Theoretical Computer Science 407 (2008) 532–544

Contents lists available at ScienceDirect

Theoretical Computer Science

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

Monotonicity in digraph search problems☆

Boting Yang*, Yi Cao

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

ARTICLE INFO

Article history:
Received 8 March 2008
Received in revised form 6 August 2008
Accepted 15 August 2008
Communicated by D.-Z. Du

Keywords:
Digraph searching
Pursuit-and-evasion problem
Monotonicity
NP-completeness

ABSTRACT

In this paper, we study the monotonicity and complexity of five digraph search problems: directed searching, mixed directed searching, internal directed searching, internal strong searching, and internal weak searching. In the first three search problems, both searchers and intruder must follow the edge directions when they move along edges. In the internal strong search problem, the intruder must move in the edge directions but searchers need not. In the internal weak search problem, searchers must move in the edge directions but the intruder need not. There are three actions for searchers in the first two search problems: placing, removing and sliding, and there are only two actions for searchers in the last three internal search problems: placing and sliding. Note that the internal strong searching is a "strong" version of the internal directed searching, the internal weak searching is a "weak" version of the internal directed searching, and the internal edge searching is an analogy of the internal directed searching on undirected graphs. We prove that the first three problems are monotonic and the last two problems are non-monotonic, respectively. It is interesting that the internal directed searching is monotonic while the internal strong searching, the internal weak searching and the internal edge searching are all non-monotonic. We also show that the first four problems are NP-complete and the last problem is NP-hard. We solve the open problem on whether a non-monotonic searching problem can be NP-complete.

1. Introduction

Many real-world problems can be naturally modeled by graph search problems. Examples include: capturing intruders in a building, clearing a complex system of interconnected pipes which are contaminated by some noxious gas, and killing a computer virus in a network system. The meaning of a cleared or contaminated edge varies with the problems. For example, in the problem of capturing intruders, a cleared edge means that there is no intruder hiding along this edge, while a contaminated edge means that there may be some intruders hiding along this edge.

In general, a *graph* or *digraph search problem* is to find the minimum number of searchers required to capture an intruder hiding in a graph or digraph. In the edge search problem introduced in [14], there are three types of actions for searchers, i.e., placing, removing and sliding, and an edge is cleared only by a sliding action in a proper way. In the node search problem introduced in [11], there are only two types of actions for searchers, i.e., placing and removing, and an edge is cleared if both end vertices are occupied by searchers. Kirousis and Papadimitriou [11] showed that the node search number is equal to the pathwidth plus one. Bienstock and Seymour [5] introduced the mixed search problem that combines the edge search and node search problems. Thus, in the mixed search problem, an edge is cleared if both end vertices are occupied by searchers

☆ A preliminary version of part of this work was presented in the Proceedings of the 4th International Conference on Theory and Applications of Models of Computation, 2007.

* Corresponding author. Tel.: +1 306 585 4774; fax: +1 306 585 4745.
E-mail addresses: boting@cs.uregina.ca (B. Yang), caoyi200@cs.uregina.ca (Y. Cao).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.08.025

Table 1
Monotonicity and NP-completeness of search problems

Search problem	Monotonicity	Complexity
Directed search	Monotonic	NP-complete
Mixed directed search	Monotonic	NP-complete
Internal directed search	Monotonic	NP-complete
Internal strong search	Non-monotonic	NP-complete
Internal weak search	Non-monotonic	NP-hard
Internal edge search	Non-monotonic	NP-hard

or cleared by a sliding action in a proper way. In the mixed search problem, searchers have three actions: placing, removing and sliding. Barrière et al. [4] introduced the internal edge search problem, in which searchers have only two actions: placing and sliding. A survey of graph searching results can be found in [1,8].

When studying search problems from a computational complexity viewpoint, we are interested in deciding the search number of a graph. Megiddo et al. [14] showed that the edge search problem is NP-hard. This problem belonging to the NP class follows from the monotonicity result of [12] in which LaPaugh showed that recontamination of edges cannot reduce the number of searchers needed to clear a graph. Monotonicity is a very important issue in graph search problems. Bienstock and Seymour [5] proposed a method that gives a succinct proof for the monotonicity of the mixed search problem, which implies the monotonicity of the edge search problem and the node search problem. Fomin and Thilikos [7] provided a general framework that can unify monotonicity results in a unique min–max theorem.

An undirected graph is not always sufficient in representing all the information of a real-world problem. For example, directed edges are required if the graph models one-way streets in a road system. Johnson et al. [10] generalized the concepts of tree-decomposition and treewidth to digraphs and introduced a digraph search problem accordingly. Reed [15] defined another treewidth on digraphs. Safari [16] introduced a new parameter of digraphs, d-width, which is related to the directed treewidth of a digraph. Evans and Safari [6] identified the class of digraphs whose directed treewidth and d-width are both equal to one. Barat [3] generalized the cops-and-robber game to digraphs. He proved that an optimal monotonic search strategy for a digraph needs at most one more searcher than the search number of the digraph. There are several other digraph searching models studied in [9,13]. In all of the above digraph search problems, both searchers and intruders can stay only on vertices. Yang and Cao [18] introduced the strong search model in which the intruder must follow the edge directions but searchers need not when they move along edges. Yang and Cao [19] also introduced the weak search model in which searchers must follow the edge directions but the intruder need not when they move along edges. They proved that both the strong search model and the weak search model are monotonic. In [20] Yang and Cao introduced the directed vertex separation and investigated the relations between different digraph search models, directed vertex separation, and directed pathwidth. In the digraph search models in [18,19], there are three types of actions for searchers: placing, removing and sliding. Alspach et al. [2] proposed three digraph search models, the internal directed search, the internal strong search and the internal weak search, in which searchers cannot be removed from digraphs. In all digraph search problems in [2,18, 19], both searchers and intruders can stay on both vertices and edges.

The main problem left unresolved in [2] is the monotonicity issue of the three internal search problems on digraphs. In this paper, we prove that the internal directed search problem is monotonic, and the internal strong and the internal weak search problems are non-monotonic. These results are surprising because, while the internal directed searching is monotonic, the internal edge searching, which is an analogy of the internal directed searching on undirected graphs, is non-monotonic [4], the strong version of the internal directed searching (i.e., internal strong searching) is non-monotonic, and the weak version of the internal directed searching (i.e., internal weak searching) is also non-monotonic. In this paper, we also introduce and study the directed searching and mixed directed searching. The definitions of these problems are given later in Section 2. We prove the monotonicity of the mixed directed search problem by using the method proposed by Bienstock and Seymour [5]. We also prove the monotonicity of the directed search problem. From these monotonicity results, we prove that the mixed directed search, directed search, and internal directed search problems are NP-complete, respectively. In the open problem session of the first GRASTA workshop[1], Dimitrios Thilikos proposed an open problem on whether a non-monotonic searching problem can be NP-complete. We solve this problem by showing that the internal strong search problem is NP-complete although it is non-monotonic. For the internal weak search problem, we can only prove that it is NP-hard, and we conjecture that it is also NP-complete. Table 1 summarizes these results.

This paper is organized as follows. In Section 2, we give some definitions and notation. In Section 3, we prove the monotonicity of the mixed directed search problem. In Section 4, we prove the monotonicity of the directed search problem. In Section 5, we prove that the internal directed search problem is monotonic and both internal strong and internal weak search problems are non-monotonic. In Section 6, we show the NP-completeness results. Finally, we conclude this paper in Section 7.

[1] The First Workshop on Graph Searching, Theory and Applications (GRASTA), Crete, Greece, October 9–12, 2006.

2. Definitions and notation

All graphs and digraphs in this paper contain at least one edge. Throughout this paper, we use D to denote a digraph, (u, v) to denote a directed edge with tail u and head v, and $u \rightsquigarrow v$ to denote a directed path from u to v. An *undirected path* between two vertices $u, v \in V(D)$ is defined as a path between u and v in the underlying graph of D, and is denoted as $u \sim v$. We assume that before the first action is carried out in any search model studied in this paper, D contains one intruder and no searchers, and all edges of D are *contaminated*. This means that the intruder is invisible and may hide on any edge or vertex of D initially. In any search model, a *search strategy* is a sequence of searchers' actions such that the final action leaves all edges of D *uncontaminated* (or *cleared*), which means that the intruder is captured. The digraph D is *cleared* if all of its edges are cleared.

In the *internal directed search model*, both searchers and intruder must move in the edge directions. The intruder can move at a great speed at any time from vertex u to vertex v along a directed path $u \rightsquigarrow v$ that contains no searchers. There are two types of actions for searchers: (1) placing a searcher on a vertex, and (2) sliding a searcher along an edge from its tail to its head. A contaminated edge (u, v) can be cleared in one of two ways by a sliding action: (1) sliding a searcher from u to v along (u, v) while at least one searcher is located on u, or (2) sliding a searcher from u to v along (u, v) while all edges with head u are already cleared. A cleared edge will be *recontaminated* if there is a directed path containing this cleared edge and a contaminated edge pointing to the cleared edge such that there are no searchers stationing on any internal vertex of this directed path. The minimum number of searchers needed to clear D in this model is called the *internal directed search number* of D, denoted by $\mathrm{ids}(D)$.

