

Maximum vertex occupation time and inert fugitive: recontamination does help

Dariusz Dereniowski*

Department of Algorithms and System Modeling,
Gdańsk University of Technology, Poland

deren@eti.pg.gda.pl

Abstract: Given a simple graph G , we consider the node search problem with inert fugitive. We are interested in minimizing the maximum vertex occupation time, i.e. the maximum number of steps in which a vertex is occupied by a searcher during a search of G . We prove that a search program which does not allow a recontamination may not find an optimal solution to this problem. Moreover, the difference between the minimum maximum vertex occupation time computed by a monotone search program and a program without such a restriction may be arbitrarily large.

Keywords: combinatorial problems, inert fugitive, monotone search, node search, recontamination

1 Introduction

There are several models of the graph searching problem according to the assumptions concerning the fugitive and the searchers. We assume that the fugitive is *fast* (when he moves he can traverse an unguarded path of arbitrary length), *invisible* (the searchers can use only the history of their moves to deduce a potential position of the fugitive), and *omniscient* (has a complete knowledge and always chooses the most advantageous for him position). In the *standard search* the fugitive can always change his location while in the *inert search* the fugitive can move only just before a searcher is about to capture him. We may distinguish three ways the searchers can clear a graph: *edge search*, where an edge is cleared by sliding a searcher along the edge; *node search*, where an edge is declared clear if both of its endpoints are occupied by searchers; *mixed search*, where an edge can be cleared according to both of the above rules.

*Partially supported by the Foundation for Polish Science (FNP) and by MNiSW grant N516 029 31/2941

Kirousis and Papadimitriou introduced the standard node search problem [5, 6]. Earlier, the standard edge search problem has been studied independently by Parsons [9] and Petrov [10]. The mixed search problem has been defined in [1]. The inert node search was introduced in [2] and the description of the inert edge and mixed search can be found in [14].

Let us recall some optimization criteria studied in the context of the inert search problem. One of the most interesting parameters in various search problems is the minimum number of searchers required to search a graph. It turns out that this number is related to the treewidth of a graph [2]. Another parameter is the *cost*, defined in [3] as the sum of the number of searchers used, where the sum is taken over all moves of the search strategy. The cost is related to the problem of finding minimum number of edges, which one has to add to the graph in order to obtain a chordal supergraph, a parameter called *fill-in*, as noted in [4]. In this paper we are interested in a parameter called *minimum occupation time* introduced in [4] where it has been proved to be equal to the treewidth of a graph, a generalization of bandwidth. Some properties and lower bounds for the treewidth can be found in [11].

Recontamination is one of the most important aspects of the search problems. A direct consequence of monotonicity of a search is that the problem belongs to NP. Furthermore, it usually turns out that in such cases the optimization parameter of the search problem is related to a well known graph parameter, see e.g. [2, 3, 4, 6, 12]. LaPaugh [7] proved the monotonicity property for minimizing the number of searchers in the standard edge search problem. Then, this result has been extended to standard node search [6] and standard mixed search [1]. It is also known that there exist optimal monotone search in the case of inert node search [2, 13] or inert edge and inert mixed search [14]. The monotonicity of the standard node search of the smallest cost has been proved in [3]. A proof which gives monotonicity for minimizing the number of searchers both for inert and standard node search is given in [8].

In this paper we address the problem of the monotonicity of inert search programs minimizing the maximum vertex occupation time. In Section 2 we introduce the necessary notation and give a formal statement of the problem. One of the open questions stated in [4] is if recontamination can help to search a graph. In this paper we give a positive answer to this question by proving that for any integer k there exist a simple graph G for which the difference between the maximum vertex occupation time of each monotone search program and an optimal non-monotone search program is at least k . The construction of such a family of graphs is given in Section 3.

2 Preliminaries

Given a simple graph $G = (V(G), E(G))$, a vertex $v \in V(G)$ is *clear* if it cannot contain the fugitive, otherwise v is *contaminated*. If a vertex is occupied by a searcher then we say that it is *guarded*. We say that $P \subseteq G$ is an *unguarded path* if all the internal vertices of P are contaminated.

