A Faster Algorithm to Recognize Even-Hole-Free Graphs

Hsien-Chih Chang∗ Hsueh-I Lu†

January 14, 2012

Abstract

We study the problem of determining whether an $n$-node $m$-edge graph has an even hole, i.e., an induced simple cycle consisting of an even number of nodes. Conforti, Cornuéjols, Kapoor, and Vušković gave the first polynomial-time algorithm for the problem, which runs in $O(n^{40})$ time. Later, Chudnovsky, Kawarabayashi, and Seymour reduced the running time to $O(n^{31})$. The best previously known algorithm for the problem, due to da Silva and Vušković, runs in $O(n^{19})$ time. In this paper, we solve the problem in time $O(n^{11})$.

1 Introduction

A hole is an induced simple cycle consisting of at least four nodes. A hole is even (respectively, odd) if it consists of an even (respectively, odd) number of nodes. This paper studies the problem of determining whether a graph has even holes. Let $n$ (respectively, $m$) be the number of nodes (respectively, edges) of the input graph. Conforti, Cornuéjols, Kapoor, and Vušković [13, 17] gave the first polynomial-time algorithm for the problem, which runs in $O(n^{40})$ time [6]. Later, Chudnovsky, Kawarabayashi, and Seymour [6] reduced the running time to $O(n^{31})$. Chudnovsky et al. [6] also observed that the running time can be further reduced to $O(n^{15})$ as long as prisms can be detected efficiently, but Maffray and Trotignon [30] showed that detecting prisms is NP-hard. The best previously known algorithm for the problem, due to da Silva and Vušković [21], runs in $O(n^{19})$ time. Based upon new structural results and algorithmic techniques, we solve the problem in $O(n^{11})$ time, as stated in the following theorem.

Theorem 1.1. It takes $O(m^2n^7)$ time to determine if an $n$-node $m$-edge graph has even holes.

Even-hole-free graphs are extensively studied in the literature [1, 14–16, 19–21, 28, 29]. See [40] for a recent survey. Even-hole-free planar graphs [33] can be recognized in $O(n^3)$ time. Determining whether a graph has an even (respectively, odd) hole that contains a particular node is NP-complete [2]. The celebrated strong perfect graph theorem of Chudnovsky, Robertson, Seymour, and Thomas [7] states that a graph $G$ is perfect if and only if neither $G$ nor the complement of $G$ has odd holes. Although perfect graphs can be recognized in $O(n^9)$ time [4], the tractability of recognizing odd-hole-free graphs remains open (see, e.g., [24]). Polynomial-time algorithms for detecting odd holes are known for planar graphs [23], claw-free graphs [27, 37],

∗Email: hcsoso@gmail.com. Department of Computer Science and Information Engineering, National Taiwan University.
†Email: hil@csie.ntu.edu.tw. Web: www.csie.ntu.edu.tw/~hil. Department of Computer Science and Information Engineering, National Taiwan University. This author also holds joint appointments in the Graduate Institute of Networking and Multimedia and the Graduate Institute of Biomedical Electronics and Bioinformatics, National Taiwan University. Address: 1 Roosevelt Road, Section 4, Taipei 106, Taiwan, ROC.
and graphs with bounded clique numbers [18]. Graphs without holes (i.e., chordal graphs) can be recognized in \(O(m + n)\) time [35,36,38]. Graphs without holes consisting of five or more nodes (i.e., weakly chordal graphs) can be recognized in \(O(m^2 + n)\) time [31,32]. It takes \(O(n^2)\) time to detect a hole that contains any \(o((\log n / \log \log n)^{2/3})\) given nodes in an \(O(1)\)-genus graph [25,26].

According to the categorization of Vušković [40, Section 4], our result is a decomposition-based algorithm [4,5,11,12,14,15,17,28,34,37]. (See [6,8] for other approaches.) Our algorithm has two phases. The “cleaning phase” (see Lemma 2.1 and §3) either ensures that the input graph has even holes or produces a collection of graphs such that at least one graph in the set satisfies some “tracking” property (see Property 1) needed in the next phase. The “decomposition phase” (see Lemma 2.2 and §4) is based upon a structural characterization of even-hole-free graphs, which basically says that a graph either belongs to some basic class or can be decomposed into smaller graphs. According to the structural characterization, our algorithm recursively decomposes \(G\) into smaller graphs until each small graph belongs to the basic class. Due to the tracking property ensured by the cleaning phase, the collection of small graphs has even holes if and only if \(G\) has even holes. Detecting even holes in each small graph in the basic class becomes easy.

The rest of the paper is organized as follows. Section 2 gives the framework of our proof which has two phases: (a) Lemma 2.1 summarizes the cleaning phase, (b) Lemma 2.2 summarizes the decomposition phase, and (c) Theorem 1.1 is proved by Lemmas 2.1 and 2.2. Section 3 proves Lemma 2.1. Section 4 proves Lemma 2.2. Section 5 concludes the paper.

2 Framework of our proof

Unless clearly specified otherwise, all graphs throughout the paper are simple. Let \(|S|\) be the cardinality of set \(S\). Let \(V(G)\) consist of the nodes in graph \(G\). Let \(H\) be a subgraph of \(G\). Let \(N_H(v)\) consist of the neighbors in \(H\) of node \(v\) in \(G\). Therefore, \(N_H(v) = N_G(v) \cap V(H)\). Let \(G[H]\) be the subgraph of \(G\) induced by \(V(H)\). Subgraphs \(H\) and \(H'\) of graph \(G\) are adjacent in \(G\) if some node of \(H\) and some node of \(H'\) are adjacent in \(G\). Subgraphs \(H\) and \(H'\) of graph \(G\) are separate in \(G\) if \(H\) and \(H'\) are disjoint and not adjacent in \(G\). For any set \(S\) of nodes, let \(G \setminus S = G[V(G) \setminus S]\).

Let \(C\) be a hole of \(G\). A node \(x \in V(G) \setminus V(C)\) is a major node [6] of \(C\) in \(G\) if at least three nodes in \(N_C(x)\) are independent in \(G\). Let \(M_G(C)\) consist of the major nodes of \(C\) in \(G\). For any node \(x \in V(G) \setminus V(C)\) that is adjacent to \(C\) in \(G\), one can verify that \(x\) is not a major node of \(C\) in \(G\) if and only if (1) \(G[N_C(x)]\) has exactly one connected component, which has at most four nodes or (2) \(G[N_C(x)]\) has exactly two connected components, each of which has at most two nodes. Let \(N^i_j(C)\) with \(1 \leq i \leq 4\) consist of the nodes \(x \in V(G) \setminus V(C)\) that are adjacent to \(C\) such that \(G[N_C(x)]\) has exactly one connected component, which has \(i\) nodes. Let \(N^{i,j}_{G}(C)\) with \(1 \leq i \leq j \leq 2\) consist of the nodes \(x \in V(G) \setminus V(C)\) that are adjacent to \(C\) such that \(G[N_C(x)]\) has exactly two connected components, one of which has \(i\) nodes and the other of which has \(j\) nodes. Therefore, for each hole \(C\) of \(G\), the nodes adjacent to \(C\) in \(G\) can be partitioned into the following eight disjoint sets: \(N^1_G(C), N^2_G(C), N^3_G(C), N^4_G(C), N^{1,1}_G(C), N^{1,2}_G(C), N^{2,2}_G(C),\) and \(M_G(C)\).

We say that \((H, u_1, u_2, u_3)\) is a quadruple of graph \(G\) if \(H\) is an induced subgraph of \(G\) with \(\{u_1, u_2, u_3\} \subseteq V(H)\). Hole \(C\) of \(G\) is clean if \(M_G(C) = N^{2,2}_G(C) = \emptyset\). A triple \((u_1, u_2, u_3)\) of nodes is a tracker of hole \(C\) if \((u_1, u_2, u_3)\) is a path of \(C\). Quadruple \((H, u_1, u_2, u_3)\) of \(G\) is tracking if \(H\) has a clean shortest even hole \(C\) such that \((u_1, u_2, u_3)\) is a tracker of \(C\). Below is a property of a set \(Q\) of quadruples of graph \(G\):
Property 1. Graph $G$ has even holes if and only if set $Q$ contains tracking quadruples of $G$.

The subgraph of $G$ induced by four distinct nodes $b_1, \ldots, b_4$ is a diamond [17, 28] of $G$ if $G[(b_1, \ldots, b_4)]$ consists of a four-node cycle $(b_1, \ldots, b_4)$ and an edge $(b_2, b_4)$. An induced subgraph of $G$ is a beetle of $G$ if it consists of (1) a diamond induced by a sequence $b_1, \ldots, b_4$ of four nodes and (2) an induced tree $T$ with three leaves $b_1, b_2$, and $b_3$ such that $T \setminus \{b_1, b_2, b_3\}$ and $b_4$ are separate in $G$. One can verify that a graph has even holes if it has beetles.