In the *internal strong search model*, the intruder must move in the edge directions but searchers need not. The intruder can move at a great speed at any time from vertex u to vertex v along a directed path $u \rightsquigarrow v$ that contains no searchers. There are two types of actions for searchers: (1) placing a searcher on a vertex, and (2) sliding a searcher along an edge from one end vertex to the other. A contaminated edge (u, v) can be cleared in one of three ways by a sliding action: (1) sliding a searcher from u to v along (u, v) while at least one searcher is located on u, (2) sliding a searcher from u to v along (u, v) while all edges with head u are already cleared, or (3) sliding a searcher from v to u along the edge (u, v). A cleared edge will be *recontaminated* if there is a directed path containing this cleared edge and a contaminated edge pointing to the cleared edge such that there are no searchers stationing on any internal vertex of this directed path. The minimum number of searchers needed to clear D in this model is called the *internal strong search number* of D, denoted by $\mathrm{iss}(D)$.

In the *internal weak search model*, searchers must move in the edge directions but the intruder need not. The intruder can move at a great speed at any time from vertex u to vertex v along an undirected path $u \sim v$ that contains no searchers. There are two types of actions for searchers: (1) placing a searcher on a vertex, and (2) sliding a searcher along an edge from its tail to its head. A contaminated edge (u, v) can be cleared in one of two ways by one sliding action: (1) sliding a searcher from u to v along (u, v) while at least one searcher is located on u, or (2) sliding a searcher from u to v along (u, v) while all edges incident with u except (u, v) are already cleared. A cleared edge will be *recontaminated* if there is an undirected path containing this cleared edge and a contaminated edge such that there are no searchers stationing on any internal vertex of this undirected path. The minimum number of searchers needed to clear D in this model is called the *internal weak search number* of D, denoted by $\mathrm{iws}(D)$.

In *directed* and *mixed directed search models*, both searchers and intruder must move in the edge directions when they slide along edges. The intruder can move at a great speed at any time from vertex u to vertex v along a directed path $u \rightsquigarrow v$ that contains no searchers. A cleared edge will be *recontaminated* if there is a directed path containing this cleared edge and a contaminated edge such that there are no searchers stationing on any internal vertex of this directed path.

In the directed search model, there are three types of actions for searchers: (1) placing a searcher on a vertex, (2) removing a searcher from a vertex, and (3) sliding a searcher along an edge from its tail to its head. A contaminated edge (u, v) can be cleared in one of two ways by a sliding action: (1) sliding a searcher from u to v along (u, v) while at least one searcher is located on u, or (2) sliding a searcher from u to v along (u, v) while all edges with head u are already cleared. The minimum number of searchers needed to clear D in the directed search model is called the *directed search number* of D, denoted by $\mathrm{ds}(D)$.

In the mixed directed search model, there are four types of actions for searchers: (1) placing a searcher on a vertex that is not occupied, (2) removing a searcher from a vertex, (3) clearing an edge whose two end vertices are occupied, and (4) sliding a searcher from u to v along an edge (u, v) satisfying that before the sliding, all in-edge of u are cleared and v is not occupied. A contaminated edge (u, v) can be cleared by action (3) or (4). The minimum number of searchers needed to clear D in the mixed directed search model is called the *mixed directed search number* of D, denoted by $\mathrm{xds}(D)$.

In the above five search models, one difference is whether searchers and intruder obey the edge direction when they move along edges. Another difference is whether searchers can be removed from digraphs. We summarize these differences in Table 2.

We say that a vertex in D is *occupied* at some moment if at least one searcher is located on this vertex at this moment. Any searcher that is not on D at some moment is said *free* at this moment. Note that all searchers are initially free.

Let $\mathcal{M}$ be one of the above five search models on digraphs and $s_{\mathcal{M}}(D)$ denote the search number of digraph D under the model $\mathcal{M}$. Let S be a search strategy of $\mathcal{M}$ and A_i be the set of cleared edges immediately after the ith action. S is *monotonic* if $A_i \subseteq A_{i+1}$ for each i. We say that the model $\mathcal{M}$ is *monotonic* if for any digraph D, there exists a monotonic search strategy that can clear D using $s_{\mathcal{M}}(D)$ searchers. We say that the model $\mathcal{M}$ is *non-monotonic* if there exists a digraph D such that any

Table 2
Differences between search models

Search models	Directed or mixed directed	Int. directed	Int. strong	Int. weak
Searchers obey direction	Yes	Yes	No	Yes
Intruder obeys direction	Yes	Yes	Yes	No
Searchers can be removed	Yes	No	No	No

Table 3
Notation of search numbers

Search problem	Search number
Directed search	ds
Monotonic directed search	mds
Mixed directed search	xds
Internal directed search	ids
Monotonic internal directed search	mids
Internal strong search	iss
Monotonic internal strong search	miss
Internal weak search	iws
Monotonic internal weak search	miws
Node search	ns

monotonic search strategy requires more than $s_{\mathcal{M}}(D)$ searchers to clear D. Monotonicity can be defined similarly for other search models. Table 3 summarizes the notation used in this paper.

3. Monotonicity of the mixed directed search model

In this section, we will show that the mixed directed search model is monotonic, which means that recontamination does not help to reduce the mixed directed search number of a digraph. We will extend the method proposed by Bienstock and Seymour [5]. We first define the exposed vertex set.

Definition 3.1. Let D be a digraph. For an edge set $X \subseteq E(D)$, a vertex in $V(D)$ is an *exposed vertex* with respect to X if it is the tail of an edge in X and the head of an edge in $E(D) - X$. The set of all exposed vertices with respect to X is denoted by $\partial(X)$.

In the directed or mixed directed search model, it follows from Definition 3.1 that at any moment, an exposed vertex with respect to the cleared edge set must be occupied by a searcher. Thus, such an exposed vertex is also called a *critical vertex*. The following result appeared in [3] (we give a proof for completeness).

Lemma 3.2. *For any* $X, Y \subseteq E(D)$, $|\partial(X \cap Y)| + |\partial(X \cup Y)| \leq |\partial(X)| + |\partial(Y)|$.

Proof. We have to prove that every exposed vertex counted on the left-hand side is also counted at least as many times on the right-hand side. If $v \in \partial(X \cap Y)$ and $v \in \partial(X \cup Y)$, then there is an edge $e \in X \cap Y$ with tail v and there is an edge $e' \notin X \cup Y$ with head v. Thus $e \in X$ with tail v and $e' \notin X$ with head v, and $e \in Y$ with tail v and $e' \notin Y$ with head v. Therefore $v \in \partial(X)$ and $v \in \partial(Y)$. Similarly, if $v \in \partial(X \cap Y)$, then $v \in \partial(X)$ or $v \in \partial(Y)$; and if $v \in \partial(X \cup Y)$, then $v \in \partial(X)$ or $v \in \partial(Y)$. ■

Definition 3.3. Let D be a digraph. A *campaign* in D is a sequence $(X_0, X_1, \ldots, X_\ell)$ of subsets of $E(D)$ such that $X_0 = \emptyset$, $X_\ell = E(D)$ and $|X_i - X_{i-1}| \leq 1$, for $1 \leq i \leq \ell$. The *width* of the campaign is defined as $\max_{0 \leq i \leq n} |\partial(X_i)|$. A campaign is *progressive* if $X_0 \subseteq X_1 \subseteq \cdots \subseteq X_\ell$ and $|X_i - X_{i-1}| = 1$, for $1 \leq i \leq \ell$.

Similar to [3,5], we have the following lemma for the progressive campaign (we give a proof for completeness).

Lemma 3.4. *If there is a campaign in D of width at most k, then there is a progressive campaign in D of width at most k.*

Proof. Choose a campaign $(X_0, X_1, \ldots, X_\ell)$ of width at most k such that it satisfies two conditions: (i) $\sum_{i=0}^{n}(|\partial(X_i)| + 1)$ is minimum, and (ii) subject to (i), $\sum_{i=0}^{n} |X_i|$ is minimum. We will show that $(X_0, X_1, \ldots, X_\ell)$ is progressive.

If $|X_j - X_{j-1}| = 0$, $1 \leq j \leq \ell$, then $X_j \subseteq X_{j-1}$ and $(X_0, \ldots, X_{j-1}, X_{j+1}, \ldots, X_\ell)$ is a campaign of width at most k, contradicting condition (i). Thus $|X_j - X_{j-1}| \neq 0$. Since $|X_j - X_{j-1}| \leq 1$, we know that $|X_j - X_{j-1}| = 1$, $1 \leq j \leq \ell$.

We now prove that $X_{j-1} \subseteq X_j$, $1 \leq j \leq \ell$. If $|\partial(X_{j-1} \cup X_j)| < |\partial(X_j)|$, then $|\partial(X_{j-1} \cup X_j)| < k$ and $(X_0, \ldots, X_{j-1}, X_{j-1} \cup X_j, X_{j+1}, \ldots, X_\ell)$ is a campaign of width at most k, contradicting condition (i). Thus, we have $|\partial(X_{j-1} \cup X_j)| \geq |\partial(X_j)|$, $1 \leq j \leq \ell$. From Lemma 3.2, we know that $|\partial(X_{j-1} \cap X_j)| \leq |\partial(X_{j-1})|$. Hence, $(X_0, \ldots, X_{j-2}, X_{j-1} \cap X_j, X_j, \ldots, X_\ell)$ is a campaign of width at most k. From condition (ii), we have $|X_{j-1} \cap X_j| \geq |X_{j-1}|$ which implies that $X_{j-1} \subseteq X_j$.