A search program Π is a sequence of pairs $(A_0, Z_0), (A_1, Z_1), \dots, (A_m, Z_m)$, where $A_i \subseteq V(G), Z_i \subseteq A_i$ are, respectively, the sets of clear and guarded vertices in a step $i, i = 0, \dots, m$. The initial state is $A_0 = Z_0 = \emptyset$. In the final state we require all the vertices to be cleared, i.e. $A_m = V(G)$. For each $i = 1, \dots, m$ there exists a vertex $v_i \in V(G)$ and a set $U_i \subseteq A_{i-1}$ such that $\{v_i\} = A_i \setminus A_{i-1}$ and $Z_i = U_i \cup \{v_i\}$. The set $A_i \setminus Z_i$ contains all the vertices $x \in A_{i-1}$ such that there is no unguarded path connecting v_i and x in step i . In other words, we clear the vertex v_i in the i th step by placing a searcher at v_i while the other searchers are placed at the vertices in Z_i in order to protect the remaining vertices in A_i from possible recontamination in this step. If for each $i = 1, \dots, m$ it holds $A_{i-1} \subseteq A_i$ then the search program Π is *monotone*. Note that $m = |V(G)|$ for each monotone search program, because $A_i = A_{i-1} \cup \{v_i\}$, for $i = 1, \dots, m$. For $u \in V(G)$ define

$$g(\Pi, u) = |\{i \in \{1, \dots, m\} : u \in Z_i\}|,$$

i.e. $g(\Pi, u)$ equals the number of steps in which u is occupied by a searcher. Then we define the *vertex occupation time of Π* as

$$\text{ot}(\Pi) = \max\{g(\Pi, u) : u \in V(G)\},$$

and the *vertex occupation time of G* as

$$\text{ot}(G) = \min\{\text{ot}(\Pi) : \Pi \text{ is a search program for } G\}.$$

Finally let

$$\text{mot}(G) = \min\{\text{ot}(\Pi) :$$

$\Pi \text{ is a monotone search program for } G\}.$

Note that in the case of a monotone search program Π we have $A_i = A_{i-1} \cup \{v_i\}, i = 1, \dots, m$, so in order to describe the search program Π we may use a permutation $\pi: \{1, \dots, m\} \rightarrow V(G), \pi(i) = v_i$ for $i = 1, \dots, m, m = |V(G)|$. Observe that if the vertices $v_1, \dots, v_{i-1}, i \leq m$, have been already cleared and we place in the i th step a searcher at a vertex v_i , then the set Z_i of guarded vertices must contain v_i and all the vertices $v_j, j < i$ such that there exists a path P connecting v_i and v_j , where all the internal vertices of P belong to $\{v_{i+1}, \dots, v_m\}$. This however is not true in the case of a search program Π' which is not monotone. In order to extend this notation to such cases let

$$\pi' : \{1, \dots, M\} \rightarrow \{v_i, \overline{v}_i : i = 1, \dots, |V(G)|\},$$

$M \geq m$, where $\pi'(i) = v_k$ means placing a searcher at a contaminated vertex v_k while $\pi'(i) = \overline{v}_k$ indicates that from now on the vertex v_k is treated as if it was contaminated until it is cleared again. The set Z_i , as before, contains v_k and the cleared vertices v_j , such that there exists an unguarded path P connecting v_k to v_j . A path P is unguarded in this case if each internal vertex v_l of P satisfies one of the conditions

$$v_l \notin \{\pi'(1), \dots, \pi'(i-1)\}, \tag{1}$$

$$\pi'(t) = \overline{v_l} \text{ and } v_l \notin \{\pi'(t+1), \dots, \pi'(i-1)\} \quad (2)$$

for some $t < i$. The condition (1) means that v_l has been not cleared yet, while (2) says that v_l has been not cleared again since it has been declared contaminated in a step t . Note that $\pi'(i) = \overline{v_k}$ does not imply any movement of searchers or the fugitive, but it means that v_k will not be protected by searchers from possible recontamination. Since this cannot force the fugitive to move, we have $Z_i = \emptyset$, which means that a move of this type does not contribute to $\text{ot}(G)$.

Definition 1 For a monotone search program Π and $X \subseteq V(G)$ let $\text{first}_\pi(X)$ be the vertex $v \in X$ satisfying $\pi^{-1}(v) < \pi^{-1}(u)$ for each $u \in X, u \neq v$.