Lemma 2.1. It takes $O(m^2n^7)$ time to complete either one of the following tasks for any $n$-node $m$-edge graph $G$. Task 1: Ensuring that $G$ has even holes. Task 2: (a) Ensuring that $G$ has no beetles and (b) obtaining a set $Q$ of $O(mn^3)$ quadruples of $G$ that satisfies Property 1.

Lemma 2.2. Given (a) an $n$-node $m$-edge graph $G$ that has no beetles and (b) a set $Q$ of quadruples of $G$ that satisfies Property 1, it takes $O(mn^3 \cdot |Q|)$ time to determine whether $G$ has even holes.

Proof of Theorem 1.1. We apply Lemma 2.1 on the input $n$-node $m$-edge graph $G$ in $O(m^2n^7)$ time. If $G$ is ensured to have even holes, then the theorem is proved. Otherwise, $G$ has no beetles and we have a set $Q$ of $O(mn^3)$ quadruples of $G$ that satisfies Property 1. By Lemma 2.2, one can determine whether $G$ has even holes in time $O(mn^3 \cdot |Q|) = O(m^2n^6)$. The theorem is proved.

3 The cleaning phase

This section proves Lemma 2.1. The idea is to generate a set of quadruples of $G$ such that one of them is a tracking quadruple. Specifically, after determining whether $G$ has beetles, we construct polynomially many node sets according to the structure of a shortest even hole $C$. One of the sets $S$ contains all the “bad nodes” $M_G(C) \cup N^{2,2}_G(C)$ and $S$ is disjoint from $C$. By removing $S$ from $G$ there is a clean shortest even hole in $G$. Also we guess the position of a tracker of $C$ by trying all possible combinations of three nodes. By combining the guesses of $S$ and of the trackers, we can reduce the number of quadruples generated. Finally we only have to prove that the set of quadruples satisfies Property 1.

A $k$-hole is a $k$-node hole. A clique of $G$ is a complete subgraph of $G$. A clique of $G$ is maximal if it is not contained by other cliques of $G$. We need the following four lemmas.

Lemma 3.1 (Chudnovsky and Seymour [8, 9]). Given a set $S$ of three nodes of an $n$-node graph $G$, it takes $O(n^4)$ time to determine whether $G$ has an induced subtree that contains $S$.

Lemma 3.2 (Farber [22, Proposition 2], da Silva and Vušković [20, Section 2]). If an $n$-node $m$-edge graph $G$ has no 4-holes, then (a) $G$ has $O(n^2)$ maximal cliques, and (b) it takes $O(mn^2)$ time to either ensure that $G$ has even holes or compute the maximal cliques of $G$.

Lemma 3.3 (Chudnovsky et al. [6, Lemma 4.2]). If $C$ is a shortest even hole of a graph $G$ that has no 4-holes, then $N^{2,2}_G(C) \subseteq N_G(u) \cap N_G(v)$ holds for some edge $(u, v)$ of $C$.

Lemma 3.4. For any shortest even hole $C$ of a graph $G$ that has no 4-holes, if $G[M_G(C)]$ is not a clique of $G$, then $M_G(C) \subseteq N_G(u)$ holds for some node $u$ of $C$.

We first prove Lemma 2.1 using Lemmas 3.1, 3.2, 3.3, and 3.4.

Proof of Lemma 2.1. We first show that it takes $O(m^2n^7)$ time to determine whether $G$ has beetles. If there is a sequence $b_1, \ldots, b_7$ of nodes such that (a) the subgraph induced by $b_1, \ldots, b_4$ is a diamond of $G$, (b) edges $(b_1, b_5), (b_2, b_6)$, and $(b_3, b_7)$ are in $G$, and (c) $\{b_5, b_6, b_7\}$ is contained
by an induced tree in $G \setminus S$ with $S = (N_G(b_1) \cup \cdots \cup N_G(b_4)) \setminus \{b_5, b_6, b_7\}$, then output that $G$ has beetles; otherwise, output that $G$ has no beetles. By Lemma 3.1, the procedure takes $O(m^2n^7)$ time. To see the correctness, consider the case that the subgraph induced by $b_1, \ldots, b_4$ is a diamond of $G$ and $G$ has edges $(b_1, b_2), (b_2, b_3), (b_3, b_4)$. If some induced tree $T'$ of $G \setminus S$ contains $\{b_5, b_6, b_7\}$, then $T = G[T' \cup \{b_1, b_2, b_3\}] \setminus \{(b_1, b_2), (b_2, b_3)\}$ is an induced tree of $G$ such that (a) $b_1, b_2,$ and $b_3$ are leaves of $T$ and (c) $T \setminus \{b_1, b_2, b_3\}$ and $\{b_4\}$ are separate in $G$, implying that $G[\{b_1, \ldots, b_4\}] \cap T$ is a beetle of $G$. If $\{b_5, b_6, b_7\}$ is not contained by any induced tree of $G \setminus S$, then $G$ has no beetles containing the diamond of $G$ induced by $b_1, \ldots, b_4$. Therefore, it takes $O(m^2n^7)$ time to determine whether $G$ has beetles. We also spend $O(n^4)$ time to determine whether $G$ has 4-holes. If $G$ has 4-holes or beetles, then $G$ has even holes. The lemma is proved. The rest of the proof assumes that $G$ has no 4-holes and beetles.

We next apply Lemma 3.2 in $O(mn^2)$ time to either ensure that $G$ has even holes or compute the $O(n^2)$ maximal cliques of $G$. If $G$ is ensured to have even holes, then the lemma is proved. Otherwise, let $Q_1$ consist of the quadruples of $G$ of the form $(G \setminus S_1, u_1, u_2, u_3)$ with

$$S_1 = S_1(v_1, v_2, u_1, u_2, u_3) = (N_G(v_1) \cap N_G(v_2)) \cup (N_G(u_2) \setminus \{u_1, u_3\}),$$

where $(u_1, u_2)$ and $(v_1, v_2)$ are edges of $G$ and $u_3$ is a node of $G$. Let $Q_2$ consist of the quadruples of $G$ of the form $(G \setminus S_2, u_1, u_2, u_3)$ with

$$S_2 = S_2(K, u_1, u_2) = V(K) \cup (N_G(u_1) \cap N_G(u_2)),$$

where $K$ is a maximal clique of $G$, $(u_1, u_2)$ is an edge of $G$, and $u_3$ is a node of $G$. Let $Q = Q_1 \cup Q_2$. We have $|Q| \leq |Q_1| + |Q_2| = O(mn^2)$. Since the maximal cliques of $G$ are available, $Q$ can be computed in $O(mn^3 \cdot O(n)$ time. It remains to show that $Q$ satisfies Property 1. If $Q$ has a tracking quadruple $(H, u_1, u_2, u_3)$ of $G$, then the induced subgraph $H$ of $G$ has even holes, implying that $G$ has even holes. To see the only-if direction of Property 1, suppose that $G$ has even holes. Let $C$ be a shortest even hole of $G$. One can verify that if node set $S$ satisfies $M_G(C) \cup N_{G}^2(C) \subseteq S \subseteq V(G)$ and $V(C) \cap S = \emptyset$, then $C$ is a clean hole of $G \setminus S$.

**Case 1:** $M_G(C) \subseteq N_G(u_2)$ holds for some node $u_2$ of $C$. Let $u_1$ and $u_3$ be the neighbors of $u_2$ in $C$. By $\{u_1, u_3\} \subseteq V(C)$ and $M_G(C) \cap V(C) = \emptyset$, we have $M_G(C) \subseteq N_G(u_2) \setminus \{u_1, u_3\}$. By Lemma 3.3, $M_G(C) \cup N_{G}^2(C) \subseteq S_1$ holds for edges $(v_1, v_2)$ and $(u_1, u_2)$ of $G$ and node $u_3$ of $G$. By the choices of $u_1$ and $u_3$, we have $(N_G(u_2) \setminus \{u_1, u_3\}) \cap V(C) = \emptyset$. Since $C$ is an even hole of $G$ and $(v_1, v_2)$ is an edge of $C$, we have $N_G(v_1) \cap N_G(v_2) \cap V(C) = \emptyset$. Therefore, $S_1 \cap V(C) = \emptyset$. Let $H = G \setminus S_1$. We have that (a) $(H, u_1, u_2, u_3)$ is a quadruple of $G$, (b) $C$ is a clean even hole of $H$, and (c) $(u_1, u_2, u_3)$ is a path of $C$ which implies that $(u_1, u_2, u_3)$ is a tracker of $C$. Therefore $(H, u_1, u_2, u_3)$ is a tracking quadruple of $G$ in $Q_1$.