By Definition 3.3, we know that $(X_0, X_1, \ldots, X_\ell)$ is progressive. ■

In the remainder of this section, we will prove that the mixed directed search problem is monotonic. The searchers' actions of the mixed directed searching look more complicated than those in other models. It is necessary to describe them precisely as follows.

Definition 3.5. Let $S = (s_1, s_2, \ldots, s_\ell)$ be a mixed directed search strategy for a digraph D. For $1 \le i \le \ell$, let A_i be the set of cleared edges and Z_i be the set of occupied vertices immediately after action s_i such that $\partial(A_i) \subseteq Z_i$. Let $A_0 = Z_0 = \emptyset$. Each action s_i, $1 \le i \le \ell$, is one of the following four types:

(a) (*placing a searcher on* v) $Z_i = Z_{i-1} \cup \{v\}$ for some vertex $v \in V(D) - Z_{i-1}$ and $A_i = A_{i-1}$ (note that each vertex in Z_i has exactly one searcher);
(b) (*removing a searcher from* v) $Z_i = Z_{i-1} - \{v\}$ for some vertex $v \in Z_{i-1}$ and $A_i = \{e \in A_{i-1}$: for any directed path $u \rightsquigarrow w$ containing e and an edge $e' \in E(D) - A_{i-1}$ such that w is the head of e and u is the tail of e', the path $u \rightsquigarrow w$ has an internal vertex in $Z_i\}$;
(c) (*node-search-clearing* e) $Z_i = Z_{i-1}$ and $A_i = A_{i-1} \cup \{e\}$ for some edge $e = (u, v) \in E(D)$ with both ends u and v in Z_{i-1};
(d) (*edge-search-clearing* e) $Z_i = (Z_{i-1} - \{u\}) \cup \{v\}$ and $A_i = A_{i-1} \cup \{e\}$ for some edge $e = (u, v) \in E(D)$ with $u \in Z_{i-1}$ and $v \in V(D) - Z_{i-1}$ and every (possibly 0) edge with head u belongs to A_{i-1}.

From Definition 3.5, we know that at most one edge can be cleared in one action and each vertex is occupied by at most one searcher at any time. Note that recontamination in the mixed directed search problem is caused only by removing actions. In (c) and (d), if $e \in A_{i-1}$, then we say this action is *superfluous*. Adding or deleting superfluous actions will not affect the number of searchers used in a search strategy, however, sometimes allowing superfluous actions may make arguments simple.

The following lemma establishes a relation between the mixed directed searching of D and a campaign in D.

Lemma 3.6. *Let D be a digraph without multiple edges. If* $\mathrm{xds}(D) \le k$, *then there is a campaign in D of width at most $k - 1$.*

Proof. Let $S = (s_1, s_2, \ldots, s_m)$ be a mixed directed search strategy of D using at most k searchers. For $1 \le i \le m$, let A_i be the set of cleared edges and Z_i be the set of occupied vertices immediately after s_i, and let $A_0 = Z_0 = \emptyset$. We first normalize S such that the normalized search strategy can also clear D using at most k searchers. The normalized search strategy may contain some superfluous actions, but this will not increase the number of searchers required to clear D.

The normalization is conducted by inserting some node-search-clearing actions after each placing action and edge-search-clearing action. Specifically, for each $1 \le i \le m$, if $Z_i - Z_{i-1}$ is empty, i.e., s_i is a removing or node-search-clearing action, then we leave s_i unchanged; otherwise, $Z_i - Z_{i-1}$ contains a vertex v, i.e., s_i is a placing or edge-search-clearing action such that v is occupied just after s_i, then let $E_v^1 = \{(u, v) \in E(D)$: $u \in Z_{i-1}\}$, $E_v^2 = \{(v, u) \in E(D)$: $u \in Z_{i-1}$ and all edges with head u except (v, u) are in $A_{i-1}\}$, and $E_v^3 = \{(v, u) \in E(D)$: $u \in Z_{i-1}$ and $(v, u) \notin E_v^2\}$, and we then insert a subsequence of node-search-clearing actions between s_i and s_{i+1}, such that each edge in E_v^1 is cleared first, then each edge in E_v^2, and finally each edge in E_v^3 (edges in the same set are cleared in an arbitrary order). After the normalization, we obtain a new sequence of actions that contains each old action and some new node-search-clearing actions. It is easy to see that this new sequence of actions, denoted by $(s'_1, s'_2, \ldots, s'_\ell)$, is still a mixed directed search strategy of D using at most k searchers.

For $1 \le i \le \ell$, let A'_i be the set of cleared edges and Z'_i be the set of occupied vertices immediately after s'_i, and let $A'_0 = Z'_0 = \emptyset$. Since $\partial(A'_i) \subseteq Z'_i$, $|Z'_i| \le k$, and $|A'_i - A'_{i-1}| \le 1$, $1 \le i \le \ell$, we know that $(A'_0, A'_1, \ldots, A'_\ell)$ is a campaign in D of width at most k.

We now show that the campaign $(A'_0, A'_1, \ldots, A'_\ell)$ can be converted into a campaign $(X_0, X_1, \ldots, X_\ell)$ of width at most $k - 1$. For each i from 0 to ℓ, if $|\partial(A'_i)| \le k - 1$, then let $X_i = A'_i$. If $|\partial(A'_i)| = k$, then $\partial(A'_i) = Z'_i$. Let v be the last vertex occupied by a searcher in Z'_i. Recall that just after v receives a searcher in a placing or edge-search-clearing action, the following actions clear all edges in E_v^1, E_v^2 and E_v^3 by node-search-clearing. Note that at the step when an edge $(u, v) \in E_v^1$ is cleared, v is not a critical vertex at this step. When an edge $(v, u) \in E_v^2$ is cleared, since D has no multiple edges, all edges with head u are cleared and thus u is not a critical vertex. Hence, when $|\partial(A'_i)| = k$, s'_i must be a node-search-clearing action that clears an edge in E_v^3. Therefore, each vertex in Z'_i has at least one contaminated edge with tail not in Z'_i. Let s'_j be the first removing action after s'_i. Such an action must exist; otherwise, D will not be cleared because each vertex in Z'_i has at least one contaminated edge with tail not in Z'_i. Let $R = A'_{j-1} - A'_j$ and $X_p = A'_p - R$ for $i \le p \le j$. Since $|A'_p - A'_{p-1}| \le 1$, $i \le p \le j$, we know that $|X_p - X_{p-1}| \le 1$ for $1 \le p \le j$. Suppose that s'_j removes the searcher on w. Then all edges with tail w must be contaminated immediately after s'_j, which means that A'_j contains no edges with tail w. Hence, X_p contains no edges with tail w for $i \le p \le j$. Thus $w \notin \partial(X_p)$ and $|\partial(X_p)| \le k - 1$ for $i \le p \le j$. We then consider A'_{j+1} and construct X_{j+1}. We can continue this process and finally we obtain a campaign $(X_0, X_1, \ldots, X_\ell)$ in D of width at most $k - 1$. ■

Lemma 3.7. *For a digraph D, if $(X_0, X_1, \ldots, X_\ell)$ is a progressive campaign in D of width at most $k - 1$, then there is a monotonic mixed directed search strategy that clears D using at most k searchers such that the edges of D are cleared in the order $e_1, e_2, \ldots, e_\ell$, where $e_i = X_i - X_{i-1}$, $1 \le i \le \ell$.*

Proof. We construct a monotonic mixed directed search strategy inductively. Suppose that we have cleared edges $e_1, \ldots, e_{j-1}$, $2 \le j \le \ell$, in order, and that no other edges have been cleared yet. Let $e_j = (u, v)$ and $C_{j-1} = \{p \in V(D)$: p has no in-edge or all in-edges of p belong to $X_{j-1}\}$. Before we clear (u, v), we may remove searchers such that each vertex

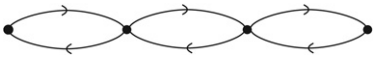

Fig. 1. $\mathrm{xds}(D) = 2$ and $\mathrm{ds}(D) = 3$.

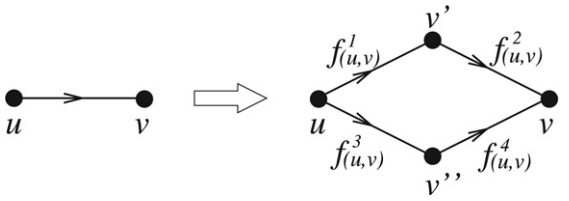

Fig. 2. Converting D into D^*.

in $\partial(X_{j-1})$ is occupied by one searcher and all other searchers are free. If $|\{u, v\} \cup \partial(X_{j-1})| \leq k$, we may place free searchers on both ends of e_j and execute node-search-clearing. Assume that $|\{u, v\} \cup \partial(X_{j-1})| > k$. Since $|\partial(X_{j-1})| \leq k-1$, it follows that $|\partial(X_{j-1})| = k-1$ and $\{u, v\} \cap \partial(X_{j-1}) = \emptyset$. Thus, we have one free searcher. We now prove that $u \in C_{j-1}$. If $u \notin C_{j-1}$, then $u \in \partial(X_j)$ and $|\partial(X_j)| = k$, which contradicts the condition that $(X_0, X_1, \ldots, X_\ell)$ has width at most $k-1$. Thus, u has no contaminated in-edges and we can place the free searcher on u and then slide the searcher from u to v along (u, v) to clear e_j by edge-search-clearing. ■

From Lemmas 3.4, 3.6 and 3.7, we have the following result.