Definition 2 Let Π be any search program for G and let $v \in V(G)$. Define $\text{next}_\pi(v)$ to be the multiset containing i copies of v , where i is the number of times v has been cleared, and i copies of a vertex $u \in V(G) \setminus \{v\}$ assuming that i times occurred a situation of clearing the vertex u and guarding v in the same step.

Observe that $\text{next}_\pi(v)$ contains at most one occurrence of a vertex if Π is monotone. From the definition of next_π it immediately follows that $|\text{next}_\pi(v)| = g(\Pi, v)$ for $v \in V(G)$, and consequently

$$\text{ot}(\Pi) = \max\{|\text{next}_\pi(v)| : v \in V(G)\}. \quad (3)$$

Definition 3 Given a monotone search program Π for a graph G , let $\text{prev}_\pi(v)$ be the set of such vertices $u \in V(G)$ that during clearing u the vertex v is unguarded and there exists an unguarded path connecting u and v .

3 The gap between *mot* and *ot*

In this section we give an example of a family of graphs G_i such that, given an integer k , there exists an integer $i \geq 0$ for which the inequality holds

$$\text{mot}(G_i) - \text{ot}(G_i) \geq k. \quad (4)$$

Let

$$N = 24k, l = 8 \quad (5)$$

and define a graph G_N as follows. $V(G_N) = V_1 \cup V_2 \cup X \cup \{s_1, s_2\}$, where $V_i = C_1^i \cup \dots \cup C_{2l}^i$, $i = 1, 2$, $X = X_1 \cup \dots \cup X_{3l-1}$. Let

$$|C_j^i| = N/2 \text{ for } i = 1, 2, j = 1, \dots, 2l, \quad (6)$$

and let

$$|X_i| = N/6 \text{ for } i = 1, \dots, 3l - 1. \quad (7)$$

Now we define the edge set $E(G_N)$ of G_N . For $C \in \{C_j^i : i = 1, 2, j = 1, \dots, 2l\} \cup \{X_i : i = 1, \dots, 2l - 1\}$ we have $\{u, v\} \in E(G_N)$ for all $u, v \in C$, i.e. each set

C_j^i , $i = 1, 2, j = 1, \dots, 2l$, and X_i , $i = 1, \dots, 3l - 1$ is a clique. In addition, we use a notation $(Y, Z) = \{\{y, z\} : y \in Y, z \in Z\}$ for two vertex sets Y and Z . Then, except for the edges in the above cliques we have that $(X_i, X_{i+1}) \subseteq E(G_N)$, $i = 1, \dots, 3l - 2$, $(C_j^i, C_{j+1}^i) \subseteq E(G_N)$ for $i = 1, 2, j = 1, \dots, 2l - 1$, $(\{s_1\}, C_l^1 \cup C_{l+1}^1 \cup X_1) \subseteq E(G_N)$ and $(\{s_2\}, C_l^2 \cup C_{l+1}^2 \cup X_{3l-1}) \subseteq E(G_N)$. Fig. 1 depicts the graph G_N (dashed lines represent edges between each pair of vertices from the sets in opposite endpoints of the line).

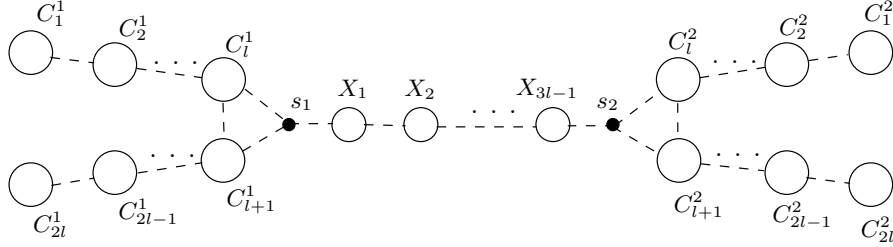


Figure 1: The graph G_N

Assume for the remaining part of this section that Π is a monotone search program for G_N satisfying

$$\text{ot}(\Pi) \leq N + k, \quad (8)$$

and π is the corresponding permutation of the vertices of G_N .