**Case 2:** $M_G(C) \not\subseteq N_G(u)$ holds for all nodes $u$ of $C$. By Lemma 3.4, $G[M_G(C)]$ is a clique of $G$. Let $K$ be a maximal clique of $G$ with $M_G(C) \subseteq V(K)$. By Lemma 3.3, $M_G(C) \cup N_{G}^2(C) \subseteq S_2$ holds for edges $(u_1, u_2)$ of $G$. We have $V(K) \cap V(C) = \emptyset$, since otherwise by $M_G(C) \cap V(C) = \emptyset$, each node $u$ in $V(K) \cap V(C)$ would have $M_G(C) \subseteq V(K) \setminus \{u\} \subseteq N_G(u)$, a contradiction. It follows from $(N_G(u_1) \cap N_G(u_2)) \cap V(C) = \emptyset$ that $S_2 \cap V(C) = \emptyset$. Let $H = G \setminus S_2$. Let $u_3$ be the neighbor of $u_2$ in $C$ other than $u_1$. We know that (a) $(H, u_1, u_2, u_3)$ is a quadruple of $G$, (b) $C$ is a clean hole of $H$, and (c) $(u_1, u_2, u_3)$ is a path of $C$ which implies that $(u_1, u_2, u_3)$ is a tracker of $C$. Therefore $(H, u_1, u_2, u_3)$ is a tracking quadruple of $G$ in $Q_2$. The lemma is proved.  

The rest of the section proves Lemma 3.4. Let $int(P)$ of path $P$ be $P \setminus \{u, v\}$, where $u$ and $v$ are the endpoints of $P$. For each hole $C$ of graph $G$, there are two orders for the nodes of $C$ around $C$. Let an arbitrary but fixed one of the two orders be the clockwise order of nodes around $C$. For any two distinct nodes $u_1$ and $u_2$ of $C$, let $C(u_1, u_2)$ be the path of $C$ from $u_1$ to $u_2$ in clockwise order around $C$. An ordered pair $(u_1, u_2)$ of two adjacent nodes $u_1$ and $u_2$ in
C is a gate [6] of C in graph G for an ordered pair \( \langle x_1, x_2 \rangle \) of two distinct nodes \( x_1 \) and \( x_2 \) in \( M_G(C) \) if the following conditions hold:

**Condition G1:** \( C(u_1, u_2) \) equals edge \( (u_1, u_2) \).

**Condition G2:** \( G \) contains edges \( (u_1, x_2) \) and \( (u_2, x_1) \).

**Condition G3:** \( G \) contains at least one of edges \( (u_1, x_1) \) and \( (u_2, x_2) \).

**Condition G4:** There is a node \( u_3 \) in \( int(C(u_2, u_1)) \) such that \( x_1 \) (respectively, \( x_2 \)) is not adjacent to \( int(C(u_2, u_3)) \) (respectively, \( int(C(u_3, u_1)) \)) in \( G \).

We need the following two lemmas to prove Lemma 3.4.

**Lemma 3.5** (Chudnovsky et al. [6, Lemmas 2.3 and 2.4]). The following statements hold for any shortest even hole of a graph \( G \) that has no 4-holes.

1. If \( x_1 \) and \( x_2 \) are two distinct nodes of \( M_G(C) \) that are not adjacent in \( G \), then \( G \) has a gate of \( C \) for \( \langle x_1, x_2 \rangle \).
2. If \( x_1, x_2, \) and \( x_3 \) are three distinct nodes of \( M_G(C) \) such that \( G[\{x_1, x_2, x_3\}] \) has at most one edge, then \( \{x_1, x_2, x_3\} \subseteq N_G(u) \) holds for some node \( u \) of \( C \).

**Lemma 3.6.** Let \( C \) be a shortest even hole of a graph \( G \) that has no 4-holes. If \( x_1, x_2, \) and \( x_3 \) are three distinct nodes of \( M_G(C) \) such that \( G[\{x_1, x_2, x_3\}] \) has at most two edges, then \( \{x_1, x_2, x_3\} \subseteq N_G(u) \) holds for some node \( u \) of \( C \).

Proof. By Lemma 3.5(2), it suffices to prove the lemma for the case that \( G[\{x_1, x_2, x_3\}] \) has exactly two edges, say, \( (x_1, x_3) \) and \( (x_2, x_3) \). Since \( x_1 \) and \( x_2 \) are not adjacent in \( G \), Lemma 3.5(1) ensures that \( G \) has a gate \( \langle u_1, u_2 \rangle \) of \( C \) for \( \langle x_1, x_2 \rangle \). By Conditions G2 and G3 of gate \( \langle u_1, u_2 \rangle \), \( N_G(x_1) \cap N_G(x_2) \cap V(C) \neq \emptyset \). Let \( u \) be an arbitrary node in \( N_G(x_1) \cap N_G(x_2) \cap V(C) \). We know that \( u \) is adjacent to \( x_3 \) in \( G \), since otherwise \( (x_1, u, x_2, x_3) \) would be a 4-hole of \( G \), a contradiction. Therefore, \( u \) is a node of \( C \) in \( N_G(x_1) \cap N_G(x_2) \cap N_G(x_3) \). The lemma is proved.

We are ready to prove Lemma 3.4 using Lemmas 3.5 and 3.6.

Proof of Lemma 3.4. Since \( G[M_G(C)] \) is not a clique, there are two distinct nodes \( x_1 \) and \( x_2 \) of \( M_G(C) \) that are not adjacent in \( G \). By Lemma 3.5(1), \( G \) has a gate \( \langle u_1, u_2 \rangle \) of \( C \) for \( \langle x_1, x_2 \rangle \). We first show

\[
N_G(x_1) \cap N_G(x_2) \subseteq \{u_1, u_2\}. \tag{1}
\]

As ensured by Condition G4 of gate \( \langle u_1, u_2 \rangle \), let \( u_3 \) be a node in \( int(C(u_2, u_1)) \) such that \( x_1 \) is not adjacent to \( int(C(u_2, u_3)) \) in \( G \) and \( x_2 \) is not adjacent to \( int(C(u_3, u_1)) \) in \( G \). By Condition G1 of gate \( \langle u_1, u_2 \rangle \), \( N_G(x_1) \cap N_G(x_2) \subseteq \{u_1, u_2, u_3\} \). We know \( u_3 \notin N_G(u_1) \cap N_G(u_2) \), since otherwise \( G[\{x_1, x_2, u_1, u_3\}] \) or \( G[\{x_1, x_2, u_2, u_3\}] \) would be a 4-hole of \( G \) by Conditions G2 and G3, a contradiction. Therefore, Equation (1) holds. We next show

\[
M_G(C) \subseteq N_G(u_1) \cup N_G(u_2). \tag{2}
\]

If \( M_G(C) = \{x_1, x_2\} \), then \( M_G(C) \subseteq N_G(u_1) \cup N_G(u_2) \) follows from Condition G2 of gate \( \langle u_1, u_2 \rangle \). If \( x_3 \) is a node of \( M_G(C) \) other than \( x_1 \) and \( x_2 \), then Lemma 3.6 ensures \( \{x_1, x_2, x_3\} \subseteq N_G(u) \) for some node \( u \) of \( C \). By Equation (1), \( u \in N_G(x_1) \cap N_G(x_2) \subseteq \{u_1, u_2\} \), implying \( x_3 \in N_G(u_1) \cup N_G(u_2) \). Therefore, Equation (2) holds.