Lemma 3.8. *Given a digraph D that has no multiple edges, the following are equivalent:*

(i) $\mathrm{xds}(D) \leq k$;
(ii) *there is a campaign in D of width at most* $k-1$;
(iii) *there is a progressive campaign in D of width at most* $k-1$; *and*
(iv) *there is a monotonic mixed directed search strategy that clears D using at most k searchers.*

From Lemma 3.8, we can prove the monotonicity of the mixed directed search model.

Theorem 3.9. *Given a digraph D, if* $\mathrm{xds}(D) = k$, *then there is a monotonic mixed directed search strategy that clears D using k searchers.*

Proof. If D has no multiple edges, then the result follows from Lemma 3.8. Otherwise, let D' be a digraph obtained from D by replacing all edges with the same tail and the same head by a single edge. Notice that D' has no multiple edges. We first show that $\mathrm{xds}(D') = k$. For a mixed directed search strategy that clears D using k searchers, it is also a mixed directed search strategy that clears D' using k searchers (some actions are superfluous). Thus, $\mathrm{xds}(D') \leq k$. From Lemma 3.8, there is a monotonic mixed directed search strategy S that clears D' using at most k searchers. We now extend S to a monotonic mixed directed search strategy for D. For each node-search-clearing action s in S, if s clears an edge in D' which has multiplicity m, $m \geq 2$, in D, then we insert $m-1$ node-search-clearing actions just after s so that all the multiple edges can be cleared. For each edge-search-clearing action t in S which slides a search λ from u to v along (u, v), if (u, v) has multiplicity m, $m \geq 2$, in D, then just after t, we insert $m-1$ triples of actions: "removing λ from v", "placing λ on u" and "edge-search-clearing one of the multiple edges with tail u and head v". These actions can clear all the multiple edges with tail u and head v. By the extension of S, we obtain a monotonic mixed directed search strategy that clears D using at most k searchers. Note that $\mathrm{xds}(D) = k$. Therefore, there is a monotonic mixed directed search strategy that clears D using k searchers. ■

4. Monotonicity of the directed search model

In Section 3, we have proved that the mixed directed search problem is monotonic. In this section we will prove that the monotonicity of the mixed directed search problem implies the monotonicity of the directed search problem. The following lemma describes a relationship between the directed searching and the mixed directed searching.

Lemma 4.1. *If D is a digraph, then* $\mathrm{ds}(D) - 1 \leq \mathrm{xds}(D) \leq \mathrm{ds}(D)$.

The two equalities in Lemma 4.1 can be achieved. For example, for a directed path, both search numbers equal 1, and for the digraph D in Fig. 1, it is easy to see that $\mathrm{xds}(D) = 2$ and $\mathrm{ds}(D) = 3$.

From Lemma 4.1, we know that the difference between $\mathrm{ds}(D)$ and $\mathrm{xds}(D)$ is not a fixed constant. It is not easy to use this lemma to prove the monotonicity of the directed search model. However, we can transform D into another digraph D^* such that $\mathrm{ds}(D) = \mathrm{xds}(D^*)$. Then we can use this relation to prove the monotonicity of the directed search model.

Theorem 4.2. *For a digraph D, let* D^* *be a digraph obtained from D by replacing each edge* $(u, v) \in E(D)$ *by two directed paths* (u, v', v) *and* (u, v'', v). *For* $(u, v) \in E(D)$, *let* $f^1_{(u,v)} = (u, v')$, $f^2_{(u,v)} = (v', v)$, $f^3_{(u,v)} = (u, v'')$ *and* $f^4_{(u,v)} = (v'', v)$ *(see Fig. 2). The following are equivalent:*

(i) $\mathrm{ds}(D) \leq k$;
(ii) $\mathrm{xds}(D^*) \leq k$;

(iii) *there is a progressive campaign* $(X_0, X_1, \ldots, X_\ell)$ *in* D^* *of width at most* $k-1$ *such that for each* $(u, v) \in E(D)$, $m_1 < m_2$ *and* $m_3 < m_4$, *where* m_i, $1 \le i \le 4$, *is the subscript of* X_{m_i} *that is the first set containing* $f^i_{(u,v)}$;
(iv) *there is a monotonic mixed directed search strategy that clears* D^* *using at most* k *searchers such that for each* $(u, v) \in E(D)$, $f^1_{(u,v)}$ *is cleared before* $f^2_{(u,v)}$ *and* $f^3_{(u,v)}$ *is cleared before* $f^4_{(u,v)}$; *and*
(v) $\mathrm{mds}(D) \le k$.

Proof. (i)⇒(ii). Let $(s_1, s_2, \ldots, s_\ell)$ be a directed search strategy of D using at most k searchers. We will inductively construct a mixed directed search strategy $(S'_1, S'_2, \ldots, S'_\ell)$ of D^* using at most k searchers, where S'_i is a subsequence of actions corresponding to s_i. Since s_1 is a placing action, let S'_1 be the same placing action. Suppose that we have constructed $S'_1, S'_2, \ldots, S'_{j-1}$ such that the following two conditions are satisfied: (1) the set of occupied vertices immediately after s_h, $1 \le h \le j-1$, is the same as the set of occupied vertices immediately after the last action in S'_h, and (2) if $(u, v) \in E(D)$ is cleared immediately after s_h, $1 \le h \le j-1$, then the corresponding four edges $f^i_{(u,v)} \in E(D^*)$, $1 \le i \le 4$, are also cleared immediately after the last action in S'_h.

We now construct S'_j. If s_j is a placing action that places a searcher on an unoccupied vertex, S'_j will take the same action. If s_j is a placing action that places a searcher on an occupied vertex, S'_j will be empty. If s_j is a removing action that removes the only searcher from a vertex, S'_j will take the same action. If s_j is a removing action that removes a searcher from a vertex occupied by at least two searchers, S'_j will be empty. If s_j is a sliding action that slides a searcher from vertex u to vertex v along edge (u, v) to clear the contaminated edge (u, v), we have two cases.

Case 1. All edges with head u are cleared in D immediately before s_j. By the hypothesis, the vertex $u \in V(D^*)$ is also occupied and all edges with head u in D^* are also cleared immediately after the last action in S'_{j-1}. If v is not occupied, then we can construct S'_j as follows: edge-search-clearing (u, v'), edge-search-clearing (v', v), removing the searcher from v, placing the searcher on u, edge-search-clearing (u, v'') and edge-search-clearing (v'', v). If v is occupied, then we can construct S'_j as follows: edge-search-clearing (u, v'), node-search-clearing (v', v), removing the searcher from v', placing the searcher on u, edge-search-clearing (u, v''), node-search-clearing (v'', v), and removing the searcher from v''.

Case 2. At least one edge with head u is contaminated in D immediately before s_j. We know that there is at least one searcher on u while performing s_j, which implies that u is occupied by at least two searchers immediately before s_j. By the hypothesis, the vertex $u \in V(D^*)$ is also occupied and we have at least one free searcher immediately after the last action in S'_{j-1}. If v is not occupied, then we can construct S'_j as follows: placing a searcher on v', node-search-clearing (u, v'), edge-search-clearing (v', v), removing the searcher from v, placing the searcher on v'', node-search-clearing (u, v'') and edge-search-clearing (v'', v). If v is occupied, then we can construct S'_j as follows: placing a searcher on v', node-search-clearing (u, v'), node-search-clearing (v', v), removing the searcher from v', placing the searcher on v'', node-search-clearing (u, v''), node-search-clearing (v'', v), and removing the searcher from v''.

If s_j is a sliding action that slides a searcher from vertex u to vertex v along edge (u, v) but does not clear the contaminated edge (u, v), we know that immediately before s_j, u is occupied by only one searcher and at least one edge with head u is contaminated. By the hypothesis, the vertex $u \in V(D^*)$ is also occupied immediately after the last action in S'_{j-1}. If v is occupied, then S'_j consists of only one action: removing the searcher from u. If v is not occupied, then S'_j consists of two actions: removing the searcher from u and placing it on v.

It is easy to see that $(S'_1, S'_2, \ldots, S'_\ell)$ can clear D^* using at most k searchers.

(ii)⇒(iii). Since D^* has no multiple edges, by Lemma 3.8, there is a progressive campaign $(X_0, X_1, \ldots, X_\ell)$ in D^* of width at most $k-1$. We can modify this campaign to satisfy the requirement of (iii). By Definition 3.3, we know that m_1, m_2, m_3 and m_4 have different values. We have four cases regarding the smallest value.

