Theoretical Computer Science 507 (2013) 100–113

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Fast–mixed searching and related problems on graphs

Boting Yang *

Department of Computer Science, University of Regina, Regina, Saskatchewan S4S 0A2, Canada

A B S T R A C T

In this paper, we introduce the fast–mixed search model, which is a combination of the fast search model and the mixed search model. Given a graph G in which a fugitive hides on vertices or along edges, the fast–mixed search problem is to find the minimum number of searchers to capture the fugitive in the fast–mixed search model. We establish a relation between the fast–mixed search problem and the induced-path cover problem. We also consider relations between the fast–mixed search problem and other graph search problems. We prove that Fast–Mixed Searching From Leaves is NP-complete, and it remains NP-complete for planar graphs with maximum degree 3. We also prove that Fast–Mixed Searching Between Leaves is NP-complete, and it remains NP-complete for graphs with maximum degree 4. We present linear-time algorithms for computing the fast–mixed search number and optimal search strategies of some classes of graphs, including trees, cacti, and proper interval graphs.

1. Introduction

Given a graph in which a fugitive hides on vertices or along edges, the objective of a graph search problem is usually to find the minimum number of searchers required to capture the fugitive. The edge search problem and the node search problem are two major graph search problems. The edge search problem was introduced by Megiddo et al. [9]. They showed that determining the edge search number of a graph is NP-hard. They also gave a linear-time algorithm to compute the edge search number of a tree. The node search problem was introduced by Kirousis and Papadimitriou [6]. They showed that the node search number is equal to the pathwidth plus one and that the edge search number and node search number differ by at most one. Both search problems are monotonic [1,7].

Bienstock and Seymour [1] introduced the mixed search problem that combines the edge search and the node search problems. In the mixed search problem, the invisible fugitive can move at a great speed at any time from a vertex to another vertex along a searcher-free path between these two vertices. There are three actions for searchers: *placing* (place a searcher on a vertex), *removing* (remove a searcher from a vertex) and *sliding* (slide a searcher from one endpoint of an edge to the other along the edge). The fugitive is captured on an edge either when a searcher slides along the edge or when both endpoints of the edge are occupied by searchers. The fugitive is also captured when the fugitive and a searcher occupy the same vertex. The mixed search problem is monotonic [1]. The minimum number of searchers required to capture the fugitive is the *mixed search number* of G, denoted by $\mathrm{ms}(G)$. To illustrate the mixed search strategy, let us consider a tree T_{2k}, $k \geq 2$, shown in Fig. 1, which consists of $2k$ paths xx_i, $1 \leq i \leq 2k$. Suppose that T_{2k} contains an invisible fugitive. We can place one searcher on the vertex x and then use another searcher to move from x to x_i along each path xx_i, $1 \leq i \leq 2k$. It is easy to see that this strategy must use all three actions, i.e., placing, removing and sliding. We can show that $\mathrm{ms}(T_{2k}) = 2$.

Dyer et al. [2] introduced the fast search problem, in which the invisible fugitive can move at a great speed at any time from a vertex to another vertex along a searcher-free path between these two vertices. There are two actions for searchers, namely, *placing* and *sliding*. The fugitive is captured on an edge or a vertex only when a searcher slides along the edge or occupies the vertex. In the fast search problem, every edge can be traversed exactly once by a searcher, and it is cleared by a sliding action. The minimum number of searchers required to capture the fugitive is the *fast search number* of G, denoted by $\mathrm{fs}(G)$. Some recent development on the fast search problem can be found in [13,15]. For the tree T_{2k} in Fig. 1, we can place

* Tel.: +1 306 585 4774; fax: +1 306 585 4745.
E-mail address: boting@cs.uregina.ca.

0304-3975/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.tcs.2013.04.015

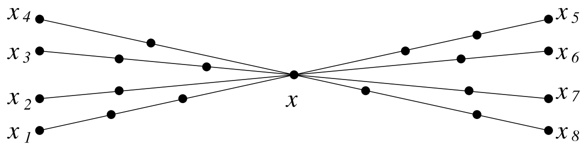

Fig. 1. A tree T_{2k} with $k = 4$.

k searchers on leaves x_i, $1 \le i \le k$, and move them to other leaves x_{k+i}, $1 \le i \le k$, along different paths to clear all edges on the tree. It is easy to see that this strategy uses both placing and sliding actions and every edge is traversed exactly once. We can show that $\mathrm{fs}(T_{2k}) = k$.

In this paper, we introduce the fast–mixed search model, which combines the fast search and the mixed search models. Let G be a connected graph with no loops or multiple edges. In the fast–mixed search model, G initially contains no searchers and it contains only one fugitive who hides on vertices or along edges. The fugitive is invisible to searchers, and he can move at a great speed at any time from one vertex to another vertex along a searcher-free path between the two vertices. An edge (resp. a vertex) where the fugitive may hide is said to be *contaminated*, while an edge (resp. a vertex) where the fugitive cannot hide is said to be *cleared*. A vertex is said to be *occupied* if it has a searcher on it. There are two types of actions for searchers in each step of the fast–mixed search model:

- *placing* a searcher on a contaminated vertex; and
- *sliding* a searcher along a contaminated edge uv from u to v if v is contaminated and all edges incident on u except uv are cleared.

These two actions are different from those in the fast search model or the mixed search model because of the conditions on performing them. In the fast (or mixed) search model, a vertex can be occupied by two or more searchers at any moment, but in the fast–mixed search problem, every vertex is occupied by at most one searcher. In the fast (or mixed) search model, if a vertex is occupied by two searchers and at least two contaminated edges are incident on it, we can slide one searcher along a contaminated edge, but this cannot happen in the fast–mixed search problem.

In the fast–mixed search model, a contaminated edge becomes cleared if both endpoints are occupied by searchers or if a searcher slides along it from one endpoint to the other. An *fms-strategy* of G is a sequence of actions such that the final action leaves all edges of G cleared. The graph G is *cleared* if all edges are cleared. The minimum number of searchers required to clear G (i.e., to capture the fugitive) is the *fast–mixed search number* of G, denoted by $\mathrm{fms}(G)$. An fms-strategy of G that uses $\mathrm{fms}(G)$ searchers is called an *optimal fms-strategy*. For the tree T_{2k} in Fig. 1, we can place $2k - 1$ searchers on leaves x_i, $1 \le i \le 2k - 1$, and move them to vertices adjacent to the vertex x, and then move one of them to the leaf x_{2k} to clear the remaining edges on the tree. In this strategy, $2k - 2$ edges incident on x are cleared because both endpoints of these edges are occupied by searchers at some moment when a searcher moves to x and all other edges are cleared by sliding actions. We can show that $\mathrm{fms}(T_{2k}) = 2k - 1$ by Theorem 7.3.

The fast–mixed search problem has a close relation with the induced-path cover problem. In [8], Le et al. proved that it is NP-complete to decide whether or not the vertex set of a connected graph can be partitioned into two subsets, each of which induces a path. In [11], Pan and Chang gave linear-time algorithms for the induced-path cover problem on block graphs whose blocks are complete graphs, cycles or complete bipartite graphs.

This paper is organized as follows. In Section 2, we give definitions and notation. In Section 3, we give characterizations of graphs that can be cleared by k searchers. We also establish a relation between the fast–mixed search problem and the induced-path cover problem. In Section 4, we give lower bounds on the fast–mixed search number. In Section 5, we consider relations between fast–mixed searching and fast searching, and relations between fast–mixed searching and mixed searching. In Section 6, we give NP-completeness results. In Section 7, we show linear-time algorithms for computing the fast–mixed search number of some classes of graphs including trees, cacti and proper interval graphs. Finally, we conclude this paper in Section 8.

2. Preliminaries

Throughout this paper, we only consider finite connected graphs with no loops or multiple edges. We use $G = (V, E)$ to denote a graph with vertex set V and edge set E, and we also use $V(G)$ and $E(G)$ to denote the vertex set and edge set of G respectively. We use uv (or sometimes $\{u, v\}$) to denote an edge with endpoints u and v. Definitions omitted here can be found in [14].

For a graph $G = (V, E)$ and a vertex $v \in V$, the *degree* of v, denoted by $\deg_G(v)$, is the number of edges incident on v. The vertex set $\{u : uv \in E\}$ is the *neighborhood* of v, denoted as $N_G(v)$. If there is no ambiguity, we use $\deg(v)$ and $N(v)$ without subscripts. Let $\delta(G) = \min\{\deg(v) : v \in V(G)\}$. For a subset $V' \subseteq V$, we use $G[V']$ to denote the subgraph induced by V', which consists of all vertices of V' and all of the edges that connect vertices of V' in G. For a vertex $v \in V$, we use $G - v$ to denote the subgraph induced by $V \setminus \{v\}$.

A *path* is a sequence $v_0v_1 \dots v_k$ of distinct vertices of G with $v_{i-1}v_i \in E$ for all i $(1 \le i \le k)$. It is denoted by $v_0v_1 \dots v_k$ or $v_0 \sim v_k$. The *length* of a path is the number of edges on the path. A path in G is called an *induced path* if the subgraph induced by its vertex set is a path.