We prove the lemma by showing \( M_G(C) \subseteq N_G(u_1) \) or \( M_G(C) \subseteq N_G(u_2) \). If \( x_1 \) and \( u_1 \) are not adjacent in \( G \), then Equation (1) implies \( N_G(x_1) \cap N_G(x_2) \subseteq \{u_2\} \). For any node \( x_3 \) other than \( x_1 \) and \( x_2 \) of \( M_G(C) \), Lemma 3.6 ensures that \( \{x_1, x_2, x_3\} \subseteq N_G(u) \) holds for some node
u of C. Therefore, \( u \in N_G(x_1) \cap N_G(x_2) \subseteq \{u_2\} \), implying \( M_G(C) \subseteq N_G(u_2) \). The lemma is proved. If \( x_2 \) and \( u_2 \) are not adjacent in \( G \), then Equation (1) implies \( N_G(x_1) \cap N_G(x_2) \subseteq \{u_1\} \). For any node \( x_3 \) other than \( x_1 \) and \( x_2 \) of \( M_G(C) \), Lemma 3.6 ensures that \( \{x_1, x_2, x_3\} \not\subseteq N_G(u) \) holds for some node \( u \) of \( C \). Therefore, \( u \in N_G(x_1) \cap N_G(x_2) \subseteq \{u_1\} \), implying \( M_G(C) \subseteq N_G(u_1) \). The lemma is proved. The rest of the proof assumes that \( G \) has edges \( \{x_1, u_1\} \) and \( \{x_2, u_2\} \). Assume for contradiction that there are two distinct nodes \( x_3 \) and \( x_4 \) of \( M_G(C) \) with \( x_3 \not\notin N_G(u_1) \) and \( x_4 \not\notin N_G(u_2) \). Since \( G \) has edges \( \{x_1, u_1\} \) and \( \{x_2, u_2\} \), we know that \( x_1, x_2, x_3, \) and \( x_4 \) are four distinct nodes of \( M_G(C) \). By \( x_3 \not\in N_G(u_1) \), we have \( x_3 \not\in N_G(x_1) \cap N_G(x_2) \), since otherwise \( G[\{x_1, x_2, x_3, u_1\}] \) would be a 4-hole of \( G \). Let \( x \) be a node in \( \{x_1, x_2\} \) with \( x \not\in N_G(x_3) \). By Lemma 3.6, \( \{x, x_3, x_4\} \subseteq N_G(u) \) holds for some node \( u \) of \( C \). By definitions of \( x_3 \) and \( x_4 \), we have \( u \not\in \{u_1, u_2\} \). By \( x_3 \not\in N_G(u_1) \) and Equation (2), \( x_3 \in N_G(u_2) \). Thus, \( \{x, u_2, x_3, u\} \) is a cycle of \( G \). Since \( x \not\in N_G(x_3) \) and \( G[\{x, u_2, x_3, u\}] \) cannot be a 4-hole of \( G \), we have \( u_2 \in N_G(u) \). Therefore, \( \{u_1, u_2, u, x_4\} \) is a cycle of \( G \). Since \( u_1 \) and \( u \) are the two distinct neighbors of \( u_2 \) in hole \( C \), we have \( u \not\in N_G(u_1) \). By \( u_2 \not\in N_G(x_4) \), \( G[\{u_1, u_2, u, x_4\}] \) is a 4-hole of \( G \), a contradiction. The lemma is proved.

\section{The decomposition phase}

This section proves Lemma 2.2. First we give an overview to the decomposition phase. As mentioned in the introduction, the decomposition process is guided by a structural characterization of even-hole-free graphs. This is the technique used by Conforti, Cornuèjols, Kapoor, and Vušković [16, 17]. They provided a complex structural theorem for even-hole-free graphs, together with the first known polynomial-time algorithm. A much simpler characterization is proved by da Silva and Vušković [21], which is also the one used in this paper. (See Lemma 4.9 in §4.2.) One of the decompositions used in the structural result is defined as follows. For any node \( s \) of graph \( H \), let \( N_H(s) = N_H(s) \cup \{s\} \). Subset \( S \) of \( V(H) \) is a star-cut\(^1\) of \( H \) if \( H \setminus S \) is disconnected and \( S \subseteq N_H(s) \) holds for some node \( s \) of \( S \).

\begin{lemma}
Given (a) an \( n \)-node \( m \)-edge graph \( G \) having no beetles and (b) a quadruple \((H, u_1, u_2, u_3)\) of \( G \), it takes \( O(mn^3) \) time to complete one of the following three tasks. Task 1: Ensuring that \( G \) has even holes. Task 2: Ensuring that \((H, u_1, u_2, u_3)\) is not a tracking quadruple of \( G \). Task 3: Obtaining an \( O(n) \)-node \( O(m) \)-edge graph \( H' \) that satisfies the following conditions.

\begin{itemize}
\item Condition S1: \( H' \) has no star-cuts.
\item Condition S2: If \( H' \) has even holes, then \( H \) has even holes.
\item Condition S3: If \( H' \) has no even holes, then \((H, u_1, u_2, u_3)\) is not a tracking quadruple of \( G \).
\end{itemize}

\begin{lemma}
Given an \( n \)-node \( m \)-edge graph \( H \) that has no star-cuts, it takes \( O(mn^3) \) time to determine whether \( H \) has even holes.
\end{lemma}

We first prove Lemma 2.2 using Lemmas 4.1 and 4.2.

\begin{proof}[Proof of Lemma 2.2]
We first apply Lemma 4.1 on each quadruple of \( Q \) in overall \( O(mn^3 \cdot |Q|) \) time. If \( G \) is ensured to have even holes via Task 1 for some quadruple of \( Q \), then the lemma is proved. Let \( H \) consist of the graphs \( H' \) obtained from the quadruples \((H, u_1, u_2, u_3)\) of \( Q \) via Task 3. If \( H = \emptyset \), then Task 2 ensures that \( Q \) has no tracking quadruples. By Property 1 of \( Q \), \( G \) has no even holes, and the lemma is proved. It remains to consider the case that \( H \neq \emptyset \). By Condition S1, each graph \( H' \) of \( H \) has \( O(n) \) nodes and \( O(m) \) edges and has no star-cuts. By Condition S2, if a graph \( H' \) of \( H \) has even holes, then \( G \) has even holes since the induced

\(^1\)Star-cuts are called \textit{star-cutsets} in the literature [10].
subgraph $H$ of $G$ has. By Condition S3, if all graphs $H'$ of $\mathcal{H}$ have no even holes, then the given $Q$ has no tracking quadruples, which follows from Property 1 of $Q$ that $G$ has no even holes. In summary, $\mathcal{H}$ is a non-empty set of star-cut-free graphs such that $\mathcal{H}$ contains a graph that has even holes if and only if $G$ has even holes. By Lemma 4.2, whether $\mathcal{H}$ contains a graph that has even holes can be determined in overall time $O(mn^3 \cdot |\mathcal{H}|) = O(mn^3 \cdot |Q|)$. The lemma is proved.

The rest of the section proves Lemmas 4.1 and 4.2.

### 4.1 Star-cuts

This subsection proves Lemma 4.1. The main problem of decomposing a star-cut is that the existence of even holes may not be preserved during the decompositions. However, a clean even hole will remain intact (see Lemma 4.3 and proof of Lemma 4.1). Another problem is that even if we successfully decompose a star-cut, the whole decomposition process may take too much time, since the number of small graphs constructed may be exponentially large. Conforti et al. [17, Lemma 3.18] resolved the problem by refining the blocks. This guarantees that the total number of small graphs constructed is polynomial. While this is a solution, some side effects are introduced. First, it becomes much harder to prove that a small graph preserves clean even holes. Second, after the decomposition, instead of producing one graph, a whole family of graphs are generated, and this significantly increases the overall time complexity of the decomposition phase. In fact, all the decomposition-based algorithms using star-cuts suffer from this problem. We overcome this problem by using trackers. During the decomposition, we decide the position of a clean shortest even hole by a tracker of $C$ (see Lemma 4.4). Therefore we can apply the remaining decomposition procedure on a specific small graph and discard all the others.

Node $x$ dominates [17] node $y$ in graph $H$ if $x \neq y$ and $N_H[y] \subseteq N_H[x]$. Node $y$ is dominated in $H$ if some node of $H$ dominates $y$ in $H$. A star-cut $S$ of graph $H$ is full if $S = N_H[s]$ holds for some node $s$ of $S$. Star-cuts are hard to find and there are no known polynomial-time algorithms\(^2\). By removing all the dominated nodes, star-cuts become full star-cuts (see Lemma 4.5). Full star-cuts can be detected in $O(mn^2)$ time in an $n$-node $m$-edge graph. We need the following three lemmas.

**Lemma 4.3.** If $(H, u_1, u_2, u_3)$ is a quadruple of an $n$-node $m$-edge graph $G$ that has no beetles, then it takes $O(mn^2)$ time to obtain a quadruple $(H', u'_1, u'_2, u'_3)$ of $G$ that satisfies the following conditions.\(^3\)

- **Condition D1:** $H'$ has no dominated nodes.
- **Condition D2:** $H'$ is an induced subgraph of $H$.
- **Condition D3:** if $(H, u_1, u_2, u_3)$ is a tracking quadruple of $G$, then so is $(H', u'_1, u'_2, u'_3)$.

**Lemma 4.4.** If $(H, u_1, u_2, u_3)$ is a tracking quadruple of a graph $G$ that has no beetles and $S$ is a full star-cut in $H$, then one of the following conditions holds.

