, \ldots, a_k$, and replace the edges $\{(b_i, t)\}_{i=1}^{k-1}$ with an arbitrary backward-subtree that respects the lexicographic order $b_1, \ldots, b_{k-1}, t$.

The serial-parallel width of $G_{SP(k)}$ is exactly k, where
$\{(s, b_1), (a_2, b_2), \ldots, (a_{k-1}, b_{k-1}), (a_k, t)\}$ are the serial-parallel edges.

The graph $G_{SP(k)}$ was used under different names in [2, 30, 28], usually to derive examples of games with high equilibrium costs.

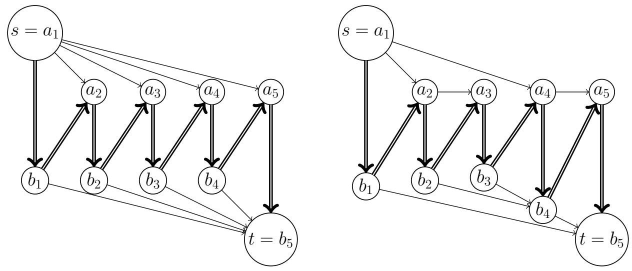


Figure 2 The left figure is the graph $G_{SP(5)}$, and the right figure is a variant of it. For convenience, the long path in each graph appears in double lines, and the forward- and backward-trees in thin lines.

▶ **Lemma 17.** *If S is a set of parallel [serial] edges in a 2-terminal graph G', then after any sequence of d-embedding steps on G', the set S is still parallel [resp., serial]. In particular, if G' is d-embedded in G then $PW(G) \geq PW(G')$ and $SPW(G) \geq SPW(G')$.*

Proof sketch. For serial sets the statement is obvious.

Consider a sequence of J d-embedding operations on $G^0 = G'$ that ends in $G^J = G$. Suppose that S is parallel. Let C^0 be a minimal cut in $G^0 = G'$ containing S. We show by induction that after every step $j \leq J$ there is a minimal cut C^j in G^j, such that $C^{j-1} \subseteq C^j$.

Assume by induction that C^{j-1} is a minimal cut in G^{j-1}. The graph G^j differs from G^{j-1} either by a single added edge, or by a single split vertex. By Lemma 9 split steps do not change the set of paths, and thus $C^j = C^{j-1}$ is still a minimal cut. Thus suppose G^j differs by an addition step of an edge $e = (a, b)$. Either C^{j-1} is still a cut in G^j, or e connects a node a reachable from s to a node b with a path to t. In the latter case, $C^j = C^{j-1} \cup \{e\}$ is a cut. To see that C^j is minimal suppose we remove an edge $e' \neq e$. If $C' = C^j \setminus \{e'\}$ is a cut in G^j, then $C' \setminus \{e\} = C^{j-1} \setminus \{e'\}$ is a cut in G^{j-1}, in contradiction to the induction hypothesis that C^{j-1} is minimal. In either case, S is still contained in a minimal cut C^j after every operation, and in particular contained in a minimal cut C^J of $G^J = G$. ◀

4.1 Characterization of graphs with bounded serial-parallel width

Before we get to our main theorem we start with a characterization of parallel sets.

▶ **Proposition 18** (Parallel sets characterization). *Let $G = \langle V, E, s, t \rangle$ be a TDAG, and a set of k edges $S \subseteq E$, where for each $e_i \in S$, $e_i = (a_i, b_i)$. The following conditions are equivalent: (1) S is parallel; (2) there is a forward-subtree T_s in G with root s and leafs $\{a_i\}_{i \leq k}$, and a backward-subtree T_t in G with leaf t and roots $\{b_i\}_{i \leq k}$; (3) there is a sequence of d-minor operations that deletes or contracts all edges except S.*

Proof. "$1 \Rightarrow 2$": Suppose that S is parallel, then it is contained in a minimal cut C. Let G_C be graph G without the edges of C. Let T_s be all vertices reachable from s in G_C, and T_t all vertices from which t is reachable and let $G(X)$ be the subgraph of G induced by $X \subseteq V$. $T_s \cap T_t = \emptyset$ as otherwise there is a path from s to t in G_C. Also, $a_i \in T_s$ for all i, as otherwise the edge e_i can be removed from C and $C \setminus \{e_i\}$ is still a cut. Likewise for $b_i \in T_t$. Since G is a TDAG, and $G(T_s)$ contains a path from s to every a_i, then $G(T_s)$ is w.l.o.g. a forward-tree. Similarly for T_t.