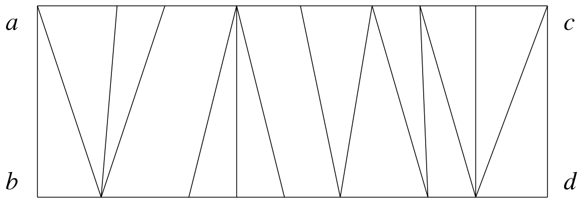

Fig. 2. A bipolar outerplanar graph with two polar edges *ab* and *cd*.

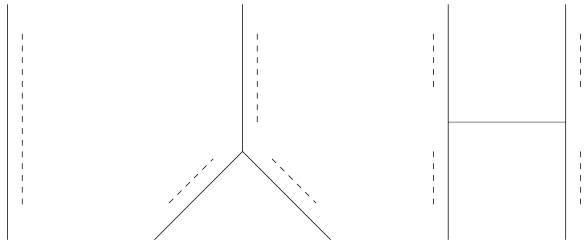

Fig. 3. Trees with fms ≤ 2, where any edge marked by a dashed line can be replaced by a path of length at least one.

For a graph G, a *path cover* is a set of vertex-disjoint paths that contain all the vertices of G. An *induced-path cover* is a path cover in which every path is an induced path of G. The induced-path cover problem is to find a minimum number of vertex-disjoint induced-paths that contain all the vertices of G. This minimum number is called the *induced-path number* of G, denoted by $\mathrm{ipc}(G)$.

A graph is said to be *outerplanar* if it can be drawn in the plane without crossing edges such that all vertices are incident to the unbounded face of the drawing. A biconnected outerplanar graph has a planar embedding in which the boundary of the unbounded face is a simple cycle that includes all the vertices. The edges on the boundary of the unbounded face are called *boundary edges*, and the remaining edges are called *chords*. An outerplanar graph is *bipolar* if it is biconnected and there are two boundary edges *ab* and *cd* such that every chord has one endpoint on the path $a \sim c$ and the other endpoint is on the path $b \sim d$ (see Fig. 2). Edges *ab* and *cd* are called *polar edges* and vertices a, b, c, d are called *polar vertices*.

A graph G is a *k-tree* if and only if either G is a complete graph with k vertices, or G has a vertex v such that $N(v)$ induces a k-clique, and $G - v$ is a k-tree. A vertex v of a k-tree G is called *simplicial* if $N(v)$ induces a k-clique.

3. Characterizations and fms-path covers

We first consider graphs that can be cleared by two searchers. For trees, we have the following characterization.

Lemma 3.1. *For a tree T, the following are equivalent:*

(i) $\mathrm{fms}(T) \leq 2$.

(ii) *All vertices of T have degree less than 4; at most two vertices have degree 3; and if T has two vertices of degree 3, then these two vertices must be adjacent.*

(iii) *T is one of the graphs in Fig. 3.*

Proof. It is easy to see that (ii) ⇒ (iii) and (iii) ⇒ (i). We now show that (i) ⇒ (ii). Suppose that $\mathrm{fms}(T) \leq 2$. If T has a vertex v of degree at least 4, then $\mathrm{fms}(T) > 3$ because at most two incident edges of v can be cleared by sliding a single searcher and all other incident edges are cleared by different searchers. This is a contradiction. If $\delta(T) = 3$ and T has at least three vertices of degree 3, then T has at least 5 leaves. Since each leaf is associated with a searcher starting or ending on the leaf, we have $\mathrm{fms}(T) \geq 3$. This is a contradiction. If $\delta(T) = 3$ and T has two non-adjacent vertices of degree 3, then two searchers cannot move from leaf to leaf to traverse all vertices. This is a contradiction. □

Definition 3.2. A graph is called a *ladder* if it can be obtained from a bipolar outerplanar graph by attaching at most one path to each polar vertex (if two polar vertices coincide, then this vertex can have at most two paths attached to it). Referring to Fig. 6(a), a graph is called a *standard ladder* if it can be obtained from a grid $G_{2\times n}$ ($n \geq 2$) by attaching four edges to the four corner vertices of $G_{2\times n}$ respectively.

Lemma 3.3. *For any connected graph G that is not a tree,* $\mathrm{fms}(G) = 2$ *if and only if G is a ladder.*

Proof. If G is a ladder, it is easy to see that we can use two searchers to clear G. On the other hand, since G is not a tree, we know that $\mathrm{fms}(G) \geq 2$. Thus $\mathrm{fms}(G) = 2$.

If $\mathrm{fms}(G) = 2$, let P_1 and P_2 be two paths traversed by the two searchers respectively. Then P_1 and P_2 must be induced paths containing all vertices of G and the edges between them have no intersections. Since G is connected and is not a tree, G must be a ladder. □

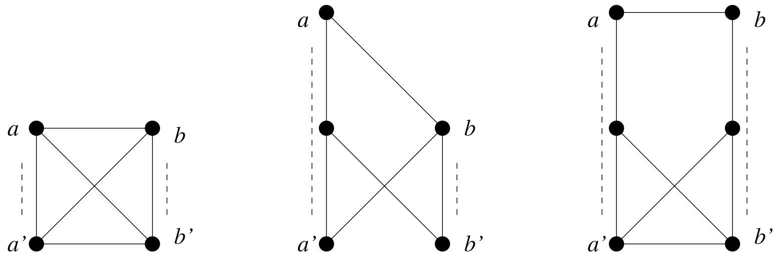

Fig. 4. The forbidden subgraphs in Corollary 3.9, where any edge marked by a dashed line can be replaced by a path of length at least one.

Before we give a characterization of graphs G with $\mathrm{fms}(G) = k$ from a graph drawing point of view, we generalize ladders as follows.

Definition 3.4. A graph $G = (V, E)$ is called a *k-stack* if k is the smallest integer such that G can be drawn in the following way:

1. Draw k vertex-disjoint paths $P_1, \ldots, P_k$ such that $\cup_{i=1}^{k} V(P_i) = V$ and $\cup_{i=1}^{k} E(P_i) \subseteq E$ (note that P_i may have length 0). Suppose that all paths are drawn by parallel vertical straight line segments, and let p_i, $1 \le i \le k$, be a moving point on P_i. Initially, each p_i is on the bottom vertex of P_i. Draw edges between the moving points if there is an edge between them in G.
2. Repeat the following process until all moving points are located on the top vertices of all vertical paths: Pick a moving point p_i $(1 \le i \le k)$ that has an above neighbor v on P_i satisfying that each vertex in $N_G(v)$ either contains a moving point p_j $(1 \le j \le k)$ or has not been traversed by a moving point; move p_i up to the vertex v; and then draw edges between v and p_j $(1 \le j \le k, j \ne i)$ if there is an edge between them in G.

From Definitions 3.2 and 3.4, we know that a ladder is always a 2-stack, but a 2-stack is a ladder if it is connected and has at least one cycle.

Theorem 3.5. *For any connected graph G, $\mathrm{fms}(G) = k$ if and only if G is a k-stack.*

Proof. If G is a k-stack, from Definition 3.4, we have k vertex-disjoint induced paths $P_1, \ldots, P_k$ that include all vertices of G. If each moving point is simulated by a searcher, then k searchers can clear G. Thus $\mathrm{fms}(G) \le k$. Since k is the smallest integer for the drawing, we know that $\mathrm{fms}(G) \ge k$. Hence $\mathrm{fms}(G) = k$.

If $\mathrm{fms}(G) = k$, we can draw the graph by replacing each searcher by a moving point and k is the smallest number. Thus, G is a k-stack. □

We now establish a relation between the fast–mixed search problem and the induced-path cover problem.

Theorem 3.6. *For a graph $G = (V, E)$ that can be cleared by k searchers in an fms-strategy S, let $V_1, \ldots, V_k$ be k subsets of V such that each vertex in V_i, $1 \le i \le k$, is visited by the same searcher in the fms-strategy S. Then $V_1, \ldots, V_k$ form a partition of V and each V_i induces a path.*

Proof. In an fms-strategy of G, for each vertex, either exactly one searcher is placed on it or exactly one searcher slides to it. Thus $V_1, \ldots, V_k$ form a partition of V. Since no searcher can jump from one vertex to another, we know that the subgraph $G[V_i]$ (induced by V_i) is connected. If there are two nonadjacent vertices in $G[V_i]$ that are adjacent in G, then both vertices must be occupied by different searchers at some moment. This is a contradiction. Hence, $G[V_i]$ is an induced path in G. □

Definition 3.7. In Theorem 3.6, each induced path $G[V_i]$ is called an *fms-path* with respect to S and the set $\{G[V_1], \ldots, G[V_k]\}$ of fms-paths is called an *fms-path cover* of G with respect to S.

Corollary 3.8. *For any graph $G = (V, E)$, the number of actions in any fms-strategy is $|V|$.*

Proof. Consider an fms-strategy S of G that uses k searchers. Let $\{P_1, \ldots, P_k\}$ be an fms-path cover of G with respect to S. From the proof of Theorem 3.6, we know that each path P_i is cleared by a searcher sliding from one end to the other. If P_i is a vertex, then a searcher is placed on it and we let the placing action correspond to this vertex; otherwise, a searcher is placed on one end of P_i, say u, and the searcher slides to the other end along P_i. Then let the placing action correspond to u and let each sliding action correspond to the destination vertex of each edge on P_i. Therefore, the total number of actions in S is $|V|$. □

We now consider the subgraph induced by the vertices on any two fms-paths. Fig. 4 illustrates three families of forbidden subgraphs.