- **Condition B1:** There are two distinct nodes $s_1$ and $s_2$ of $S$ and two distinct connected components $B_1$ and $B_2$ of $H \setminus S$ such that (a) $s_1$ and $s_2$ are not adjacent in $H$ and (b) both $s_1$ and $s_2$ are adjacent to both $B_1$ and $B_2$ in $H$.

- **Condition B2:** (a) If $\{u_1, u_2, u_3\} \subseteq S$, then there is a unique connected component $B$ of $H \setminus S$ such that both $u_1$ and $u_3$ are adjacent to $B$ in $H$. (b) If $\{u_1, u_2, u_3\} \nsubseteq S$, then there is a unique connected component $B$ of $H \setminus S$ such that $\{u_1, u_2, u_3\} \subseteq V(B) \cup S$.

---

\(^2\)Conforti et al. [17, before Definition 1.10].

\(^3\)A similar lemma with stronger requirements can be found in Conforti et al. [17, Lemma 1.14].
Moreover, if Condition B2 holds, then \((H[V(B) \cup S], u_1, u_2, u_3)\) is a tracking quadruple of \(G\).

**Lemma 4.5** (Conforti et al. [17, Lemmas 1.13 and 1.14]). Let \(G\) be a graph that has no dominated nodes. If \(G\) has star-cuts, then \(G\) has full star-cuts.

We first prove Lemma 4.1 using Lemmas 4.3, 4.4, and 4.5.

**Proof of Lemma 4.1.** Let \(Q_0\) be the initial given quadruple \((H, u_1, u_2, u_3)\) of \(G\). Our algorithm iteratively updates \(H\), \(u_1\), \(u_2\), and \(u_3\) by the following steps until one of the three tasks is completed.

**Step 1:** Let \((H', u'_1, u'_2, u'_3)\) be a quadruple of \(G\) obtained from \((H, u_1, u_2, u_3)\) as ensured by Lemma 4.3. If \(H'\) has no full star-cuts, then complete Task 3 by outputting \(H'\). Otherwise, let \((H, u_1, u_2, u_3) = (H', u'_1, u'_2, u'_3)\) and proceed to Step 2.

**Step 2:** Let \(S\) be an arbitrary full star-cut of \(H\). If there are two distinct nodes \(s_1\) and \(s_2\) of \(S\) and two distinct connected components \(B_1\) and \(B_2\) of \(H \setminus S\) such that (a) \(s_1\) and \(s_2\) are not adjacent in \(H\) and (b) both \(s_1\) and \(s_2\) are adjacent to both \(B_1\) and \(B_2\) in \(H\), then complete Task 1 by outputting that \(G\) has even holes. Otherwise, proceed to Step 3.

**Step 3:** If \(\{u_1, u_2, u_3\} \subseteq S\), determine if there is a unique connected component \(B\) of \(H \setminus S\) satisfying that both \(u_1\) and \(u_2\) are adjacent to \(B\) in \(H\); if \(\{u_1, u_2, u_3\} \not\subseteq S\), determine if there is a unique connected component \(B\) of \(H \setminus S\) satisfying that \(\{u_1, u_2, u_3\} \subseteq V(B) \cup S\). In either case, if there is such a unique \(B\), then replace \(H\) with \(H[V(B) \cup S]\) and proceed to the next iteration of the loop. Otherwise, complete Task 2 by outputting that \(Q_0\) is not a tracking quadruple of \(G\).

We first show that if \(Q_0\) is a tracking quadruple of \(G\), then each intermediate \(Q = (H, u_1, u_2, u_3)\) throughout the loop remains a tracking quadruple of \(G\). The statement can be proved by verifying the following claim: *Each update to \(Q\), either in Step 1 or 3, does not affect the condition that \(Q\) is a tracking quadruple of \(G\).* If \(Q\) is updated by Step 1, then the claim follows from Condition D3 of Lemma 4.3. If \(Q\) is updated by Step 3, then it follows from the definition of Step 2 that Condition B1 of Lemma 4.4 does not hold. Therefore, Condition B2 of Lemma 4.4 holds, and the claim follows from Lemma 4.4.

We next ensure the correctness. *Case 1:* Step 1 outputs \(H'\). By Condition D1 of Lemma 4.3, we know that \(H'\) has no dominated nodes. Since \(H'\) has no full star-cuts, Lemma 4.5 ensures that \(H'\) has no star-cuts. Condition S1 holds. Condition S2 follows from Condition D2 of Lemma 4.3. Condition S3 follows from Condition D3 of Lemma 4.3. Therefore, Task 3 is indeed completed. *Case 2:* Step 2 outputs that \(G\) has even holes. Let \(P_1\) be a shortest path between \(s_1\) and \(s_2\) in \(H[V(B_1) \cup \{s_1, s_2\}]\). Let \(P_2\) be a shortest path between \(s_1\) and \(s_2\) in \(H[V(B_2) \cup \{s_1, s_2\}]\). One can verify that at least one of the three cycles of graph \(P_1 \cup P_2 \cup \{(s, s_1), (s, s_2)\}\) is an even hole of \(H\), implying that \(G\) has even holes. Therefore, Task 1 is indeed completed. *Case 3:* Step 3 outputs that \(Q_0\) is not a tracking quadruple of \(G\). By definitions of Steps 2 and 3, both Conditions B1 and B2 of Lemma 4.4 do not hold. Lemma 4.4 and the above claim ensure that \(Q_0\) is not a tracking quadruple of \(G\). Therefore, Task 2 is indeed completed.

By Condition D2 of Lemma 4.3, Step 1 does not increase the number of nodes of \(H\). If Step 3 updates \(H\), the number of nodes in \(H\) is decreased by at least one since \(S\) is a cut. Therefore, the loop has \(O(n)\) iterations. By Lemma 4.3 and the fact that detecting full star-cuts in \(H'\) takes \(O(mn)\) time, Step 1 takes \(O(mn^2)\) time. Step 2 takes \(O(mn^2)\) time: For each of the \(O(n^2)\) pairs of nonadjacent nodes \(s_1\) and \(s_2\) in \(S\), it takes \(O(m)\) time to determine whether they have two distinct common neighboring connected components of \(H \setminus S\). Step 3 takes \(O(m)\) time. Therefore, the overall running time is \(O(mn^2)\). The lemma is proved.

The rest of the subsection proves Lemmas 4.3 and 4.4.
4.1.1 Proving Lemma 4.3

We need the following two lemmas to prove Lemma 4.8, which is used by the proof of Lemma 4.3.

Lemma 4.6 (Chudnovsky et al. [6, Lemma 2.2]). Let \( C \) be a shortest even hole of graph \( G \). If \( v \) is a major node of \( C \) in \( G \), then \( |N_C(v)| \) is even.

Lemma 4.7. If \( C \) is a shortest even hole of a graph \( G \) that has no beetles, then each node of \( V(G) \setminus V(C) \) that is adjacent to \( C \) in \( G \) belongs to exactly one of the following disjoint sets: \( N_G^{1,1}(C) \), \( N_G^{2,2}(C) \), \( N_G^{1,2}(C) \), \( N_G^{2,1}(C) \), and \( M_G(C) \).

Proof. Let \( x \) be a node of \( V(G) \setminus V(C) \) that is adjacent to \( C \) in \( G \). Assume that \( x \) is not in \( M_G(C) \). We prove that \( x \) is in one of the other five sets. Since \( x \) is not a major node of \( C \), any independent set in \( G[N_C(x)] \) has size at most 2. Thus, one of the following conditions holds. Condition 1: \( G[N_C(x)] \) has exactly one connected component, which has at most four nodes. Condition 2: \( G[N_C(x)] \) has exactly two connected components, each of which has at most two nodes. If Condition 1 holds, then \( x \) is in one of the sets \( N_G^i(C) \) for \( 1 \leq i \leq 4 \). If \( |N_C(v)| = 2 \), then \( G[C \cup \{x\}] \) is a hole in \( G \), a contradiction. Thus, \( x \) is in one of the following sets: \( N_G^{1,1}(C) \), \( N_G^{2,2}(C) \), and \( N_G^{1,2}(C) \). If Condition 2 holds, then \( x \) is in one of the following sets: \( N_G^{1,1}(C) \), \( N_G^{2,2}(C) \), and \( N_G^{1,2}(C) \). If \( x \) is in \( N_G^{1,2}(C) \), let the nodes \( N_C(x) = \{u_1, u_2, u_3\} \) form the order of \( u_1, u_2, u_3 \) in \( C \). One of the induced cycles \( G[N_C(x)] \) contains a connected component of size at least 3. Therefore, \( x \) belongs to \( N_G^3(C) \). This implies that \( (u, x, w) \) is a path in \( G \) and \( x \) has no other neighbors in \( C \). Thus, \( C' \) is a shortest even hole of \( G \).