Let us first describe in an informal way where monotone search programs fail. Regardless of the ordering of the vertices in V_i , each set C_j^i , $j \neq 1, j \neq 2l$, contains a vertex, which has to be guarded for at least N turns, because two consecutive sets C_j^i, C_{j+1}^i form a clique on N vertices. Moreover, roughly speaking, the cliques in V_i have to be cleared one by one according to increasing (or decreasing, but this case is symmetric) values of their indices j . A situation where we start clearing a subset V_i while some vertices in $V(G_N) \setminus V_i$ are already clear must occur, because there are two disjoint sets V_i . So, during clearing C_1^i, \dots, C_l^i we must avoid a recontamination in $V(G_N) \setminus V_i$. To protect the clear vertices in $V(G_N) \setminus V_i$ during clearing V_i we must either guard them or clear all the vertices of some separator disconnecting those vertices and V_i (and then guard the vertices of the separator in the following steps). We are forced to clear some separators in $V(G_N) \setminus V_i$ during clearing V_i , because of the length of the sequence C_1^i, \dots, C_l^i . Since the cleared vertices in V_i with the minimum distance to the separator have to be guarded during clearing the separator in order to avoid recontamination in V_i , this separator cannot be of size bigger than k . There are only two such separators in G_N and we will show that this is not enough, because of the size of each set C_j^i and $l = 8$.

Inequality (8) will lead us to a contradiction in Lemma 3, but in Lemmas 1 and 2 we will analyze some properties of Π with bound (8). On the other hand we will describe in Lemma 4 a non-monotone search program Π' with $\text{ot}(\Pi') = N + 1$. We start by proving that the vertices $\text{first}_\pi(C_j^i)$, $i \in \{1, 2\}$,

$j = 1, \dots, 2l$ have to be cleared in a specific order by each monotone search program II.

Lemma 1 *We have that $\text{first}_\pi(V_i) \in C_1^i \cup C_{2l}^i$ for $i = 1, 2$. Moreover, two following implications hold*

$$\begin{aligned} \text{first}_\pi(V_i) \in C_1^i &\Rightarrow \\ \pi^{-1}(\text{first}_\pi(C_j^i)) &< \pi^{-1}(\text{first}_\pi(C_{j+1}^i)), \end{aligned}$$

$$\begin{aligned} \text{first}_\pi(V_i) \in C_{2l}^i &\Rightarrow \\ \pi^{-1}(\text{first}_\pi(C_j^i)) &> \pi^{-1}(\text{first}_\pi(C_{j+1}^i)), \end{aligned}$$

for each $i = 1, 2$ and $j = 1, \dots, 2l - 1$.

Proof: The case when $v = \text{first}_\pi(V_i) \in C_j^i$ for $1 < j < 2l$ is not possible: otherwise we would have $C_{j-1}^i \cup C_j^i \cup C_{j+1}^i \subseteq \text{next}_\pi(v)$, because all the vertices in $(C_{j-1}^i \cup C_j^i \cup C_{j+1}^i) \setminus \{v\}$ are adjacent to v and cleared later than v . This implies, by (5) and (6), that $|\text{next}_\pi(v)| \geq 3N/2 > N + k$, which contradicts (3) and (8).

Let $v = \text{first}_\pi(V_i) \in C_1^i$. If j is the smallest index satisfying inequality $\pi^{-1}(\text{first}_\pi(C_j^i)) > \pi^{-1}(\text{first}_\pi(C_{j+1}^i))$ then, as before, $C_{j-1}^i \cup C_j^i \cup C_{j+1}^i \subseteq \text{next}_\pi(\text{first}_\pi(C_j^i))$, which is not possible. The case when $v \in C_{2l}^i$ is analogous. \square

Since the cases $\text{first}_\pi(V_i) \in C_1^i$ and $\text{first}_\pi(V_i) \in C_{2l}^i$ are symmetric, we assume in the following that $\text{first}_\pi(V_i) \in C_1^i$ for $i = 1, 2$. Now we prove that, regardless of the ordering according to which the vertices in V_i , $i = 1, 2$, are cleared, each vertex $\text{first}_\pi(C_j^i)$, $j = 1, \dots, 2l - 1$, has to be guarded for at least N turns, even when we restrict the search program to the subgraph induced by V_i . We also show that at least $N/2 - 2k$ vertices of each clique C_1^i, \dots, C_{l-1}^i , $i = 1, 2$, have to be cleared before clearing s_i .