"$2 \Rightarrow 3$": The union of $G(T_s), G(S)$, and $G(T_t)$ is a valid subgraph G' of G of which S is a minimal cut: for any e_i there is a path $s - a_i - b_i - t$. Since G' is a valid subgraph of G, then by Lemma 8 it is d-embedded and thus a d-minor of G. Then, since all nodes in T_s

have indegree at most 1, we can backward-contract all of T_s to a single node s. Similarly, we forward-contract all of T_t to the node t, and we are left with a graph that has two nodes whose only edges are S.

"$3 \Rightarrow 1$": By Theorem 11 we can consider the reverse sequence of d-embedding operations from $G^0 = G_{P(k)}$ to $G^J = G$. By Lemma 17, the set S is still parallel after every operation and in particular in G. ◀

We get a characterization of graphs with bounded parallel width as a simple corollary.

▶ **Theorem 19.** *For any TDAG G and $k \geq 2$, $PW(G) \geq k$ if and only if $G_{P(k)}$ is a d-minor of G.*

Proof. "$\Rightarrow$": Consider some parallel set S of size k. By Prop. 18 there is a sequence of d-minor operations that ends with a graph whose only edges are S. This graph is $G_{P(k)}$. "$\Leftarrow$": Follows directly from Lemma 17 and Thm. 11, since $PW(G_{P(k)}) = k$. ◀

▶ **Theorem 20** (Main Theorem)**.** *For any TDAG G and $k \geq 2$, $SPW(G) \geq k$ if and only if some variant of $G_{SP(k)}$ is a d-minor of G.*

Proof. "$\Rightarrow$": Consider the graph G. Suppose that $SPW(G) \geq k$, then there is a set $S = \{e_1, \ldots, e_k\}$ that is part of a minimal cut C between s and t. Denote $e_i = (a_i, b_i)$.

By Prop. 18, G has a forward-subtree T_s with root s and leafs $\{a_i\}_{i \leq k}$, and a backward-subtree T_t in G with leaf t and roots $\{b_i\}_{i \leq k}$. Also, by definition of the parallel width, there is a *simple* $s - t$ path p' containing S, w.l.o.g. in lexicographic order.

We now describe a series of d-minor steps on G that will result in a variant of $G_{SP(k)}$. Delete all edges and vertices that are not part of p', T_s or T_t. This leaves us with a graph G' that is a valid subgraph of G and thus, by Lemma 8 and Thm. 11, is also a d-minor of G.

p' is composed of a sequence of subpaths between vertices $s, y_1, x_2, y_2 \ldots, x_{k-1}, y_{k-1}, x_k, t$, where each x_i is the first intersection of $[p'_{b_{i-1}a_i}]$ with T_s. Thus x_i is an ancestor of (or coincides with)a_i in T_s. Similarly, $\{y_i\}_{i=1}^{k-1}$ are on the backward-subtree T_t, where y_i is the last intersection of $[p'_{b_i a_{i+1}}]$ and T_t. Denote by $A_i \subseteq \{a_2, \ldots, a_k\}$ all leafs of the subtree of T_s rooted at x_i, and by $B_i \subseteq \{b_1, \ldots, b_{k-1}\}$ all roots of the subtree of T_t whose leaf is y_i. In particular, $a_i \in A_i$, and $a_j \notin A_i$ for $j < i$, as otherwise there is a cycle $x_i - a_j - b_j - y_j - x_i$. Likewise, $b_i \in B_i$ and $b_j \notin B_i$ for $j > i$.