Case 1. $m_1 = \min\{m_1, m_2, m_3, m_4\}$. We already have $m_1 < m_2$. If $m_3 > m_4$, then, for each $m_1 + 1 \le i \le m_3$, replace X_i by $X'_i = X_{i-1} \cup \{f^3_{(u,v)}\}$. Since $X'_{m_3} = X_{m_3-1} \cup \{f^3_{(u,v)}\} = X_{m_3}$ and $v'' \notin \partial(X'_i)$, $m_1 + 1 \le i \le m_3$, it is easy to see that the updated campaign is still a progressive campaign in D^* of width at most $k-1$. Let the updated campaign still be denoted by $(X_0, X_1, \ldots, X_\ell)$ and m_i $(1 \le i \le 4)$ is the subscript of X_{m_i} that is the first set containing $f^i_{(u,v)}$ in the updated campaign. Thus, we have $m_1 < m_2$ and $m_3 = m_1 + 1 < m_4$ in the updated campaign.

Case 2. $m_2 = \min\{m_1, m_2, m_3, m_4\}$. For each $m_2 \le j \le m_1$, replace X_j by $X'_j = X_{j-1} \cup \{f^1_{(u,v)}\}$. After this modification, the first set containing $f^1_{(u,v)}$ is X'_{m_2} and the first set containing $f^2_{(u,v)}$ is X'_{m_2+1}. Since $f^2_{(u,v)} \in X_i$ and $f^1_{(u,v)} \notin X_i$ for $m_2 \le i \le m_1 - 1$, we know that $v' \in \partial(X_i)$, $m_2 \le i \le m_1 - 1$. Thus, for $m_2 \le i \le m_1 - 1$, we have $\partial(X'_i) \subseteq (\partial(X_i) - \{v'\}) \cup \{u\}$ and $|\partial(X'_i)| \le |\partial(X_i)|$. We also know that $X'_{m_1} = X_{m_1-1} \cup \{f^1_{(u,v)}\} = X_{m_1}$. It follows that the updated campaign is still a progressive campaign in D^* of width at most $k-1$. Let the updated campaign still be denoted by $(X_0, X_1, \ldots, X_\ell)$ and m_i $(1 \le i \le 4)$ is the subscript of X_{m_i} that is the first set containing $f^i_{(u,v)}$ in the updated campaign. Thus, we have $m_1 < m_2$. Then we can use the method described in Case 1 to achieve $m_3 < m_4$ by modifying the campaign if necessary.

Case 3. $m_3 = \min\{m_1, m_2, m_3, m_4\}$. We already have $m_3 < m_4$ and we can use the method described in Case 1 to modify the campaign such that $m_1 < m_2$ in the updated campaign.

Case 4. $m_4 = \min\{m_1, m_2, m_3, m_4\}$. We can use the method described in Case 2 to modify the campaign such that $m_3 < m_4$ and $m_1 < m_2$ in the updated campaign.

For each $(u, v) \in E(D)$, we can repeat the above procedure for the corresponding four edges $f^i_{(u,v)} \in E(D^*)$, $1 \le i \le 4$. Finally, we can obtain a campaign as required.

(iii)⇒(iv). Let $(X_0, X_1, \ldots, X_\ell)$ be the progressive campaign described in (iii). The monotonic mixed directed search strategy constructed in Lemma 3.7 can use at most k searchers to clear $f^1_{(u,v)}$ before $f^2_{(u,v)}$ and to clear $f^3_{(u,v)}$ before $f^4_{(u,v)}$ for each $(u, v) \in E(D)$.

(iv)⇒(v). Let $S = (s_1, s_2, \ldots, s_\ell)$ be a monotonic mixed directed search strategy of D^* satisfying the condition of (iv). We will construct a monotonic directed search strategy S' of D using at most k searchers. For each edge $(u, v) \in E(D)$, without loss of generality, suppose that S clears $f^1_{(u,v)}$ before $f^3_{(u,v)}$. For each i from 1 to ℓ, consider s_i. If s_i is a placing or removing action on a vertex that is also in $V(D)$, S' will take the same action. If s_i is a placing or removing action on a vertex in $V(D^*) - V(D)$, S' will do nothing. If s_i is an edge-search-clearing action that clears edge $f^1_{(u,v)}$, then in S', we can clear $(u, v) \in E(D)$ in the same way as s_i does. If s_i is a node-search-clearing action that clears edge $f^1_{(u,v)}$ by the searcher α on u and the searcher β on v', then in S', we know that α is also on u and β is free. Thus, we can place β on u and then clear $(u, v) \in E(D)$ by sliding β from u to v along (u, v). If s_i clears edge $f^2_{(u,v)}, f^3_{(u,v)}$ or $f^4_{(u,v)}$, S' will do nothing. It is easy to see that S' can clear D using at most k searchers.

(v)⇒(i). It is trivial. ■

5. Monotonicity of internal search models

In this section, we show the monotonicity of the internal directed searching and the non-monotonicity of the internal strong searching and internal weak searching.

Definition 5.1. In the process of clearing a digraph D under the directed or internal directed search model, a vertex in D is *cleared* if all its in-edges and out-edges are cleared, *partially cleared* if all its in-edges (possibly it has no in-edge) are cleared and at least one of its out-edges is contaminated, *critical* if at least one of its in-edges is contaminated and at least one of its out-edges is cleared, and *contaminated* if at least one of its in-edges and all of its out-edges are contaminated. A directed or internal directed search strategy is called *optimal* if it clears D using $\mathrm{ds}(D)$ or $\mathrm{ids}(D)$ searchers, respectively.

From Theorem 4.2, we have the following result.

Lemma 5.2. *For any digraph, there always exists an optimal directed search strategy satisfying the following four conditions:* (1) *every edge is cleared exactly once;* (2) *after each action of placing a searcher on a vertex, the next steps must clear all out-edges of this vertex;* (3) *after each action of sliding a searcher along edge* (u, v) *from* u *to* v*, the next action must be removing this searcher from* v*; and* (4) *searchers are removed from a vertex immediately after all in-edges incident on this vertex have been cleared.*

Proof. For a digraph D, it follows from Theorem 4.2 that there exists a monotonic directed search strategy that clears D using $\mathrm{ds}(D)$ searchers. Among all such search strategies of D, there exists a monotonic optimal search strategy $S = (s_1, s_2, \ldots, s_\ell)$ such that the number of actions in S is minimum. Thus, S does not contain any redundant actions, such as the action of placing a searcher on vertex v is followed immediately by an action of removing a searcher from v. We now modify S to obtain a new monotonic optimal search strategy S' of D. For each i from 1 to ℓ, we modify s_i in the following ways.

Case 1. s_i is a placing action, say, placing a searcher on vertex u. If all out-edges of u are cleared, then we delete s_i in the modified strategy. Otherwise, we have three subcases regarding s_{i+1}. (1) If s_{i+1} is removing a searcher from a vertex or sliding a searcher along an edge from a vertex that is not u, then we can swap the action s_i with the action s_{i+1} and relabeling their subscripts in the increasing order. (2) If s_{i+1} is placing a searcher λ on a vertex, then for each out-edge (u, v) of u, we insert the following three actions between s_i and s_{i+1}: "place λ on u", "slide λ from u to v" and "remove λ from v". (3) If s_{i+1} is sliding a searcher λ from vertex u to another vertex, then for each out-edge (u, v) of u, we insert the following three actions between s_i and s_{i+1}: "slide λ from u to v", "remove λ from v" and "place λ on u". After each inserted sliding action, if a vertex becomes cleared or partially cleared, then remove searchers from this vertex immediately.

Case 2. s_i is a sliding action, say, sliding a searcher from u to v along (u, v). If (u, v) has been cleared by the inserted actions, then delete s_i. Otherwise, we have not placed any searcher on u before the action s_i. Thus, all in-edges of u are cleared and we have a free searcher, say λ, in S'. We replace s_i by the following actions: for each out-edge (u, w) of u, we insert the three actions: "place λ on u", "slide λ from u to w" and "remove λ from w". After each inserted sliding action, if a vertex becomes cleared or partially cleared, then remove searchers from this vertex immediately.

Case 3. s_i is a removing action, say, removing a searcher from u. Because S has minimum number of actions, from the previous modification, we know that u contains at most one searcher in S', and we also know that if u contains one searcher in S', then u must be critical. Thus, we need to delete s_i in the modified strategy.

It is easy to see that S' satisfies the four conditions in the lemma. ■

A digraph is *strong* (or *strong connected*) if there is a directed path between any two vertices on the digraph. We first prove the monotonicity of the internal directed searching on strong digraphs.

Lemma 5.3. *For any strong digraph* D, $\mathrm{ds}(D) = \mathrm{ids}(D) = \mathrm{mids}(D)$.

