Exclusive Graph Searching vs. Pathwidth

Euripides Markou, Nicolas Nisse, Stéphane Pérennes

Exclusive Graph Searching vs. Pathwidth

Euripides Markou, Nicolas Nisse, Stéphane Pérennes

RESEARCH REPORT
N° 8523
April 2014
Project-Teams COATI

ISSN 0249-6399 ISRN INRIA/RR--8523--FR+ENG

Exclusive Graph Searching vs. Pathwidth*

Euripides Markou[†], Nicolas Nisse[‡], Stéphane Pérennes[§]

Project-Teams COATI

Research Report n° 8523 — April 2014 — 26 pages

Abstract: In Graph Searching, a team of *searchers* aims at capturing an invisible *fugitive* moving arbitrarily fast in a graph. Equivalently, the searchers try to clear a *contaminated* network. The problem is to compute the minimum number of searchers required to accomplish this task. Several variants of Graph Searching have been studied mainly because of their close relationship with the *pathwidth* of a graph.
Blin *et al.* defined the *Exclusive Graph Searching* where searchers cannot *"jump"* and no node can be occupied by more than one searcher. In this paper, we study the complexity of this new variant. We show that the problem is NP-hard in planar graphs with maximum degree 3 and it can be solved in linear-time in the class of cographs. We also show that *monotone* Exclusive Graph Searching is NP-complete in split graphs where Pathwidth is known to be solvable in polynomial time. Moreover, we prove that monotone Exclusive Graph Searching is in P in a subclass of star-like graphs where Pathwidth is known to be NP-hard.
Hence, the computational complexities of monotone Exclusive Graph Searching and Pathwidth cannot be compared. This is the first variant of Graph Searching for which such a difference is proved.

Key-words: graph searching; pathwidth; computational complexity; monotone strategies; exclusivity property

*

† University of Thessaly, Lamia, Greece
‡ Inria, & Univ. Nice Sophia Antipolis, CNRS, I3S, Sophia Antipolis, France
§ Univ. Nice Sophia Antipolis, CNRS, I3S, Sophia Antipolis, France

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Exclusive Graph Searching vs. Pathwidth

Résumé : Dans les jeux de capture (Graph Searching), une équipe d'agents doit capturer un fugitif invisible se déplaçant rapidement dans un graphe. De façon équivalente, les agents doivent nettoyer un réseau contaminé. Le problème est de calculer le nombre minimum d'agents nécessaires pour accomplir cette tache. Plusieurs variantes de ces jeux ont été étudiés pour leur lien avec la pathwidth des graphes.

Blin *et al.* ont défini le jeu de capture exclusif dans lequel les agents ne peuvent ni "sauter", ni occuper un sommet à plusieurs. Dans ce rapport, nous étudions la complexité de cette nouvelle variante. Nous montrons que le problème est NP-difficile dans les graphes planaires avec degré au plus 3 et qu'il peut être résolu en temps linéaire dans la classe des cographes. Nous montrons également que la variante monotone du jeu exclusif est NP-complet dans la classe des split-graphes où la pathwidth est NP-complet. De plus, nous prouvons que le jeu exclusif monotone peut être calculé en temps polynomial dans une classe de star-like graphes où calculer la pathwidth est NP-complet.

Les complexités de la pathwidth et du jeu de capture exclusif monotone ne peuvent donc être comparées. Il s'agit de la première variante de jeu de capture où une telle différence est avérée.

Mots-clés : jeux de capture exclusif, pathwidth, complexité

1 Introduction

In *Graph Searching* [Bre67, Par78], a team of *searchers* aims at clearing a *contaminated* network. Many variants have been studied that differ with respect to the moves allowed to the searchers, the ways of clearing the graph and the constraints imposed to the *search strategies* (see the survey [FT08]). In each variant, the main problem consists of computing the minimum number of searchers, called *search number* of G, required to clear the graph G.

One of the main motivations for studying Graph Searching arises from the fact that it provides an algorithmic interpretation of *path-decompositions* of graphs [RS83, KP86]. For instance, the *node-search number* of any graph equals its *pathwidth*, plus one [KP86, BS91] and any other "classical" variant differs from pathwidth up to a constant ratio (see Related Work). Since computing the pathwidth is NP-hard in many graph classes (e.g., [Gus93]), a polynomial-time algorithm for computing some "classical" variant of search number in one of these classes would provide a polynomial-time approximation algorithm for pathwidth. To the best of our knowledge, no graph class is known where the complexities of pathwidth and some "classical" variant of Graph Searching are different.

Recently, Blin *et al.* introduced a new variant, namely *Exclusive Graph Searching*, that appears to be very different from the previous ones [BBN12, BBN13]. In this paper, we study the computational complexity of this new variant. In particular, we prove that the computational complexities of monotone Exclusive Graph Searching and Pathwidth cannot be compared.

Exclusive Graph Searching. An *exclusive search strategy* [BBN12, BBN13] consists of first placing k searchers at distinct nodes of a graph $G = (V, E)$. Then, at each step, a searcher at some node $v \in V$ can slide along an edge $\{v, u\} \in E$ only if node u is not yet occupied by another searcher. By definition, any exclusive search strategy satisfies the *exclusivity* property: at any step, any node is occupied by at most one searcher. Initially, all edges of G are contaminated and an edge $e \in E$ is cleared if either a searcher slides along it or if both endpoints of e are occupied simultaneously. An edge e is *recontaminated* if there is a path, free of searchers, from e to another contaminated edge. A strategy is *winning* if eventually all edges of G become clear. As an example, a winning exclusive strategy in a n-node star (a tree with $n-1$ leaves) consists of: 1) first placing searchers at $n-2$ distinct leaves (i.e., all but one leaf, say v), and then 2) sliding a searcher from a leaf to the center of the star and then along the last contaminated edge (to v). It is easy to see that there are no winning strategies using $\leq n-3$ searchers in an n-node star.

The *exclusive search-number* of G, $xs(G)$, is the minimum number k such that there is a winning strategy using k searchers to clear G. A strategy is *monotone* if no edge is ever recontaminated. The *monotone-exclusive search-number* of G, $mxs(G)$, is the smallest k such that there is a winning monotone strategy using k searchers to clear G. By definition, $xs(G) \leq mxs(G)$ for any graph G. Note that this inequality may be strict [BBN13]. If $mxs(G) = xs(G)$ for any graph G in some class $\mathcal{C}$ of graph, Exclusive Graph Searching is said *monotone* in $\mathcal{C}$.

In [BBN13], the question of the complexity of computing xs in arbitrary graphs was left open, as well as the question of whether there exists a graph class in which computing the exclusive search-number could provide a polynomial-time approximation of pathwidth. In this paper, we answer the first question and further investigate the second one.

Our results[1]**.** We first show that computing the exclusive search-number is NP-hard in the class of *planar graphs with maximum degree* 3 (Sec. 2).

Then, we focus on *star-like* graphs (Section 3). We first show that Exclusive Graph Searching is not monotone in star-like graphs. Then, we show that the computational complexities of

[1] We postpone the formal definitions of the graph classes mentioned here to the corresponding sections.

monotone Exclusive Graph Searching and pathwidth differ in star-like graphs. More precisely, in Section 3.1, we show that monotone Exclusive Graph Searching can be computed in polynomial time in a subclass of star-like graphs where pathwidth is known to be NP-complete [Gus93], and in Section 3.2 we show that computing the monotone exclusive search-number is NP-complete in *split graphs* (where the pathwidth can be solved in polynomial-time [Gus93]). This is the first variant of Graph Searching where such a difference arises.

Finally, we show that Exclusive Graph Searching is monotone and can be computed in linear-time in the class of *cographs* (Section 4), where pathwidth can also be computed in polynomial time.

Our results are summarized in Table 1. We leave as open problems the question of whether Exclusive Graph Searching is monotone in split graphs and the question of if there are graph classes where Exclusive Graph Searching provides a polynomial-time approximation of pathwidth.

Related Work. *Graph Searching* has been introduced by Breish for modeling the rescue of a lost speleologist by a team of searchers in a network of caves [Bre67]. Later on, Parsons formalized Graph Searching as a game to clear contaminated networks [Par78]. Formally, in *edge Graph Searching*, the searchers can be placed at nodes of a graph, removed from nodes or may slide along edges. Any edge of the graph is cleared when a searcher slides along it. Kirousis and Papadimitriou defined *node Graph Searching* in which searchers can only be placed at and removed from nodes, and edges are cleared only when both their endpoints are simultaneously occupied [KP86]. This variant has been introduced because the corresponding monotone search-number of a graph is equal to its pathwidth plus one [KP86]. Then, Bienstock and Seymour defined the *mixed Graph Searching* in order to prove the monotonicity of previous variants [BS91]. In mixed Graph Searching, the allowed moves are similar as in edge Graph Searching but an edge e is cleared when a searcher slides along it or when both endpoints of e are simultaneously occupied. The search numbers corresponding to the three above mentioned variants are known as *edge-*, *node-* and *mixed-search numbers*, denoted by es, ns and s respectively.