Lemma 2 *For $i = 1, 2$ it holds*

$$|\text{next}_\pi(\text{first}_\pi(C_j^i)) \cap V_i| \geq N, \quad j = 1, \dots, 2l - 1, \quad (9)$$

$$|C_j^i \cap \text{prev}_\pi(s_i)| \geq N/2 - 2k, \quad j = 1, \dots, l - 1. \quad (10)$$

Proof: By Lemma 1, $C_{j+1}^i \subseteq \text{next}_\pi(\text{first}_\pi(C_j^i))$ for each $j = 1, \dots, 2l - 1$. Moreover, by the definition we have $C_j^i \subseteq \text{next}_\pi(\text{first}_\pi(C_j^i))$. So, (6) implies (9). Let $C \subseteq C_j^i$ satisfy

$$\forall v \in C \quad \pi^{-1}(v) > \pi^{-1}(\text{first}_\pi(C_{j+1}^i)) \quad (11)$$

for some $j \in \{1, \dots, l-1\}$. By Lemma 1 we obtain $\text{first}_\pi(C_{j+2}^i) \in \text{next}_\pi(\text{first}_\pi(C_{j+1}^i))$ which implies that $C_{j+2}^i \subseteq \text{next}_\pi(\text{first}_\pi(C_{j+1}^i))$. Clearly, $C_{j+1}^i \subseteq \text{next}_\pi(\text{first}_\pi(C_{j+1}^i))$. By (11), $C \subseteq \text{next}_\pi(\text{first}_\pi(C_{j+1}^i))$. So, $|\text{next}_\pi(\text{first}_\pi(C_{j+1}^i))| \geq N + |C|$. Thus, by (3) and (8),

$$|C| \leq k. \quad (12)$$

This proves that there are at least $N/2 - k$ vertices $v \in C_j^i \setminus C$ such that $\pi^{-1}(v) < \pi^{-1}(\text{first}_\pi(C_{j+1}^i))$. By Lemma 1 we have that the path $v, \text{first}_\pi(C_{j+1}^i), \dots, \text{first}_\pi(C_l^i), s_i$ is unguarded during clearing the above vertices v . If $v \in \text{next}_\pi(s_i) \cap (C_j^i \setminus C) \neq \emptyset$, i.e. there exists a vertex v in $C_j^i \setminus C$ cleared after s_i , then by Lemma 1, inequality (11), and the fact that $j + 1 \leq l$ we get $\text{first}_\pi(C_l^i) \in \text{next}_\pi(s_i)$. So, $C_l^i \cup C_{l+1}^i \subseteq \text{next}_\pi(s_i)$. Since, $|C_l^i \cup C_{l+1}^i| = N$, (8) implies that at most k vertices $v \in C_j^i \setminus C$ belong to $\text{next}_\pi(s_i)$. By (12) we have $|C_j^i \setminus C| \geq N/2 - k$, which implies that during clearing at least $N/2 - 2k$ vertices $v \in C_j^i$ the vertex s_i is contaminated, which implies that $v \in \text{prev}_\pi(s_i)$. This gives (10). \square

Lemma 3 $\text{mot}(G_N) > N + k$.

Proof: Assume (8) for a contradiction. Suppose that for $v \notin V_i$ we have

$$\pi^{-1}(v) < \pi(\text{first}_\pi(V_i)) \text{ for some } i \in \{1, 2\}, \quad (13)$$