Proof. We first prove $\mathrm{mids}(D) \le \mathrm{ds}(D)$. Let $\mathrm{ds}(D) = k$ and S be an optimal directed search strategy that satisfies the four conditions in Lemma 5.2. It follows from condition (1) in Lemma 5.2 that S is monotonic. We will construct a monotonic

internal directed search strategy T that uses k searchers to clear D. For convenience, let $D_T = D$, and for any vertex v in D, let $v_T = v$ such that D_T and v_T associate with T while D and v associate with S. S has three different actions, that is, placing, removing and sliding. Let $S = \{s_1, s_2, \ldots, s_\ell\}$ and X_i, $1 \leq i \leq \ell$, be the set of cleared edges just after the action s_i. Let T_i, $1 \leq i \leq \ell$ be a subsequence of actions in T that corresponds to the action s_i, and Y_i, $1 \leq i \leq \ell$, be the set of cleared edges just after the last action of T_i. Note that D (D_T) contains no searchers initially. For the internal directed searching, when a searcher is placed on a vertex, it cannot be removed from the digraph. So, the only action for a searcher located on D_T is sliding. If s_1 is placing a searcher on vertex v, then let T_1 contain only one action, that is, placing a searcher on vertex v_T. It is easy to see that $X_1 = Y_1 = \emptyset$. Assume that $X_{i-1} \subseteq Y_{i-1}$[2] for $2 \leq i \leq \ell$. We now show that $X_i \subseteq Y_i$. We have three cases for s_i, $2 \leq i \leq \ell$.

Case 1. The action s_i is a removing action that removes a searcher from a vertex u. Then we have two subcases regarding the state of u.

1.1 If u is cleared, partially cleared, or critical just before the removing action s_i, let $T_i = \emptyset$, that is, T_i does nothing. Notice that if u is critical just before s_i, then u is occupied by at least two searchers because S is monotonic. Thus, $X_i = X_{i-1} \subseteq Y_{i-1} = Y_i$.

1.2 If u is contaminated just before the removing action s_i, then it follows from Lemma 5.2 that u contains only one searcher, say λ, just before the removing action. Let λ_T be the only searcher on u_T just after the last action in T_{i-1}. Let $P(u_T, v_T)$ be a directed path such that v_T is a critical vertex and every internal vertex on $P(u_T, v_T)$ is contaminated. Let T_i be a sequence of sliding actions that move the searcher λ_T on u_T along the edges of $P(u_T, v_T)$ from u_T to v_T. When λ_T slides along $P(u_T, v_T)$, since each internal vertex on $P(u_T, v_T)$ is contaminated and there is no searchers occupying these vertices, no edge is cleared or recontaminated. Since S is monotonic, we know that $X_i = X_{i-1} \subseteq Y_{i-1} = Y_i$.

Case 2. The action s_i is a placing action that places a searcher on vertex v. If v_T is cleared or the number of searchers on v_T just after the last action of T_{i-1} is greater than or equal to the number of searchers on v just before the placing action s_i, then we set $T_i = \emptyset$. It is easy to see that $X_i = X_{i-1} \subseteq Y_{i-1} = Y_i$. Otherwise, we have two subcases regarding the number of searchers on D_T.

2.1. D_T contains less than k searchers just after the last action of T_{i-1}. Then place a new searcher on v_T. So T_i contains only this placing action. In this case, we have $X_i = X_{i-1} \subseteq Y_{i-1} = Y_i$.

2.2. D_T contains k searchers just after the last action of T_{i-1}. Let $U = \{x_T \in V(D_T)$: just after the last action of T_{i-1}, x_T is a cleared or partially cleared vertex containing at least one searcher, or a critical vertex containing at least two searchers$\}$. If $U = \emptyset$, then D_T has k critical vertices, and so does D. This is a contradiction. Thus, $u \neq \emptyset$. Since D is strong, for every vertex $x_T \in U$, there is a shortest directed path from x_T to v_T, denoted $P(x_T, v_T)$. Let u_T be a vertex in U such that every internal vertex of $P(u_T, v_T)$ does not belong to U. Let λ_T be a searcher on u_T. So T_i contains a sequence of sliding actions that move λ_T along the edges on $P(u_T, v_T)$ from u_T to v_T. For any edge (a_T, b_T) on $P(u_T, v_T)$, there are three subcases when λ_T slides from a_T to b_T.

2.2.1. $a_T = u_T$. Then (a_T, b_T) is cleared and will not be recontaminated.

2.2.2. a_T is an internal vertex on $P(u_T, v_T)$ and there is a searcher on a_T when λ_T sliding along (a_T, b_T). In this case, a_T must be a critical vertex just before λ_T sliding along (a_T, b_T). Since a is also a critical vertex in D just before s_i, it follows from Lemma 5.2 that all out-edges of a are cleared. Thus, both (a, b) and (a_T, b_T) are cleared just before λ_T sliding along (a_T, b_T).

2.2.3. a_T is an internal vertex on $P(u_T, v_T)$ and there is no searcher on a_T when λ_T sliding along (a_T, b_T). In this case, if (a, b) is cleared just before s_i, so is (a_T, b_T). If (a, b) is contaminated just before s_i and (a_T, b_T) is also contaminated just after λ_T sliding along (a_T, b_T), then there is no recontamination. If (a, b) is contaminated just before s_i and (a_T, b_T) is cleared just after λ_T sliding along (a_T, b_T), by the definition of clearing an edge in the internal directed search, all in-edges of a_T are cleared just before λ_T sliding along (a_T, b_T). Thus, (a_T, b_T) will not be recontaminated.

From cases 2.2.1, 2.2.2 and 2.2.3, we know that when λ_T slides along the edges on $P(u_T, v_T)$ from u_T to v_T, some edges may be cleared and no edges are recontaminated. Hence, $X_i = X_{i-1} \subseteq Y_i$.

Case 3. The action s_i is a sliding action that slides a searcher along an edge (u, v) from u to v. If (u_T, v_T) is cleared just after the last action of T_{i-1}, then $T_i = \emptyset$; otherwise, T_i contains only one sliding action that clears (u_T, v_T) by moving a searcher along (u_T, v_T) from u_T to v_T. Since S is monotonic and $X_{i-1} \subseteq Y_{i-1}$, there is no recontamination when the searcher slides along (u_T, v_T) from u_T to v_T. Thus, $X_i \subseteq Y_i$.

From the above three cases, we know that T is a monotonic internal directed search strategy that uses k searchers to clear D_T. Hence, we have $\text{mids}(D) \leq \text{ds}(D)$. It is easy to see that $\text{ds}(D) \leq \text{ids}(D) \leq \text{mids}(D)$. Therefore, $\text{ds}(D) = \text{ids}(D) = \text{mids}(D)$. ■

Similar to Lemma 5.3, we can prove the monotonicity of the internal strong searching on strong digraphs. Let $\text{ss}(D)$ denote the strong search number of digraph D [18].

[2] When we compare edge set X_j with Y_j, $1 \leq j \leq \ell$, we consider Y_j as a subset of $E(D)$ because $D_T = D$.

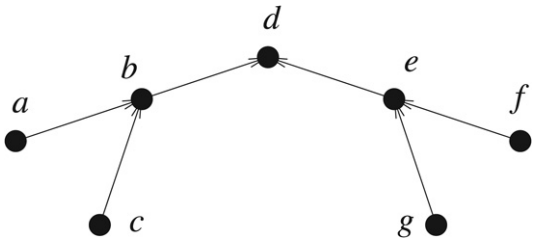

Fig. 3. $\mathrm{iss}(D_1) = 1$ and $\mathrm{miss}(D_1) = 2$.

Lemma 5.4. *For any strong digraph D,* $\mathrm{ss}(D) = \mathrm{iss}(D) = \mathrm{miss}(D)$.

Proof. Since the strong search problem is monotonic [18], there always exists an optimal strong search strategy satisfying the four conditions in Lemma 5.2. Since the intruder in the internal strong search problem behaves in the same way as the intruder in the internal directed search problem, the proof of this lemma is the same as that of Lemma 5.3 if we replace all terms related to the internal directed search by the corresponding terms in the internal strong search and add one more case as follows.

Case 4. If the action s_i is a sliding action that slides a searcher along an edge (u, v) from v to u, then T_i contains only one sliding action that slides a searcher along (u_T, v_T) from v_T to u_T. Since S is monotonic and $X_{i-1} \subseteq Y_{i-1}$, there is no recontamination when the searcher slides along (u_T, v_T) from v_T to u_T. Thus, $X_i \subseteq Y_i$. ■

We now prove the monotonicity of the internal directed searching on general digraphs.

Theorem 5.5. *For any digraph D,* $\mathrm{ids}(D) = \mathrm{mids}(D)$.