Note that the indegree of all nodes in T_s is 1, except for $\{x_i\}_{i=2}^{k}$ whose indegree is 2 (one edge from the parent in T_s, and one from the predecessor node on p'), and s whose indegree is 0. We thus backward-contract all edges in T_s that do not point to some x_i. We get a forward-subtree $\hat{T}_s$:

- The root of $\hat{T}_s$ is $s = x_1$, and its nodes are $\{x_i\}_{i=2}^{k}$;
- Each path $x_i - a_i$ in G' becomes a single node $x_i = a_i$ in $\hat{G}$;
- The subtree rooted by x_i in T_s becomes a subtree in $\hat{T}_s$ over nodes A_i maintaining their order, i.e., children have higher index than their parent. For example, in Fig. 2 on the right, $A_4 = \{a_4, a_5\}$ and a_4 is a parent of a_5.

We similarly contract T_t to $\hat{T}_t$ on nodes $\{y_i\}$.

The last step is to contract every subpath $[p'_{y_i x_{i+1}}]$ to a single edge (y_i, x_{i+1}). Denote the union of these edges by $\hat{F}$, so that $S \cup \hat{F}$ is the path we get after contracting p'.

We get that the contracted graph $\hat{G}' = S \cup \hat{F} \cup \hat{T}_s \cup \hat{T}_t$ is isomorphic to a variant of $G_{SP(k)}$. More specifically, s and t are mapped to themselves, each x_i for $i = 2, \ldots, k$ in $\hat{G}$ is mapped to a_i in $G_{SP(k)}$, and each y_i for $i = 1, \ldots, k-1$ in $\hat{G}'$ is mapped to b_i in $G_{SP(k)}$. For each $i = 2, \ldots, k$, let j be the maximal index such that x_j is an ancestor of x_i in T_s. If such

j exists, then the parent of a_i in $\hat{G}'$ is a_j, and otherwise its parent is $s = a_1$. The parent of a_i in $G_{SP(k)}$ is the closest ancestor x_j of the node x_i in T_s (and similarly for the child of b_i).

"$\Leftarrow$": Follows directly from Lemma 17 and Thm. 11, since SPW for any variant of $G_{SP(k)}$ is k. ◀

Since $G_{SP(k)}$ has $2k$ vertices, we get that $SPW(G) \leq \frac{|V|}{2}$. Another corollary of Theorem 20 is a generalization of the lower bounds on negative externality from [2, 28]. These papers show how instances with high externality (depending on k) can be constructed from any variant of $G_{SP(k)}$. By Theorem 20 this is true for *any graph* G with $SPW(G) \geq k$.

4.2 Series-parallel graphs

Series-parallel 2-terminal graphs have been long studied in contexts such as electric circuits [9], complexity of graph algorithms [37], and also routing games [29, 11].

▶ **Definition 21** (Series-parallel graph [10, 16]). A *[directed] series-parallel graph* is a 2-terminal graph $\langle V, E, s, t\rangle$, and is either a single edge (s, t), or is composed recursively by one of the two following steps:

Serial composition. Combine two [directed] 2-terminal graphs $\langle V_1, E_1, s_1, t_1\rangle, \langle V_2, E_2, s_2, t_2\rangle$ serially by merging t_1 with s_2.

Parallel composition. Combine two [directed] 2-terminal graphs $\langle V_1, E_1, s_1, t_1\rangle$, $\langle V_2, E_2, s_2, t_2\rangle$ in parallel by merging s_1 with s_2, and t_1 with t_2.

Our last result in this section is showing that directed series-parallel graphs (DSP) characterize exactly the 2-terminal graphs with serial-parallel width of 1.

▶ **Proposition 22** ([14]). *Let G be a 2-terminal directed graph. Then G is a DSP if and only if the directed Braess graph G_B is not h-embedded in G.*

Proposition 22 and the relation between h-embeddings and d-embeddings yield the following.

▶ **Theorem 23.** *Let G be a TDAG, and let $k \geq 2$. The following conditions coincide. (1) G is a directed series-parallel graph. (2) The directed Braess graph G_B is not d-embedded in G. (3) $SPW(G) = 1$.*

Proof. Note that G_B has no hubs, as all vertices have at most 3 neighbors. Thus by Prop. 10, G_B is d-embedded in G if and only if it is h-embedded (as $|J| = 0$, $\mathcal{G}$ contains only G_B itself). Thus we get (1) $\iff$ (2).

(2) $\iff$ (3) follows as a special case from Thm. 20. ◀

5 Computational Problems

We first ask whether we can efficiently decide when a directed graph is 2-terminal.