Next we show that \( M_G(C') = N_G^{2,2}(C') = \emptyset \), which implies that \( C' \) is clean in \( G \). Assume for contradiction that there is a node \( z \) in \( M_G(C') \setminus N_G^{2,2}(C') \). (Note that \( z \neq v \) since \( v \) is in \( N_G^2(C') \).) Since \( C \setminus \{v\} = C' \setminus \{x\} \), if \( z \) is adjacent to both \( x \) and \( v \) or none of them, then \( M_G(C') = M_G(C) \) and \( N_G^{2,2}(C') = N_G^{2,2}(C) \). Since \( C \) is clean in \( G \), \( M_G(C') = N_G^{2,2}(C') = \emptyset \). This implies that \( M_G(C') = N_G^{2,2}(C') = \emptyset \), contradicting the previous assumption. Therefore \( z \) is adjacent to exactly one of the nodes in \( v, x \). Case 1: \( z \in M_G(C') \). By Lemma 4.6, \( |N_{C'}(z)| \geq 4 \). Since \( M_G(C) = N_G^{2,2}(C) = \emptyset \), \( M_G(C') = N_G^{2,2}(C') \). Assume that \( z \) is adjacent to \( v \) but not to \( x \). Then \( |N_C(z)| > |N_{C'}(z)| \), a contradiction. Assume that \( z \) is adjacent to \( x \) but not to \( v \). Then \( |N_C(z)| = |N_{C'}(z)| - 1 \), and we have \( |N_C(z)| = 3 \). By Lemma 4.7, \( z \) is in \( N_G^3(C) \). Since \( z \) is not adjacent to \( v \), \( G[C \cup \{x, z\} \setminus \{v\}] \) is a hole in \( G \), a contradiction. Case 2: \( z \in N_G^{2,2}(C') \). We have \( |N_{C'}(z)| \) even and thus \( |N_C(z)| \) is odd. If \( z \) is adjacent to \( v \) but not to \( x \), then \( z \) is in \( M_G(C) \), a contradiction. If \( z \) is adjacent to \( x \) but not to \( v \), then \( z \) is in \( N_G^{1,2}(C) \), contradicting to Lemma 4.7. Therefore \( M_G(C') = N_G^{2,2}(C') = \emptyset \). The lemma is proved.

We are ready to prove Lemma 4.3 using Lemma 4.8.

Proof of Lemma 4.3. Let \( Q_0 \) be the initial given quadruple \( (H, u_1, u_2, u_3) \) of \( G \). Our algorithm iteratively updates \( H, u_1, u_2 \) and \( u_3 \) by the following steps.
Step 1: If none of the pairs of distinct nodes \((x, y)\) in \(H\) satisfy \(N_H[y] \subseteq N_H[x]\), then output \((H', u'_1, u'_2, u'_3) = (H, u_1, u_2, u_3)\). Otherwise, let \((x, y)\) be a pair of nodes satisfying \(N_H[y] \subseteq N_H[x]\) and proceed to Step 2.

Step 2: Replace \(H\) with \(H \setminus \{y\}\). If \(y\) is equal to node \(u_i\) for some \(i \in \{1, 2, 3\}\), then replace \(u_i\) with \(x\). After \((H, u_1, u_2, u_3)\) is updated, proceed to the next iteration of the loop.

We prove the correctness of the algorithm. Step 1 outputs \((H', u'_1, u'_2, u'_3)\). By definition of Step 1, \(H' = H\) has no dominated nodes. Condition 1 holds. Condition 2 follows from the update operation in Step 2. As for Condition 3, we show that if \(Q_0\) is a tracking quadruple of \(G\), then each intermediate \(Q = (H, u_1, u_2, u_3)\) throughout the loop remains a tracking quadruple of \(G\). The statement can be proved by verifying the following claim: Each update to \(Q\) in Step 2 does not affect the condition that \(Q\) is a tracking quadruple of \(G\).

First we prove that \(H \setminus \{y\}\) has a clean shortest even hole \(C'\). Since \(Q\) is a tracking quadruple, there is a clean shortest even hole \(C\) of \(H\), and \((u_1, u_2, u_3)\) is a tracker of \(C\) in \(H\). Therefore \((u_1, u_2, u_3)\) is a path of \(C\). If \(y\) is not on \(C\), then \(C' = C\) is a clean shortest even hole of \(H \setminus \{y\}\), and \((u_1, u_2, u_3)\) is a path of \(C'\). Therefore \((u_1, u_2, u_3)\) is a tracker of \(C'\) in \(H \setminus \{y\}\). This proves that \((H \setminus \{y\}, u_1, u_2, u_3)\) is a tracking quadruple of \(G\). The rest of the proof assumes that node \(y\) is on \(C\). Assume node \(x\) is on \(C\). There is a neighbor of \(y\) on \(C\) that is not adjacent to \(x\), which contradicts to the fact that \(N_H[y] \subseteq N_H[x]\). Thus \(x\) is not on \(C\). By Lemma 4.8, \(C' = H[C \cup \{x\} \setminus \{y\}]\) is a clean shortest even hole of \(H\). Since \(C' \subseteq H \setminus \{y\}\), \(C'\) is a clean shortest even hole of \(H \setminus \{y\}\). Next we prove that \((u_1, u_2, u_3)\) is a tracker of \(C'\) in \(H \setminus \{y\}\). If \(y \notin \{u_1, u_2, u_3\}\), then \((u_1, u_2, u_3)\) is a path of \(C'\). If \(y \in \{u_1, u_2, u_3\}\), since \(x\) dominates \(y\) and \((u_1, u_2, u_3)\) is a path of \(C\), after replacing \(y\) with \(x\), the updated \((u_1, u_2, u_3)\) is a path of \(C'\). This proves that \((H \setminus \{y\}, u_1, u_2, u_3)\) is a tracking quadruple of \(G\).

Step 2 decreases the number of nodes in \(H\) by one. Therefore, the loop has \(O(n)\) iterations. By the fact that detecting dominated nodes in \(H\) takes \(O(mn)\) time, Step 1 takes \(O(mn)\) time. Step 2 takes \(O(1)\) time. Therefore, the overall running time is \(O(mn^2)\). The lemma is proved.

4.1.2 Proving Lemma 4.4

Proof of Lemma 4.4. Let \(Q = (H, u_1, u_2, u_3)\) be the given tracking quadruple of \(G\), and \(S\) be the given full star-cut in \(H\). We prove that one of Conditions B1 and B2 holds. Since \(Q\) is a tracking quadruple of \(G\), there is a clean shortest even hole \(C\) in \(G\) such that \((u_1, u_2, u_3)\) is a tracker of \(C\). Since \(S\) is a full star-cut, \(S = N_H[s]\) holds for some node \(s\) in \(S\).

First we prove that either Condition B1 holds, or the following claim is true: There is a unique connected component \(B\) of \(H \setminus S\) that intersects \(C\), and \(|S \cap V(C)| \leq 3\). If \(s\) is on \(C\), since \(s\) and the neighbors of \(s\) in \(C\) form a path on \(C\), i.e., \(C[S \cap V(C)]\) is a path on \(C\), there is a unique connected component \(B\) of \(H \setminus S\) that intersects \(C\). Since \(S = N_H[s]\) and \(s \in S \cap V(C)\), we have \(|S \cap V(C)| \leq 3\). If \(s\) is not on \(C\), we consider the following two cases. Case 1: There are at least two connected components of \(H \setminus S\) that intersect \(C\). Therefore \(S \cap V(C)\) has at least two components. Since \(C\) is clean, \(M_H(C) = N_H^{2,2}_H(C) = \emptyset\). By Lemma 4.7 and the fact that \(S \cap V(C)\) has at least two components, node \(s\) is in \(N_H^1(C)\). Let \(s_1\) and \(s_2\) be the two nodes in \(N_C(s)\), and let \(B_1, B_2\) be the two connected components of \(H \setminus S\) that intersect \(C\). Then there are two distinct nodes \(s_1\) and \(s_2\) of \(S\) and two distinct connected components \(B_1\) and \(B_2\) of \(H \setminus S\) such that (a) \(s_1\) and \(s_2\) are not adjacent in \(H\) and (b) both \(s_1\) and \(s_2\) are adjacent to both \(B_1\) and \(B_2\) in \(H\). Therefore Condition B1 holds. Case 2: There are no connected components of \(H \setminus S\) that intersect \(C\). Therefore \(V(C) \subseteq S = N_H[s]\). Since \(C\) is clean, \(M_H(C) = N_H^{2,2}_H(C) = \emptyset\). Since \(S \cap V(C)\) has only one component, by Lemma 4.7 node \(s\) is in one of the following sets: \(N_H^1(C), N_H^2(C), N_H^{2,2}_H(C)\). We have \(|V(C)| \leq 3\), which contradicts to the fact that \(C\) is a hole.
Therefore there is a unique connected component \( B \) of \( H \setminus S \) that intersects \( C \). By Lemma 4.7, node \( s \) is in one of the following sets: \( N_H^1(C), N_H^2(C), \) and \( N_H^3(C) \). Thus \( |S \cap V(C)| \leq 3 \). Now the claim follows, which implies that \( C \setminus [S \cap V(C)] \) has only one component, and \( B \) is the unique connected component of \( H \setminus S \) such that \( V(C) \subseteq (V(B) \cup S) \).