and v is chosen in such a way that the length of a shortest path P connecting s_i and v is minimum. Note that if $v = s_i$ then this path is of length 0. Such a vertex v must exist, because if $\text{first}_\pi(V(G)) \in V_1$ then $i = 2$, and $i = 1$ otherwise. By Lemma 1, $\text{first}_\pi(V_i) = \text{first}_\pi(C_1^i)$. Each shortest path connecting v and $\text{first}_\pi(C_1^i)$ is unguarded during clearing $\text{first}_\pi(C_1^i)$, because of the choice of v . Since, by Lemma 2(10) and by (5), $|(C_1^i \cup C_2^i \cup C_3^i) \cap \text{prev}_\pi(s_i)| \geq 3(N/2 - 2k) > N + k$ we have that there can be no unguarded path between v and $\text{first}_\pi(C_3^i)$ when $\text{first}_\pi(C_4^i)$ is cleared. In order to guard each such path we must clear all vertices of some separator S disconnecting v and $\text{first}_\pi(C_3^i)$ in G_N . So we have that for each $u \in S$, $\pi^{-1}(\text{first}_\pi(C_1^i)) < \pi^{-1}(u) < \pi^{-1}(\text{first}_\pi(C_4^i))$, because all the vertices in S are contaminated when $\text{first}_\pi(C_1^i)$ is cleared which follows from the minimality of the length of P . By Lemma 2(9), $|S| \leq 3k$, because during clearing each vertex in S at least one of the vertices $\text{first}_\pi(C_1^i), \text{first}_\pi(C_2^i), \text{first}_\pi(C_3^i)$ must be guarded. So, by (5) and the definition of G_N , $S = \{s_1\}$ or $S = \{s_2\}$. If $S \neq \{s_i\}$ then we repeat the same argument for the vertex u in S and for the corresponding separator $\{s_i\}$. There are two separators S of cardinality at most $3k$ in G_N which means that s_i is cleared before $\text{first}_\pi(C_7^i)$ is cleared – a contradiction with Lemma 2(10). Thus, the vertex v in (13) cannot exist. \square

The proof of Lemma 4 contains a description of a search strategy Π' satisfying $\text{ot}(\Pi') \leq N + 1$. The program avoids the difficulty occurring in the case of monotone search strategies in the following way. After clearing V_1 and $\{s_1\} \cup X$, the program starts clearing V_2 . However, each vertex in $\{s_1\} \cup X$

is guarded for at most $N/6$ turns during clearing V_2 . We accomplish this by recontaminating the cliques in X : when the vertices in X_j have been guarded for $N/6$ steps during clearing a subset of V_2 (to protect the clear vertices in $V_1 \cup \{s_1\} \cup X_1 \cup \dots \cup X_j$) then we allow a contamination of X_j and we guard X_{j-1} during the next $N/6$ steps of the search. The chain X is defined to be long enough. At the end we clear the vertices in X again.

Lemma 4 $\text{ot}(G_N) \leq N + 1$.

Proof: In the following we define a sequence π' , containing the vertices of G and the symbols \bar{v} for $v \in V(G)$, describing a search program Π' . We use a notation that a set $S \subseteq V(G)$ (respectively $\bar{S} \subseteq V(G)$) stands in a sequence for all the vertices in S (\bar{v} , where $v \in S$, respectively) ordered arbitrarily. First we clear the vertices in $V_1 \cup \{s_1\} \cup X$ according to the ordering:

$$C_1^1, C_2^1, \dots, C_{2l}^1, s_1, X_1, X_2, \dots, X_{3l-1}.$$

Note that π' is monotone until this point. For $\pi_1 = C_1^2, \dots, C_l^2$ we define $S_i(\pi_1) = \{\pi_1((i-1)N/6+1), \dots, \pi_1(iN/6)\}$, $i = 1, \dots, 3l$, to be the i th continuous block of size $N/6$ in π_1 . Observe that

$$\bigcup_{i=1, \dots, 3l} S_i(\pi_1) = \bigcup_{i=1, \dots, l} C_i^2.$$

Then, in π' we clear the vertices in V_2 in the following way

$$S_1(\pi_1), \overline{X_{3l-1}}, S_2(\pi_1), \overline{X_{3l-2}}, \dots, S_{3l-1}(\pi_1), \\ \overline{X_1}, S_{3l}(\pi_1), s_2, C_{l+1}^2, \dots, C_{2l}^2.$$

Finally, we have to clear the recontaminated vertices in X :

$$X_1, X_{3l-1}, X_2, X_{3l-2}, \dots, X_{3l/2-1}, X_{3l/2+1}, X_{3l/2}.$$

One can prove that all the vertices of G_N are clear when we apply the above search program.