▶ **Proposition 24.** *It is $\mathcal{NP}$-complete to decide if a directed graph is 2-terminal, but in $\mathcal{P}$ if the graph is acyclic.*

The next two natural computational questions accept as input 2-terminal graphs G and G'.

IsDMinor: Is G' a d-minor of G?

IsDEmbedded: Is G' d-embedded in G?

The complexity may depend on whether the graphs are TDAGs (in which case the questions coincide), and also on whether G' is a fixed graph of size k. We write down some of our results explicitly, and summarize all of them in Table 1.

Table 1 The computational complexity of problems we study. Results without references either follow from other results in the table or from known results.
* - IsDEmbedded is easy if the minor G' is acyclic.

	2-terminal graph		TDAG	
	any k	fixed k	any k	fixed k
IsSerial	$\mathcal{NP}$-c	$\mathcal{NP}$-c [P. 25]	$\mathcal{P}$	$\mathcal{P}$
IsParallel	?	?	?	$\mathcal{P}$ [P. 26]
IsSerialParallel	$\mathcal{NP}$-c	$\mathcal{NP}$-c [P. 25]	?	$\mathcal{P}$ [P. 26]
MaxSerial	$\mathcal{NP}$-c	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$
MaxParallel	$\mathcal{NP}$-c	?	$\mathcal{NP}$-c [P. 27]	$\mathcal{P}$ [C. 28]
MaxSerialParallel	?	?	?	$\mathcal{P}$ [C. 28]
IsDMinor	$\mathcal{NP}$-c	?	$\mathcal{NP}$-c	$\mathcal{P}$
IsDEmbedded	$\mathcal{NP}$-c	$\mathcal{P}$ *	$\mathcal{NP}$-c	$\mathcal{P}$

5.1 Testing properties of edge sets

We are interested in the following questions on a given 2-terminal graph $G = \langle V, E, s, t\rangle$ and a set $S = \{(a_i, b_i)\}_{i\leq k}$ of k edges:
IsSerial: Is there an $s - t$ path containing S?
IsParallel: Is S parallel?
IsSerialParallel: Is S both serial and parallel?
Note that since all of these properties are phrased in terms of existence, containment in $\mathcal{NP}$ is trivial.

Our main tool in many of the results, both positive and negative, will be the m-VertexDisjointPaths problem: given a directed graph $G = \langle V, E\rangle$ and m pairs of vertices $\{(x_i, y_i)\}_{i\leq m}$, find whether there are vertex-disjoint paths $x_i - y_i$ in G for all $i \leq m$. This problem is equivalent to that of checking if a graph G' is h-embedded in G [12], yet using it for our problems requires some modifications. The problem is $\mathcal{NP}$-complete even when G is a DAG [39], and $\mathcal{NP}$-complete for $m = 2$ in general directed graphs [12]. In contrast, it is in $\mathcal{P}$ when G is a DAG *and* m is fixed [12].

▶ **Proposition 25.** IsSerial *and* IsSerialParallel *are $\mathcal{NP}$-complete even for $k = 3$.*

For $k = 1$ every instance is a 'yes' instance, as any single edge is part of a simple path and part of a minimal cut.

The most tricky part is the complexity of identifying a parallel set. Using some of the structural results obtained in the previous sections, we can show the following.

▶ **Proposition 26.** IsParallel *is in $\mathcal{P}$ for TDAGs and fixed k.*

Proof. Denote $e_i = (a_i, b_i)$ for any $e_i \in S$. Denote $A = \{a_1, \ldots, a_k\}$ and $B = \{b_1, \ldots, b_k\}$. By Prop. 18, it suffices to decide if G contains a forward-subtree T_s to all of A, and a backward-subtree T_t from all of B to t. Note that T_s contains at most $k - 1$ "junctions", i.e., nodes with outdegree greater than one (including s). Suppose first that we guess what these vertices are and what is their hierarchy, and denote them by $X = \{s = x_1, \ldots, x_{k'}\}$ and relations T_X. We similarly guess a set Y of junctions in T_t and the relations among them T_Y. Our algorithm works as follows:

- For every x_j with degree d_j in T_X, split x_j into $d_j + 1$ nodes such that one of them x_j^0 retains all incoming edges (entry port), and each of the other $x_j^{v_j}$ (exit port) retains all outgoing edges. v_j is the first node from $X \cup A$ downward from x_j on T_s.