To prove the lemma, we consider the following two cases. Case 1: \( \{u_1, u_2, u_3\} \subseteq S \cap V(C) \) and \( \{u_1, u_2, u_3\} \) is a path of \( C \), the above claim ensures that there is a unique connected component \( B \) of \( H \setminus S \) such that \( V(C) \subseteq (V(B) \cup S) \), and \( S \cap V(C) = \{u_1, u_2, u_3\} \). Consider the neighbors of \( u_1 \) and \( u_3 \) on \( C \) other than \( u_2 \), i.e., \( N = N_C(u_1) \cup N_C(u_3) \). Since \( S \cap V(C) = \{u_1, u_2, u_3\} \), \( N \cap S = \emptyset \); it follows that \( N \not\subseteq (V(B) \cup S) \), and both \( u_1 \) and \( u_3 \) are adjacent to \( B \) in \( H \). If there is another connected component \( B' \) of \( H \setminus S \) other than \( B \) such that both \( u_1 \) and \( u_3 \) are adjacent to \( B' \) in \( H \), then let \( s_1 = u_1, s_2 = u_3, B_1 = B, \) and \( B_2 = B' \). There are two distinct nodes \( s_1 \) and \( s_2 \) of \( S \) and two distinct connected components \( B_1 \) and \( B_2 \) of \( H \setminus S \) such that (a) \( s_1 \) and \( s_2 \) are not adjacent in \( H \) and (b) both \( s_1 \) and \( s_2 \) are adjacent to both \( B_1 \) and \( B_2 \) in \( H \). Therefore Condition B1 holds. Otherwise \( B \) is the unique connected component of \( H \setminus S \) such that both \( u_1 \) and \( u_3 \) are adjacent to \( B \) in \( H \), and Condition B2(a) holds.

Case 2: \( \{u_1, u_2, u_3\} \not\subseteq S \). By the above claim, let \( B \) be the unique connected component of \( H \setminus S \) such that \( V(C) \subseteq (V(B) \cup S) \). Since \( \{u_1, u_2, u_3\} \subseteq V(C) \), \( \{u_1, u_2, u_3\} \subseteq V(B) \cup S \). Assume there is another connected component \( B' \) of \( H \setminus S \) other than \( B \) such that \( \{u_1, u_2, u_3\} \subseteq V(B') \cup S \). Since \( (V(B) \cup S) \cap (V(B') \cup S) = S \), we have \( \{u_1, u_2, u_3\} \subseteq S \), a contradiction. Therefore \( B \) is the unique connected component of \( H \setminus S \) such that \( \{u_1, u_2, u_3\} \subseteq V(B) \cup S \). Condition B2(b) holds.

If Condition B2 holds, we verify that \( (H[V(B) \cup S], u_1, u_2, u_3) \) is a tracking quadruple of \( G \). Since \( V(C) \subseteq (V(B) \cup S), H[V(B) \cup S] \) has a clean shortest even hole \( C \). Also \( \{u_1, u_2, u_3\} \) is a path of \( C \), which implies that \( \{u_1, u_2, u_3\} \) is a tracker of \( C \). This proves that \( (H[V(B) \cup S], u_1, u_2, u_3) \) is indeed a tracking quadruple of \( G \). The lemma is proved.

\[ \square \]

4.2 2-joins

This subsection proves Lemma 4.2. A disjoint partition \((V_1, V_2)\) of the nodes of graph \( H \) is a non-path 2-join [3, 39] of \( H \) with link sets \((V_{1,1}, V_{1,2}, V_{2,1}, V_{2,2})\) if the following conditions hold.\(^4\)

\begin{align*}
\text{Condition J1:} & \quad |V_1| \geq 3 \text{ and } |V_2| \geq 3. \\
\text{Condition J2:} & \quad V_{1,i} \neq \emptyset, V_{2,i} \neq \emptyset, V_{1,i} \cap V_{2,i} = \emptyset, \text{ and } V_{1,i} \cup V_{2,i} \subseteq V_i \text{ hold for each } i \in \{1, 2\}. \\
\text{Condition J3:} & \quad \text{Each node of } V_{1,1} \text{ and each node of } V_{2,1} \text{ are adjacent in } H. \text{ Each node of } V_{2,1} \text{ and each node of } V_{2,2} \text{ are adjacent in } H. \text{ There are no other edges between } V_{1,i} \text{ and } V_{2,i}. \\
\text{Condition J4:} & \quad \text{For each } i \in \{1, 2\}, H[V_i] \text{ is not an induced path } P \text{ between a node of } V_{1,i} \text{ and a node of } V_{2,i} \text{ with } V(\text{int}(P)) \subseteq V_i \setminus (V_{1,i} \cup V_{2,i}). \quad \text{\(^5\)}
\end{align*}

Graph \( H \) is an extended clique tree if there are two distinct nodes \( x \) and \( y \) of \( H \) such that each biconnected component of \( H \setminus \{x, y\} \) is a clique. We need the following two lemmas.

**Lemma 4.9** (da Silva and Vušković [21, Corollary 1.3]).

1. Given an \( n \)-node \( m \)-edge graph \( H \), it takes \( O(mn^3) \) time to complete one of the following tasks.
   Task 1: Determining whether \( H \) has even holes. Task 2: Ensuring that \( H \) is not an extended clique tree.

2. A graph that has no even holes, star-cuts, and non-path 2-joins is an extended clique tree.

\(^4\)A disjoint partition \((V_1, V_2)\) with link sets \((V_{1,1}, V_{1,2}, V_{2,1}, V_{2,2})\) satisfying Conditions J1 to J3 is called a 2-join, which will not be used in this paper.

\(^5\)The 2-joins in [17, 21] are defined to be non-path.
Lemma 4.10. Given an \( n \)-node \( m \)-edge graph \( H \) having no star-cuts, it takes \( O(mn^3) \) time to complete one of the following tasks. Task 1: Ensuring that \( H \) has even holes. Task 2: Obtaining an \( O(n) \)-node \( O(m) \)-edge graph \( H' \) having no 4-holes, star-cuts, and non-path 2-joins such that \( H' \) has even holes if and only if \( H \) has even holes.

We first prove Lemma 4.2 using Lemmas 4.9 and 4.10.

Proof of Lemma 4.2. By Lemma 4.10, it suffices to show that given an \( n \)-node \( m \)-edge graph \( H \) having no 4-holes, star-cuts, and non-path 2-joins, it takes \( O(mn^3) \) time to determine whether \( H \) has even holes. We apply Lemma 4.9(1) on \( H \). If whether \( H \) has even holes is determined, then the lemma is proved. Otherwise, \( H \) is not an extended clique tree. By Lemma 4.9(2), \( H \) has even holes. The lemma is also proved.

The rest of the subsection proves Lemma 4.10. In order to decompose a (non-path) 2-join, we need one more property. Non-path 2-join \( (V_1, V_2) \) of graph \( H \) with link sets \( (V_{1,1}, V_{1,2}, V_{2,1}, V_{2,2}) \) is connected [39] if it satisfies

Condition J5: For each \( i \in \{1, 2\} \), there is an induced path \( P_i \) of \( H[V_i] \) between a node of \( V_{i,1} \) and a node of \( V_{i,2} \) such that \( V(\text{int}(P_i)) \subseteq V_i \setminus (V_{i,1} \cup V_{i,2}) \).\(^6\)

Let \((V_1, V_2)\) be a connected non-path 2-join with link sets \((V_{1,1}, V_{1,2}, V_{2,1}, V_{2,2})\). Let \( P_1 \) (respectively, \( P_2 \)) be an induced path of \( H[V_1] \) (respectively, \( H[V_2] \)) as ensured by Condition J5. Let \( p_1 \) and \( p_2 \) be two integers defined as follow: If \( |V(\text{int}(P_1))| \) is even, then let \( p_1 = 4 \); otherwise, let \( p_1 = 3 \). If \( |V(\text{int}(P_2))| \) is even, then let \( p_2 = 4 \); otherwise, let \( p_2 = 3 \). The blocks [17, 39] of \((V_1, V_2)\) for \( P_1 \) and \( P_2 \) are the graphs \( H_1 \) and \( H_2 \) defined as follows.

