Blazing a Trail via Matrix Multiplications: A Faster Algorithm for Non-Shortest Induced Paths

Yung-Chung Chiu
Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan

Hsueh-I Lu ✉
Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan

Abstract

For vertices u and v of an n-vertex graph G, a *uv-trail* of G is an induced uv-path of G that is not a shortest uv-path of G. Berger, Seymour, and Spirkl [*Discrete Mathematics* 2021] gave the previously only known polynomial-time algorithm, running in $O(n^{18})$ time, to either output a uv-trail of G or ensure that G admits no uv-trail. We reduce the complexity to the time required to perform a poly-logarithmic number of multiplications of $n^2 \times n^2$ Boolean matrices, leading to a largely improved $O(n^{4.75})$-time algorithm.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases Induced subgraph, induced path, non-shortest path, dynamic data structure

Digital Object Identifier 10.4230/LIPIcs.STACS.2022.23

Funding *Hsueh-I Lu*: MOST grants 110–2221–E–002–075–MY3 and 107–2221–E–002–032–MY3.

1 Introduction

Let G be an n-vertex undirected and unweighted graph. Let $V(G)$ consist of the vertices of G. For any subgraph H of G, let $G[H]$ be the subgraph of G induced by $V(H)$. A subgraph H of G is *induced* if $G[H] = H$. That is, an induced subgraph of G is a subgraph of G that can be obtained by deleting a set of vertices together with its incident edges from G. Various kinds of induced subgraphs are involved in the deepest results of graph theory and graph algorithms. One of the most prominent examples concerns the *perfection* of G that the chromatic number of each induced subgraph H of G equals the clique number of H. A graph is *odd* (respectively, *even*) if it has an odd (respectively, even) number of edges. A *hole* of G is an induced cycle of G having at least four edges. The seminal Strong Perfect Graph Theorem of Chudnovsky, Robertson, Seymour, and Thomas [16, 21], conjectured by Berge in 1960 [4, 5, 6], ensures that the perfection of a graph G can be determined by detecting an odd hole in G or its complement $\bar{G}$. Based on the theorem, the first known polynomial-time algorithms for recognizing perfect graphs take $O(n^{18})$ [30] and $O(n^9)$ [13] time. The $O(n^9)$-time version can be implemented to run in $O(n^{8.373})$ time via Boolean matrix multiplications [52, §6.2].

Detecting a class of induced subgraphs can be much more difficult than detecting its counterpart that need not be induced. For instance, detecting a path spanning three prespecified vertices is tractable (via, e. g., [50, 58]), but the *three-in-a-path* problem that detects an induced path spanning three prespecified vertices is NP-hard (see, e. g., [43, 52]). Cycle detection has a similar situation. Detecting a cycle of length three, which has to be induced, is the classical triangle detection problem that can also be solved efficiently by Boolean matrix multiplications. Although it is tractable to detect a cycle of length at least four spanning two prespecified vertices (also via, e. g., [50, 58]), the *two-in-a-cycle* problem that detects a hole spanning two prespecified vertices is NP-hard (and so are the corresponding

one-in-an-even-cycle and one-in-an-odd-cycle problems) [8, 9]. See, e. g., [57, §3.1] for graph classes on which the two-in-a-cycle problem is tractable. Detecting a tree spanning an arbitrary set of prespecified vertices is easy via computing the connected components of G. Detecting an induced tree spanning an arbitrary set of prespecified vertices is NP-hard [42]. The *three-in-a-tree* problem that detects an induced tree spanning three prespecified vertices was first shown to be solvable in $O(n^4)$ time [20] and then in $O(n^2 \log^2 n)$ time [52]. The tractability of the corresponding k-in-a-tree problem for any fix $k \geq 4$ is unknown. See [54] for an $O(n^4)$-time algorithm for the k-in-a-tree problem in a graph of girth at least k.

As for subgraph detection without the requirement of spanning prespecified vertices, detecting a cycle is straightforward. Even and odd cycles are also long known to be efficiently detectable (see, e. g., [3, 32, 63]). While detecting a hole (i. e., recognizing chordal graphs) is solvable in $O(n^2)$ time [59, 60, 61], detecting an odd (respectively, even) hole is more difficult. There are early $O(n^3)$-time algorithms for detecting odd and even holes in planar graphs [46, 56], but the tractability of detecting an odd hole was open for decades (see, e. g., [22, 24, 27]) until the recent major breakthrough of Chudnovsky, Scott, Seymour, and Spirkl [19]. Their $O(n^9)$-time algorithm is later implemented to run in $O(n^8)$ time [52], immediately implying the currently best known algorithm for recognizing perfect graphs based on the Strong Perfect Graph Theorem. It is also recently known that a shortest odd hole, if any, can be found in $O(n^{14})$ time [18]. As for detecting even holes, the first polynomial-time algorithm, running in about $O(n^{40})$ time, appeared in 1997 [23, 25, 26]. It takes a line of intensive efforts to bring down the complexity to $O(n^{31})$ [14], $O(n^{19})$ [31], $O(n^{11})$ [10], and finally $O(n^9)$ [52]. The tractability of finding a shortest even hole, open for 16 years [14, 48], is resolved by a newly announced $O(n^{31})$-time algorithm [12]. See [17] (respectively, [29]) for detecting an odd (respectively, even) hole with a prespecified length lower bound. See [1, 15] for the first polynomial-time algorithm for finding an independent set of maximum weight in a graph having no hole of length at least five. See [33] for upper and lower bounds on the complexity of detecting an $O(1)$-vertex induced subgraph.

The *two-in-a-path* problem that detects an induced path spanning two prespecified vertices is equivalent to determining whether the two vertices are connected. On the other hand, the corresponding two-in-an-odd-path and two-in-an-even-path problems are NP-hard [8, 9], although each of them admits an $O(n^7)$-time algorithm when G is planar [49]. See [35, 37, 55] for how an induced even uv-path of G affects the perfection of G. See [51] for a conjecture by Erdős on how an induced uv-path of G affects the connectivity between u and v in G. Finding a longest uv-path in G that has to (respectively, need not) be induced is NP-hard [39, GT23] (respectively, [39, ND29]). See [41, 47] for longest or long induced paths in special graphs. The presence of long induced paths in G affects the tractability of coloring G [40]. See also [1] for the first polynomial-time algorithm for finding a minimum feedback vertex set of a graph having no induced path of length at least five. Detecting a non-shortest uv-path in G is easy. A k-th shortest uv-path in G can also be found in near linear time [34]. See [44] for listing induced paths and holes. See [11, §4] for the parameterized complexity of detecting an induced path with a prespecified length. Detecting an induced uv-path in a directed graph G is NP-complete (even if G is planar) [36] and $W[1]$-complete [43]. However, the tractability of detecting a non-shortest induced uv-path in an undirected graph G was unknown until the recent result of Berger, Seymour, and Spirkl [7].

Let $\|G\|$ denote the number of edges in G. A path with end-vertices u and v is a *uv-path*. If P is a path with $\{u, v\} \subseteq V(P)$, then let $P[u, v]$ denote the uv-path of P. A uv-path P of G is *shortest* if G admits no uv-path Q with $\|Q\| < \|P\|$, so each shortest uv-path of G is induced. We call an induced uv-path of G that is not a shortest uv-path of G a *uv-trail* of G.

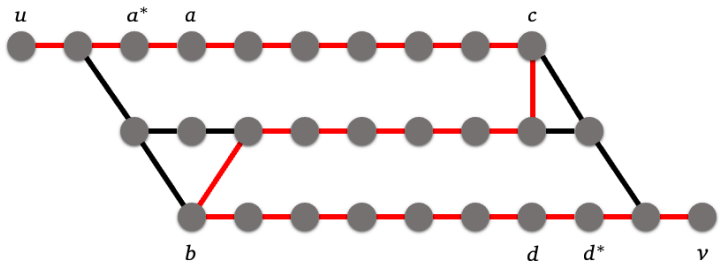

Figure 1 The red uv-path P is the only uv-trail of the uv-straight graph G. The twist pair of P is (c, b). The twist of P is 6. $P[a^*, c]$ and $P[b, d^*]$ form a pair of wings for the quadruple (a, b, c, d) of $V(G)$ in G.