- Connect x_j^0 to all of $x_j^{v_j}$.
- Similarly split each $y_j \in Y$ to multiple entry ports and a single exit port.
- Find vertex-disjoint paths from each exit port to the entry port of one child in T_X or T_Y, respectively. E.g. from $x_j^{v_j}$ to a_i if $v_j = a_i$ for some $i \leq k$, or to $x_{j'}^0$ if $v_j = x_{j'}$ for some $j' \leq k'$.

Consider the algorithm above. The total number of edges in each tree T_X, T_Y is at most $2k$, so the total number of paths we seek in each iteration is less than $4k$. Such paths, if exist, can be found in time $|V|^{O(k^2)}$ due to the result of [12].

If such vertex-disjoint paths exist, then merging back all copies of each junction will provide us with a disjoint forward-subtree T_s and backward-tree T_t. In the other direction, if such trees exist and use junctions X and Y respectively, then the paths between every two junctions are vertex-disjoint except in the junctions themselves. Since we split each junction, these paths will be fully vertex disjoint. Thus the algorithm will always find trees T_s, T_t using junctions X, Y, if such exist.

The total number of iterations is the number of ways to select $2k$ vertices out of $|V|$, times the number of trees we can try on each set of size $2k$ (less than $(2k)^{(2k)}$ by Cayley's formula), so in total no more than $|V|^{O(k^2)}$ iterations.

The total runtime is $|V|^{O(k^2)}$ which is polynomial for fixed k.

In the full version of the paper we have shown a polynomial time algorithm to determine if S is serial(even polynomial in k for TDAGs). Hence, we can check whether S is serial-parallel by checking each property separately. ◀

5.2 Testing width properties of graphs

Given 2-terminal graph G and an integer k we study:
MaxSerial: Is there a serial set S of size $\geq k$?
MaxParallel: Is $PW(G) \geq k$?
MaxSerialParallel: Is $SPW(G) \geq k$?

▶ **Proposition 27.** MaxParallel *is $\mathcal{NP}$-complete even on TDAGs.*

Proof. MaxParallel problem is in $\mathcal{NP}$. Given any 2-terminal directed graph $G = \langle V, E \rangle$ and a set S of edges in E we can easily check whether S is an $s - t$ cut; if S is indeed an $s - t$ cut, then by deleting the edges in S there is no directed path from s to t and this can be easily verified via Dijkstra algorithm .

To show completeness we reduce from MaxDiCut on DAGs [24]. In an instance of MaxDiCut problem we are given a directed acyclic graph $G = \langle V, E \rangle$ and an integer k, and we are asked if there is a partition of V into two sets V_1 and V_2 so that the cardinality of the edge set $C = \{(u, v) \in E | u \in V_1, v \in V_2\}$ is at least k. We construct a 2-terminal DAG G' as follows. We add the vertex s and we connect it with every vertex $v \in V$ via an edge directed from s to v. Furthermore, we add the vertex t and we connect it with every vertex $v \in V$ via an edge directed from v to t. Clearly, G' is a 2-terminal graph. Furthermore, it is not hard to see that no directed cycles were created. Thus, G' is a 2-terminal DAG. We will prove that there exists a directed cut of size k in G if and only if there exists an $s - t$ directed cut of size $|V| + k$ in G'.

Firstly, assume that in G there exists a partition of V into V_1 and V_2 such that the size of C, i.e., the number of directed edges from V_1 to V_2, is k. Then, the set S that contains C, the edges from the vertices of V_1 to t and the edges from s to vertices of V_2, is a minimal $s - t$ cut. Observe, $|S| = |C| + |V_1| + |V_2| = k + |V|$. To see why S is an $s - t$ cut, observe that there is no path of the form $s - v - t$ with $v \in V$, because one of the edges (s, v) and (v, t) is missing. The only other way to reach t from s is to go from s to some vertex of V_1, move to V_2, and then reach t. But every edge from V_1 to V_2 is in C, hence there is no such $s - t$ path. Furthermore, S is minimal since for any edge (u, v) in C there is clearly a path $s - u - v - t$ in G' that does not contain any other edge in S, and for any other edge in $S \setminus C$ there is an $s - t$ path of length three that does not use any other in S.