Corollary 3.9. *For an fms-strategy S of a graph G that uses k searchers, $k \ge 2$, let P be an fms-path cover of G with respect to S. For any two paths $P_1, P_2 \in P$, let H be the subgraph of G induced by vertices $V(P_1) \cup V(P_2)$. Then H has the following properties:*

(i) *H has one of the three patterns:* (a) *a forest consisting of two disjoint paths,* (b) *a tree consisting of two adjacent degree-3 vertices and all other vertices having degree one or two; and* (c) *a ladder.*

(ii) *H does not contain any graph in Fig. 4, where $a, a' \in V(P_1)$ and $b, b' \in V(P_2)$.*

Proof. (i) From Lemmas 3.1 and 3.3, we know that H must have one of the above three patterns.

(ii) Since all graphs in Fig. 4 are not forests or trees, we only need to check if any one of them is a ladder. From Definition 3.2, any ladder contains a bipolar outerplanar subgraph. Note that if a graph has a subgraph that is a subdivision of K_4 or $K_{2,3}$, then this graph is not outerplanar. In Fig. 4, since the graphs on the left and right are subdivisions of K_4 and the graph in the middle is a subdivision of $K_{2,3}$, we know that all of them are forbidden subgraphs for ladders. □

4. Lower bounds

We first give two lower bounds on the fast–mixed search number which are related to the minimum vertex degree of the graph.

Lemma 4.1. *For any graph G with ℓ leaves,* $\mathrm{fms}(G) \geq \lceil \ell/2 \rceil$.

Proof. For each leaf v of G, there must exist a searcher who either is placed on v or slides to v. If a searcher slides to v, then the only incident edge of v is cleared and the searcher cannot move along this edge again by the definition of sliding. The searcher must then stay on v until the end of the search process. Note that it is possible that a searcher starts at a leaf and stops at another leaf. Thus, $\mathrm{fms}(G) \geq \lceil \ell/2 \rceil$. □

From Theorem 3.6, we have the following lemma.

Lemma 4.2. *Let G be a graph with components $G_1, \ldots, G_k$. Then* $\mathrm{fms}(G) \geq \sum_{i=1}^{k} \delta(G_i)$.

The following lower bound is related to the numbers of vertices and edges.

Theorem 4.3. *For a graph G with n vertices and m edges,*

$$\mathrm{fms}(G) \geq n - \frac{1}{2}\sqrt{4n^2 - 4n - 8m + 1} - \frac{1}{2},$$

where the equality is held for any k-tree that has exactly two simplicial vertices.

Proof. Let $k = \mathrm{fms}(G)$ and S be an optimal fms-strategy of G. Let $\mathcal{P}$ be an fms-path cover of G with respect to S. Initially, k searchers are placed on one end of each path in $\mathcal{P}$. The number of edges they can clear is at most $k(k-1)/2$. For each sliding action along an edge in a path of $\mathcal{P}$, the number of edges the searcher can clear is at most k. Note that the total number of edges in $\mathcal{P}$ is $n-k$. Thus, the total number of edges k searchers can clear is at most $k(k-1)/2 + (n-k)k$. Hence $m \leq k(k-1)/2 + (n-k)k$, which implies that $k^2 + (1-2n)k + 2m \leq 0$. Therefore, $k \geq n - \frac{1}{2}\sqrt{4n^2 - 4n - 8m + 1} - \frac{1}{2}$.

If G is a k-tree that has exactly two simplicial vertices, then $m = k(k-1)/2 + (n-k)k$. It follows from Corollary 7.9 that $\mathrm{fms}(G) = n - \frac{1}{2}\sqrt{4n^2 - 4n - 8m + 1} - \frac{1}{2}$. □

Recall that we use $\mathrm{ipc}(G)$ to denote the induced-path number of G. From Theorem 3.6, we have the following lower bound.

Lemma 4.4. *For any graph G,* $\mathrm{fms}(G) \geq \mathrm{ipc}(G)$.

Since every fms-strategy is also a mixed search strategy, we have the following result.

Lemma 4.5. *For any graph G,* $\mathrm{fms}(G) \geq \mathrm{ms}(G)$.

It is easy to see that the family of graphs $\{G : \mathrm{fms}(G) \leq k\}$ is not subgraph-closed for any integer $k \geq 1$. Furthermore, the family of graphs $\{G$: G is connected and $\mathrm{fms}(G) \leq k\}$ is not subgraph-closed either for any integer $k \geq 2$. This can be shown by the graph in Fig. 5, in which $\mathrm{fms}(G) = 2$ but $\mathrm{fms}(H) = 3$. Although the fast–mixed search number of a graph may be less than that of its subgraph, we can show that the fast–mixed search number of a graph is always greater than or equal to that of its subgraphs that are grids or cliques.

Corollary 4.6. (i) *If clique K_k ($k \geq 2$) is a minor of G, then* $\mathrm{fms}(G) \geq k - 1$.

(ii) *If grid $G_{k\times\ell}$ ($2 \leq k \leq \ell$) is a minor of G, then* $\mathrm{fms}(G) \geq k$.

Proof. Since the family of graphs $\{G$: $\mathrm{ms}(G) \leq k\}$ is minor-closed, from Lemma 4.5 we know that $\mathrm{fms}(G) \geq \mathrm{ms}(G) \geq \mathrm{ms}(K_k) = k - 1$ if K_k is a minor of G. Similarly, $\mathrm{fms}(G) \geq \mathrm{ms}(G) \geq \mathrm{ms}(G_{k\times\ell}) = k$ if $G_{k\times\ell}$ is a minor of G. □

5. Relations to fast searching and mixed searching

We first consider the relationship between fast–mixed searching and fast searching.

On one hand, there are graphs whose fast–mixed search number is arbitrarily bigger than their fast search number. Consider a complete bipartite graph $K_{1,n}$, where $n \geq 2$. We can show that $\mathrm{fms}(K_{1,n}) = n - 1$ while $\mathrm{fs}(K_{1,n}) = \lceil \frac{n}{2} \rceil$. On the other hand, there are graphs whose fast–mixed search number is arbitrarily smaller than its fast search number. Consider the standard ladder L_n (see Fig. 6(a)). It is easy to see that $\mathrm{fms}(L_n) = 2$, but $\mathrm{fs}(L_n) = n + 2$.

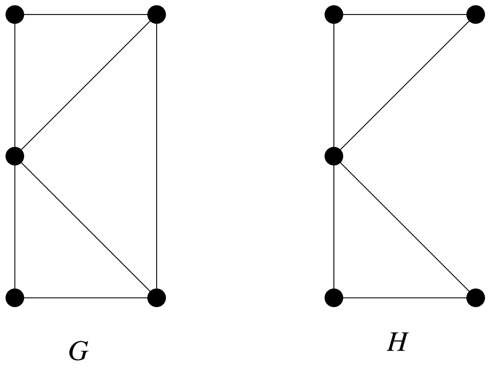

Fig. 5. H is a subgraph of G with $\mathrm{fms}(H) > \mathrm{fms}(G)$.

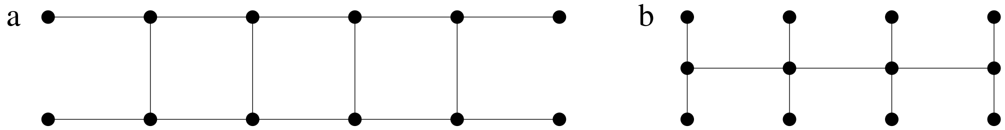

Fig. 6. (a) A standard ladder L_4 with $\mathrm{fms}(L_4) = 2$ and $\mathrm{fs}(L_4) = 6$. (b) A caterpillar H_4 with $\mathrm{fms}(H_4) = 4$ and $\mathrm{ms}(H_4) = 2$.

With regard to complexity, the fast–mixed search problem is also different from the fast search problem. It may seem that finding the fast–mixed search number is easier than finding the fast search number for sparse graphs, but this may not be true. From [13], we know that the fast search number of graphs with maximum degree 3 can be found in polynomial time. However, we will show that the problem of finding the fast–mixed search number of planar graphs with maximum degree 3 is NP-complete if all searchers start from leaves.

We now consider the relationship between fast–mixed searching and mixed searching. From Lemma 4.5, we know that the fast–mixed search number is always greater than or equal to the mixed search number for any graph. However, there are graphs whose optimal fms-strategies look very different from their optimal mixed search strategies. Consider a caterpillar H_n with a spine on n vertices, $n \geq 3$, such that each vertex on the spine has two pendent edges (see Fig. 6(b)). The optimal fms-strategy of H_n is to place n searchers on the bottom n leaves and slide them to the n vertices on the spine, and then slide them to the top n leaves. But the optimal mixed search strategy of H_n is to place a searcher on one end vertex of the spine and move it to the other end vertex of the spine passing through all degree-4 vertices, and meanwhile place another searcher on each leaf one by one. We can generalize this observation to the following.