- \( H_1 \) consists of (a) \( H[V_1] \), (b) a \( p_2 \)-node path \( P_2 \) between nodes \( v_{2,1} \) and \( v_{2,2} \) such that \( \text{int}(P_2) \) is separate from \( H[V_1] \), (c) edges \((v_{2,1}, v)\) for all nodes \( v \) of \( V_{1,1} \), and (d) edges \((v_{2,2}, v)\) for all nodes \( v \) of \( V_{1,2} \).

- \( H_2 \) consists of (a) \( H[V_2] \), (b) a \( p_1 \)-node path \( P_1 \) between nodes \( v_{1,1} \) and \( v_{1,2} \) such that \( \text{int}(P_1) \) is separate from \( H[V_2] \), (c) edges \((v_{1,1}, v)\) for all nodes \( v \) of \( V_{2,1} \), and (d) edges \((v_{1,2}, v)\) for all nodes \( v \) of \( V_{2,2} \).

When decomposing 2-joins using blocks, if we find a 2-join which is already in the form of a block, then the decomposition process may not stop. This is the main reason why we use non-path 2-joins.

Lemma 4.11 (Charbit et al. [3]). Given an \( n \)-node \( m \)-edge graph \( H \), it takes \( O(mn^2) \) time to complete one of the following tasks. Task 1: Ensuring that \( H \) has no non-path 2-joins. Task 2: Obtaining a non-path 2-join of \( H \).

Lemma 4.12 (Conforti et al. [17], da Silva et al. [21, Theorem 10.10], Trotignon et al. [39, Lemmas 3.2 and 3.8]). If \((V_1, V_2)\) is a non-path 2-join of a graph \( H \) having no 4-holes and star-cuts, then \((V_1, V_2)\) is a connected non-path 2-join. For any induced paths \( P_1 \) and \( P_2 \) as ensured by Condition J5, blocks \( H_1 \) and \( H_2 \) of \((V_1, V_2)\) for \( P_1 \) and \( P_2 \) satisfy the following conditions.

Condition T1: Both \( H_1 \) and \( H_2 \) have no star-cuts.

Condition T2: \( H \) has no even holes if and only if both \( H_1 \) and \( H_2 \) have no even holes.

Condition T3: \(|H_1| \geq 4\) and \(|H_2| \geq 4\).

\(^6\) The 2-joins in [21] are defined to be connected. In [17] this condition is proved as a property after decomposing star-cuts.
It remains to prove Lemma 4.10 using Lemmas 4.11 and 4.12.

Proof of Lemma 4.10. Let $H_0$ be the initial given graph. We first determine whether $H_0$ has 4-holes in $O(n^4)$ time. If $H_0$ is ensured to have 4-holes, then the lemma is proved by completing Task 1. Therefore we assume $H_0$ has no 4-holes in the following proof. Let $H = H_0$. We construct a table by computing whether $K$ has even holes to each graph $K$ with at most 20 edges in $O(1)$ time. And we iteratively update $H$ by the following steps until a graph $H'$ is outputted.

Step 1: Repeat the following step if there exists a connected component $B$ of $H$ satisfying $|E(B)| \leq 20$: Determine whether $B$ has even holes using the table constructed above; if $B$ has even holes, then complete Task 1 by outputting that $H$ has even holes; otherwise remove $B$ from $H$. If there are no connected components $B$ of $H$ satisfying $|E(B)| \leq 20$, proceed to Step 2.

Step 2: Apply Lemma 4.11 on the current $H$. If $H$ has no non-path 2-joins, then complete Task 2 by outputting $H$. Otherwise, let $(V_1, V_2)$ be a non-path 2-join of $H$ with link sets $(V_{1,1}, V_{1,2}, V_{2,1}, V_{2,2})$. For each $i \in \{1, 2\}$, let $P_i$ be an induced path of $H[V_i]$ between a node of $V_{i,1}$ and a node of $V_{i,2}$ such that $V(int(P_i)) \subseteq V_i \setminus (V_{i,1} \cup V_{i,2})$. Replace $H$ with the disjoint union of the blocks $H_1$ and $H_2$ of $(V_1, V_2)$ for $P_1$ and $P_2$ and proceed to the next iteration.

By definition of the above procedure, the outputted graph has no non-path 2-join. In Step 2 the blocks are well-defined, since by Lemma 4.12 any non-path 2-join $(V_1, V_2)$ of $H$ is a connected non-path 2-join. First we show that each intermediate $H$ throughout the loop satisfies the following claim: Each update to $H$, either in Step 1 or Step 2, does not affect the conditions that (1) $H$ has no star-cuts and (2) $H$ has even holes if and only if $H_0$ has even holes. If $H$ is updated in Step 1, then Condition (1) follows from the fact that $B$ is an induced subgraph of $H$ and Condition (2) follows from definition of Step 1 that $B$ has no even holes. If $H$ is updated in Step 2, then Condition (1) follows from Condition T1 of Lemma 4.12 and Condition (2) follows from Condition T2 of Lemma 4.12. Therefore the claim follows.

We prove the correctness of the algorithm. Case 1: Step 1 outputs $H$. By definition of Step 1, $H$ is ensured to have even holes. Therefore, Task 1 is indeed completed. Case 2: Step 2 outputs $H$. $H$ has no 4-holes since $H_0$ does not. By the above claim, $H$ has no star-cuts and $H$ has even holes if and only if $H_0$ has even holes. By definition of Step 2, $H$ has no non-path 2-joins. Since in each iteration the number of edges is increased by at most 7 by definition of blocks, the number of connected components is increased by at most one, and by Condition T3 of Lemma 4.12 the number of edges in each connected component of $H$ is at least 8, the number of connected components $k$ in $H$ satisfies the following inequality: $m + 7k \geq 8k$. This implies that $k = O(m)$, and the size of $E(H)$ is bounded by $m + 7k = O(m)$. A similar argument proves that $|V(H)| = O(n)$. Therefore, Task 2 is indeed completed.

Step 1 takes $O(mn^2)$ time by Lemma 4.11. Step 2 takes $O(m)$ time, since blocks of a connected non-path 2-join can be constructed in $O(m)$ time. Therefore each iteration of the above procedure takes $O(mn^2)$ time. There are at most $O(n)$ iterations since the number of connected components in $H$ is increased by at most one in each iteration, and the number of connected components of $H$ is $O(m)$. The overall running time is $O(mn^3)$. The lemma is proved.

5 Concluding remarks

We show that whether a graph $G$ has even holes can be determined in $O(m^2n^7)$ time. By a new structural result for major nodes of a shortest even hole, graph $G$ can be cleaned much effi-
ciently than the original method in [6, Lemma 2.5]. And we introduce the notion of “trackers” as a new algorithmic technique, which helps us to keep track of the position of a shortest even hole in a star-cut. This significantly reduces the huge time complexity of decomposing star-cuts in [17, Section 3] and [21, Theorem 10.9]. We believe that the use of trackers can enhance the performance of other induced subgraph detection algorithms based on decomposing star-cuts. By arbitrarily deleting nodes on the input graph and calling our even-hole recognition algorithm, an even hole can be found with an additional $O(n)$ factor in running time.

The bottleneck of our algorithm is the step of detecting beetles in the proof of Lemma 2.1. The full strength of beetles are used in Lemma 4.8. Instead of using beetles, we can introduce the following notion. Let $N^3_G(C)$ consist of nodes $u$ in $N^3_G(C)$ that is adjacent to some other node $v$ in $N^3_G(C)$ such that $N_C(u)$ and $N_C(v)$ are disjoint. We can prove the following lemma.

**Lemma 5.1.** If $C$ is a shortest even hole of a graph $G$ that has no 4-holes, then each node of $M_G(C)$ is adjacent to each node of $N^{rs}_G(C)$ in $G$.

Assume that we can clean the nodes of $N^3_G(C)$ in the cleaning phase using Lemma 5.1 without increasing the time complexity of Lemma 2.1. We have to deal with $N^3_G(C)$ in Lemma 4.8. This can be done if more induced subgraphs which guarantee the existence of even holes are detected in the cleaning phase. The time complexity would then be reduced. We conjecture that whether an $n$-node graph has even holes can be determined in $O(n^{10})$.

**Acknowledgement**

We thank Gerard J. Chang for helpful discussion on the number of maximal cliques in graphs.

**References**