See Figure 1 for an example. A graph admitting no uv-trail is *uv-trailless*. Berger, Seymour, and Spirkl [7] gave the formerly only known polynomial-time algorithm, running in $O(n^{18})$ time, to either output a uv-trail of G or ensure that G is uv-trailless. Their result leads to an $O(n^{21})$-time algorithm [28] to determine whether all holes of G have the same length. We improve the time of finding a uv-trail to $O(n^{4.75})$ as summarized in the following theorem, where the $\tilde{O}$ notation hides poly-logarithmic factors and $M(m) = O(m^{2.373})$ [2, 53, 62] denotes the time of multiplying two $m \times m$ Boolean matrices.

▶ **Theorem 1.** *For any two vertices u and v of an n-vertex graph G, it takes $\tilde{O}(M(n^2))$ time to either obtain a uv-trail of G or ensure that G is uv-trailless.*

Theorem 1 immediately reduces the time of recognizing a graph with all holes the same length from $O(n^{21})$ to $O(n^{7.75})$.

Technical overview

Berger et al.'s and our algorithms are based on the following "guess-and-verify" approach. A subroutine B taking an ℓ-tuple of $V(G)$ as the only argument is a *uv-trailblazer of degree ℓ* for G if running B on all ℓ-tuples of $V(G)$ always reports a uv-trail of G unless G is uv-trailless. We call an ℓ-tuple of $V(G)$ on which B reports a uv-trail of G a *trail marker* for B. An $O(f(n))$-time uv-trailblazer of degree ℓ for G immediately implies the following $O(n^\ell \cdot f(n))$-time *trailblazing algorithm* for G: Run B on each ℓ-tuple $(a_1, \ldots, a_\ell)$ of $V(G)$ to either obtain a uv-trail of G or ensure that $(a_1, \ldots, a_\ell)$ is not a trail marker for B. If none of the $O(n^\ell)$ iterations produces a uv-trail of G, then report that G is uv-trailless.

A graph H is *uv-straight* [7] if $\{u, v\} \subseteq V(H)$ and each vertex of H belongs to at least one shortest uv-path of H. For instance, the graph in Figure 1 is uv-straight. Berger et al.'s algorithm starts with an $O(n^3)$-time preprocessing step (see Lemma 2) that either reports a uv-trail of G or obtains a uv-straight graph H with $V(H) \subseteq V(G)$ such that (a) a uv-trail of G can be obtained from a uv-trail of H in $O(n^2)$ time and (b) if H is uv-trailless, then so is G. If no uv-trail is reported by the preprocessing, then the main procedure runs an $O(n^{18})$-time trailblazing algorithm on the uv-straight graph H based on an $O(n^4)$-time

degree-14 uv-trailblazer for H. As for postprocessing, if a uv-trail of H is obtained by the main procedure, then report a uv-trail of G obtainable in $O(n^2)$ time as ensured by the preprocessing. Otherwise, report that G is uv-trailless.

Our $O(n^{4.75})$-time algorithm adopts the preprocessing and postprocessing steps of Berger et al., while reducing the preprocessing time from $O(n^3)$ to $O(M(n))$ (see Lemma 6). For the benefit of the main procedure, we run a second preprocessing step, taking $O(n^{4.75})$ time, to compute a static data structure from which a pair of "wings" that are some disjoint paths in H, if any, for each quadruple of $V(H)$ can be obtained in $O(n)$ time (see Lemma 7). Our main procedure is also a trailblazing algorithm, based on a faster uv-trailblazer of a much lower degree for H: We reduce the time from $O(n^4)$ to $O(n^2 \log^2 n)$ and largely bring down the degree from 14 to 2. Thus, the main procedure runs in $O(n^4 \cdot \log^2 n)$ time, even faster than the second preprocessing step.

The key to our improved uv-trailblazer is a new observation, described by Lemma 5, on any shortest uv-trail P of a uv-straight graph G. Specifically, Berger et al.'s algorithm looks for a uv-trail in G that consists of (1) a shortest us-path S of G containing 7 guessed vertices and a shortest tv-path T of G containing another 7 guessed vertices such that S and T are anticomplete in G and (2) a shortest st-path Q of $G_{S,T} = G - (N_G[S \cup T] \setminus N_G[\{s,t\}])$. Lemma 5 ensures that much fewer guessed vertices on S and T suffice to guarantee that Q stays intact in $G_{S,T}$. To illustrate the usefulness of Lemma 5, we show in §2 that three lemmas of Berger et al. [7] (i. e., Lemmas 2, 3, and 4) together with Lemma 5 already yield an $O(n^2)$-time uv-trailblazer of degree 4 for G, leading to a simple $O(n^6)$-time trailblazing algorithm on G. More precisely, if a and b (respectively, c and d) are the vertices that are farthest apart from each other in P having the minimum identical distance to u (respectively, v) in G, then (a,b,c,d) is a trail marker for an $O(n^2)$-time uv-trailblazer for G: Due to the symmetry between u and v in G, Lemma 5 guarantees an $O(n^2)$-time obtainable uv-trail of G that contains the precomputed pair of "wings" for this (a,b,c,d).

Our proof of Theorem 1 in §3 further displays the usefulness of Lemma 5. We show that the aforementioned vertices a and b in P actually form a trail marker (a,b) for an $O(n^2 \log^2 n)$-time uv-trailblazer for G. Dropping both c and d from the trail marker (a,b,c,d) of §2 inevitably increases the time of the uv-trailblazer for G. We manage to keep the time of a degree-two uv-trailblazer as low as $O(n^2 \log^2 n)$ via the dynamic data structure of Holm, de Lichtenberg, and Thorup [45] supporting efficient edge updates and connectivity queries for G (see Lemma 8). To make our proof of Theorem 1 in §3 self-contained, a simplified proof of Lemma 4 is included in §2. Since Lemmas 2 and 3 are implied by Lemmas 6 and 7, which are proved in §3, our proof for the $O(n^6)$-time algorithm in §2 is also self-contained.

2 A simple $O(n^6)$-time algorithm

Let G be a connected graph containing vertices u and v. For any vertices x and y of G, let $d_G(x,y) = \|P\|$ for a shortest xy-path P of G. Let $h(x) = d_G(u,x)$ be the *height* of a vertex x in G. If xy is an edge of G, then $|h(x) - h(y)| \leq 1$. For any subgraph H of G, (i) let $G - H = G[V(G) \setminus V(H)]$, (ii) let $N_G(H)$ consist of the vertices $y \in V(G - H)$ adjacent to at least one vertex of H in G, and (iii) let $N_G[H] = N_G(H) \cup V(H)$. For any $x \in V(G)$, let $G - x = G - \{x\}$, let $N_G(x) = N_G(\{x\})$, and let $N_G[x] = N_G[\{x\}]$. X and Y are *adjacent* (respectively, *anticomplete*) in G if $N_G(X) \cap V(Y) \neq \varnothing$ (respectively, $N_G[X] \cap V(Y) = \varnothing$).