Theorem 5.1. *Given a graph G that contains at least one edge, let G' be a graph obtained from G by adding two pendent edges on each vertex. Then* $\mathrm{ms}(G') \leq \mathrm{ms}(G) + 1$ *and* $\mathrm{fms}(G') = |V(G)|$.

Proof. Let S be an optimal monotonic mixed search strategy of G. We can obtain a mixed search strategy of G' by adding $2|V(G)|$ placing actions. That is, for each vertex of G' that is also a vertex of G, when it is occupied first time by a searcher, we use an additional searcher to clear the two pendent edges of the vertex by two placing actions on leaves. Thus, $\mathrm{ms}(G') \leq \mathrm{ms}(G) + 1$. For the fast–mixed search of G', it is easy to see that $\mathrm{fms}(G') \leq |V(G)|$. It follows from Lemma 4.1 that $\mathrm{fms}(G') = |V(G)|$. □

From Theorem 5.1, we know that the optimal mixed search strategy of G' is very close to that of G because their search numbers differ by at most one. However, the optimal fms-strategy of G' is always using $|V(G)|$ searchers to clear the two pendent edges on each vertex of G, which is independent of the optimal fms-strategy of G.

For some planar graphs, the optimal fms-strategy can be very different from the optimal mixed search strategy, the optimal fast search strategy, and the optimal induced-path cover. For example, the planar graph G in Fig. 7 can be cleared using 6 searchers in the mixed search model, that is, using 5 searchers to clear the five rows and one more searcher to clear all degree-2 vertices on columns. In the fast search model, we can clear G using 10 searchers who clear the seven internal columns and three internal rows and using another 2 searchers to clear the bottom row, rightmost column, leftmost column, and top row [13]. In the fast–mixed search model, we can clear G using 9 searchers who clear the nine columns respectively. For the induced-path cover of G, the five induced paths, which correspond to the five dashed paths, contains all vertices of G.

6. NP-completeness

We study the following three problems in this section.

Fast–Mixed Searching
Instance: A graph $G = (V, E)$ and a nonnegative integer k.
Question: Is $\mathrm{fms}(G) \leq k$?

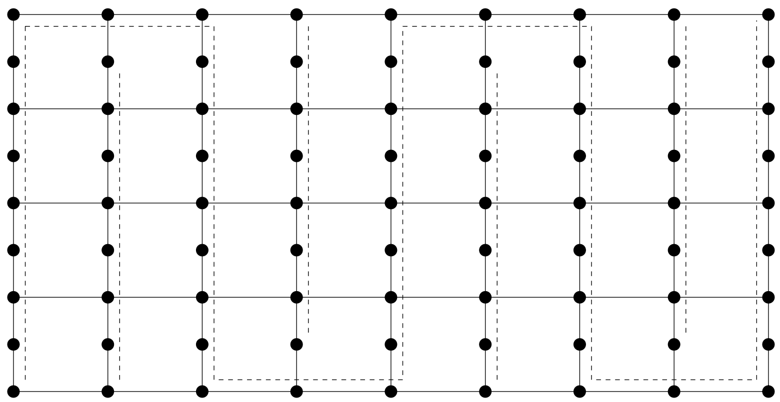

Fig. 7. A planar graph G with $\mathrm{fms}(G) = 9$, $\mathrm{fs}(G) = 12$, $\mathrm{ms}(G) = 6$, and $\mathrm{ipc}(G) \leq 5$.

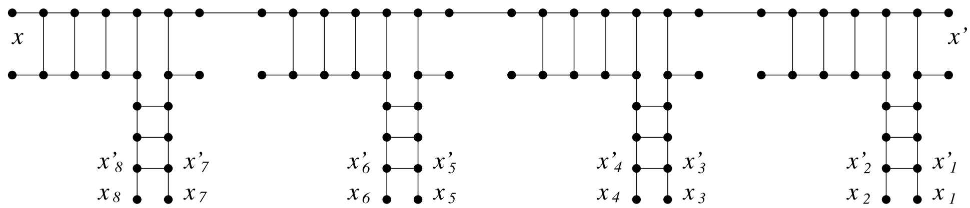

Fig. 8. A variable gadget G^k with four legs ($k = 4$).

FAST–MIXED SEARCHING BETWEEN LEAVES
Instance: A graph $G = (V, E)$ with ℓ leaves.
Question: Is $\mathrm{fms}(G) = \lceil \ell/2 \rceil$?

FAST–MIXED SEARCHING FROM LEAVES
Instance: A graph $G = (V, E)$ and a nonnegative integer k.
Question: Is there an fms-strategy that uses k searchers to clear G such that each searcher starts from a leaf of G?

We will prove the NP-completeness of these problems by reductions either from POSITIVE PLANAR 1-IN-3-SAT or from POSITIVE PLANAR 2-IN-4-SAT. The positive planar 1-in-3-SAT (resp. 2-in-4-SAT) problem can be described as follows. Let ϕ be a boolean formula in the conjunctive normal form (CNF) with m clauses $\{c_1, \ldots, c_m\}$ and n variables $\{x_1, \ldots, x_n\}$ such that each clause contains exactly three (resp. four) literals. The incident graph of ϕ is the bipartite graph with vertex set $\{c_1, \ldots, c_m, x_1, \ldots, x_n\}$ and edge set $\{c_i x_j$: clause c_i contains variable $x_j\}$. We say that ϕ is *planar* if its incident graph is planar, and ϕ is *positive* if it contains no negations of variables. A truth assignment of ϕ is 1-in-3 (resp. 2-in-4) satisfying if each clause has exactly one (resp. two) true literals, and ϕ is *1-in-3* (resp. *2-in-4*) *satisfiable* if there is a 1-in-3 (resp. 2-in-4) satisfying truth assignment. POSITIVE PLANAR 1-IN-3-SAT (or 2-IN-4-SAT) is defined as follows.

POSITIVE PLANAR 1-IN-3-SAT (resp. POSITIVE PLANAR 2-IN-4-SAT)
Instance: A positive planar boolean formula ϕ in CNF such that each clause contains exactly three (resp. four) literals.
Question: Is ϕ 1-in-3 (resp. 2-in-4) satisfiable?

Let G^k be a graph with k "legs" as illustrated in Fig. 8. We call graph G^k a *variable gadget*, which corresponds to a boolean variable that appears k times in a CNF-formula. We first show a property of variable gadgets as follows.

Lemma 6.1. *Let G^k be a variable gadget as illustrated in Fig. 8. Then* $\mathrm{fms}(G^k) = 2k+1$. *Furthermore, for any optimal fms-strategy of G^k, if a searcher slides from x to its neighbor, then for each leaf x_i ($1 \leq i \leq 2k$) there is a searcher sliding from x'_i to x_i; and if a searcher slides to x from its neighbor, then for each leaf x_i ($1 \leq i \leq 2k$) there is a searcher sliding from x_i to x'_i.*

Proof. Refer to Fig. 8. We first give an fms-strategy to clear G^k using $2k + 1$ searchers. Place $2k$ searchers on vertices x_i ($1 \leq i \leq 2k$), slide them from all x_i to their neighbors and then keep sliding them up to the second row. We have cleared all edges on "legs". We then place one searcher on vertex x' and move it to vertex x along the top row edges; in the meantime, we move the $2k$ searchers on the second row accordingly to clear all edges between the top row and the second row. Finally, each of the $2k$ searchers arrives at a different leaf, occupying all the leaves on the second row. Thus, $\mathrm{fms}(G^k) \leq 2k+1$. On the other hand, it follows from Lemma 4.1 that $\mathrm{fms}(G^k) \geq 2k+1$. Therefore, $\mathrm{fms}(G^k) = 2k+1$. Suppose that there is an optimal fms-strategy S, in which a searcher slides from x to its neighbor. Since S is an optimal fms-strategy and $\mathrm{fms}(G^k) = 2k + 1$, we know that S must use $2k + 1$ searchers to clear G^k. Since G^k has $4k + 2$ leaves, every searcher must start from a leaf and end at another leaf. If there is a searcher sliding from x_i to x'_i for some i, then we need at least $2k + 2$ searchers to clear G^k.

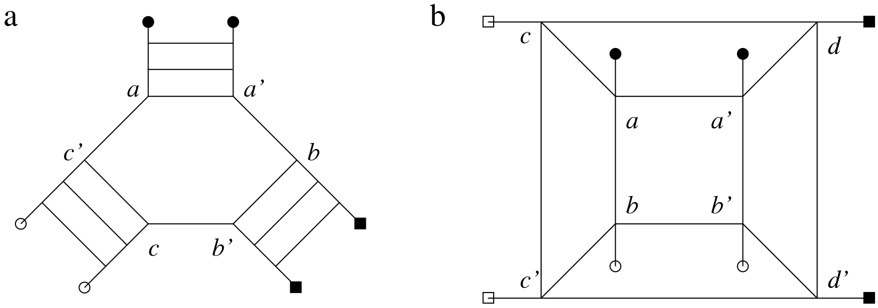

Fig. 9. (a) A clause gadget G_c in Theorem 6.3. (b) A clause gadget H_c in Theorem 6.5.