Now we prove that $\text{ot}(\Pi') \leq N + 1$ which, by (3), is equivalent to proving that $|\text{next}_{\pi'}(v)| \leq N + 1$ for each $v \in V(G)$. If $v \in C_j^i$, $i = 1, 2$, then $\text{next}_{\pi'}(v) \subseteq C_j^i \cup C_{j+1}^i \cup \{s_i\}$ for $j < 2l$, and $\text{next}_{\pi'}(v) \subseteq C_j^i$ for $j = 2l$. So, $|\text{next}_{\pi'}(v)| \leq N + 1$ for $v \in V_1 \cup V_2$. For s_1 we have $\text{next}_{\pi'}(s_1) = \{s_1, s_2\} \cup X_1 \cup X_1 \cup S_{3l}(\pi_1)$. This is in particular a multiset in which every vertex in X_1 appears twice. So, $|\text{next}_{\pi'}(s_1)| \leq N/2 + 2$. If $v \in X_i$ then $\text{next}_{\pi'}(v) \subseteq X_{i+1} \cup X_{i+1} \cup X_i \cup X_i \cup X_{3l-i} \cup S_{3l-i}(\pi_1)$ for $i \leq (3l-2)/2$, and $\text{next}_{\pi'}(v) \subseteq X_{i-1} \cup X_{i+1} \cup X_i \cup X_i \cup X_{3l-i+1} \cup S_{3l-i}(\pi_1)$ for $i > (3l-2)/2$. In both cases $|\text{next}_{\pi'}(v)| \leq N$. Finally, $\text{next}_{\pi'}(s_2) = X_1 \cup X_{3l-1} \cup C_{l+1}^2 \cup \{s_2\}$, which implies that $|\text{next}_{\pi'}(s_2)| = 5N/6 + 1$. \square

Lemmas 3 and 4 imply the following.

Theorem 1 *For each $k > 0$ there exists a graph G for which $\text{mot}(G) - \text{ot}(G) \geq k$.* \square

4 Conclusions

The problem of computing $\text{mot}(G)$ is NP-hard for cobipartite graphs G [4]. A direct consequence of Theorem 1 is that we cannot conclude that the problem of finding maximum vertex occupation time is in NP in general. An interesting question is for which classes of graph the equality $\text{ot}(G) = \text{mot}(G)$ holds? By the construction of the graphs G_N we get that the monotonicity property does not hold for chordal graphs of bounded diameter.

References

- [1] D. Bienstock and P. Seymour. Monotonicity in graph searching. *J. Algorithms*, 12(2):239–245, 1991.
- [2] N.D. Dendris, L.M. Kirousis, and D.M. Thilikos. Fugitive-search games on graphs and related parameters. *Theor. Comp. Sci.*, 172(1):233–254, 1997.
- [3] F.V. Fomin and P.A. Golovach. Graph searching and interval completion. *SIAM J. Discrete Math.*, 13(4):454–464, 2000.
- [4] F.V. Fomin, P. Heggernes, and J.A. Telle. Graph searching, elimination trees, and a generalization of bandwidth. *Algorithmica*, 41(2):73–87, 2004.
- [5] L.M. Kirousis and C.H. Papadimitriou. Interval graphs and searching. *Discrete App. Math.*, 55:181–184, 1985.
- [6] L.M. Kirousis and C.H. Papadimitriou. Searching and pebbling. *Theor. Comput. Sci.*, 47(2):205–218, 1986.
- [7] A.S. LaPaugh. Recontamination does not help to search a graph. *J. ACM*, 40(2):224–245, 1993.
- [8] F. Mazoit and N. Nisse. Monotonicity of non-deterministic graph searching. *Theoretical Comp. Sci.*, 399:169–178, 2008.
- [9] T.D. Parsons. Pursuit-evasion in a graph. In *Theory and Applications of Graphs, Lecture Notes in Mathematics*, volume 642, pages 426–441. Springer-Verlag, 1978.
- [10] N.N. Petrov. A problem of pursuit in the absence of information on the pursued. *Differentsial'nye Uravneniya*, 18:1345–1352, 1982.
- [11] D. Rautenbach. Lower bounds on treespan. *Inf. Process. Lett.*, 96(2):67–70, 2005.
- [12] A.L. Rosenberg and I.H. Sudborough. Bandwidth and pebbling. *Computing*, 31(2):115–139, 1983.
- [13] P.D. Seymour and R. Thomas. Graph searching and a min-max theorem for tree-width. *J. Comb. Theory Ser. B*, 58(1):22–33, 1993.

- [14] Y.C. Stamatiou and D.M. Thilikos. Monotonicity and inert fugitive search games. In *6th Twente Workshop on Graphs and Combinatorial Optimization, Electronic Notes in Disc. Math.*, volume 3, page 184, 1999.