▶ **Lemma 2** (Berger et al. [7, Lemma 2.2]). *For any vertices u and v of an n-vertex connected graph G, it takes $O(n^3)$ time to obtain (1) a uv-trail of G or (2) a uv-straight graph H with $V(H) \subseteq V(G)$ such that (a) a uv-trail of G is $O(n^2)$-time obtainable from that of H and (b) if H is uv-trailless, then so is G.*

A path of G is *monotone* [7] if all of its vertices have distinct heights in G. A monotone xy-path of G is a shortest xy-path of G. The converse may not hold. A shortest xy-path of G with $\{x, y\} \cap \{u, v\} \neq \varnothing$ is monotone. A monotone a^*c-path W_1 of G containing a vertex a and a monotone bd^*-path W_2 of G containing a vertex d with

$$h(a^*) + 1 = h(a) = h(b) \leq h(c) = h(d) = h(d^*) - 1$$

form a pair (W_1, W_2) of *wings* for the quadruple (a, b, c, d) of $V(G)$ in G if

$$d_{G[W_1 \cup W_2]}(a^*, d^*) > \|W_1\| + \|W_2\|,$$

that is, $W_1 - c$ (respectively, W_1) and W_2 (respectively, $W_2 - b$) are anticomplete in G. An (a, b, c, d) is *winged* in G if G admits a pair of wings for (a, b, c, d). See also Figure 1 for an example.

▶ **Lemma 3** (Berger et al. [7, Lemma 2.1]). *It takes $O(n^6)$ time to compute a data structure from which the following statements hold for any quadruple (a, b, c, d) of $V(G)$ for an n-vertex graph G:*

1. *It takes $O(1)$ time to determine whether (a, b, c, d) is winged in G.*
2. *If (a, b, c, d) is winged in G, then it takes $O(n)$ time to obtain a pair of wings for (a, b, c, d) in G.*

We comment that Lemma 2.1 of Berger et al. [7] is slightly different from Lemma 3, but their proof is easily adjustable into one for Lemma 3. See also §3 for a proof of Lemma 7, which implies and improves upon Lemma 3.

Let G be a uv-straight graph. If $h(s) - h(t)$ is maximized by the vertices s and t of a uv-path P of G such that $P[u, s]$ is a shortest us-path of G and $P[t, v]$ is a shortest tv-path of G, then the *twist* [7] of P is $h(s) - h(t)$ and we call (s, t) the *twist pair* of P. See also Figure 1 for an example. If (s, t) is the twist pair of a uv-path P of G, then $P[u, s]$ and $P[t, v]$ are disjoint if and only if P is a non-shortest uv-path of G. The next lemma is also needed in §3. To make our proof of Theorem 1 in §3 self-contained, we include a proof of Lemma 4 simplified from that of Berger et al. [7, Lemma 2.3].

▶ **Lemma 4** (Berger et al. [7, Lemma 2.3]). *If (s, t) is the twist pair of a shortest uv-trail P of a uv-straight graph G, then $h(s) \geq h(x) \geq h(t)$ holds for each vertex x of $P[s, t]$.*

Proof. Let $I = V(P[s, t]) \setminus \{s, t\}$. Let s^* (respectively, t^*) be the neighbor of s (respectively, t) in $P[s, t]$. By definition of (s, t), we have $h(s^*) \leq h(s)$ and $h(t^*) \geq h(t)$. If $I = \varnothing$, then $(s^*, t^*) = (t, s)$ implies the lemma. Otherwise, it suffices to prove $h(s) \geq h(x) \geq h(t)$ for each $x \in I$. If $h(x) > h(s)$ were true for the $x \in I$ maximizing the lexicographical order of $(h(x), d_{P[s,t]}(x, t))$, then the concatenation of $P[u, x]$ and a shortest xv-path of G is a uv-trail (containing s^*) of G shorter than P. If $h(x) < h(t)$ were true for the $x \in I$ minimizing the lexicographical order of $(h(x), d_{P[s,t]}(x, t))$, then the concatenation of a shortest ux-path of G and $P[x, v]$ is a uv-trail (containing t^*) of G shorter than P. ◀

A monotone uc-path S of G with $h(c) = h(s)$ is a *sidetrack* for a uv-trail P of G with twist pair (s, t) if satisfying the following *Conditions S*.

S1: $d_{G[S \cup T]}(u, v) > \|S\| + \|T\|$ holds for a monotone tv-path T of G.
S2: The vertex a of S with $h(a) = h(t)$ is on the monotone subpath $P[u, s]$.

The inequality of Condition S1 is equivalent to the statement that $S - c$ (respectively, S) and T (respectively, $T - t$) are anticomplete in G. Thus, $S[a^*, c]$ and $T[t, d^*]$ form a pair of wings for (a, t, c, d) in G, where a^* is the vertex of S with $h(a^*) = h(a) - 1$ and dd^* is the

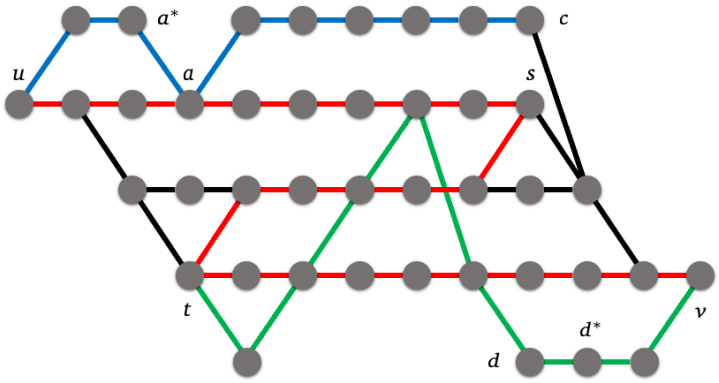

Figure 2 The blue uc-path is a sidetrack S for the red uv-trail P of the uv-straight graph G. Each of $P[t,v]$ and the green tv-path can be a monotone tv-path T satisfying Condition S1.

edge of T with $h(s) = h(d) = h(d^*) - 1$. See Figure 2 for an example. The key to our largely improved uv-trailblazers in §2 and §3 is the following lemma, whose proof is illustrated in Figure 3.

▶ **Lemma 5.** *If S is a sidetrack for a shortest uv-trail P of a uv-straight graph G with twist pair (s,t), then*

$$d_{G[S\cup P[s,t]]}(u,t) \geq d_P(u,t).$$

Proof. Condition S1 implies a monotone tv-path T of G with $d_{G[S\cup T]}(u,v) > \|S\| + \|T\|$. Assume for contradiction a shortest ut-path Q of $G[S \cup P[s,t]]$ with $\|Q\| < d_P(u,t)$, implying $d_{G[Q\cup T]}(u,v) < \|P\|$. By $t \notin V(S)$, Q contains an edge xy with $x \in V(S)$ and $y \in V(P[s,t])$ that minimizes $d_{P[s,t]}(y,t)$. Let R be a shortest uv-path of $G[Q \cup T]$. If x were not in $V(R)$, then $N_G(S[u,x] - x) \cap V(T) \neq \varnothing$, violating Condition S1. Hence, R contains x and, thus, y. Since R is an induced uv-path of G with $\|R\| < \|P\|$, we have $\|R\| = h(v)$, implying that R is monotone. By $d_R(u,x) < d_R(u,y)$,

$$h(x) + 1 = h(y). \tag{1}$$

By $\|Q\| + \|P[t,v]\| < \|P\|$, the concatenation of Q and $P[t,v]$ is a non-induced uv-path of G, implying that $G[Q \cup P[t,v]]$ contains a monotone uv-path R'. Let $x'y'$ be the edge of R' with $x' \in V(S) \cap V(Q)$ and $y' \in V(P[t,v])$ that maximizes $h(y')$. By $d_{R'}(u,x') < d_{R'}(u,y')$,

$$h(x') + 1 = h(y'). \tag{2}$$

We know $h(x') \neq h(t) - 1$ or else $y' = t$ violates Condition S1. We know $h(x') \neq h(t)$ or else Condition S2 violates that P is induced. By $h(x') \geq h(t) + 1$ and Equation (2), y' and t are anticomplete in G. Let t' be the vertex closest to y in $P[y,t]$ with $h(t') = h(t)$, implying that y' and t' are anticomplete in G no matter whether $t' = t$ or not. By $h(x) \geq h(x') \geq h(t) + 1$ and Lemma 4, the concatenation P' of a shortest ut'-path of G, $P[t',y]$, the edge yx, and a shortest xv-path of $G[S[x',x] \cup P[y',v]]$ is an induced uv-path of G shorter than P. By Equation (1) and $d_{P'}(u,x') < d_{P'}(u,y')$, we have that P' is a uv-trail of G, contradicting the definition of P. ◀