This is a contradiction. Therefore, if a searcher slides from x to its neighbor, then for each leaf x_i ($1 \le i \le 2k$) there is a searcher sliding to x_i from its neighbor.

Similarly, we can show that if a searcher slides to x from its neighbor in an optimal fms-strategy, then for each leaf x_i ($1 \le i \le 2k$) there is a searcher sliding from x_i to its neighbor. □

Let G_c be a graph illustrated in Fig. 9(a). We call graph G_c a *clause gadget*, which corresponds to a clause c in a CNF-formula. Note that G_c has three pairs of leaves and each pair is marked using the same pattern, i.e., solid circles, solid squares or hollow circles. Each pair of leaves corresponds to a literal in c. G_c has the following property.

Lemma 6.2. *Let G_c be a graph as illustrated in Fig. 9(a). Then* $\text{fms}(G_c) = 4$ *and in any optimal fms-strategy of G_c, either two searchers start from a pair of leaves marked with the same pattern and four searchers end on the other two pairs of leaves, or two searchers end on a pair of leaves marked with the same pattern and four searchers start from the other two pairs of leaves.*

Proof. Refer to Fig. 9(a). Because of the symmetric structure of G_c, we only need to consider the following two cases in optimal fms-strategies.

1. Place two searchers on the two solid-circled leaves. Slide them to vertices a and a', and then to c' and b respectively. Place two searchers on vertices b' and c. Finally, slide all searchers to the four leaves and G_c is cleared.
2. Place four searchers on the two solid-circled leaves and two solid-squared leaves. Slide them to vertices a, a', b, b'. Slide the searchers on a and b' along ac' and $b'c$ to c' and c respectively. Slide the searchers on c and c' to the two hollow-circled leaves and G_c is cleared.

It is not hard to see that for any other fms-strategy of G_c, e.g., three searchers start from three leaves and three searchers end on the other three leaves, we need at least five searchers to clear G_c. □

Theorem 6.3. Fast–Mixed Searching From Leaves *is NP-complete, and it remains NP-complete for planar graphs with maximum degree 3.*

Proof. It is easy to verify that the problem is in NP. To show NP-hardness, we will construct a reduction from Positive Planar 1-in-3-SAT which is NP-complete [12]. Let ϕ be an instance of Positive Planar 1-in-3-SAT with m clauses $\{c_1, \ldots, c_m\}$ and n variables $\{x_1, \ldots, x_n\}$. We now construct a planar graph with maximum degree 3, which is an instance of Fast–Mixed Searching From Leaves. For each clause c_j, $1 \le j \le m$, we construct the corresponding clause gadget G_{c_j} such that leaves marked using the same pattern (e.g., solid circles) correspond to the same literal in c_j (see Fig. 9(a)). For each variable x that appears k times in ϕ, we construct the corresponding variable gadget G_x^k such that leaves x_{2i-1} and x_{2i} correspond to the i-th occurrence of the variable x (see Fig. 8). If the i-th occurrence of the variable x is in clause c_j, then we use a standard ladder to connect leaves x_{2i-1} and x_{2i} to the leaves marked using the same pattern in G_{c_j}. For each clause gadget, all leaves become degree-2 vertices after they are linked by three standard ladders. In polynomial time, we can construct a planar graph with maximum degree 3, denoted by G_ϕ (see Fig. 10). Let $s = n + 4m$. We next show that the positive planar formula ϕ is 1-in-3 satisfiable if and only if there is an fms-strategy that uses s searchers to clear G_ϕ such that each searcher starts from a leaf.

Suppose that the planar positive formula ϕ is 1-in-3 satisfiable. Consider a 1-in-3 satisfying truth assignment of ϕ. For each variable x whose value is false and which appears k times in ϕ, we clear the variable gadget G_x^k by sliding a searcher from x to x' to clear all edges on the top row. In the meantime, we slide $2k$ searchers from each leaf on the second row to clear edges on the second row, edges between top and second rows, and edges on all legs, until each leaf x_j, $1 \le j \le 2k$, is occupied by a searcher. We then slide $2k$ searchers along the standard ladders to clause gadgets. After we clear all variable gadgets that correspond to variables with false value, and all standard ladders linking to them, each clause gadget has exactly four searchers on two pairs of leaves, which correspond to the two false literals in the clause. For each clause gadget G_c, we slide the four searchers to the 6-cycle in the middle, and then slide two of them to their contaminated neighbors on the 6-cycle. Then we clear all edges on the 6-cycle and slide the two searchers to the remaining pair of leaves marked with the same pattern. Note that this pair of leaves correspond to the true literal. Then we slide all $2m$ searchers along standard ladders to variable gadgets, which correspond to variables with true value. Finally, we clear each true variable gadget G_x^k by sliding $2k$ searchers to clear all leg edges, in the meantime sliding a searcher from x' to x to clear all edges on the top row, edges on

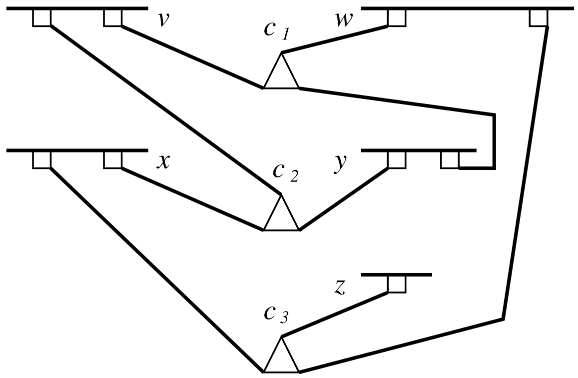

Fig. 10. The graph G_ϕ constructed for $\phi = c_1 \wedge c_2 \wedge c_3$, where $c_1 = (v \vee w \vee y)$, $c_2 = (v \vee x \vee y)$ and $c_3 = (w \vee x \vee z)$, where each square represents a leg of a variable gadget and each triangle represents a clause gadget.

the second row, and edges between top and second rows. Thus, we use s searchers to clear G_ϕ such that each searcher starts from a leaf.

Conversely, suppose that there is an fms-strategy S that uses s searchers to clear G_ϕ such that each searcher starts from a leaf. Note that G_ϕ has $2n + 6m$ leaves and every leaf v must be associated with a searcher who either slides from v to its neighbor or slides to v from its neighbor. Since $s = n + 4m$, we know that $n + 2m$ searchers end on leaves. From the structure of G_c and the three standard ladders connecting to it, there must be two searchers starting from the middle 6-cycle or ending on it. If two searchers start from the middle 6-cycle of G_c, they must move to two leaves of variable gadgets through ladders; if two searchers end on the middle 6-cycle, they must move from two leaves of variable gadgets through ladders. Since $n + 4m$ searchers start from leaves and $n + 2m$ of them end on the remaining leaves, we know that there must be two searchers ending in the middle 6-cycle of each clause gadget. It follows from Lemma 6.1 that each variable gadget that has k legs must be cleared by $2k + 1$ searchers such that either there is a searcher sliding from x to its neighbor and there is a searcher sliding from x_i' to x_i $(1 \le i \le 2k)$, or there is a searcher sliding to x from its neighbor and there is a searcher sliding from x_i to x_i' $(1 \le i \le 2k)$. Thus, in the fms-strategy S, for the $n + 4m$ searchers that start from leaves of variable gadgets, $2m$ of them end on leaves of other variable gadgets and another $2m$ of them end on clause gadgets. For each clause gadget, there are four searchers coming from variable gadgets, so we assign the two literals and their corresponding variables to be false and we assign the remaining literal and its corresponding variable to be true. In this way, we can find a 1-in-3 satisfying truth assignment of ϕ. Hence, ϕ is 1-in-3 satisfiable. □

In the remainder of this section, we prove that Fast–Mixed Searching Between Leaves and Fast–Mixed Searching are NP-complete. Let H_c be a clause gadget as illustrated in Fig. 9(b), which corresponds to a clause c. Note that H_c has four pairs of leaves and each pair is marked using the same pattern, i.e., solid squares, hollow squares, solid circles and hollow circles. Each pair of leaves corresponds to a literal in c. H_c has the following property.

Lemma 6.4. *Let H_c be a graph as illustrated in Fig. 9(b). Then* $\mathrm{fms}(H_c) = 4$ *and in any optimal fms-strategy of H_c, four searchers must start from two pairs of leaves marked with two patterns.*

Proof. Refer to Fig. 9(b). Since H_c has 8 leaves, it follows from Lemma 4.1 that $\mathrm{fms}(H_c) \ge 4$. We will describe fms-strategies in which four searchers start from four leaves and end on the other four leaves to clear H_c. Thus, $\mathrm{fms}(H_c) = 4$. Because of the symmetric structure of the cube, we only need to consider the following two cases in optimal fms-strategies.

1. Place four searchers on the two solid-circled leaves and the two hollow-circled leaves. Slide them one by one to vertices a, a', b, b', and then slide them to vertices c, d, c', d'. Finally, slide them to the four leaves and H_c is cleared.
2. Place four searchers on the two solid-circled leaves and the two hollow-squared leaves. Slide them one by one to vertices a, a', c, c'. Slide the searchers on a and c along ab and cd to b and d respectively. Slide the searchers on a' and c' along $a'b'$ and $c'd'$ to b' and d' respectively. Finally, slide the four searchers to the four leaves and H_c is cleared.