Proof. Let $D_1, \ldots, D_N$ be all strong components of D in an acyclic ordering, which is a linear ordering of all strong components such that if $(u, v) \in E(D)$, $u \in V(D_i)$ and $v \in V(D_j)$, then $i \leq j$. Let $\mathrm{ids}(D) = k$ and S be an internal directed search strategy that uses k searchers to clear D. Let $A_S = (e_1, e_2, \ldots, e_m)$ be the edge sequence of D such that S clears all edges of D in this order. Note that some edges may appear more than one time in A_S because S may be non-monotonic. From A_S, we select all edges of D_1 and let them form a subsequence, denoted by $A_S(D_1)$, which keep the original ordering in A_S. If the subsequence $A_S(D_1)$ is not the prefix of A_S, there must be two adjacent elements e_i and e_{i+1} in A_S such that $e_i \notin A_S(D_1)$ and $e_{i+1} \in A_S(D_1)$. Since each searcher cannot be removed from D and it can only move along the direction of an edge, the searcher who clears e_i cannot move to any vertex of D_1 after e_i is cleared. So, this searcher cannot be used to clear any edge of D_1 after e_i is cleared and it cannot be used to block the intruder because the intruder cannot move from any D_j, $j > 1$, to D_1. If we swap the two actions in S that clear e_i and e_{i+1}, the number of searchers required by the new strategy is still k. We can keep doing such swap operations until the prefix of the cleared edge sequence is $A_S(D_1)$. After all edges of D_1 are cleared, they cannot become recontaminated because the intruder must follow edge directions. Since D_1 is strong, from Lemma 5.3, we can use a monotonic internal directed search strategy to clear D_1 and replace $A_S(D_1)$ by the corresponding cleared edge sequence. Let S_1 be the modified directed search strategy that uses k searchers to clear D by monotonically clearing D_1 first and then clearing the remaining edges of D. Similarly, we can rearrange the actions of S_1 such that, after D_1 is cleared, we clear all out-going edges of D_1 and then clear D_2 monotonically. This rearrangement does not increase the number of searchers. We can repeat this rearrangement process until D_N is cleared monotonically. Hence, we obtain a monotonic internal directed search strategy that uses k searchers to clear D. Therefore, $\mathrm{mids}(D) \leq \mathrm{ids}(D)$. Since $\mathrm{ids}(D) \leq \mathrm{mids}(D)$, we have $\mathrm{ids}(D) = \mathrm{mids}(D)$. ■

We now show the non-monotonicity of the internal strong searching. From Lemma 5.4, we need to consider digraphs that are not strong, such as acyclic digraphs. Let D_1 be the digraph illustrated in Fig. 3. The following internal search strategy can clear D_1 using one searcher: A searcher is first placed on a, then it slides from a to b, b to c, c to b, b to d, d to e, e to f, f to e, e to g, g to e, and e to d. Thus, $\mathrm{iss}(D_1) = 1$. Note that in this strategy, when the searcher slides from e to f, the edge (e, d) is recontaminated. It is easy to see that $\mathrm{miss}(D_1) = 2$. Hence, we have the following result.

Theorem 5.6. *For the digraph* D_1 *depicted in Fig. 3,* $\mathrm{iss}(D_1) < \mathrm{miss}(D_1)$.

Note that for any connected acyclic digraph, the internal strong search number is always 1. But there are some acyclic digraphs, the monotonic internal strong search number is $\Omega(\log n)$, where n is the number of vertices in the digraph. For example, let T be the orientation of a complete rooted binary tree with n vertices such that for each edge connecting a child with its parent, the child is the tail and the parent is the head of this edge. We can show that $\mathrm{iss}(T) = 1$ and $\mathrm{miss}(T) = \log_2(n + 1) - 1$. Thus, the ratio of the monotonic internal strong search number to the internal strong search number may be arbitrarily large.

Contrary to the internal strong searching, we can show that the internal weak search problem is monotonic on acyclic graphs.

Lemma 5.7. *For any acyclic digraph D,* $\mathrm{ids}(D) = \mathrm{iws}(D) = \mathrm{miws}(D)$.

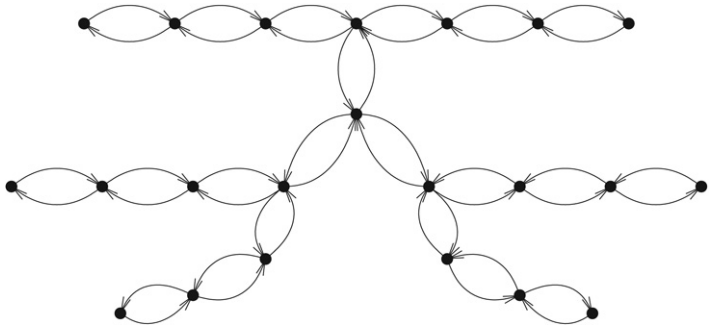

Fig. 4. $\mathrm{iws}(D_2) = 4$ and $\mathrm{miws}(D_2) = 5$.

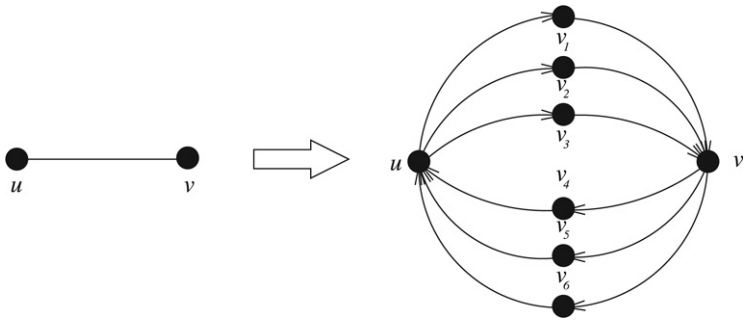

Fig. 5. Replacing edge uv in G with six directed paths.

Proof. Let $\mathrm{ids}(D) = k$. In the proof of Theorem 5.5, if each strong component is a vertex, say v_i, $1 \le i \le N$, then the monotonic internal directed search strategy obtained in the end of the proof is a strategy that uses k searchers to clear all vertices of D in the acyclic ordering $v_1, \ldots, v_N$. Note that for each vertex v_i, $1 \le i \le N$, with out-degree $\delta^+(v_i)$, we need to slide $\delta^+(v_i)$ searchers along each out-edge to clear vertex v_i. If we need to place searchers on v_i, we can arrange all placing actions before all sliding actions. Thus, such a monotonic internal directed search strategy is also a monotonic internal weak search strategy, which uses k searchers to clear D. Thus, $\mathrm{miws}(D) \le \mathrm{ids}(D)$. Since $\mathrm{ids}(D) \le \mathrm{iws}(D) \le \mathrm{miws}(D)$, we have $\mathrm{ids}(D) = \mathrm{iws}(D) = \mathrm{miws}(D)$. ■

From Lemma 5.7, we need to consider digraphs that are not acyclic to prove the non-monotonicity of the internal weak search problem. Let D_2 be the digraph illustrated in Fig. 4. We can show that $\mathrm{iws}(D_2) = 4$ and $\mathrm{miws}(D_2) = 5$. Thus, we have the following result.

Theorem 5.8. *For the digraph D_2 depicted in Fig. 4,* $\mathrm{iws}(D_2) < \mathrm{miws}(D_2)$.

Unlike the ratio of the monotonic internal strong search number to the internal strong search number which may be arbitrarily large, we conjecture that the ratio of the monotonic internal weak search number to the internal weak search number is less than 2.

6. NP-completeness results

Kirousis and Papadimitriou [11] proved that the node search problem is NP-complete. In this section, we will establish relations between the node search number and the directed/mixed directed/internal directed/internal strong/internal weak search numbers. Using these relations, we prove that all five digraph search problems are NP-hard. From Theorems 3.9, 4.2 and 5.5, we then prove the directed/mixed directed/internal directed search problems belong to the NP class. We also prove that the internal strong search problem belongs to the NP class, although it is not monotonic.

For a connected graph G, the minimum number of searchers needed to clear G in the node search model is the *node search number* of G, denoted by $\mathrm{ns}(G)$.

For any connected graph G, let D_G be a digraph obtained from G by adding three length two directed cycles on each vertex of G, and replacing each edge $uv \in E(G)$ with six directed paths (u, v_1, v), (u, v_2, v) (u, v_3, v), (v, v_4, u), (v, v_5, u) and (v, v_6, u) (see Fig. 5). We have the following relations between search numbers on G and D_G.

Theorem 6.1. *For any connected graph G and its corresponding digraph D_G described above,* $\mathrm{ds}(D_G) = \mathrm{xds}(D_G) = \mathrm{iss}(D_G) = \mathrm{ids}(D_G) = \mathrm{iws}(D_G) = \mathrm{ns}(G) + 1$.

Proof. Since $\mathrm{iss}(D_G) \le \mathrm{ids}(D_G) \le \mathrm{iws}(D_G)$ and $\mathrm{xds}(D_G) \le \mathrm{ds}(D_G)$, we only need to show that $\mathrm{iws}(D_G) \le \mathrm{ns}(G) + 1$, $\mathrm{ns}(G) \le \mathrm{iss}(D_G) - 1$, $\mathrm{ds}(D_G) \le \mathrm{ns}(G) + 1$, and $\mathrm{ns}(G) \le \mathrm{xds}(D_G) - 1$.

We first show $\mathrm{iws}(D_G) \leq \mathrm{ns}(G)+1$. Let S be a monotonic node search strategy that clears G using k searchers. Notice that S is a sequence of placing and removing actions. We can construct an internal weak search strategy S' by inserting actions into S as follows. Initially, let S' be obtained from S by deleting all removing actions from S. For each placing action s in S that places a searcher on vertex u, if the number of searchers on D_G is less than k, then we keep the same placing action in S'; otherwise, there is a vertex in D_G that contains one more searcher than the corresponding vertex in G, we then insert actions into the current S' just after s by sliding this searcher along a directed path to the vertex u on D_G. After u is occupied by the searcher in S', we insert actions into S' to clear the three length two directed cycles incident on u by using an additional searcher. Let E_s be the set of new cleared edges in G caused by action s. If $E_s \neq \emptyset$, then for each edge $uv \in E_s$, we insert actions into the current S' such that we can use an additional searcher to clear the six directed paths corresponding to uv. Thus, S' can clear D_G using $k+1$ searchers. Therefore, $\mathrm{iws}(D_G) \leq \mathrm{ns}(G)+1$. Similarly, we can prove $\mathrm{ds}(D_G) \leq \mathrm{ns}(G)+1$.