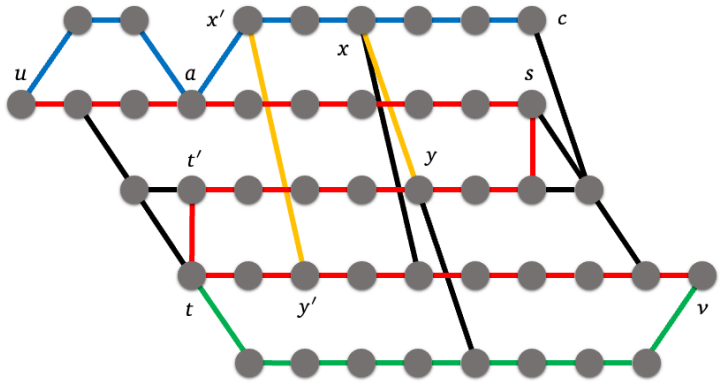

Figure 3 An illustration for the proof of Lemma 5. The red path denotes a shortest uv-trail P of the uv-straight graph G. The blue monotone path denotes a sidetrack S for P. The green path denotes a monotone path T satisfying Condition S1.

We are ready to describe and justify an $O(n^6)$-time algorithm that either reports a uv-trail of G or ensures that G is uv-trailless. Let G be connected without loss of generality.

Our $O(n^6)$-time algorithm

Apply Lemma 2 in $O(n^3)$ time to either report a uv-trail of G as stated in Lemma 2(1) or make G a uv-straight graph satisfying Conditions (a) and (b) of Lemma 2(2) with respect to the original G. If no uv-trail is reported in the previous step, then apply Lemma 3 to obtain the data structure D for the winged quadruples of G in $O(n^6)$ time. With the standard $O(n^2)$-time postprocessing readied by the preprocessing, it remains to show an $O(n^2)$-time degree-4 uv-trailblazer for the uv-straight graph G, which immediately leads to an $O(n^6)$-time trailblazing algorithm that either reports a uv-trail of G or ensures that G is uv-trailless.

Let B be the following $O(n^2)$-time subroutine, taking a quadruple (a, b, c, d) of $V(G)$ as the argument: Determine in $O(1)$ time from the data structure D whether (a, b, c, d) is winged in G. If not, then exit. Otherwise, obtain in $O(n)$ time from D a pair (W_1, W_2) of wings for (a, b, c, d) in G. Since G is uv-straight, it takes $O(n^2)$ time to obtain a monotone uc-path S of G containing W_1 and a monotone bv-path T of G containing W_2. Obtain in $O(n^2)$ time the subgraph $G_{c,b}$ of G induced by

$$\{x \in V(G) : h(b) \leq h(x) \leq h(c)\} \setminus ((N_G[S - c] \cup N_G[T - b]) \setminus \{c, b\}).$$

If c and b are not connected in $G_{c,b}$, then exit. Otherwise, report the concatenation $P_{c,b}$ of (i) the uc-path S, (ii) a shortest cb-path of $G_{c,b}$, and (iii) the bv-path T.

By definition of S, T, and $G_{c,b}$, the uv-path $P_{c,b}$ of G reported by $B(a, b, c, d)$ is induced in G, which is not monotone by $h(b) \leq h(c)$. Thus, $P_{c,b}$ is a uv-trail of G.

Let P be an arbitrary unknown shortest uv-trail of G with twist pair (s, t). Let a (respectively, d) be the vertex of the monotone $P[u, s]$ (respectively, $P[t, v]$) with $h(a) = h(t)$ (respectively, $h(d) = h(s)$). See Figure 4 for an illustration. The rest of the section shows that (a, t, s, d) is a trail marker for B.

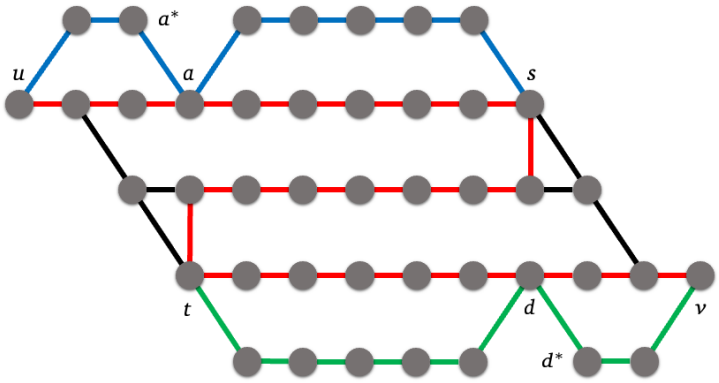

Figure 4 An illustration for the proof that B is a uv-trailblazer of degree four. The red path denotes a shortest uv-trail of the uv-straight graph G. The blue and green paths denote a monotone us-path and a monotone tv-path of G containing a precomputed pair of wings for (a, t, s, d) that need not coincide with P except at a, t, s, and d.

Observe that $P[a^*, s]$ and $P[t, d^*]$ with the neighbor a^* of a in $P[u, a]$ and the neighbor d^* of d in $P[d, v]$ form a pair of wings for (a, t, s, d) in G. Thus, the quadruple (a, t, s, d) is winged in G. The monotone us-path S of G containing W_1 is a sidetrack for P, since the monotone tv-path T of G containing W_2 satisfies Conditions S1 and S2 for S. Due to the symmetry between u and v in G, the monotone vt-path T of the vu-straight graph G is also a sidetrack for the shortest vu-trail P of G with twist pair (t, s), since the monotone su-path S of G satisfies Conditions S1 and S2 for T. Lemma 4 guarantees $h(t) \leq h(x) \leq h(s)$ for each vertex x of $P[s, t]$. By Lemma 5, $P[s, t] - \{s, t\}$ is anticomplete to both $S - s$ and $T - t$, implying that $P[s, t]$ is a path of $G_{s,t}$. Since s and t are connected in $G_{s,t}$, the subroutine call $B(a, t, s, d)$ outputs a uv-trail $P_{s,t}$ of G in $O(n^2)$ time. Hence, (a, t, s, d) is indeed a trail marker of B.

As a matter of fact, $P_{s,t}$ is a shortest uv-trail of G due to $\|P_{s,t}\| = \|P\|$. Since the preprocessing and postprocessing may ruin the shortestness of the reported uv-trail, we have an $O(n^6)$-time algorithm on an n-vertex general (respectively, uv-straight) graph G that either reports a general (respectively, shortest) uv-trail of G or ensures that G is uv-trailless.

3 An $O(n^{4.75})$-time algorithm

This section gives a self-contained proof of Theorem 1. The *product* of $m \times m$ Boolean matrices A and B is the $m \times m$ Boolean matrix C such that $C(i, k) = \mathit{true}$ if and only if $A(i, j) = B(j, k) = \mathit{true}$ holds for an index j. The following lemma implies and improves upon Lemma 2, which takes $O(n^3)$ time to obtain a uv-trail of G from a uv-trail of H.