It is not hard to see that for any fms-strategy of H_c in which the four searchers are not placed on two pairs of leaves marked with two patterns, we need at least five searchers to clear H_c. □

We now use variable gadgets G_x^k and clause gadgets H_c to show the NP-completeness result of Fast–Mixed Searching Between Leaves. The reduction is similar to the one used in the proof of Theorem 6.3.

Theorem 6.5. Fast–Mixed Searching Between Leaves *is NP-complete, and it remains NP-complete for graphs with maximum degree 4.*

Proof. We will show that the problem is NP-hard by a reduction from Positive Planar 2-in-4-SAT, which is NP-complete [5]. Let ϕ be an instance of Positive Planar 2-in-4-SAT with m clauses $\{c_1, \ldots, c_m\}$ and n variables $\{x_1, \ldots, x_n\}$. For each clause c_j, $1 \le j \le m$, we construct the corresponding clause gadget H_{c_j} such that leaves marked using the same pattern correspond

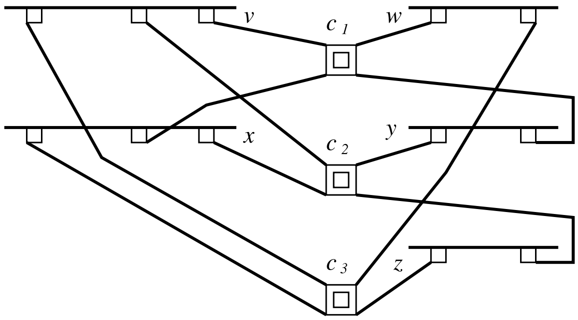

Fig. 11. The graph H_ϕ constructed for $\phi = c_1 \wedge c_2 \wedge c_3$, where $c_1 = (v \vee w \vee x \vee y)$, $c_2 = (v \vee x \vee y \vee z)$ and $c_3 = (v \vee w \vee x \vee z)$, where each square represents a leg of a variable gadget and each double square represents a clause gadget.

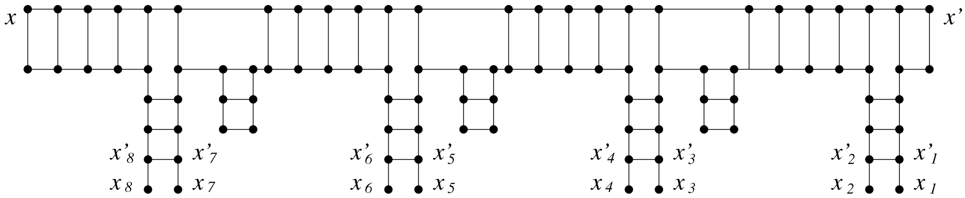

Fig. 12. A variable gadget G^k with four legs ($k = 4$).

to the same literal in c_j (see Fig. 9(b)). For each variable x that appears k times in ϕ, we construct the corresponding variable gadget G_x^k such that leaves x_{2i-1} and x_{2i} correspond to the i-th occurrence of the variable x (see Fig. 8). If the i-th occurrence of the variable x is in clause c_j, then we use a standard ladder to connect leaves x_{2i-1} and x_{2i} to the leaves marked using the same pattern in H_{c_j}. For each clause gadget, all leaves become degree-2 vertices after they are linked by four standard ladders. In polynomial time, we can construct a graph with maximum degree 4, denoted by H_ϕ (see Fig. 11). We then show that the positive planar formula ϕ is 2-in-4 satisfiable if and only if $\mathrm{fms}(H_\phi) = n + 4m$.

Suppose that the positive planar formula ϕ is 2-in-4 satisfiable. Consider a 2-in-4 satisfying truth assignment of ϕ. For each variable x whose value is true and which appears k times in ϕ, we clear the variable gadget G_x^k using $2k + 1$ searchers. We then slide $2k$ searchers along the standard ladders to clause gadgets. After we clear all variable gadgets that correspond to variables with true value and all standard ladders linking to them, each clause gadget has four searchers on two pairs of leaves marked with two patterns, which correspond to the two true literals in the clause. From Lemma 6.4, we can clear each clause gadget H_c using the four searchers such that the other two pairs of leaves are occupied when H_c is cleared. Note that these two pairs of leaves correspond to the two false literals. Then we slide all $4m$ searchers along standard ladders to variable gadgets, which correspond to variables with false value. Finally, we clear each false variable gadget G_x^k using $2k + 1$ searchers. Thus, we can use $n + 4m$ searchers to clear H_ϕ. On the other hand, since H_ϕ has $2n + 8m$ leaves, it follows from Lemma 4.1 that we need at least $n + 4m$ searchers to clear H_ϕ. Hence, $\mathrm{fms}(H_\phi) = n + 4m$.

Conversely, suppose that $\mathrm{fms}(H_\phi) = n + 4m$. Since H_ϕ has $2n + 8m$ leaves, we know that every leaf v must be associated with a searcher who either starts from v or ends on v. Thus, each variable gadget that has k legs must be cleared by $2k + 1$ searchers and no searcher can stay in any vertex of clause gadgets at the end of the whole searching process. Hence, from Lemma 6.4, each clause gadget must be cleared by four searchers and these four searchers must start from two pairs of leaves marked with two patterns and end at the other two pairs of leaves. We can assign true to the two literals corresponding to the two pairs of leaves where four searchers start, and assign false to the two literals corresponding to the two pairs of leaves where four searchers end. Since $\mathrm{fms}(H_\phi) = n + 4m$, we can find a 2-in-4 satisfying truth assignment of ϕ. Thus, ϕ is 2-in-4 satisfiable. □

Note that the leaves in variable gadgets just make the argument easy because each leaf is associated with a searcher. If we modify variable gadgets in a way as illustrated in Fig. 12, then the reduction graph in Fig. 11 becomes biconnected with maximum degree 4, which is denoted by H'_ϕ. We still can show that the positive planar formula ϕ is 2-in-4 satisfiable if and only if $\mathrm{fms}(H'_\phi) \le n + 4m$. Therefore, we have the following result.

Corollary 6.6. Fast–Mixed Searching *is NP-complete, and it remains NP-complete even for biconnected graphs with maximum degree 4.*

From the structure of the reduction graph in the proof of Theorem 6.5, we can obtain a similar result for the induced-path cover problem. Given a graph G with k leaves, it is easy to see that $\mathrm{ipc}(G) \ge \lceil k/2 \rceil$. The problem is to decide if $\mathrm{ipc}(G) = \lceil k/2 \rceil$.

Corollary 6.7. *Given a graph G with k leaves, the problem of determining whether there are $\lceil k/2 \rceil$ vertex-disjoint induced-paths that contain all the vertices of G is NP-complete. It remains NP-complete for graphs of maximum degree 4.*

7. Special classes of graphs

First we have the following results for complete graphs and grids.

Lemma 7.1. (i) *For a complete graph K_n ($n \geq 2$),* $\mathrm{fms}(K_n) = n - 1$.
(ii) *For a grid $G_{k\times\ell}$ with k rows and ℓ columns ($2 \leq k \leq \ell$),* $\mathrm{fms}(G_{k\times\ell}) = k$.

Proof. From Lemma 4.5, we have $\mathrm{fms}(K_n) \geq \mathrm{ms}(K_n) = n - 1$ and $\mathrm{fms}(G_{k\times\ell}) \geq \mathrm{ms}(G_{k\times\ell}) = k$. It is easy to see that we can clear K_n using $n - 1$ searchers, and clear $G_{k\times\ell}$ using k searchers in the fast–mixed search model. □

Although a complete graph can have many more edges than its spanning complete bipartite subgraphs (e.g., a star $K_{1,n-1}$), their fast–mixed search numbers differ by only 1.

Lemma 7.2. *For a complete bipartite graph $K_{m,n}$, if $n \geq m \geq 1$ and $n > 1$, then* $\mathrm{fms}(K_{m,n}) = m + n - 2$.

Proof. Let (U, V) with $|U| = m$ and $|V| = n$ be a partition of the vertex set of $K_{m,n}$ such that all edges are between U and V. Let $u \in U$ and $v \in V$. Place $n + m - 2$ searchers on each vertex of $K_{m,n}$ except u and v. Slide a searcher from a vertex in V to the vertex u, and then slide this searcher from u to v. Now the graph becomes cleared. Thus $\mathrm{fms}(K_{m,n}) \leq m + n - 2$.

We now show that $\mathrm{fms}(K_{m,n}) \geq m + n - 2$. Let $\mathcal{P}$ be an fms-path cover of $K_{m,n}$. For a complete bipartite graph, it is easy to see that each induced path has at most 2 edges. If there are two paths in $\mathcal{P}$ such that one of them contains two edges and the other contains at least one edge, then the subgraph induced by the vertices of the two paths contains the graph in the middle of Fig. 4. This contradicts case (ii) of Corollary 3.9. If there are three paths in $\mathcal{P}$ such that each of them contains exactly one edge, the subgraph induced by the vertices of the three paths cannot be cleared by three searchers. This is a contradiction. Hence, there are only three cases for $\mathcal{P}$: only one path in $\mathcal{P}$ has length 2 and all others have length 0, exactly two paths in $\mathcal{P}$ have length 1 and all others have length 0, or at most one path in $\mathcal{P}$ has length 1 and all others have length 0. Thus $|\mathcal{P}| \geq m + n - 2$, and therefore $\mathrm{fms}(K_{m,n}) = m + n - 2$. □

For trees, we can show the following relation between the fast–mixed search and the induced-path cover.