These three variants are very close one from each other (note that both edge and node strategies are mixed strategies). In particular, for any graph G, $ns(G)-1 \leq es(G) \leq ns(G)+1$ and $s(G) \leq ns(G) \leq s(G)+1$ [KP86, BS91](all inequalities are tight). For all these variants, there are simple graph transformations allowing to compute one of these parameters from another one [KP86, BS91]. For instance, $s(G^+) = es(G)$ for any graph G where G^+ is obtained from G by subdividing[2] each edge once [KP86].

An important property of Graph Searching is the *monotonicity*. Each of the node-, edge- and mixed Graph Searching variants is monotone. That is, for any graph G, there is a monotone mixed (resp., node, resp., edge) strategy that clears G using $s(G)$ (resp., $ns(G)$, resp., $es(G)$) searchers [BS91]. The monotonicity property is very important, in particular because it is the corner stone of the link between the node search number of a graph and its pathwidth. More precisely, for any graph G, $ns(G) = pw(G) + 1$ [Bie91].

The problem of computing the edge search number has been shown to be NP-complete in the class of planar graphs with maximum degree 3 [MS88]. As mentioned above $s(G^+) = es(G)$ in any graph G [KP86] and this reduction from edge search to mixed search preserves planarity and maximum degree. Moreover, in the resulting graph G^+, the set of vertices with degree at least three induces an independent set. Altogether, it gives:

Theorem 1 *[MS88, KP86] The problem of computing the mixed search number is NP-complete in the class of planar graphs with degree* ≤ 3 *where the set of vertices with degree exactly* 3 *induces an independent set.*

[2] The subdivision of an edge $e = uv$ consists in adding a new node x and replacing e by two edges ux and xv.

	pathwidth pw (node-search ns)	edge-search es	mixed-search s	exclusive-search **[this paper]**
planar graphs with bounded maximum degree	NP-complete [MS88, KP86]			**NP-hard** (xs) (Section 2)
split graphs	P [Gus93]	P [PTK$^+$00]	linear [FHM10]	**NP-complete** (mxs) (Sec. 3.2)
star-like graphs with ≥ 2 peripheral nodes per peripheral clique	NP-complete [Gus93]	?	?	**P** (mxs) (Section 3.1)
cographs	P [BM93]	linear [GHM12]	P [HM08]	**linear** (Sec. 4) $xs = mxs$

Table 1: Summary of the complexity results.

The pathwidth problem and the variants of Graph Searching have been studied in many particular graph classes. To the best of our knowledge, no classes of graphs are known where the computational complexities of these problems are different. Pathwidth, edge-search number and mixed-search number can be computed in polynomial-time in trees [Sko03, CHM12]. All these parameters can also be computed in polynomial-time in cographs [BM93, GHM12], in split-graphs [Gus93, FHM10] or in permutation graphs [HM08]. On the other hand, pathwidth is NP-hard in star-like graphs [Gus93]. Moreover, pathwidth cannot be approximated up to an additive constant in arbitrary graphs (unless P=NP) [DKL87]. It is a long standing open problem to answer whether there is a class of graphs where the complexities of pathwidth and mixed- (or edge-) search number are different.

Blin *et al.* defined *Exclusive Graph Searching* [BBN12, BBN13] to address two somewhat unrealistic assumptions of edge- (node-, mixed-) search strategies. In previous variants, searchers are enable to “jump" from one node of the graph to another, potentially far away, node. Second, several searchers may occupy simultaneously the same node. Therefore, in Exclusive Graph Searching, searchers are only allowed to slide along edges and must satisfy the exclusivity constraint. Notice that any exclusive strategy is a mixed one (hence $s(G) \leq xs(G)$ for any graph G). However, the results in [BBN12, BBN13] show that Exclusive Graph Searching seems to behave differently from the previous variants. Indeed, in a graph G, $xs(G)$ may differ exponentially from the pathwidth $pw(G)$ of G. For instance, $pw(T) = O(\log n)$ for any n-node tree T [MHG$^+$88], while $xs(S) = mxs(S) = n - 2$ for any n-node star S. The main result in [BBN13] is that $xs(T)$ can be computed in polynomial-time in any tree T. Finally, it is shown that $pw(G) \leq xs(G) \leq mxs(G) \leq (\Delta - 1)(pw(G) + 1)$ in any graph G with maximum degree Δ [BBN13]. It is also shown that Exclusive Graph Searching is not *monotone* in the class of trees, i.e., there are trees T such that $xs(T) < mxs(T)$ [BBN13]. Exclusive Graph Searching has also been studied in a distributed setting in [DDSN$^+$13, DNN14].

2 NP-hardness of Exclusive Search Number in planar graphs with maximum degree 3

In this section, we prove that the problem of computing the exclusive search number is NP-hard in planar graphs with maximum degree 3. For our purpose, we reduce to our problem the problem of computing the mixed search number of planar graphs with maximum degree 3 where no two nodes with degree 3 are adjacent (this problem is NP-hard by Theorem 1).

The construction part of the reduction consists into replacing any node of degree three by a triangle. Exclusive search differs from mixed search because searchers can only slide and therefore, because of the exclusivity property, the searchers have to avoid to meet other searchers at the same node. Intuitively, the triangles allow the searchers to bypass each other.

Let $G = (V, E)$ be any planar graph with maximum degree 3 and such that the nodes with degree exactly 3 induce an independent set. Let $T \subseteq V$ be the set of nodes with degree exactly 3. T is an independent set and all nodes in $V \setminus T$ have degree at most two. The planarity of G will not be used below, but it is well preserved by our reduction.

We construct $G^{\triangle}$ from G as follows. For any node $v \in T$ with neighbors a, b, c (note that $\{a, b, c, \} \subseteq V \setminus T$), we replace v by a triangle with nodes v_a, v_b, v_c and we add edges av_a, bv_b and cv_c.

Clearly, $G^{\triangle}$ can be constructed from G in polynomial time (with respect to G's size) and it is planar and has maximum degree 3. We prove that any monotone mixed-strategy for G can be transformed into an exclusive strategy for $G^{\triangle}$ without increasing the number of searchers. This part of the proof is quite technical because the obtained exclusive strategy is not monotone and we need to control the recontamination. Conversely, from any exclusive strategy for $G^{\triangle}$, it is easy to define a mixed-strategy for G using the same number of searchers. Therefore:

Theorem 2 *For any planar graph G with maximum degree 3 and no two adjacent nodes with degree exactly 3, $s(G) = xs(G^{\triangle})$.*

From Theorems 1 and 2, we get:

Corollary 1 *The problem of computing the exclusive search number is NP-hard in the class of planar graphs with maximum degree 3.*

3 Exclusive Graph Searching in star-like graphs

In this section, we study the complexity of computing the exclusive search number in star-like graphs. Surprisingly, our results are somehow “orthogonal" to the ones concerning pathwidth in this class of graphs.

3.1 Star-like graphs: When Exclusive Graph Searching is easier than Pathwidth

A connected graph $G = (V, E)$ is a *star-like graph* if V can be covered by cliques $C_0, C_1, \cdots, C_r$ such that, for any $i, j \leq r$ with $i \neq j$, $C_i \cap C_j \subseteq C_0$. Said differently, a graph is a star-like graph if it is chordal and its clique-tree is a star. A graph is *k-star-like* if $c_i = |C_i \setminus C_0| \leq k$ for any $1 \leq i \leq r$. C_0 is called the central clique and any node in $V \setminus C_0$ is called *peripheral*.

We start with a simple remark:

Lemma 1 *Exclusive Graph Searching is not monotone in star-like graphs.*

Proof. Let G be the star-like graph formed by two peripheral cliques which are triangles and one edge (the central clique) connecting them. It is easy to see that $xs(G) = 2 < mxs(G) = 3$.
■ ■

In [Gus93], Gustedt proved that computing the pathwidth of star-like graphs is NP-hard. A simple look at his reduction shows that he actually proved that computing the pathwidth is

NP-hard in the class of star-like graphs where the peripheral cliques have at least 2 peripheral nodes, i.e., $|C_i \setminus C_0| \geq 2$ for any $0 < i \leq r$. We prove that the monotone exclusive search number can be computed in polynomial-time in this class of graphs.

Theorem 3 *Let G be a star-like graph with cliques $(C_0, \cdots, C_r)$ such that $|C_i \setminus C_0| > 1$ for any $0 < i \leq r$, that is each non central clique has at least two peripheral nodes.*

1. *Either there is an edge $\{u,v\} \in E(C_0)$ that does not belong to any peripheral clique, and $mxs(G) = |V(G)| - r - 1$,*
2. *or $mxs(G) = |V(G)| - r$.*

Corollary 2 *The monotone exclusive search number can be computed in polynomial time in the class of star-like graphs where each peripheral clique has at least 2 peripheral nodes.*

3.2 Split graphs: When Exclusive Graph Searching is harder than Pathwidth

We now focus on 1-star-like graphs, also called *split graphs*. In other words, a connected graph $G = (V, E)$ is a split graph if V can be partitioned into C and I where C induces a clique and I is an independent set.