▶ **Lemma 6.** *For any vertices u and v of an n-vertex connected graph G, it takes $O(M(n))$ time to obtain (1) a uv-trail of G or (2) a uv-straight graph H with $V(H) \subseteq V(G)$ such that (a) a uv-trail of G can be obtained from a uv-trail of H in $O(n^2)$ time and (b) if H is uv-trailless, then so is G.*

Proof. We adopt the proof of Berger et al. [7, Lemma 2.2] with slight simplification and improvement. It takes $O(n^2)$ time to obtain the maximal set $F \subseteq V(G)$ such that $G[F]$ is uv-straight. If $F = V(G)$, then the lemma is proved by returning $H = G$. The rest of

the proof assumes $F \subsetneq V(G)$. It takes $O(M(n))$ time to determine whether some connected component K of $G-F$ admits nonadjacent vertices x and y of $N_G(K) \subseteq F$ with $h(x) < h(y)$. If there is such a (K, x, y), then a shortest uv-path of $G[P_x \cup K \cup P_y]$ for any shortest ux-path P_x and yv-path P_y of G is a uv-trail of G obtainable in $O(n^2)$ time, proving the lemma. Otherwise, let H be the union of the uv-straight $G[F]$ and the $O(M(n))$-time obtainable graph H' with $V(H') = F$ (via contracting each connected component of $G - F$ into a single vertex and then squaring the adjacency matrix) such that distinct vertices x and y are adjacent in H' if and only if $\{x, y\} \subseteq N_G(K)$ holds for a connected component K of $G - F$. Observe that each edge xy of H' with $h(x) \neq h(y)$ is also an edge of $G[F]$. By $|h(x)-h(y)| \leq 1$ for all edges xy of H', H remains uv-straight and $d_H(u, x) = h(x)$ holds for each $x \in F$. To see Condition (a), for any given uv-trail Q of H, let P be an $O(n^2)$-time obtainable non-monotone uv-path of G obtained from Q by replacing each edge xy of Q not in $G[F]$ with a shortest xy-path P_{xy} of $G - (F \setminus \{x, y\})$. If P were not induced, then there is an edge zz' of $G[P]$ not in P with $z \in V(P_{xy})$ and $z' \in V(P_{x'y'})$ for distinct edges xy and $x'y'$ of Q that are not in $G[F]$. Thus, $\{x, y, x', y'\} \subseteq N_G(K)$ holds for some connected component K of $G-F$. By definition of H', $H[\{x, y, x', y'\}]$ is complete, contradicting that Q is an induced path of H. Thus, P is a uv-trail of G, proving Condition (a). As for Condition (b), let P be a uv-trail of G. For any distinct vertices x and y of P such that $P[x, y]$ is a maximal subpath of P contained by $G[\{x, y\} \cup K]$ for some connected component K of $G - F$, $P[x, y]$ is an induced xy-path of $G[\{x, y\} \cup K]$. The path Q obtained from P by replacing each such $P[x, y]$ by the edge xy of H' is an induced uv-path of H. If Q were a shortest uv-path of H, then $|h(x) - h(y)| = 1$ holds for each edge xy of Q, implying that each edge xy of Q is an edge of P, contradicting that P is a uv-trail of G. ◀

The bottleneck of our algorithm for Theorem 1 comes from the following lemma, which implies and improves upon Lemma 3 that takes $O(n^6)$ time.

▶ **Lemma 7.** *It takes $\tilde{O}(M(n^2))$ time to compute a data structure from which the following statements hold for any quadruple (a, b, c, d) of $V(G)$ for an n-vertex graph G:*

1. *It takes $O(1)$ time to determine whether (a, b, c, d) is winged in G.*
2. *If (a, b, c, d) is winged in G, then it takes $O(n)$ time to obtain a pair of wings for (a, b, c, d) in G.*

Proof. The lemma holds clearly for the quadruples (a, b, c, d) of $V(G)$ with $h(c) \leq h(a) + 1$. The rest of the proof handles those with $h(a)+2 \leq h(c)$. A pair of wings for such an (a, b, c, d) must be anticomplete in G. It takes $O(n^4)$ time to obtain the $n^2 \times n^2$ Boolean matrix A such that $A((a, b), (c, d)) = \mathit{true}$ if and only if (i) $h(a) = h(b) \leq h(c) = h(d) \leq h(a) + 1$ and (ii) G admits a pair of anticomplete wings for (a, b, c, d). The transitive closure $C = A^n$ of A can be obtained in $O(M(n^2) \cdot \log n)$ time via obtaining A^{2^i} in the i-th iteration. That is, $C((a, b), (c, d)) = \mathit{true}$ if and only if (i) $h(a) = h(b) \leq h(c) = h(d)$ and (ii) G admits a pair of anticomplete wings for (a, b, c, d) in G. Statement 1 is proved. Statement 2 is immediate from the $\tilde{O}(M(n^2))$-time obtainable $n^2 \times n^2$ witness matrix W for C by, e. g., Galil and Margalit [38]. That is, if $C((a, b), (c, d)) = \mathit{true}$ and $h(a) + 2 \leq h(c)$ hold, then $W((a, b), (c, d))$ is a vertex pair (x, y) that satisfies $h(a) < h(x) < h(c)$ and $C((a, b), (x, y)) = C((x, y), (c, d)) = \mathit{true}$. ◀

The following dynamic data structure for a graph supports efficient edge updates and connectivity queries.

▶ **Lemma 8** (Holm, de Lichtenberg, and Thorup [45])**.** *There is a data structure for an initially empty n-vertex graph that supports each edge insertion and edge deletion in amortized $O(\log^2 n)$ time and answers whether two vertices are connected in $O(\log n / \log \log n)$ time.*

We are ready to prove Theorem 1. Assume without loss of generality that G is connected.

Our $O(n^{4.75})$-time algorithm

Apply Lemma 6 in $O(M(n))$ time to either report a uv-trail of G as in Lemma 6(1) or make G a uv-straight graph satisfying Conditions (a) and (b) of Lemma 6(2) with respect to the original G. If no uv-trail is reported in the previous step, then apply Lemma 7 in $\tilde{O}(M(n^2))$ time to obtain the data structure D for the winged quadruples of $V(G)$ in G. It remains to show an $O(n^2 \log^2 n)$-time degree-two uv-trailblazer for the uv-straight graph G based on the precomputed D which already spends $O(n^{4.75})$ time. We proceed in two phases. Phase 1 shows that we already have an $O(n^3)$-time degree-two uv-trailblazer for G. Phase 2 then reduces the time to $O(n^2 \log^2 n)$ via Lemma 8.

Phase 1. Let B_1 be the $O(n^3)$-time subroutine, taking a pair (a, b) of $V(G)$ as the only argument, that runs the following $O(n^2)$-time procedure for each vertex c of G: Determine from D in $O(n)$ time whether G admits a winged quadruple (a, b, c, d_c) of $V(G)$ for some d_c. If not, then exit. Otherwise, obtain from D in $O(n)$ time a pair (W_1, W_2) of wings for an arbitrary winged (a, b, c, d_c). Since G is uv-straight, it takes $O(n^2)$ time to obtain a monotone uc-path S_c of G containing W_1 and a monotone bv-path T_c of G containing W_2. Obtain in $O(n^2)$ time the subgraph G_c of G induced by

$$(\{x \in V(G) : h(b) \leq h(x) \leq h(c)\} \setminus (N_G[S_c - c] \setminus \{c\})) \cup V(T_c).$$

If the vertices c and b are not connected in G_c, then exit. Otherwise, report the $O(n^2)$-time obtainable concatenation P_c of the uc-path S_c of G and a shortest cv-path of G_c.

By definition of S_c, T_c, and G_c, the uv-path P_c of G reported by $B_1(a, b)$ for any c is induced in G. Since the height of each neighbor of c in G_c is at most $h(c)$, P_c is not monotone. Thus, P_c is a uv-trail of G. Let P be an arbitrary unknown shortest uv-trail of G with twist pair (s, t). Let a (respectively, e) be the vertex of the monotone $P[u, s]$ (respectively, $P[t, v]$) with $h(a) = h(t)$ (respectively, $h(e) = h(s)$). See Figure 5 for an illustration. To ensure that B_1 is an $O(n^3)$-time uv-trailblazer of degree 2 for G, the rest of the phase proves that (a, t) is a trail marker for B_1 by showing that the iteration with $c = s$ reports a uv-trail P_s of G.