Theorem 7.3. *For a tree T,* $\mathrm{fms}(T) = \mathrm{ipc}(T)$.

Proof. From Lemma 4.4, we know that $\mathrm{fms}(T) \geq \mathrm{ipc}(T)$. We only need to show that $\mathrm{fms}(T) \leq \mathrm{ipc}(T)$. We will show this inequality by induction. If $\mathrm{ipc}(T) = 1$, then T is a path and it is easy to see that $\mathrm{fms}(T) = 1$. Suppose that $\mathrm{fms}(T) \leq \mathrm{ipc}(T)$ for any tree T with $\mathrm{ipc}(T) \leq k$.

We now consider a tree T with $\mathrm{ipc}(T) = k + 1$. Let $\mathcal{P}$ be an optimal induced-path cover of T. Since $k \geq 1$, we know that T is not a path. Thus, there must exist a vertex u of degree at least 3 in T that has at least two paths, $u \sim v$ and $u \sim w$, attached on it, where v and w are leaves of T and all internal vertices of $u \sim v$ and $u \sim w$ have degree 2 in T. Note that each leaf of T is an end of some path in $\mathcal{P}$. We have two cases for $\deg(u)$.

1. $\deg(u) = 3$. If $u \sim v$ and $u \sim w$ form a path P in $\mathcal{P}$, then let T' be a tree obtained from T by deleting all vertices of P. Since $\mathcal{P} \setminus \{P\}$ is an induced-path cover of T', we know that $\mathrm{ipc}(T') \leq k$. By the induction hypothesis, $\mathrm{fms}(T') \leq k$. We can modify an optimal fms-strategy of T' to obtain an fms-strategy of T such that P is cleared by an extra searcher. Thus, $\mathrm{fms}(T) \leq k + 1$. If $u \sim v$ and $u \sim w$ do not form a path in $\mathcal{P}$, let v' be a neighbor of u on $u \sim v$ and w' be a neighbor of u on $u \sim w$. Then one of the two paths $v \sim v'$ and $w \sim w'$ must be a path in $\mathcal{P}$. Without loss of generality, suppose that $v \sim v'$ is a path in $\mathcal{P}$. Let T' be a tree obtained from T by deleting all vertices of $v \sim v'$. Similarly, we can show that $\mathrm{fms}(T) \leq \mathrm{fms}(T') + 1 \leq k + 1$.
2. $\deg(u) > 3$. In this case, there must exist a path $v \sim v'$ in $\mathcal{P}$ such that v is a leaf of T, v' is a neighbor of u, and all internal vertices of $v \sim v'$ have degree 2 in T. Let T' be a tree obtained from T by deleting all vertices of $v \sim v'$. Similarly to Case 1, we can show that $\mathrm{fms}(T) \leq \mathrm{fms}(T') + 1 \leq k + 1$.

From the above, we know that $\mathrm{fms}(T) \leq \mathrm{ipc}(T)$ for any tree T with $\mathrm{ipc}(T) = k + 1$. Therefore, $\mathrm{fms}(T) \leq \mathrm{ipc}(T)$ for any tree T. □

From [10], an optimal path cover of a tree can be computed in linear time. Note that an optimal path cover of a tree is also an optimal induced-path cover of the tree. After we obtain an optimal path cover, we can use the method on the base of the induction used in the proof of Theorem 7.3 to compute an optimal fms-strategy in linear time.

Corollary 7.4. *For any tree, the fast–mixed search number and an optimal fms-strategy can be computed in linear time.*

We now consider cacti. A *cactus* is a connected graph in which any two simple cycles have at most one vertex in common.

Theorem 7.5. *For a cactus G,* $\mathrm{fms}(G) = \mathrm{ipc}(G)$.

Proof. Similarly to the proof of Theorem 7.3, we only need to show that $\mathrm{fms}(G) \leq \mathrm{ipc}(G)$ by induction. If $\mathrm{ipc}(G) = 1$, then G is a path and it is easy to see that $\mathrm{fms}(G) = 1$. If $\mathrm{ipc}(G) = 2$, then G is a special tree in Lemma 3.1, or G is a ladder. Thus $\mathrm{fms}(G) = 2$. Suppose that $\mathrm{fms}(G) \leq \mathrm{ipc}(G)$ for any cactus G with $\mathrm{ipc}(G) \leq k$.

We now consider a cactus G with $\mathrm{ipc}(G) = k + 1$, where $k \geq 2$. Let $\mathcal{P}$ be an optimal induced-path cover of G. If there exists a vertex u in G that has at least two paths, $u \sim v$ and $u \sim w$, attached to it, where v and w are leaves of G, and all internal vertices of $u \sim v$ and $u \sim w$ have degree 2 in G, then, similar to the proof of Theorem 7.3, we can show that

$\text{fms}(G) \leq k+1$. Otherwise, since $\text{ipc}(G) \geq 3$, there must exist a cycle C in G such that, among all connected components in the graph $G - E(C)$, at most one component contains cycles. We have the following cases on components attached on C.

1. There is only one vertex of C having degree more than 2 in G. Then there must exist a path $P \in \mathcal{P}$ both of whose ends are on C. Let G' be a graph obtained from G by deleting all vertices of P. Since $\mathcal{P} \setminus \{P\}$ is an induced-path cover of G', we know that $\text{ipc}(G') \leq k$. By the induction hypothesis, $\text{fms}(G') \leq k$. We can modify an optimal fms-strategy of G' to obtain an fms-strategy of G such that P is cleared by an extra searcher. Thus, $\text{fms}(G) \leq k+1$.
2. There are vertices $u_1, \ldots, u_i \in V(C)$, $i \geq 2$, with $\deg_G(u_j) \geq 3$, $1 \leq j \leq i$. Notice that at most one of these vertices, say u_i, has a component containing cycles attached to it, each other vertex u_j $(1 \leq j \leq i-1)$ has a path $u_j \sim v_j$ attached to it, where $\deg_G(u_j) = 3$, $\deg_G(v_j) = 1$, and all internal vertices of $u_j \sim v_j$ have degree 2 in G. Then there must exist a path $P \in \mathcal{P}$ such that either $V(P)$ belongs to $V(C)$, or one end of P is on C and the other one is a leaf of G, or each end of P is a leaf of G and a subpath of P is on C. Similarly to Case 1, we can show that $\text{fms}(G) \leq k+1$.

From the above, we know that $\text{fms}(G) \leq \text{ipc}(G)$ for any cactus G with $\text{ipc}(G) = k+1$. Therefore, $\text{fms}(G) \leq \text{ipc}(G)$ for any cactus G. $\square$

From [11], we have a linear-time algorithm to compute an induced-path cover of a block graph with every block being a cycle. By modifying the algorithm in [11], we can compute an optimal fms-strategy of any cactus in linear time.

Corollary 7.6. *For any cactus, the fast–mixed search number and an optimal fms-strategy can be computed in linear time.*

Let $\{I_1, \ldots, I_n\}$ be a collection of intervals on the real line. The corresponding interval graph is $G = (V, E)$, where $V = \{I_1, \ldots, I_n\}$ and $\{I_i, I_j\} \in E$ if and only if $I_i \cap I_j \neq \emptyset$. An interval graph is called *proper* if no interval in $\{I_1, \ldots, I_n\}$ properly contains any other interval.

The fast–mixed search number can be arbitrarily larger than the induced-path number on interval graphs. For example, for an interval graph K_n, $n \geq 2$, from Lemma 7.1(i), we know that $\text{fms}(K_n) = n-1$, but $\text{ipc}(K_n) = \lceil n/2 \rceil$.

Theorem 7.7. *If $G = (V, E)$ is a proper interval graph containing at least one edge, then* $\text{fms}(G) = |V| - m$*, where m is the number of maximal cliques in G.*

Proof. If G is not connected, we can consider each component. If $m = 1$, it follows from Lemma 7.1 that $\text{fms}(G) = |V(G)| - 1$. Suppose that G is connected and contains at least 2 maximal cliques. From [3] we know that the maximal cliques of G can be ordered $C_1, C_2, \ldots, C_m$ such that for any $v \in V(C_i) \cap V(C_k)$, $1 \leq i < k \leq m$, the vertex v is contained in all C_j, $i \leq j \leq k$. Since G is a proper interval graph, the intersection of any three cliques of $C_1, C_2, \ldots, C_m$ must be empty. Thus, we can choose $m+1$ different vertices b_i $(1 \leq i \leq m+1)$ such that $b_1 \in V(C_1) \setminus V(C_2)$, $b_{i+1} \in V(C_i) \cap V(C_{i+1})$ $(1 \leq i \leq m-1)$, and $b_{m+1} \in V(C_m) \setminus V(C_{m-1})$. Place $|V(C_1)| - 1$ searchers on each vertex of $C_1 \setminus \{b_2\}$. Slide the searcher on b_1 to b_2. Place $|V(C_2) \setminus V(C_1)| - 1$ searchers on each vertex of $(V(C_2) \setminus V(C_1)) \setminus \{b_3\}$. Slide the searcher on b_2 to b_3. Then C_1 and C_2 are cleared by $|V(C_1) \cup V(C_2)| - 2$ searchers such that every vertex in $(V(C_1) \cup V(C_2)) \setminus \{b_1, b_2\}$ is occupied. We can keep this process until $C_1, \ldots, C_{m-1}$ are cleared by $|V(C_1) \cup \cdots \cup V(C_{m-1})| - (m-1)$ searchers such that every vertex in $(V(C_1) \cup \cdots \cup V(C_{m-1})) \setminus \{b_1, \ldots, b_{m-1}\}$ is occupied. Place $|V(C_m) \setminus V(C_{m-1})| - 1$ searchers on each vertex of $(V(C_m) \setminus V(C_{m-1})) \setminus \{b_{m+1}\}$. Slide the searcher on b_m to b_{m+1}. Then G is cleared by $|V(G)| - m$ searchers.