In [Gus93], Gustedt proved that Pathwidth can be computed in polynomial-time in the class of k-star-like graphs, for any fixed k. Hence, the pathwidth of split graphs is polynomially computable. In this section, we prove that:

Theorem 4 *Monotone Exclusive Graph Searching is NP-complete in split graphs.*

To prove the above theorem, we first show that we can restrict our attention to monotone exclusive search strategies with particular structure. More precisely, we prove that, for any split graph G and for any $k \geq mxs(G)$, there is a monotone exclusive search strategy clearing G and using at most k searchers that proceeds as we describe below. Such a strategy is called *simple*.

1. Initially, the searchers are placed at selected distinct nodes.
2. Then, some searchers occupying peripheral nodes of G, sequentially slide to the central clique C until all nodes of C (possibly except one) are occupied.
3. If a node v of C is unoccupied, then a searcher slides along an edge of C toward v.
4. Finally, some searchers occupying the central clique sequentially slide to the remaining contaminated peripheral nodes.

In other words, we prove:

Lemma 2 *For any split graph G and any $k \geq mxs(G)$, there is a simple strategy that clears G using at most k searchers.*

To prove Theorem 4, we define a new problem, called *Maximum Augmenting Cover (MAC)*, related to Set-Cover problem. We prove that MAC is NP-hard and then reduce it to monotone Exclusive Graph Searching in split graphs. Let us briefly discuss the MAC problem and how it is related to simple strategies.

Intuitively, in order to minimize the number of searchers capable of clearing a split graph, we need to maximize the number of searchers moved during Phases 2 and 4 of a simple strategy. Let us consider Phase 2 of a simple strategy. It consists of some s steps, in each of which a searcher slides from some peripheral node $u_i \in I$ to some node $v_i \in C$ in the central clique. Let $(u_1, v_1), (u_2, v_2), \ldots, (u_s, v_s)$ be the sequence of slidings. Notice that, since the strategy is exclusive (at most one searcher per node), the nodes in $\{u_i, v_i \mid i \leq s\}$ are pairwise distinct. Moreover, since the strategy is monotone, it is not possible to have $v_j \in N(u_i)$ for $j > i$ (where $N(u_i)$ is the set of the nodes which are adjacent to u_i). Indeed, otherwise, u_i would be recontaminated by v_j when the searcher slides along (u_i, v_i). Altogether, Phase 2 somehow defines a sequence of subsets $(N(u_i))_{i \leq s}$ such that $N(u_i) \setminus \bigcup_{j<i} N(u_j) \neq \emptyset$ for any $i > 1$. Moreover, it is desirable to have such a sequence as long as possible. This led us to define the following problem, which, we think, is interesting by itself.

Let $\mathcal{S} = (S_1, \cdots, S_r)$ be a sequence of subsets of some ground set A. For any $1 \leq i \leq r$, let $s_i = |\bigcup_{1 \leq j \leq i} S_j|$. We say that $\mathcal{S}$ is *augmenting* if the sequence $(s_i)_{1 \leq i \leq r}$ is strictly increasing.

Problem 5 Maximum Augmenting Cover (MAC).

Input: *A family* $\mathcal{S} = \{S_1, \cdots, S_r\}$ *of subsets of a set* A *and a* $k \in \mathbb{N}$.

Question: *Does there exist an augmenting sequence of length* $\geq k$ *in* $\mathcal{S}$*?*

We prove that MAC is NP-hard by showing a reduction from MIN-SAT. An instance of MIN-SAT in the boolean variables $\{v_1, \cdots, v_n\}$ is composed of a collection of clauses $\{C_1, \cdots, C_m\}$ in Conjunctive Normal Form. The goal is to decide what is the minimum number of all satisfied clauses by a truth assignment to the boolean variables. MIN-SAT is known to be NP-hard [AZ05].

Theorem 6 *MAC is NP-hard.*

We are now ready to prove Theorem 4.

Proof of Theorem 4. By monotonicity, the problem is clearly in NP. Let us prove it is NP-hard. Let $(A = \{a_1, \cdots, a_n\}, \mathcal{S} = \{S_1, \cdots, S_m\})$ be an instance of MAC. We build a split graph G as follows. Start with a clique K with vertex-set $V = \{v_1, \cdots, v_n\}$ plus two independent sets $S = \{s_1, \cdots, s_m\}$ and $U = \{u_1, \cdots, u_m\}$. For any $i \leq n, j \leq m$ such that $a_i \in S_j$, add edges $\{v_i, s_j\}$ and $\{v_i, u_j\}$.

Claim 1 *If* $(A, \mathcal{S})$ *admits an augmenting sequence of length* k*, then* $mxs(G) \leq n + 2m - 2k$.

Let $(S_1, \cdots, S_k)$ be an augmenting sequence. For any $1 \leq j \leq k$, let $a_j \in S_j \setminus \bigcup_{\ell<j} S_\ell$ and let v_j be the corresponding node of V (in particular, v_j is not adjacent to any node in $\{s_1, \cdots, s_{j-1}\}$). We consider the following strategy. Initially, place a searcher at any node in $V(G) \setminus \{s_1, \cdots, s_k, v_1, \cdots, v_k\}$. For i from 1 to k, the searcher at u_i slides to v_i. Finally, for j decreasing from k to 1, the searcher at v_j slides to s_j. This is a monotone exclusive search strategy using $n + 2m - 2k$ searchers to clear G.

Claim 2 *If* $mxs(G) \leq n + 2m - 2k$*, then* $(A, \mathcal{S})$ *admits an augmenting sequence of length* k.

Let k be the maximum integer such that $mxs(G) \leq n + 2m - 2k$. Then, $mxs(G) \in \{n + 2m - 2k, n + 2m - 2k - 1\}$.

By our characterization of monotone exclusive strategies for split graphs, there are two disjoint sets $X, Y \subseteq S \cup U$ and, possibly, an edge $e \in E(K)$ such that there is a strategy using $mxs(G)$ searchers that proceeds as follows: Initially all nodes apart from $|X|$ or $|X| + 1$ nodes of K and all nodes of Y are occupied. Sequentially, the searchers at the nodes of X slide to $|X|$ unoccupied nodes of K; then a searcher slides along e (if e exists); finally $|Y|$ searchers occupying the nodes of K sequentially slide from their positions to occupy the nodes in Y.

Hence, $mxs(G) \in \{n + 2m - |X| - |Y|, n + 2m - |X| - |Y| - 1\}$. W.l.o.g., let us assume that $|X| \geq |Y|$. We get that $|X| \geq k$. It remains to prove that we may assume that $X \subseteq S$. If $u_i \in X$, then we prove that $s_i \notin X$. Then, we can modify the strategy by setting $X \leftarrow X \cup \{s_i\} \setminus \{u_i\}$. Moreover, if $s_i \in Y$, we set $Y \leftarrow Y \cup \{u_i\} \setminus \{s_i\}$.

The property of X (from our characterization and the monotonicity of the strategy), implies that X corresponds to an augmenting sequence of length k for $(A, \mathcal{S})$. ■

Conjecture 7 *Exclusive Graph Searching is monotone in split graphs.*

4 Exclusive Graph Searching in Cographs

In this section, we study the complexity of computing the exclusive search number in the class of cographs.

A graph is a *cograph* if and only if it is P_4-free, that is, if it does not contain a P_4 (path with 4 nodes) as an induced subgraph. A graph is *trivial* if it reduces to a single node. In a graph, any connected component consisting of a single node is also called *trivial*. It is well known that a graph $G = (V, E)$ is a cograph if and only if:

- G is trivial, or
- there are two non empty cographs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ such that $G = G_1 \cup G_2$ is the disjoint union of G_1 and G_2, i.e., $V = V_1 \cup V_2$ and $E = E_1 \cup E_2$, or
- there are two non empty cographs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ such that $G = G_1 \otimes G_2$ is the "product" of G_1 and G_2, i.e., $V = V_1 \cup V_2$ and $E = E_1 \cup E_2 \cup \{uv \mid u \in V_1, v \in V_2\}$.

Moreover, such a decomposition can be computed in linear time [CPS85].

Notice that by the definition of Exclusive Graph Searching, the (monotone) exclusive search number of a graph equals the sum of the (monotone) exclusive search numbers of its connected components. Therefore, to obtain a linear time algorithm for computing the exclusive search number of a cograph, it is sufficient to compute $xs(G_1 \otimes G_2)$ from $xs(G_1)$ and $xs(G_2)$ in linear time. For this purpose, for any graph G, we define G' as follows: (1) if G is connected or has no trivial component, then $G' = G$, otherwise (2) if G is not connected and has a unique trivial component $\{v\}$, then $G' = G \setminus v$, otherwise (3) if G is not connected and has at least two trivial components $\{v\}$ and $\{w\}$, then $G' = G \setminus \{v, w\}$. We prove:

Lemma 3 *Let $G = G_1 \otimes G_2$ with G_1 and G_2 two cographs.*

$$mxs(G) = xs(G) = \min\{xs(G'_1) + |V(G_2)|, xs(G'_2) + |V(G_1)|\}$$

Since $xs(G'_i)$ can easily be deduced from $xs(G_i)$, Theorem 8 simply follows by a dynamic programming algorithm.