Let a^* be the neighbor of a in the monotone $P[u, a]$, implying $h(a^*) = h(t) - 1$. Let e^* be the neighbor of e in the monotone $P[e, v]$, implying $h(e^*) = h(s) + 1$. Since $P[a^*, s]$ and $P[t, e^*]$ form a pair of wings for (a, t, s, e) in G, there is a d_s such that (a, t, s, d_s) is winged in G. Let (W_1, W_2) be the pair of wings for (a, t, s, d_s) in G obtained from D. The monotone us-path S_s of G containing W_1 is a sidetrack for P, since the monotone tv-path T_s of G containing W_2 satisfies Conditions S1 and S2 for S_s. By Lemma 4, each vertex x of $P[s, t]$ satisfies $h(t) \leq h(x) \leq h(s)$. By Lemma 5, $S_s - s$ and $P[s, t] - s$ are anticomplete in G, implying that $P[s, t]$ is a path of G_s. Since s and t are connected in G_s, the subroutine call $B_1(a, t)$ outputs a uv-trail P_s of G in the iteration with $c = s$. Hence, (a, t) is indeed a trail marker of B. One can verify that P_s is also a shortest uv-trail of the uv-straight G, although d_s need not be e. Thus, we have an $O(n^5)$-time algorithm on an n-vertex general (respectively, uv-straight) graph G that either reports a general (respectively, shortest) uv-trail of G or ensures that G is uv-trailless.

Phase 2. Since many prefixes of a long sidetrack for a shortest uv-trail P of G remain sidetracks for P, an edge can be deleted and then inserted back $\Omega(n)$ times in Phase 1. Phase 2 avoids the redundancy by processing the sidetracks in the decreasing order of their

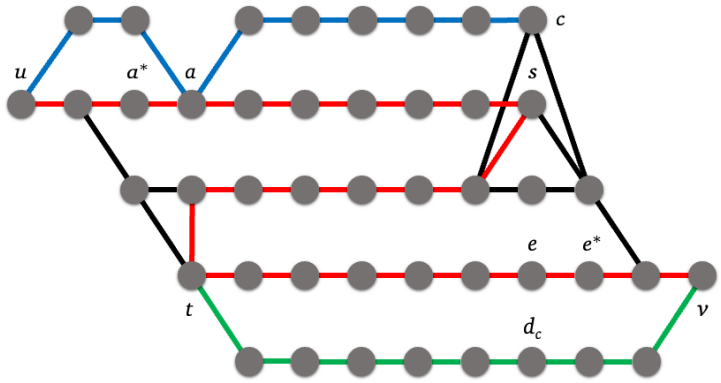

Figure 5 An illustration for the proof that B_1 is a uv-trailblazer of degree two. The red path denotes a shortest uv-trail P of the uv-straight graph G. The blue and green paths denote a monotone uc-path S_c and a monotone tv-path T_c of G containing a precomputed pair of wings for (a, t, c, d_c) that need not coincide with P except at a and t.

lengths. Let B_2 be the following subroutine that takes a pair (a, b) of $V(G)$ as the only argument. Obtain in overall $O(n^2)$ time from D each set C_i with $0 \leq i \leq h(v)$ that consists of the vertices c of G with $h(c) = i$ such that G admits a winged quadruple (a, b, c, d_c) for some vertex d_c. Let C be the union of all C_i with $0 \leq i \leq h(v)$. Obtain in overall $O(n^2)$ time from D for each vertex $c \in C$ (i) a monotone uc-path S_c of G containing a and (ii) a monotone bv-path T_c with

$$d_{G[S_c \cup T_c]}(u, v) > \|S_c\| + \|T_c\|.$$

Obtain the subgraph H of G induced by the vertices with heights at least $h(a)$ in $O(n^2 \log^2 n)$ time by the dynamic data structure of Lemma 8. Iteratively perform the following steps in the decreasing order of the indices i with $h(a) \leq i < h(v)$:

1. Delete from H the incident edges of $N_G[S_c - c]$ in G for all $c \in C_i$.
2. Insert to H the incident edges of C_i in G.
3. Delete from H all edges xy of G with $h(x) = i$ and $h(y) = i + 1$.
4. If b is not connected to any $c \in C_i$ in H, then proceed to the next iteration. Otherwise, let c be an arbitrary vertex of C_i that is connected to b in H. Exit the loop and report the $O(n^2)$-time obtainable concatenation P_c of S_c and a shortest cv-path of $G[H \cup T_c]$.

Since $S_c - c$ and $T_c - b$ are anticomplete in G and the height of each neighbor of c in H is at most $h(c)$, any arbitrary reported uv-path P_c of G is a uv-trail of G.

Throughout all iterations, the incident edges of each vertex of G is deleted $O(1)$ times by Step 1, each edge of G is updated $O(1)$ times by Steps 2 and 3, and each vertex $c \in C$ is queried $O(1)$ times by Step 4. Thus, each subroutine call $B_2(a, b)$ runs in $O(n^2 \log^2 n)$ time.

Let P be an arbitrary unknown shortest uv-trail of G with twist pair (s, t). As in Phase 1, let a (respectively, e) be the vertex of the monotone $P[u, s]$ (respectively, $P[t, v]$) with $h(a) = h(t)$ (respectively, $h(e) = h(s)$). The rest of the phase proves that (a, t) is a trail marker for B_2 by showing that an iteration with $i \geq h(s)$ in the loop of the subroutine call $B_2(a, t)$ reports a uv-trail P_c of G. See Figure 6 for an illustration.

If an iteration of $B_2(a, t)$ with $i \geq h(s) + 1$ reports a uv-trail of G (that need not be shortest), then we are done. Otherwise, we show that the iteration with $i = h(s)$ has to report a uv-trail of G. For each $c \in C$ with $h(c) \geq i$, let s_c be the unknown vertex of S_c

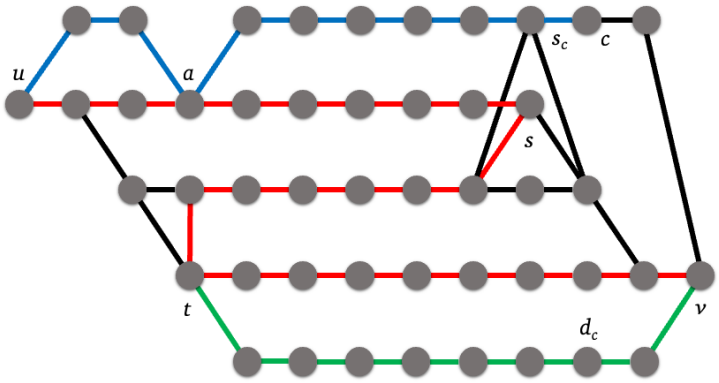

Figure 6 An illustration for the proof that B_2 is a uv-trailblazer of degree two. The red path denotes a shortest uv-trail P of the uv-straight graph G. The blue and green paths denote a monotone uc-path S_c and a monotone tv-path T_c of G containing a precomputed pair of wings for (a, t, c, d_c) that need not coincide with P except at a and t. $S_c[u, s_c]$ remains a sidetrack for P.

with $h(s_c) = i$. $S_c[u, s_c]$ remains a sidetrack for P, since T_c still satisfies Conditions S1 and S2 for $S_c[u, s_c]$. Thus, $s_c \in C_i$. By Lemma 5, $S_c[u, s_c] - s_c$ and $P[s, t] - s$ are anticomplete in G even if $S_c[u, s_c]$ need not be S_{s_c}. As a result, $P[s, t] - s$ is a path of the H at the completion of Step 1 in the i-th iteration. By $s \in C_i$ and Lemma 4, $P[s, t]$ is a path of the graph H at the completion of Step 3 in the i-th iteration. Therefore, s is a $c \in C_i$ that is connected to t in H. Step 4 in the i-th iteration has to output a (shortest) uv-trail P_c of G for some $c \in C_i$ that need not be s. Thus, we have an $O(n^{4.75})$-time algorithm that either obtains a uv-trail of G or ensures that G is uv-trailless. A reported uv-trail of G by this $O(n^{4.75})$-time algorithm need not be a shortest uv-trail of G, since we cannot afford to spend $O(n^2)$ time, as in Phase 1, for each $c \in C$ that is connected to t in the H at the $h(c)$-th iteration to obtain a shortest cv-path of $G[H \cup T_c]$.