We now show that $\mathrm{ns}(G) \leq \mathrm{iss}(D_G) - 1$. Since D_G is strong, it follows from Lemma 5.4 that there exists a monotonic internal strong search strategy S' that clears D_G using k searchers. We can construct a node search strategy S that clears G using $k-1$ searchers. For any edge uv in G, let D_{uv} be the subdigraph of D_G that consists of the six directed paths of length two between u and v (see Fig. 5). Because there are three length two directed cycles on u and v, respectively, from the structure of D_{uv} we know that there exists an action t in S' such that D_{uv} contains at least three searchers just after t. We can design actions in S that clear uv by two searchers located on u and v. For all other actions in S' with respect to D_{uv} or the three length two directed cycles on u and v respectively, S does nothing. Thus, S can clear G using $k-1$ searchers, and therefore, $\mathrm{ns}(G) \leq \mathrm{iss}(D_G) - 1$. Similarly, we can prove $\mathrm{ns}(G) \leq \mathrm{xds}(D_G) - 1$. ■

Because the node search problem is NP-complete [11], from Theorem 6.1, we can prove the following result.

Theorem 6.2. *The problem of computing the search number in the directed (resp. mixed directed, internal directed, internal strong, or internal weak) search model is NP-hard.*

From Theorems 3.9, 4.2 and 5.5, we can prove that the directed, mixed directed, and internal directed search problems belong to the NP class. Hence, we have the following results.

Theorem 6.3. *The problem of computing the search number in the directed (resp. mixed directed, or internal directed) search model is NP-complete.*

One of the major application of the monotonicity in a searching problem is to show the searching problem belonging to the NP class. Although the internal strong search problem is non-monotonic by Theorem 5.6, we can still prove that it is NP-complete, which answers an open question raised by Dimitrios Thilikos. Therefore, we have the main result of this section.

Theorem 6.4. *The problem of computing the internal strong search number is NP-complete.*

Proof. From Theorem 6.2 we only need to show that the internal strong search problem is in NP. Let D be a given connected digraph and k be a given integer. We can compute all strong components $D_1, D_2, \ldots, D_m$ of D in linear time [17]. We first show that $\mathrm{iss}(D) = \max_{1\leq i\leq m} \mathrm{iss}(D_i)$. It follows from Theorem 6 in [2] that $\mathrm{iss}(D) \geq \max_i \mathrm{iss}(D_i)$ because each D_i is a strong subdigraph of D. Thus, we only need to prove that $\mathrm{iss}(D) \leq \max_i \mathrm{iss}(D_i)$. Without loss of generality, suppose that $D_1, D_2, \ldots, D_m$ form an acyclic ordering of the strong components of D, which is a linear ordering of all strong components such that if $(u, v) \in E(D)$, $u \in V(D_i)$ and $v \in V(D_j)$, then $i \leq j$. First, we use $\mathrm{iss}(D_1)$ searchers to clear D_1. Notice that every edge with one endpoint in D_1 and the other endpoint in D_i $(i > 1)$ has its tail in D_1 and head in D_i. Thus when we clear D_1, we can leave these edges contaminated, which will not cause any recontamination. After clearing D_1, we use one searcher to clear each edge with tail in D_1 and head in D_i $(i > 1)$, and then we slide all searchers from D_1 to D_2 because D is connected. Repeat the above process with D_2, D_3, and so on until D_m is cleared. Hence, we can clear D with no more than $\max_i \mathrm{iss}(D_i)$ searchers. Therefore, $\mathrm{iss}(D) = \max_i \mathrm{iss}(D_i)$.

For each strong component D_i, $1 \leq i \leq m$, it follows from Lemma 5.4 that there exists a monotonic optimal internal strong search strategy such that every edge in D_i is cleared exactly once. Thus, a nondeterministic algorithm needs only guess an ordering of all edges in D_i. In polynomial time we can check whether we can use k searchers to clear D_i by clearing each edge of D_i in the ordering. Thus, the internal strong search problem belongs to the NP class. ■

7. Conclusion

In this paper, we investigated five digraph search problems: directed searching, mixed directed searching, internal directed searching, internal strong searching, and internal weak searching. We applied the method proposed by Bienstock and Seymour [5] to prove the monotonicity of the mixed directed searching, and then proved the monotonicity of the directed searching. For the three internal search models without "jumping", we showed that the internal directed searching is monotonic, but the internal strong searching and the internal weak searching are non-monotonic. Note that the internal strong searching is a "strong" version of the internal directed searching, and the internal weak searching is a "weak" version of the internal directed searching. The internal edge searching is an analogy of the internal directed searching on undirected graphs, which is also non-monotonic [4]. This is the reason that some researchers conjectured that the internal directed searching should be non-monotonic as well. To our surprise, we proved that it is monotonic indeed.

We also established relations for the above five digraph search problems. From these relations, we showed that these five digraph search problems are NP-hard. We showed that the directed, mixed directed and internal directed search problems are NP-complete from their monotonicity property. In particular, we also showed the internal strong search problem is NP-complete although it is non-monotonic. This solves the open problem on whether a non-monotonic searching problem can be NP-complete.

We showed by examples that the ratio of the monotonic internal strong search number to the internal strong search number may be as large as $\Omega(\log n)$, where n is the number of vertices in the digraph. We conjecture that $O(\log n)$ is an upper bound. One open problem is to find a constant upper bound for the ratio of the monotonic internal weak search number to the internal weak search number. We conjecture that this ratio is less than 2.

Another interesting problem left unresolved is whether the internal weak search problem is NP-complete.

Acknowledgments

We wish to thank Danny Dyer and Runtao Zhang for valuable discussions on this paper. The first author's research was supported in part by NSERC and MITACS.

References

[1] B. Alspach, Searching and sweeping graphs: A brief survey, Le Matematiche, 34 pp.
[2] B. Alspach, D. Dyer, D. Hanson, B. Yang, Arc searching digraphs without jumping, in: Proceedings of the 1st International Conference on Combinatorial Optimization and Applications, COCOA'07, in: Lecture Notes in Computer Science, vol. 4616, Springer, 2007, pp. 354–365.
[3] J. Barat, Directed path-width and monotonicity in digraph searching, Graphs and Combinatorics 22 (2006) 161–172.
[4] L. Barrière, P. Fraigniaud, N. Santoro, D. Thilikos, Searching is not jumping, in: Proceedings of the 29th Workshop on Graph Theoretic Concepts in Computer Science, WG'03, in: Lecture Notes in Computer Science, vol. 2880, Springer, 2003, pp. 34–45.
[5] D. Bienstock, P. Seymour, Monotonicity in graph searching, Journal of Algorithms 12 (1991) 239–245.
[6] W. Evans, M. Safari, Directed one trees, in: EuroComb 2005, DMTCS proceedings. AE, 2005, pp. 67–72.
[7] F. Fomin, D. Thilikos, On the monotonicity of games generated by symmetric submodular functions, Discrete Applied Mathematics 131 (2003) 323–335.
[8] F. Fomin, D. Thilikos, An annotated bibliography on guaranteed graph searching, manuscript.
[9] Y.O. Hamidoune, On a pursuit game on Cayley graphs, European Journal of Combinatorics 8 (1987) 289–295.
[10] T. Johnson, N. Robertson, P. Seymour, R. Thomas, Directed tree-width, Journal of Combinatorial Theory Series B 82 (2001) 138–154.
[11] L. Kirousis, C. Papadimitriou, Searching and pebbling, Theoretical Computer Science 47 (1996) 205–218.
[12] A. LaPaugh, Recontamination does not help to search a graph, Journal of ACM 40 (1993) 224–245.
[13] R.J. Nowakowski, Search and sweep numbers of finite directed acyclic graphs, Discrete Applied Mathematics 41 (1993) 1–11.
[14] N. Megiddo, S. Hakimi, M. Garey, D. Johnson, C. Papadimitriou, The complexity of searching a graph, Journal of ACM 35 (1998) 18–44.
[15] B. Reed, Introducing directed tree width, in: 6th Twente Workshop on Graphs and Combinatorial Optimization (Enschede, 1999), in: Electron. Notes Discrete Math., vol. 3, Elsevier, Amsterdam, 1999, 8 pp. (electronic).
[16] M. Safari, D-Width: A more natural measure for directed tree width, in: Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science, in: Lecture Notes in Computer Science, vol. 3618, Springer, 2005, pp. 745–756.
[17] R.E. Tarjan, Depth first search and linear graph algorithms, SIAM Journal on Computing 1 (1972) 146–160.
[18] B. Yang, Y. Cao, Monotonicity of strong searching on digraphs, Journal of Combinatorial Optimization 14 (2007) 411–425.
[19] B. Yang, Y. Cao, On the monotonicity of weak searching, in: Proceedings of the 14th International Computing and Combinatorics Conference, COCOON'08, in: Lecture Notes in Computer Science, vol. 5092, Springer, Berlin, 2008, pp. 52–61.
[20] B. Yang, Y. Cao, Digraph searching, directed vertex separation and directed pathwidth, Discrete Applied Mathematics 156 (2008) 1822–1837.