Theorem 8 *Exclusive Graph Searching is monotone in cographs and the exclusive search number of cographs can be computed in linear-time.*

5 Conclusion

We have shown that there are classes of graphs where the complexities of Exclusive Graph Searching and Pathwidth are different. An interesting open question is whether there exist classes of bounded degree graphs where Exclusive Graph Searching is polynomially computable while Pathwidth is NP-hard. In such a case, computing Exclusive Graph Searching would be a way to approximate the pathwidth [BBN13]. The question of the parameterized complexity of Exclusive Graph Searching is also interesting (note it is not closed under taking minors [BBN13]). Finally, does the problem of computing the exclusive search number of an arbitrary graph belong to NP?

References

[AZ05] A. Avidor and U. Zwick. Approximating min 2-sat and min 3-sat. *Theory Comput. Syst.*, 38(3):329–345, 2005.

[BBN12] L. Blin, J. Burman, and N. Nisse. Brief announcement: Distributed exclusive and perpetual tree searching. In *26th Int. Symp. on Distributed Computing (DISC)*, volume 7611 of *LNCS*, pages 403–404. Springer, 2012.

[BBN13] L. Blin, J. Burman, and N. Nisse. Exclusive graph searching. In *21st European Symposium on Algorithms (ESA)*, volume 8125 of *LNCS*, pages 181–192. Springer, 2013.

[Bie91] D. Bienstock. Graph searching, path-width, tree-width and related problems (a survey). *DIMACS Ser. in Discrete Mathematics and Theoretical Computer Science*, 5:33–49, 1991.

[BM93] H.L. Bodlaender and R.H. Möhring. The pathwidth and treewidth of cographs. *SIAM J. Discrete Math.*, 6(2):181–188, 1993.

[Bre67] Richard L. Breish. An intuitive approach to speleotopology. *Southwestern Cavers*, 6:72–78, 1967.

[BS91] D. Bienstock and P.D. Seymour. Monotonicity in graph searching. *J. Algorithms*, 12(2):239–245, 1991.

[CHM12] D. Coudert, F. Huc, and D. Mazauric. A distributed algorithm for computing the node search number in trees. *Algorithmica*, 63(1-2):158–190, 2012.

[CPS85] D. G. Corneil, Y. Perl, and L. K. Stewart. A linear recognition algorithm for cographs. *SIAM J. Comput.*, 14(4):926–934, 1985.

[DDSN+13] G. D'Angelo, G. Di Stefano, A. Navarra, N. Nisse, and K. Suchan. A unified approach for different tasks on rings in robot-based computing systems. In *15th Workshop on Advances in Parallel and Distributed Computational Models (APDCM)*, pages 667–676. IEEE, 2013.

[DKL87] N. Deo, M.S. Krishnamoorthy, and M.A. Langston. Exact and approximate solutions for the gate matrix layout problem. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 6(1):79–84, 1987.

[DNN14] G. D'Angelo, A. Navarra, and N. Nisse. Gathering and exclusive searching on rings under minimal assumptions. In *15th Int. Conf. on Dist. Comp. & Networking*, volume 8314 of *LNCS*, pages 149–164. Springer, 2014.

[FHM10] F.V. Fomin, P. Heggernes, and R. Mihai. Mixed search number and linear-width of interval and split graphs. *Networks*, 56(3):207–214, 2010.

[FT08] F.V. Fomin and D.M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theor. Comput. Sci.*, 399(3):236–245, 2008.

[GHM12] P.A. Golovach, P. Heggernes, and R. Mihai. Edge search number of cographs. *Discrete Applied Mathematics*, 160(6):734–743, 2012.

[Gus93] J. Gustedt. On the pathwidth of chordal graphs. *Discrete Applied Mathematics*, 45(3):233–248, 1993.

[HM08] P. Heggernes and R. Mihai. Mixed search number of permutation graphs. In *2nd Int. Workshop on Frontiers in Algorithmics (FAW)*, volume 5059 of *LNCS*, pages 196–207. Springer, 2008.

[KP86] L.M. Kirousis and C.H. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47(2):205–218, 1986.

[MHG+88] N. Megiddo, S.L. Hakimi, M.R. Garey, D.S. Johnson, and C.H. Papadimitriou. The complexity of searching a graph. *J. ACM*, 35(1):18–44, 1988.

[MS88] B. Monien and I. H. Sudborough. Min cut is np-complete for edge weighted trees. *Theor. Comput. Sci.*, 58:209–229, 1988.

[Par78] T.D. Parsons. Pursuit-evasion in a graph. In *Theory and applications of graphs (Proc. Internat. Conf., Western Mich. Univ., Kalamazoo, Mich., 1976)*, pages 426–441. Lecture Notes in Math., Vol. 642. Springer, 1978.

[PTK+00] S.-L. Peng, C. Y. Tang, M.-T. Ko, C.-W. Ho, and T.-S. Hsu. Graph searching on some subclasses of chordal graphs. *Algorithmica*, 27(3):395–426, 2000.

[RS83] N. Robertson and P.D. Seymour. Graph minors. i. excluding a forest. *J. Comb. Theory, Ser. B*, 35(1):39–61, 1983.

[Sko03] K. Skodinis. Construction of linear tree-layouts which are optimal with respect to vertex separation in linear time. *J. Alg.*, 47(1):40–59, 2003.

Appendix

1 NP-hardness of Exclusive Search Number in planar graphs with maximum degree 3

Let $\mathcal{C}$ be the set of connected planar graphs with maximum degree 3 and no edges between nodes of degree 3. Let $G = (V, E) \in \mathcal{C}$. Let $T \subseteq V$ be the set of nodes with degree exactly 3. T is an independent set and all nodes in $V \setminus T$ have degree at most two. The planarity of G will not be used below, but it is well preserved by our reduction.

We construct $G^{\triangle}$ from G as follows. For any node $v \in T$ with neighbors a, b, c (note that $N(v) = \{a, b, c, \} \subseteq V \setminus T$), we replace v by a triangle with nodes v_a, v_b, v_c and we add edges av_a, bv_b and cv_c (see Figure 1). Let $T_v = \{v_a, v_b, v_c\}$ and $E_v = \{v_a v_b, v_b v_c, v_c v_a\}$ be the set of edges of the triangle with vertex-set T_v. More formally,

$$V(G^{\triangle}) = (V \setminus T) \cup \bigcup_{v \in T} T_v$$

$$F_1 = \{uv \in E \mid u, v \in V \setminus T\} \cup \{av_a \mid a \notin T, v \in T, av \in E\}$$

$$F_2 = \bigcup_{v \in T} E_v$$

$$E(G^{\triangle}) = F_1 \cup F_2$$

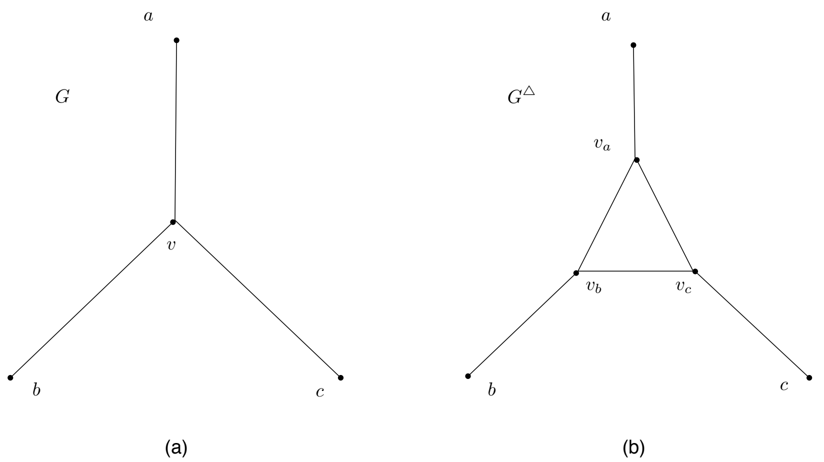

Figure 1: A node v of degree three in G (a) is transformed to a triangle T_v in $G^{\triangle}$ (b).

Note that there is a bijection between E and F_1, therefore we will identify the edges of both sets. Let $\phi_{\triangle} : E \rightarrow F_1$ be this bijection and let $\phi_{\triangle}(F) = \{\phi_{\triangle}(f) \mid f \in F\}$ for any $F \subseteq E$. Similarly, there is a bijection $\psi_{\triangle}$ between $V(G) \setminus T$ and $V(G^{\triangle}) \setminus \bigcup_{v \in T} T_v$.

Clearly, $G^{\triangle}$ can be constructed from G in polynomial time (with respect to G's size) and it is cubic and planar. We prove below that the mixed search number of G is k if and only if the exclusive search number of $G^{\triangle}$ is k.

Theorem 2 *For any $G \in \mathcal{C}$, $s(G) = xs(G^{\triangle})$.*

The proof of Theorem 2 directly follows from the following Lemmata 4 and 5.