4 Concluding remarks

We show an $O(n^{4.75})$-time algorithm for computing a uv-trail of an n-vertex undirected unweighted graph G with $\{u, v\} \subseteq V(G)$. The key to our improved algorithm is the observation regarding an arbitrary shortest uv-trail of a uv-straight graph G described by Lemma 5. The inequality of Lemma 5 is stronger than the condition that $S - c$ and $P[s, t] - s$ are anticomplete in G. As a matter of fact, the latter suffices for our uv-trailblazers in §2 and §3. Thus, a further improved uv-trailblazer might be possible if the wings for a winged quadruple can be obtained more efficiently. As mentioned in Phase 1 of §3, a shortest uv-trail, if any, of a uv-straight G can be obtained by our B_1-based trailblazing algorithm in $O(n^5)$ time. Detecting a uv-trail with length at least $2d_G(u, v)$ is NP-complete [7, Theorem 1.6]. It would be of interest to see if a shortest uv-trail or a uv-trail having length at least $d_G(u, v) + k$ for a positive $k = O(1)$ in a general G can be obtained in polynomial time. It is also of interest to see whether the one-to-all (respectively, all-pairs) version of the problem can be solved in time much lower than $O(n^{5.75})$ (respectively, $O(n^{6.75})$).

References

1 Tara Abrishami, Maria Chudnovsky, Marcin Pilipczuk, Pawel Rzazewski, and Paul D. Seymour. Induced subgraphs of bounded treewidth and the container method. In Dániel Marx, editor, *Proceedings of the 32nd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1948–1964, 2021. doi:10.1137/1.9781611976465.116.

2 Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In Dániel Marx, editor, *Proceedings of the 32nd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 522–539, 2021. doi:10.1137/1.9781611976465.32.

3 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM*, 42(4):844–856, 1995. doi:10.1145/210332.210337.

4 Claude Berge. Les problèmes de coloration en théorie des graphes. *Publications de l'Institut de statistique de l'Université de Paris*, 9:123–160, 1960.

5 Claude Berge. Färbung von Graphen, deren sämtliche bzw. deren ungerade Kreise starr sind (Zusammenfassung). *Wissenschaftliche Zeitschrift, Martin Luther Universität Halle-Wittenberg, Mathematisch-Naturwissenschaftliche Reihe*, 10:114–115, 1961.

6 Claude Berge. *Graphs*. North-Holland, Amsterdam, New York, 1985.

7 Eli Berger, Paul D. Seymour, and Sophie Spirkl. Finding an induced path that is not a shortest path. *Discrete Mathematics*, 344(7):112398.1–112398.6, 2021. doi:10.1016/j.disc.2021.112398.

8 Daniel Bienstock. On the complexity of testing for odd holes and induced odd paths. *Discrete Mathematics*, 90(1):85–92, 1991. See [9] for corrigendum. doi:10.1016/0012-365X(91)90098-M.

9 Daniel Bienstock. Corrigendum to: D. Bienstock, "On the complexity of testing for odd holes and induced odd paths" Discrete Mathematics 90 (1991) 85–92. *Discrete Mathematics*, 102(1):109, 1992. doi:10.1016/0012-365X(92)90357-L.

10 Hsien-Chih Chang and Hsueh-I Lu. A faster algorithm to recognize even-hole-free graphs. *Journal of Combinatorial Theory, Series B*, 113:141–161, 2015. doi:10.1016/j.jctb.2015.02.001.

11 Yijia Chen and Jörg Flum. On parameterized path and chordless path problems. In *Proceedings of the 22nd Annual IEEE Conference on Computational Complexity*, pages 250–263, 2007. doi:10.1109/CCC.2007.21.

12 Hou-Teng Cheong and Hsueh-I Lu. Finding a shortest even hole in polynomial time. *Journal of Graph Theory*, 2022, to appear. doi:10.1002/jgt.22748.

13 Maria Chudnovsky, Gérard Cornuéjols, Xinming Liu, Paul D. Seymour, and Kristina Vušković. Recognizing Berge graphs. *Combinatorica*, 25(2):143–186, 2005. doi:10.1007/s00493-005-0012-8.

14 Maria Chudnovsky, Ken-ichi Kawarabayashi, and Paul Seymour. Detecting even holes. *Journal of Graph Theory*, 48(2):85–111, 2005. doi:10.1002/jgt.20040.

15 Maria Chudnovsky, Marcin Pilipczuk, Michal Pilipczuk, and Stéphan Thomassé. On the maximum weight independent set problem in graphs without induced cycles of length at least five. *SIAM Journal on Discrete Mathematics*, 34(2):1472–1483, 2020. doi:10.1137/19M1249473.

16 Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The strong perfect graph theorem. *Annals of Mathematics*, 164(1):51–229, 2006. doi:10.4007/annals.2006.164.51.

17 Maria Chudnovsky, Alex Scott, and Paul Seymour. Detecting a long odd hole. *Combinatorica*, 41(1):1–30, 2021. doi:10.1007/s00493-020-4301-z.

18 Maria Chudnovsky, Alex Scott, and Paul Seymour. Finding a shortest odd hole. *ACM Transactions on Algorithms*, 17(2):13.1–13.21, 2021. doi:10.1145/3447869.

19 Maria Chudnovsky, Alex Scott, Paul Seymour, and Sophie Spirkl. Detecting an odd hole. *Journal of the ACM*, 67(1):5.1–5.12, 2020. doi:10.1145/3375720.

20 Maria Chudnovsky and Paul Seymour. The three-in-a-tree problem. *Combinatorica*, 30(4):387–417, 2010. doi:10.1007/s00493-010-2334-4.

21 Maria Chudnovsky and Paul D. Seymour. Even pairs in Berge graphs. *Journal of Combinatorial Theory, Series B*, 99(2):370–377, 2009. doi:10.1016/j.jctb.2008.08.002.

22 Maria Chudnovsky and Vaidy Sivaraman. Odd holes in bull-free graphs. *SIAM Journal on Discrete Mathematics*, 32(2):951–955, 2018. doi:10.1137/17M1131301.

23 Michele Conforti, Gérard Cornuéjols, Ajai Kapoor, and Kristina Vušković. Finding an even hole in a graph. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 480–485, 1997. doi:10.1109/SFCS.1997.646136.

24 Michele Conforti, Gérard Cornuéjols, Ajai Kapoor, and Kristina Vušković. Even and odd holes in cap-free graphs. *Journal of Graph Theory*, 30(4):289–308, 1999. doi:10.1002/(SICI)1097-0118(199904)30:4\%3C289::AID-JGT4\%3E3.0.CO;2-3.

25 Michele Conforti, Gérard Cornuéjols, Ajai Kapoor, and Kristina Vušković. Even-hole-free graphs Part I: Decomposition theorem. *Journal of Graph Theory*, 39(1):6–49, 2002. doi:10.1002/jgt.10006.

26 Michele Conforti, Gérard Cornuéjols, Ajai Kapoor, and Kristina Vušković. Even-hole-free graphs Part II: Recognition algorithm. *Journal of Graph Theory*, 40(4):238–266, 2002. doi:10.1002/jgt.10045.

27 Michele Conforti, Gérard Cornuéjols, Xinming Liu, Kristina Vušković, and Giacomo Zambelli. Odd hole recognition in graphs of bounded clique size. *SIAM Journal on Discrete Mathematics*, 20(1):42–48, 2006. doi:10.1137/S089548010444540X.