For the other direction now, consider a minimal $s - t$ cut S in G' of size $|V| + k$. Denote by A all the vertices accessible from s in $E \setminus S$, and by B all other vertices of G. The cut S contains every edge from A to B, every edge from s to B, and every edge from A to t, so in particular we get that the size of the cut defined by the partition of V to A and B in G is exactly $|S| - (|A| + |B|) = |V| + k - |V| = k$. Finally, observe that the partition defined by A and B is a directed cut for G, because otherwise there would be a directed $s - t$ path and thus S would not be an $s - t$ cut. ◀

As an immediate corollary we get that IsDMinor and IsDEmbedded are $\mathcal{NP}$-complete even on a TDAG. When $G' = \langle V', E' \rangle$ is fixed, both problems are in $\mathcal{P}$: we use the algorithm of [12] for h-embedding as a subroutine on at most $2^{|V'|^3}$ graphs due to Proposition 10.

Since by Theorems 19 and 20 finding the parallel (or serial-parallel) width is equivalent to check for excluded minors whose size is a function of k, we get the following.

▶ **Corollary 28.** MaxParallel *and* MaxSerialParallel *are in* $\mathcal{P}$ *for TDAGs and fixed* k.

6 Discussion

Many different variations of operations can be used to obtain "simple" graphs that capture the essential forbidden properties of large classes of graphs: minors, embeddings, subdivisions, etc. These operations should be rich enough to allow for a small set of forbidden graphs, but restricted enough to only capture the intended class.

We believe that d-embeddings and d-minors will turn out to be useful, beyond the applications demonstrated in the paper. For example, in [21] bad graphs for planning are identified by *undirected minors*, which mislabels many graphs due to ignoring edge directions. A tighter characterization could be obtained by d-minors.

It is interesting whether d-embeddings or d-minors can be used to characterize other classes of directed graphs, such as graphs with bounded triangular width [27] or D-width [35]. Finally, there is the question of whether a directed graph version of the Graph Minor Theorem holds for d-minors or d-embeddings [19].

References

1 I. Ashlagi, D. Monderer, and M. Tennenholtz. Two-terminal routing games with unknown active players. *Artificial Intelligence*, 173(15):1441–1455, 2009.

2 M. Babaioff, R. Kleinberg, and C. Papadimitriou. Congestion games with malicious players. In *EC*, pages 103–112. ACM, 2007.

3 A. Blum and M. Furst. Fast planning through planning graph analysis. *Artificial intelligence*, 90(1-2):281–300, 1997.

4 C. Chang and J. Slagle. An admissible and optimal algorithm for searching AND/OR graphs. *Artificial Intelligence*, 2(2):117–128, 1971.

5 B. Codenotti and M. Leoncini. *Parallel Complexity of Linear System Solution*. World Scientific, 1991.

6 D. Cohen, M. Cooper, P. Jeavons, and S. Zivny. Tractable classes of binary csps defined by excluded topological minors. In *IJCAI*, pages 1945–1951, 2015.

7 G. Cooper. The computational complexity of probabilistic inference using bayesian belief networks. *Artificial intelligence*, 42(2-3):393–405, 1990.

8 E. Demaine, M. Hajiaghayi, and K. Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *FOCS*, pages 637–646. IEEE, 2005.

9 R.J Duffin. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications*, 10(2):303–318, 1965.

10 D. Eppstein. Parallel recognition of series-parallel graphs. *Inf. and Comp.*, 98(1):41–55, 1992.

11 A. Epstein, M. Feldman, and Y. Mansour. Efficient graph topologies in network routing games. *Games and Economic Behavior*, 66(1):115–125, 2009.

12 Fortune, Hopcroft, and Wyllie. The directed subgraph homeomorphism problem. *TCS: Theoretical Computer Science*, 10, 1980.

13 V. Gogate and R. Dechter. A complete anytime algorithm for treewidth. In *UAI*, pages 201–208, 2004.

14 R. Holzman and N. Law-Yone. Network structure and strong equilibrium in route selection games. *Mathematical Social Sciences*, 46(2):193–205, 2003.