Lemma 4 *For any $G \in \mathcal{C}$, $xs(G^\triangle) \leq s(G)$.*

Proof. Let $s(G) = k \leq |V(G)|$ and let $\mathcal{S}$ be a mixed strategy for G using k searchers. By monotonicity of mixed-search [BS91], we may assume that $\mathcal{S}$ is monotone. Moreover, as mentioned above, we may assume that any node is never occupied by more than one searcher. Therefore, it is easy to see that we may assume that $\mathcal{S}$ proceeds as follows (up to a reordering of the steps of $\mathcal{S}$): first $\mathcal{S}$ places the searchers on k distinct nodes[3], then, while it remains one contaminated node, either $\mathcal{S}$ slides a searcher from a node u to an unoccupied node v, or $\mathcal{S}$ removes a searcher from a node all neighbors of which are clear and places it on a contaminated node. We call such a sliding-step or such a pair of steps (removal-placement) a *Round* of $\mathcal{S}$.

For any $i \geq 0$, let $C_i \subseteq V(G)$ be the set of clear nodes after Round i, let $E_i \subseteq E(G)$ be the set of clear edges after Round i, let $O_i \subseteq V(G)$ be the set of occupied nodes after Round i and let $R_i \subseteq C_i$ be the set of clear nodes whose all incident edges are in E_i.

In Round 0 the k searchers are initially placed at k distinct nodes by $\mathcal{S}$. Hence after the initial placement of the searchers we have: The set $O_0 \subseteq V(G)$ consists of the k vertices where the searchers are initially placed by $\mathcal{S}$. Note that O_0 is the set of nodes that are clear after these first k placements. Hence $E_0 = \{uv \in E \mid u, v \in O_0\}$ and $R_0 = \{v \in O_0 \mid N(v) \subseteq O_0\}$ is the set of vertices in O_0 with all their neighbors in O_0. Note that the searchers in R_0 are the ones that can be removed (by a pair of removal-placement steps).

We first make some general remarks. Since $\mathcal{S}$ is monotone, we have $C_i \subseteq C_{i+1}$ and $E_i \subseteq E_{i+1}$ for any $i \geq 0$. The searchers at the nodes in $R_i \cap O_i$ are exactly the searchers that may be removed (by a pair of removal-placement steps) during the next round. Finally, $C_i = R_i \cup O_i$ since nodes in $C_i \setminus R_i$ are in the *border* of the clear part and therefore must be preserved from recontamination by a searcher.

We now build an exclusive strategy $\mathcal{S}^\triangle$ for clearing $G^\triangle$ using k searchers. $\mathcal{S}^\triangle$ is divided into phases such that Phase 0 corresponds to the initialization of $\mathcal{S}$ and each Phase i ($i \geq 1$) corresponds to Round i of $\mathcal{S}$. As above, let us define, for any $i \geq 0$, $C_i^\triangle \subseteq V(G^\triangle)$ as the set of clear nodes after Phase i, $E_i^\triangle \subseteq E(G^\triangle)$ as the set of clear edges after Phase i, let $O_i^\triangle \subseteq V(G^\triangle)$ be the set of occupied nodes after Phase i and let $R_i^\triangle \subseteq V(G^\triangle)$ be the set of clear nodes whose all incident edges are in $E_i^\triangle$. To clear $G^\triangle$, each searcher of $\mathcal{S}^\triangle$ will mimic the moves of a searcher in $\mathcal{S}$. More precisely, for any $i \geq 0$, we will ensure that, after Phase i:

1. $\psi_\triangle(R_i \setminus T) \cup \bigcup_{v \in R_i \cap T} T_v \subseteq R_i^\triangle$. That is, when all edges incident to a node in $V(G)$ are clear, the same holds for the corresponding node or triangle in $G^\triangle$.
2. $O_i^\triangle \cap \psi_\triangle(V \setminus T) = \psi_\triangle(O_i \setminus T)$, i.e., for any occupied node in $V(G) \setminus T$, the corresponding node in $G^\triangle$ is also occupied.
3. $\phi_\triangle(E_i) \subseteq E_i^\triangle \cap \phi_\triangle(F_1)$. For any clear edge in $E(G)$, the corresponding edge is clear in $G^\triangle$.
4. For any $v \in (T \cap O_i)$ with $N_G(v) = \{a, b, c\}$, we have $|O_i^\triangle \cap T_v| = 1$ and

[3]Notice that $\mathcal{S}$ can always be modified so that it initially places all k searchers at k distinct nodes: The only modification is that all searchers are initially placed at the nodes they appear for the first time. It cannot happen that two searchers are placed at the same node u, since this would mean that one of them (the one that appeared later at u in the original strategy) was placed at an already cleared node. It is easy to see that this modified strategy is also monotone and clears the graph: the modified strategy may clears sooner than the original strategy some nodes and/or edges and those nodes and edges cannot be recontaminated because of the monotonicity of the original strategy.

(a) if no edges incident to v are clear or $v \in R_i$ (i.e., all edges incident to v are clear), then the searcher in $\mathcal{S}^{\triangle}$ is at some arbitrary node in T_v. Moreover, in the latter case, $x \in R_i^{\triangle}$ for any $x \in T_v$ (in particular $E_v \subseteq E_i^{\triangle}$);

(b) if exactly one edge, say $av \in E(G)$, incident to v is clear, i.e., $E_i \cap \{av, bv, cv\} = \{av\}$, then $O_i^{\triangle} \cap T_v = \{v_a\}$;

(c) if exactly two edges, say $av, bv \in E(G)$, incident to v are clear, i.e., $E_i \cap \{av, bv, cv\} = \{av, bv\}$, then $O_i^{\triangle} \cap T_v = \{v_c\}$ and $E_v \subseteq E_i^{\triangle}$.

If $\mathcal{S}^{\triangle}$ satisfies Property (1), then it clearly clears $G^{\triangle}$. Indeed, let t be the index of the last round in $\mathcal{S}$, then $R_t = V(G)$ (because $\mathcal{S}$ clears G) and $\psi_{\triangle}(R_t \setminus T) \cup \bigcup_{v \in R_t \cap T} T_v = V(G^{\triangle}) \subseteq R_t^{\triangle}$.

We finally define strategy $\mathcal{S}^{\triangle}$ and prove it satisfies these properties by induction on the number of rounds of $\mathcal{S}$.

Phase 0 proceeds as follows. First, for any $v \in O_0 \setminus T$, place one searcher at $\psi_{\triangle}(v)$ (hence Property (2) holds) and for any $v \in O_0 \cap T$, place one searcher at some node in T_v (to be explained below at which one). Note that exactly k searchers are used. For any $v \in T$, let $N(v) = \{a, b, c\}$. There are three cases depending on the number $0 \leq h \leq 3$ of v's incident edges that are in E_0.

- If $h = 0$, then the searcher is placed at an arbitrary node of T_v. Hence Property (4.a) holds for node v.

- If $h = 1$, let $av \in E_0$ and hence a is occupied (indeed, $\phi_{\triangle}^{-1}(a) \in O_0 \setminus T$ and then $a \in O_0^{\triangle}$). Then, the searcher is placed at $v_a \in T_v$ (see Figure 2). Hence Property (4.b) holds for node v. Since $\psi_{\triangle}(a)$ is also occupied, $\phi_{\triangle}(av)$ is cleared.

- If $h \in \{2, 3\}$, let $\{av, bv\} \subseteq E_0 \cap \{av, bv, cv\}$, then the searcher in T_v is placed at v_c, the searcher at $\psi_{\triangle}(b)$ slides to v_b and the searcher at $\psi_{\triangle}(a)$ slides to v_a. The edges in E_v are cleared. Then, the searchers at v_a and v_b return to $\psi_{\triangle}(a)$ and $\psi_{\triangle}(b)$ respectively (see Figure 3). Note that $\psi_{\triangle}(a)$ and $\psi_{\triangle}(b)$ have degree at most 2 and only the edges $\psi_{\triangle}(a)v_a$ and $\psi_{\triangle}(b)v_b$ might have been recontaminated but they are cleared again at the end of the phase. Hence if $h = 2$ then Property (4.c) holds for node v, while if $h = 3$ then Property (4.a) holds for node v.

Hence, at the end of Phase 0, for every pair of nodes $v \in V(G)$, $\psi_{\triangle}(v) \in V(G^{\triangle})$, where $v \in O_0 \setminus T$, Property (2) holds (that is for every occupied node of degree at most 2 in G exactly one node: the corresponding node in $G^{\triangle}$ is also occupied), and for every node $v \in O_0 \cap T$ and its corresponding triangle T_v, Property (4.a) or (4.b) or (4.c) holds (that is for every occupied node v of degree 3 in G there is exactly one occupied node in the corresponding triangle T_v in $G^{\triangle}$ so that if an edge $vx \in E(G)$ is clear then $\phi_{\triangle}(vx) \in E(G^{\triangle})$ is also clear and additionally if at least two of v's incident edges are clear in G then all edges in the corresponding triangle E_v in $G^{\triangle}$ are also clear). Therefore properties (1), (3) immediately follow.