28 Linda Cook, Jake Horsfield, Myriam Preissmann, Cléophée Robin, Paul Seymour, Ni Luh Dewi Sintiari, Nicolas Trotignon, and Kristina Vušković. Graphs with all holes the same length. *arXiv*, 2021. arXiv:2110.09970.

29 Linda Cook and Paul Seymour. Detecting a long even hole. *arXiv*, 2020. arXiv:2009.05691.

30 Gérard Cornuéjols, Xinming Liu, and Kristina Vušković. A polynomial algorithm for recognizing perfect graphs. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 20–27, 2003. doi:10.1109/SFCS.2003.1238177.

31 Murilo Vicente Gonçalves da Silva and Kristina Vušković. Decomposition of even-hole-free graphs with star cutsets and 2-joins. *Journal of Combinatorial Theory, Series B*, 103(1):144–183, 2013. doi:10.1016/j.jctb.2012.10.001.

32 Søren Dahlgaard, Mathias Bæk Tejs Knudsen, and Morten Stöckel. Finding even cycles faster via capped k-walks. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, pages 112–120. ACM, 2017. doi:10.1145/3055399.3055459.

33 Mina Dalirrooyfard, Thuy Duong Vuong, and Virginia Vassilevska Williams. Graph pattern detection: hardness for all induced patterns and faster non-induced cycles. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*, pages 1167–1178, 2019. doi:10.1145/3313276.3316329.

34 David Eppstein. Finding the k shortest paths. *SIAM Journal on Computing*, 28(2):652–673, 1998. doi:10.1137/S0097539795290477.

35 Hazel Everett, Celina M. H. de Figueiredo, Cláudia Linhares Sales, Frédéric Maffray, Oscar Porto, and Bruce A. Reed. Path parity and perfection. *Discrete Mathematics*, 165-166:233–252, 1997. doi:10.1016/S0012-365X(96)00174-4.

36 Michael R. Fellows, Jan Kratochvíl, Matthias Middendorf, and Frank Pfeiffer. The complexity of induced minors and related problems. *Algorithmica*, 13(3):266–282, 1995. doi:10.1007/BF01190507.

37 J. Fonlupt and J.P. Uhry. Transformations which preserve perfectness and H-perfectness of graphs. In Achim Bachem, Martin Grötschel, and Bemhard Korte, editors, *Bonn Workshop on Combinatorial Optimization*, volume 66 of *North-Holland Mathematics Studies*, pages 83–95. North-Holland, 1982. doi:10.1016/S0304-0208(08)72445-9.

38 Zvi Galil and Oded Margalit. Witnesses for boolean matrix multiplication and for transitive closure. *Journal of Complexity*, 9(2):201–221, 1993. doi:10.1006/jcom.1993.1014.

39 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

40 Serge Gaspers, Shenwei Huang, and Daniël Paulusma. Colouring square-free graphs without long induced paths. In Rolf Niedermeier and Brigitte Vallée, editors, *Proceedings of the 35th Symposium on Theoretical Aspects of Computer Science*, LIPIcs 96, pages 35.1–35.15, 2018. doi:10.4230/LIPIcs.STACS.2018.35.

41 Emilio Di Giacomo, Giuseppe Liotta, and Tamara Mchedlidze. Lower and upper bounds for long induced paths in 3-connected planar graphs. *Theoretical Computer Science*, 636:47–55, 2016. doi:10.1016/j.tcs.2016.04.034.

42 Petr A. Golovach, Daniël Paulusma, and Erik Jan van Leeuwen. Induced disjoint paths in AT-free graphs. In Fedor V. Fomin and Petteri Kaski, editors, *Proceedings of the 13th Scandinavian Symposium and Workshops on Algorithm Theory*, Lecture Notes in Computer Science 7357, pages 153–164, 2012. doi:10.1007/978-3-642-31155-0_14.

43 Robert Haas and Michael Hoffmann. Chordless paths through three vertices. *Theoretical Computer Science*, 351(3):360–371, 2006. doi:10.1016/j.tcs.2005.10.021.

44 Chính T. Hoàng, Marcin Kaminski, Joe Sawada, and R. Sritharan. Finding and listing induced paths and cycles. *Discrete Applied Mathematics*, 161(4-5):633–641, 2013. doi:10.1016/j.dam.2012.01.024.

45 Jacob Holm, Kristian de Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *Journal of the ACM*, 48(4):723–760, July 2001. doi:10.1145/502090.502095.

46 Wen-Lian Hsu. Recognizing planar perfect graphs. *Journal of the ACM*, 34(2):255–288, 1987. doi:10.1145/23005.31330.

47 Lars Jaffke, O-joung Kwon, and Jan Arne Telle. Mim-width I. induced path problems. *Discrete Applied Mathematics*, 278:153–168, 2020. doi:10.1016/j.dam.2019.06.026.

48 David S. Johnson. The NP-completeness column. *ACM Transactions on Algorithms*, 1(1):160–176, 2005. doi:10.1145/1077464.1077476.

49 Marcin Kaminski and Naomi Nishimura. Finding an induced path of given parity in planar graphs in polynomial time. In Yuval Rabani, editor, *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 656–670, 2012. doi:10.1137/1.9781611973099.55.

50 Ken-ichi Kawarabayashi, Yusuke Kobayashi, and Bruce A. Reed. The disjoint paths problem in quadratic time. *Journal of Combinatorial Theory, Series B*, 102(2):424–435, 2012. doi:10.1016/j.jctb.2011.07.004.

51 Matthias Kriesell. Induced paths in 5-connected graphs. *Journal of Graph Theory*, 36(1):52–58, 2001. doi:10.1002/1097-0118(200101)36:1\%3C52::AID-JGT5\%3E3.0.CO;2-N.

52 Kai-Yuan Lai, Hsueh-I Lu, and Mikkel Thorup. Three-in-a-tree in near linear time. In *Proccedings of the 52nd Annual ACM Symposium on Theory of Computing*, pages 1279–1292, 2020. doi:10.1145/3357713.3384235.

53 François Le Gall. Powers of tensors and fast matrix multiplication. In Katsusuke Nabeshima, Kosaku Nagasaka, Franz Winkler, and Ágnes Szántó, editors, *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, pages 296–303, 2014. doi:10.1145/2608628.2608664.

54 Wei Liu and Nicolas Trotignon. The k-in-a-tree problem for graphs of girth at least k. *Discrete Applied Mathematics*, 158(15):1644–1649, 2010. doi:10.1016/j.dam.2010.06.005.

55 Henry Meyniel. A new property of critical imperfect graphs and some consequences. *European Journal of Combinatorics*, 8(3):313–316, 1987. doi:10.1016/S0195-6698(87)80037-9.

56 Oscar Porto. Even induced cycles in planar graphs. In *Proceedings of the 1st Latin American Symposium on Theoretical Informatics*, pages 417–429, 1992. doi:10.1007/BFb0023845.

57 Marko Radovanovic, Nicolas Trotignon, and Kristina Vušković. The (theta, wheel)-free graphs part IV: induced paths and cycles. *Journal of Combinatorial Theory, Series B*, 146:495–531, 2021. doi:10.1016/j.jctb.2020.06.002.

58 Neil Robertson and Paul D. Seymour. Graph minors. XIII. The disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995. doi:10.1006/jctb.1995.1006.

59 D. Rose, R. Tarjan, and G. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5(2):266–283, 1976. doi:10.1137/0205021.

60 Robert Endre Tarjan and Mihalis Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13(3):566–579, 1984. see [61] for addendum. doi:10.1137/0213035.

61 Robert Endre Tarjan and Mihalis Yannakakis. Addendum: Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 14(1):254–255, 1985. doi:10.1137/0214020.

62 Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith–Winograd. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing*, pages 887–898, 2012. doi:10.1145/2213977.2214056.

63 Raphael Yuster and Uri Zwick. Finding even cycles even faster. *SIAM Journal on Discrete Mathematics*, 10(2):209–222, 1997. doi:10.1137/S0895480194274133.