We now show that $\text{fms}(G) \geq |V(G)| - m$. It follows from Corollary 3.9 that in each clique, at most one searcher can slide from one vertex to another. Thus, after G is cleared, each C_i $(1 \leq i \leq m)$ contains at least $|V(C_i)| - 1$ searchers. Hence $\text{fms}(G) \geq |V(G)| - m$. $\square$

Corollary 7.8. *For any proper interval graph, the fast–mixed search number and an optimal fms-strategy can be computed in linear time.*

Proof. From [4], we know that testing whether a given graph $G = (V, E)$ is an interval graph can be done in $O(|V| + |E|)$ time by seeking the sequence of maximal cliques $C_1, C_2, \ldots, C_m$ such that for any $v \in V(C_i) \cap V(C_k)$, $1 \leq i < k \leq m$, the vertex v is contained in all C_j, $i \leq j \leq k$. Then $\text{fms}(G) = |V| - m$, and the optimal fms-strategy described in the proof of Theorem 7.7 can be implemented in $O(|V| + |E|)$ time. $\square$

Corollary 7.9. *For a k-tree G with more than k vertices, if G has exactly two simplicial vertices, then* $\text{fms}(G) = k$.

Proof. Since a k-tree is a proper interval graph in which each clique in the sequence of maximal cliques $C_1, C_2, \ldots, C_m$ has $k+1$ vertices and $m = |V(G)| - k$. Thus $\text{fms}(G) = |V(G)| - m = k$. $\square$

A graph $G = (V, E)$ is called *fms-maximal* if $\text{fms}(G') > \text{fms}(G)$ for any graph $G' = (V, E')$, where E' is a proper superset of E (refer to Fig. 13).

From Theorem 4.3 and Corollary 7.9, we have the following result.

Corollary 7.10. *Every fms-maximal graph G with* $\text{fms}(G) = k$ *is a k-tree with exactly two simplicial vertices.*

Given two graphs G and H, the *Cartesian product* of G and H, denoted by $G \square H$, is the graph whose vertex set is the Cartesian product $V(G) \times V(H)$ of the two vertex sets $V(G)$ and $V(H)$, and in which two vertices (u, v), $(u', v') \in V(G) \times V(H)$ are adjacent in $G \square H$ if and only if $u = u'$ and v is adjacent with v' in H, or $v = v'$ and u is adjacent with u' in G. We have the following result for the Cartesian product of two graphs.

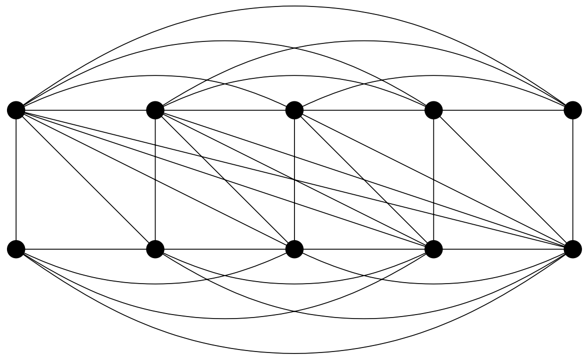

Fig. 13. An fms-maximal graph G with $\mathrm{fms}(G) = 5$.

Theorem 7.11. *For any graphs G and H,*

$$\mathrm{fms}(G\Box H) \le \min\{|V(G)|\mathrm{fms}(H), |V(H)|\mathrm{fms}(G)\}.$$

Proof. Without loss of generality, suppose that $|V(G)|\mathrm{fms}(H) \le |V(H)|\mathrm{fms}(G)$. Let S be an optimal fms-strategy of H that uses k searchers to clear H. Let $\{u_1 \sim v_1, \dots, u_k \sim v_k\}$ be an fms-path cover of H with respect to S such that each path $u_i \sim v_i$, $1 \le i \le k$, is traversed from u_i to v_i by a searcher. We now give an fms-strategy that clears $G\Box H$ using $k|V(G)|$ searchers. First place one searcher on each vertex (x, u_i), where $x \in V(G)$ and $1 \le i \le k$. In the optimal fms-strategy S for H, for each sliding action $u \to v$ along an edge uv on a path $u_i \sim v_i$, we have $|V(G)|$ sliding actions $(x, u) \to (x, v)$, $x \in V(G)$, along an edge $\{(x, u), (x, v)\}$ in $G\Box H$. Since S can clear H using k searchers, it is not hard to see that the new fms-strategy can clear $G\Box H$ using $k|V(G)|$ searchers. □

8. Conclusions

In this paper, we introduced the fast–mixed search model, which is a combination of the fast search model and the mixed search model. However, the fast–mixed search number and the optimal fms-strategy can be very different from those in fast searching and mixed searching. One motivation of the new model is to capture the fugitive as soon as possible. Another motivation is that the fms-paths of a graph G form an induced-path cover of G. We showed that $\mathrm{fms}(G) = k$ if and only if G is a k-stack, which is related to graph drawing and visualization. The fast–mixed search model may have a potential application in a generalized task scheduling problem, in which each machine can obtain information from those tasks that are being performed by machines. We gave characterizations, lower bounds, and relations to other search problems. We proved that Fast–Mixed Searching From Leaves is NP-complete even for planar graphs with maximum degree 3. We also proved that Fast–Mixed Searching Between Leaves is NP-complete for graphs with maximum degree 4. We gave linear-time algorithms for computing the fast–mixed search number of some special classes of graphs such as trees, cacti and proper interval graphs.

As a by-product, we proved that, for a graph G with k leaves, the problem of determining whether there are $\lceil k/2 \rceil$ vertex-disjoint induced-paths that contain all the vertices of G is NP-complete. It remains NP-complete for graphs of maximum degree 4.

Acknowledgments

The author would like to thank the anonymous referees for their valuable comments and suggestions, which improved the presentation of this paper.

References

[1] D. Bienstock, P. Seymour, Monotonicity in graph searching, Journal of Algorithms 12 (1991) 239–245.
[2] D. Dyer, B. Yang, O. Yaşar, On the fast searching problem, in: Proceedings of the 4th International Conference on Algorithmic Aspects in Information and Management (AAIM'08), in: Lecture notes in Computer Science, vol. 5034, Springer, 2008, pp. 143–154.
[3] P.C. Fishburn, Interval Orders and Interval Graphs, in: Wiley-Interscience Series in Discrete Mathematics, John Wiley & Sons, New York, 1985.
[4] M. Habib, R. McConnell, C. Paul, L. Viennot, Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition, and consecutive ones testing, Theoretical Computer Science 234 (2000) 59–84.
[5] J. Kára, J. Kratochvíl, D. Wood, On the complexity of the balanced vertex ordering problem, Discrete Mathematics and Theoretical Computer Science 9 (2007) 193–202.
[6] L. Kirousis, C. Papadimitriou, Searching and pebbling, Theoretical Computer Science 47 (1986) 205–218.
[7] A. LaPaugh, Recontamination does not help to search a graph, Journal of ACM 40 (1993) 224–245.
[8] H. Le, V. Le, Y. Ganjali, H. Muller, Splitting a graph into disjoint induced paths or cycles, Discrete Applied Mathematics 131 (2003) 190–212.
[9] N. Megiddo, S. Hakimi, M. Garey, D. Johnson, C. Papadimitriou, The complexity of searching a graph, Journal of ACM 35 (1988) 18–44.

[10] S. Moran, Y. Wolfstahl, Optimal covering of cacti by vertex-disjoint paths, Theoretical Computer Science 84 (1991) 179–197.
[11] J. Pan, G. Chang, Induced-path partition on graphs with special blocks, Theoretical Computer Science 370 (2007) 121–130.
[12] W. Mulzer, G. Rote, Minimum-weight triangulation is NP-hard, Journal of the ACM 55 (2008) 1–29.
[13] D. Stanley, B. Yang, Fast searching games on graphs, Journal of Combinatorial Optimization 22 (2011) 763–777.
[14] D.B. West, Introduction to Graph Theory, Prentice Hall, 1996.
[15] B. Yang, Fast edge-searching and fast searching on graphs, Theoretical Computer Science 412 (2011) 1208–1219.