Let $i \geq 0$ and assume that the properties hold at the end of Phase i. If Round i is the last one in $\mathcal{S}$, then, as previously mentioned, $G^{\triangle}$ is clear at the end of Phase i. Otherwise, we define the next Phase $i+1$ and show that the properties still hold after Phase $i+1$.

Phase $i+1$ is based on the Round $i+1$ of $\mathcal{S}$. There are two cases depending on whether Round $i+1$ consists of sliding a searcher or of a pair of removal-placement steps.

First, let us assume that Round $i+1$ consists of making a searcher slide from node u to node v.

1. Suppose that $u \in T$. Then, $v \in V(G) \setminus T$. By monotonicity of $\mathcal{S}$, uv is the single contaminated edge incident to u. Therefore, by Property (4.c), a searcher is at u_v in $G^{\triangle}$

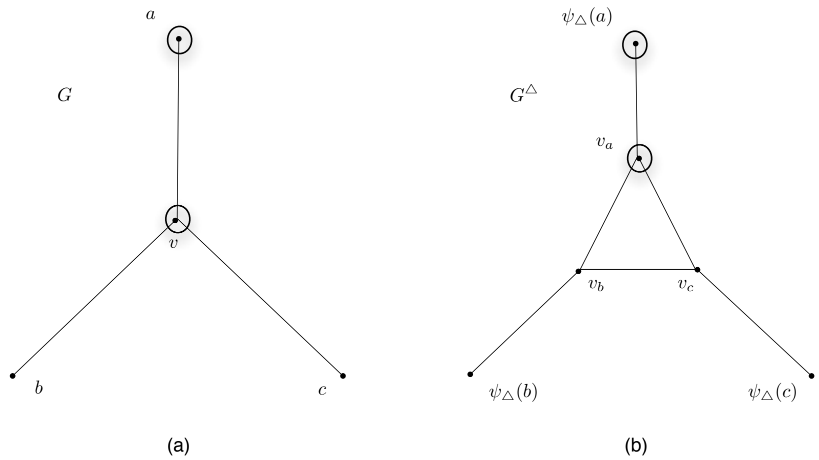

Figure 2: Let $v \in O_0 \cap T$. If only one edge, av incident to v is clear (i.e., $a \in O_0 \setminus T$) (a), then the searcher in T_v is placed at v_a (b).

and only $u_v\psi_\triangle(v)$ is contaminated. Moreover, $\psi_\triangle(v)$ is not occupied by Property (2). Therefore, Phase $i+1$ of $\mathcal{S}^\triangle$ makes the searcher at u_v to slide along $u_v\psi_\triangle(v) = \phi_\triangle(uv)$.

(a) If v is not adjacent to another node (apart from u) of degree 3 then it is easy to check that all properties are still satisfied after Phase $i+1$.

(b) If however, v *is* adjacent to another node w, different than u, of degree 3, then if w was occupied at the end of Round i (and hence by Property (4) there is exactly one searcher at w's corresponding triangle after Phase i of $\mathcal{S}^\triangle$) there are two subcases: i) No edge incident to w was clear after Round i: then the searcher at w's corresponding triangle, slides to w_v (the vertex of the w's triangle that is adjacent to v) and it is easy to see that all properties are satisfied. ii) At least one edge incident to w (different of course than vw) was clear after Round i: Suppose that at least edge wa was clear. Then either w_v was already occupied and hence Property (4.c) was true (in that case all properties hold after Phase $i+1$), or node w_a was occupied and hence Property (4.b) was true. Then in Phase $i+1$, after the sliding of the searcher at $\psi_\triangle(v)$ the same searcher slides to node w_v, then the searcher at w_a slides to w_c (where c is the remaining adjacent node of w in G), and finally the searcher at w_v slides back to $\psi_\triangle(v)$ (see Figure 4). Now all properties hold.

2. The case when $v \in T$ and $u \in V(G) \setminus T$ is similar: Phase $i+1$ of $\mathcal{S}^\triangle$ makes the searcher at $\psi_\triangle(u)$ slide along $\psi_\triangle(u)v_u$. Since v was contaminated before, it was unoccupied and all incident edges were contaminated.

(a) If there were no searchers at v's adjacent nodes $a, b \in N_G(v)$, where $a, b \neq u$ at the end of Round i of $\mathcal{S}$, then by Property (2), at the end of Phase i of $\mathcal{S}^\triangle$, nodes $\psi_\triangle(a)$, $\psi_\triangle(b)$ of $G^\triangle$ are not occupied. Hence after Phase $i+1$ all properties are still satisfied.

(b) If at least one more (apart from u) adjacent node of v, say $a \in N_G(v)$ was occupied at the end of Round i of $\mathcal{S}$, then by Property (2), at the end of Phase i of $\mathcal{S}^\triangle$, node $\psi_\triangle(a)$ of $G^\triangle$ is also occupied. In that case after the sliding of the searcher along the edge $\psi_\triangle(u)v_u$, the searcher at $\psi_\triangle(a)$ slides along the edge $\psi_\triangle(a)v_a$, then the searcher

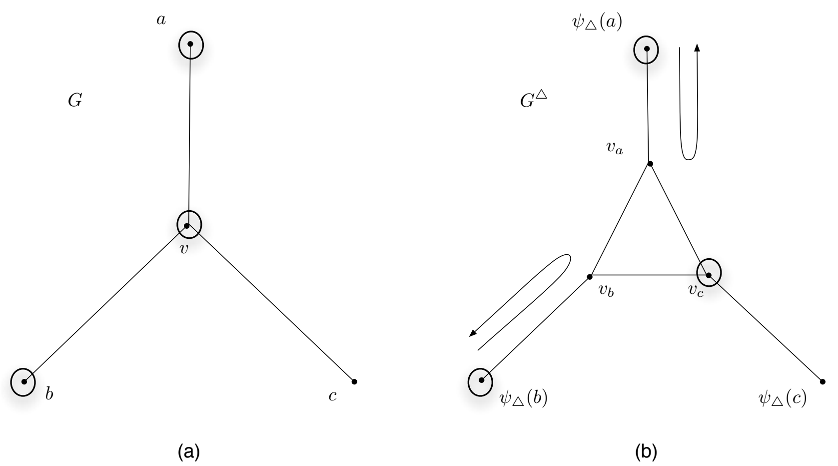

Figure 3: Let $v \in O_0 \cap T$. If at least two edges, av, bv incident to v are clear (i.e., $a, b \in O_0 \setminus T$) (a), then the searcher in T_v is placed at v_c and the searchers at $\psi_\triangle(a)$ and $\psi_\triangle(b)$ slide to v_a and v_b respectively and return (b).

at v_u slides along the edge $v_u v_c$ (where c is the remaining adjacent node of v in G), and finally the searcher at v_a slides back to $\psi_\triangle(a)$ (see Figure 5). It can be easily checked that after Phase $i+1$ all properties are still satisfied.

3. Finally, the case when $u, v \in V(G) \setminus T$ can be delt with similarly: Phase $i+1$ of $\mathcal{S}^\triangle$ makes the searcher at $\psi_\triangle(u)$ slide along $\psi_\triangle(u)\psi_\triangle(v)$. If v is not adjacent to a node of degree 3 then it is easy to check that all properties are still satisfied after Phase $i+1$. If however, v *is* adjacent to a node w of degree 3, the situation is exactly the same as in case 1.b above.

Now, assume that Round $i+1$ consists of a pair of removal-placement steps where a searcher from one node u *jumps* to another node v. Let $u \in R_i \cap O_i$ be the node from which a searcher is removed at Round $i+1$ by $\mathcal{S}$ and let $v \in V \setminus C_i$ be the node at which this searcher is placed then. Either $u \in V(G) \setminus T$ and $x = \psi_\triangle(u) \in O_i^\triangle \cap R_i^\triangle$ by Properties (1) and (2), or $u \in T$ and all nodes in T_u are in $R_i^\triangle$ by Properties (3) and (4.a), moreover, one searcher is at some node $x \in T_u$ by Property (4). Let us call the searcher at x the *current* searcher. Let $y = \psi_\triangle(v)$ if $v \notin T$ and let $y \in T_v$ otherwise. Let P be some path from x to y in $G^\triangle$. Phase $i+1$ of $\mathcal{S}^\triangle$ will mimic the jump of the searcher from u to v in $\mathcal{S}$ by a sequence of slidings along P from x to y in $G^\triangle$ in a way that, all nodes and edges that were clear before the start of the Phase $i+1$ are clear at the end of Phase $i+1$: in particular, only controlled recontamination may occur.

Notice that removing the searcher from x cannot cause any recontamination of edges in E_i since $\mathcal{S}$ is monotone and observe that placing a new searcher anywhere on $G^\triangle$ and having it slide an edge cannot cause any recontamination of edges in $E_i^\triangle$ or E_i.