15 E. Horvitz, J. Breese, and M. Henrion. Decision theory in expert systems and artificial intelligence. *International journal of approximate reasoning*, 2(3):247–302, 1988.

16 A. Jakoby, M. Liśkiewicz, and R. Reischuk. Space efficient algorithms for directed series–parallel graphs. *Journal of Algorithms*, 60(2):85–114, 2006.

17 T. Johnson, N. Robertson, P. Seymour, and R. Thomas. Directed tree-width. *Journal of Combinatorial Theory, Series B*, 82(1):138–154, 2001.

18 T. Johnson, N. Robertson, P. Seymour, and R. Thomas. Excluding a grid minor in planar digraphs. *arXiv:1510.00473*, 2015.

19 K. Kawarabayashi and S. Kreutzer. Towards the graph minor theorems for directed graphs. In *ICALP*, pages 3–10. Springer, 2015.

20 S. Kintali and Q. Zhang. Forbidden directed minors and kelly-width. *arXiv:1308.5170*, 2013.

21 J. Kleinberg and S. Oren. Time-inconsistent planning: a computational problem in behavioral economics. In *EC*, pages 547–564. ACM, 2014.

22 S. Kreutzer. Nowhere crownful classes of directed graphs. In *Encyclopedia of Algorithms*, pages 1416–1419. Springer, 2016.

23 Casimir Kuratowski. Sur le probleme des courbes gauches en topologie. *Fundamenta mathematicae*, 15(1):271–283, 1930.

24 Michael Lampis, Georgia Kaouri, and Valia Mitsou. On the algorithmic effectiveness of digraph decompositions and complexity measures. *Discrete Optimization*, 8(1):129–138, 2011. Parameterized Complexity of Discrete Optimization. `doi:10.1016/j.disopt.2010.03.010`.

25 László Lovász. Graph minor theory. *Bulletin of the American Mathematical Society*, 43(1):75–86, 2006.

26 A. Mackworth. Consistency in networks of relations. In *Readings in AI*, pages 69–78. Tioga Publ. Col., 1981.

27 K. Meer. An extended tree-width notion for directed graphs related to the computation of permanents. *Computer Science–Theory and Applications*, pages 247–260, 2011.

28 R. Meir and D. Parkes. Playing the wrong game: Bounding negative externalities in diverse populations of agents. In *AAMAS'18*, 2018. To appear.

29 I. Milchtaich. Network topology and the efficiency of equilibrium. *GEB*, 57:321–346, 2006.

30 E. Nikolova and N. Stier-Moses. The burden of risk aversion in mean-risk selfish routing. In *EC*, pages 489–506, 2015.

31 Ira Pohl. Heuristic search viewed as path finding in a graph. *Artificial intelligence*, 1(3-4):193–204, 1970.

32 N. Robertson and P. Seymour. Graph minors. II. algorithmic aspects of tree-width. *Journal of algorithms*, 7(3):309–322, 1986.

33 N. Robertson and P. Seymour. Graph minors. xx. wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004.

34 M. Rowland, A. Pacchiano, and A. Weller. Conditions beyond treewidth for tightness of higher-order lp relaxations. In *AI and Statistics*, pages 10–18, 2017.

35 M. Safari. D-width: A more natural measure for directed tree width. In *MFCS*, pages 745–756. Springer, 2005.

36 C. Shannon. The synthesis of two-terminal switching circuits. *Bell Labs Technical Journal*, 28(1):59–98, 1949.

37 K. Takamizawa, T. Nishizeki, and N. Saito. Linear-time computability of combinatorial problems on series-parallel graphs. *JACM*, 29(3):623–641, 1982.

38 P. Tang, Y. Teng, Z. Wang, S. Xiao, and Y. Xu. Computational issues in time-inconsistent planning. In *AAAI*, pages 3665–3671, 2017.

39 J. Vygen. NP-completeness of some edge-disjoint paths problems. *Discrete Applied Mathematics*, 61(1):83–90, 1995.

40 K. Wagner. Über eine eigenschaft der ebenen komplexe. *Mathematische Annalen*, 114(1):570–590, 1937.

41 Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119, 2014.