Assume the current searcher is at some node $z \in V(P)$ and that all nodes and edges that were clear after Phase i are still clear. Let w be the neighbor of z that stands between z and y in P. If w is unoccupied, then the current searcher slides to w. Then the process goes on with w instead of z. As long as w is unoccupied, moving the searcher from z to w causes no recontamination of edges in E_i as observed above. The only potential recontamination of edges in E_i could happen if a searcher in $O_i^\triangle$ (different than the one at x) moves. We show below that

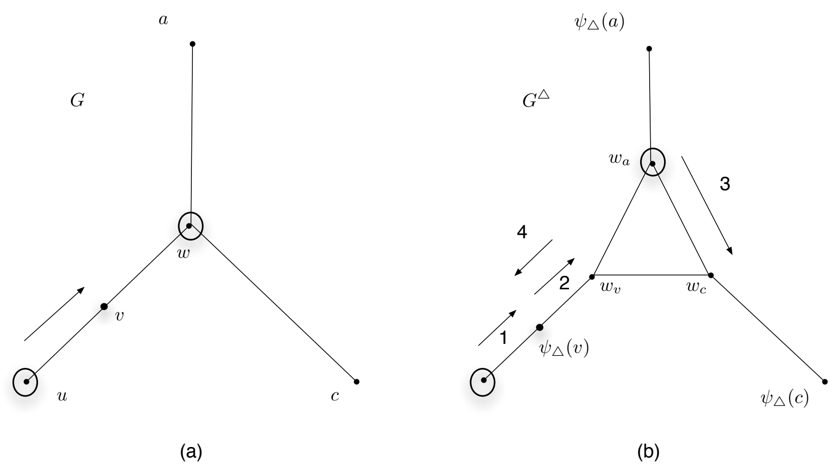

Figure 4: (a) The case when in Round $i+1$ of $\mathcal{S}$ a searcher at a node u slides towards a node v of degree 2 whose other neighbor w has degree 3 and it was already occupied at the end of Round i and the edge wa was clear. (b) The corresponding Phase $i+1$ of $\mathcal{S}^\triangle$; the numbers declare the ordering of slidings.

such recontamination can happen only to edges belonging in P and before the end of Phase $i+1$ those edges are cleared.

So let us assume that w is occupied. Note that y is not occupied so $w \neq y$ is an internal node of P and has degree at least 2.

- if w has degree 2, then the searcher at w becomes the current searcher and the process goes on, but after the first move of the searcher at w (the new current searcher), then the searcher at z (the former one) slides to w. Because w has degree 2, zw cannot be recontaminated and therefore no recontamination of edges in E_i occurs. Moreover, all nodes (except x) that were occupied after Phase i are still occupied.
- if w has three neighbors: $z, r \in P$ and s. Let $\ell \in V(G)$ be the vertex such that $w \in T_\ell$. There are several cases to be considered depending on which of the edges incident to w are contaminated and which vertices belong to T_ℓ.
  - if $T_\ell = \{w, r, s\}$. By Property (4), none of r and s are occupied. Then, the searcher at w slides to r, the one at z slides to w and the searcher at r becomes the current searcher (see Figure 6(a)). During this process, only zw could belong to E_i and it may be recontaminated but it is clear again after the slide along zw.
  - if $T_\ell = \{w, z, s\}$, then the searcher at w becomes the current searcher and the process goes on, but after the first move of the searcher at w (the new current searcher), then the searcher at z (the former one) goes to w (see Figure 6(b)). Now, only edge wr could belong to E_i and it may be recontaminated but it is clear again after the slide along zw.
  - if $T_\ell = \{w, z, r\}$. By Property (4), r is not occupied, so the current searcher at z goes to r and the process goes on (see Figure 6(c)).

Finally, suppose that the next $m \geq 1$ nodes, $r_1, r_2, ..., r_m$ in P after w are also occupied by searchers. Then the searchers, one by one (first the searcher at r_m, then the one at r_{m-1} and so

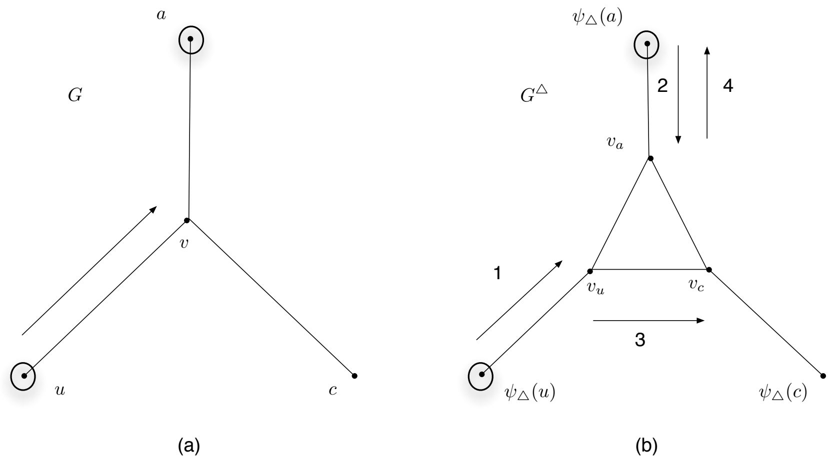

Figure 5: (a) The case when in Round $i+1$ of $\mathcal{S}$ a searcher at a node u of degree at most 2 slides towards a node v of degree 3 whose at least one more neighbor a was already occupied at the end of Round i. (b) The corresponding Phase $i+1$ of $\mathcal{S}^\triangle$; the numbers declare the ordering of slidings.

on), slide to their next node in P. Finally the searcher at w slides to r_1 and then the searcher at z slides to w. The only edges in E_i that may have been recontaminated during this procedure are edges in the subsequence of P: $< z, w, r_1, r_2, ..., r_m >$. Nevertheless, those edges are clear again at the end of this procedure as explained above.

Proceeding that way, a searcher eventually reaches y and all properties satisfied after Phase i remain valid but x is not occupied anymore while y is now occupied.

If y has degree 2 then $y = \psi_\triangle(v)$. If v is adjacent to a node h in G of degree 3, and h was occupied after Round i then Phase $i+1$ follows the same procedure as in case 1.b above (replacing w in case 1.b with h) to make sure that all properties hold.

If $v \in T$ then $y \in T_v$ and following the same procedure as in case 2 above we can clear the desired edges to satisfy Property 4. ■ ■

Lemma 5 *For any graph $G \in \mathcal{C}$, $s(G) \le xs(G^\triangle)$.*

Proof. Let $\mathcal{S}^\triangle$ be an exclusive search strategy using k searchers for $G^\triangle$. Then we can transform $\mathcal{S}^\triangle$ to a mixed strategy $\mathcal{S}$ using k searchers for G as follows. Let $I^\triangle \subseteq V(G^\triangle)$ be the set of the k nodes that are initially occupied in $\mathcal{S}^\triangle$. Let $I = \{v \in V(G) \mid (v \in T \text{ and } T_v \cap I^\triangle \neq \emptyset) \text{ or } (v \notin T \text{ and } \psi_\triangle(v) \in I^\triangle\}$, that is I is the set of nodes of G that correspond to a node in $I^\triangle$.

$\mathcal{S}$ starts by placing one searcher at each node in I. Moreover, for any $v \in T \cap I$, if two (resp., three) nodes in T_v are initially occupied by $\mathcal{S}^\triangle$, i.e., if $|T_v \cap I^\triangle| = 2$ (resp., if $|T_v \cap I^\triangle| = 3$), then two (resp., three) searchers are placed at v by $\mathcal{S}$.

Recall that F_1 is the set of edges of $G^\triangle$ that correspond to edges in G. That is F_1 is the set of edges of $G^\triangle$ that are not an edge of a triangle. Now, for each move done by $\mathcal{S}^\triangle$, if this move consists of sliding a searcher along an edge of a triangle (i.e., an edge in $E(G^\triangle) \setminus F_1$, $\mathcal{S}$ does nothing. Otherwise, if $\mathcal{S}^\triangle$ slides a searcher along an edge $e \in F_1$, then $\mathcal{S}$ slides a searcher along the corresponding edge $\phi_\triangle^{-1}(e)$.

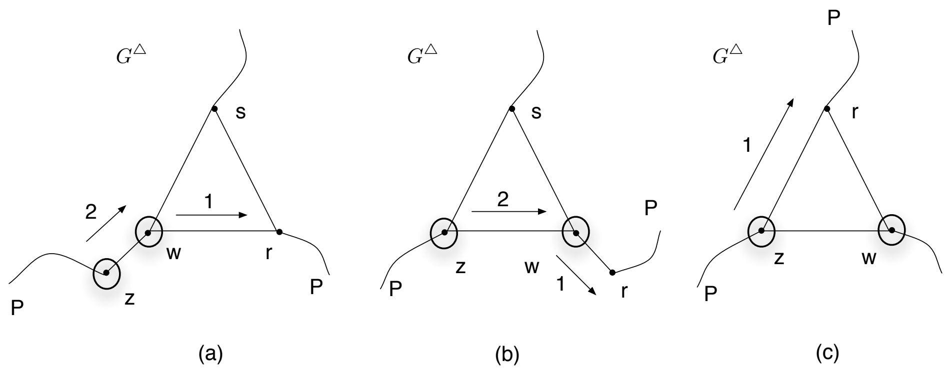

Figure 6: A searcher at node z needs to move along path P towards node w which is also occupied. The numbers in each case declare the ordering of slidings.

Such a strategy $\mathcal{S}$ is a mixed strategy that clears G using k searchers. ■ ■

2 Exclusive Graph Searching in star-like graphs

In this section, we study the complexity of computing the exclusive search number in star-like graphs. Surprisingly, our complexity results are somehow *orthogonal* to the ones concerning pathwidth in this class of graphs.

2.1 Star-like graphs

A connected graph $G = (V, E)$ is a *star-like graph* if V can be covered by cliques $C_0, C_1, \cdots, C_r$ such that, for any $i, j \leq r$, $C_i \cap C_j \subseteq C_0$. Said differently, a graph is a star-like graph if it is chordal and its clique tree is a star. A graph is *k-star-like* if $c_i = |C_i \setminus C_0| \leq k$ for any $1 \leq i \leq r$. C_0 is called the central clique and any node in $V \setminus C_0$ is called *peripherical*.

We start with two simple claims. The first one is straightforward and its proof is omitted.

Claim 3 *Let G be any graph and $v \in V(G)$. If there an exclusive strategy for G, using $xs(G)$ searchers, such that v is occupied during the whole strategy, then $xs(G \setminus \{v\}) = xs(G) - 1$.*

Claim 4 *Let G be any graph containing a clique C as subgraph. Let $\mathcal{S}$ be any exclusive strategy and s be any step of $\mathcal{S}$ such that for any step s' before s (including s), at most $|C| - 2$ nodes of C are occupied at step s'. Then, at step s, there is a contaminated edge $e \in E(C)$ with both ends unoccupied.*

Moreover, just after the first step when $|C| - 1$ searchers are in C, all edges incident to the unoccupied node are contaminated.

Proof. Assume that at most $|C| - 2$ searchers are initially placed on the nodes of C. Let us say u and $v \in C$ are not occupied. Then uv is contaminated. Now, consider any step where some edge $uv \in E(C)$ is contaminated and u and v are not occupied. Consider any sliding move along edge $xy \in E(G)$ (a searcher goes from x to y) after which at most $|C| - 2$ searchers are in C.

- If $xy \in E(C)$, w.l.o.g., $y \neq v$ (note that $x \notin \{u, v\}$). Then, after the move, $xv \in E(C)$ is contaminated and its ends are not occupied.
- If $y \notin V(C)$, clearly after the move, $uv \in E(C)$ is still contaminated and its ends are not occupied.
- If $xy \notin E(C)$ and $y \in C$, then, before the move, there were at least three nodes u, v and z that are not occupied in C. Since uv is contaminated, uz and vz are contaminated too. W.l.o.g., assume $y \notin \{u, v\}$. Then, after the move, $uv \in E(C)$ is still contaminated and its ends are not occupied. ■

■

Theorem 3 *Let G be a star-like graph with cliques $(C_0, \cdots, C_r)$ such that $|C_i \setminus C_0| > 1$ for any $0 < i \leq r$, that is each non central clique has at least two peripherical nodes.*

1. *Either there is $(u, v) \in E(C_0)$ with $\{0 < i \leq r \mid u \in C_i\} \cap \{0 < i \leq r \mid v \in C_i\} = \emptyset$ and $mxs(G) = |V(G)| - r - 1$,*
2. *or $mxs(G) = |V(G)| - r$.*

Proof. Clearly, $mxs(G) \leq |V(G)| - r$. Indeed, consider the following strategy: place a searcher at any node of the graph but one peripherical node w_i per clique C_i, $0 < i \leq r$. In particular, all nodes of C_0 are occupied. Then, sequentially for $i = 1$ to r, slide a searcher from one occupied peripherical node of C_i to w_i.

Moreover, let us assume there is an edge $(u, v) \in E(C_0)$ as defined in the case 1. Consider the following strategy: place a searcher at any node of the graph but v and one peripherical node w_i per clique C_i, $0 < i \leq r$. In particular, all nodes of $C_0 \setminus \{v\}$ are occupied. W.l.o.g., let $i \leq r$ and $\{0 < j \leq r \mid u \in C_j\} = \{1, \cdots, i\}$. Note that $v \notin C_j$ for any $j \leq i$, i.e., v is not adjacent to any node in $C_j \setminus C_0$. Then, sequentially for $j = 1$ to i, slide a searcher from one occupied peripherical node of C_j to w_j. Then, slide the searcher at u to v. Finally (if $i < r$), sequentially for $j = i + 1$ to r, slide a searcher from one occupied peripherical node of C_j to w_j. Hence, in Case 1, $mxs(G) \leq |V(G)| - r - 1$.

We now show that $mxs(G) \geq |V(G)| - r - 1$. Let us consider a monotone strategy using k searchers.

- First, let us assume that, initially, there are two nodes $u, v \in C_i \setminus C_0$ that are not occupied, for some $0 < i \leq r$. It is easy to check that the first time that u or v is occupied then the edge by which the searcher arrives in v or u must be recontaminated. Hence, initially, all peripherical nodes but at most one per clique must be occupied.

Let $\{v_1, \cdots, v_x\}$ be the nodes of C_0 that are not occupied initially. By previous remark, the number of searchers k must be at least $|V(G)| - r - x$ (all nodes are occupied but at most one peripherical node per clique and the x unoccupied nodes in C_0).

- If $x = 1$, we already have $k \geq |V(G)| - r - 1$.
- Otherwise, let us assume $x \geq 2$. Moreover, assume that these nodes $\{v_1, \cdots, v_x\}$ are ordered in the order they are occupied by the strategy, i.e., v_i is occupied before v_j for any $i < j$. We need the following remarks

– Until v_{x-1} is occupied, no searcher occupying a node u of C_0 can move. Indeed, otherwise u would be recontaminated by v_x. Therefore, for any $1 \leq i < x$, when v_i is occupied, it is by a searcher sliding from some node $y_i \in C_i \setminus C_0$ for some $1 \leq i \leq r$.
– Moreover, for any $1 \leq i < x$, all nodes of C_i (the clique containing y_i) must be initially occupied. Indeed, for purpose of contradiction, let us assume that there is a node $w \in C_i$ that is not occupied initially. Then, before v_i is occupied, no searcher can slide from $u \in C_i$ to w since otherwise, u would be recontaminated by v_i. Therefore, when a searcher slides from y_i to v_i, w is still unoccupied and contaminated. Hence, y_i is recontaminated by w, a contradiction.
– Finally, for any $1 \leq i < j < x$, $C_i \neq C_j$. Indeed, otherwise, when a searcher slides along $\{y_i, v_i\}$ to occupied v_i, y_i would be recontaminated by v_j.

All together, the above remarks imply that if $x \geq 2$ nodes are initially occupied in C_0, then at least $x-1$ peripherical cliques must have all their nodes occupied initially. Moreover, since there can be at most one peripherical node that is initially unoccupied per peripherical clique, there can be at most $r-x+1$ peripherical nodes that are initially unoccupied. In total at most $r+1$ nodes can be initially unoccupied and $k \geq |V(G)| - r - 1$.

Finally, let us show that, if $mxs(G) = |V(G)| - r - 1$, then there is $(u, v) \in E(C_0)$ with $\{0 < i \leq r \mid u \in C_i\} \cap \{0 < i \leq r \mid v \in C_i\} = \emptyset$. For purpose of contradiction, let us assume that, for any $u, v \in C_0$, there is $0 < i \leq r$ with $u, v \in C_i$. Let us consider a monotone strategy using $|V(G)| - r - 1$ searchers.

For the same reason as before, at most one peripherical node per clique can be unoccupied initially. Therefore, there is at least one in C_0 that is unoccupied. Let v be the last node of C_0 to be occupied during the strategy. Consider the configuration just before the step when v is occupied.

- We claim that all nodes of C_0 but v are occupied. Indeed, we already proved that while all nodes of C_0 but at most one are not occupied, no searcher occupying a node in C_0 can move. Moreover, once all nodes of C_0 but v are occupied, if a searcher at some node in C_0 moves, it must be to go to v since otherwise there would be some recontamination from v.

- Moreover, at most one peripherical node per clique can be unoccupied. Indeed, for purpose of contradiction, assume that there is $0 < j \leq r$ and $x, y \in C_j \setminus C_0$ that are unoccupied. Since intially, at most one node of $C_j \setminus C_0$ was unoccupied, there must be one step, before v is occupied, such that: x is unoccupied and a searcher goes from y to a node $z \in C_0$. However, it means that during this step, v recontaminates y via z and x, a contradiction.

Therefore, just before v is occupied, there are exactly $|C_0| - 1$ searchers occupying the nodes of C_0 and, for any $0 < i \leq r$, at most one node of $C_i \setminus C_0$ is unoccupied. Since the number of searchers is $|V(G)| - r - 1$, this implies that there is exactly one unoccupied node in $C_i \setminus C_0$, for any $0 < i \leq r$.

Now, consider the searcher that slides from some node u to occupy v. If $u \in C_i \setminus C_0$, since there is an unoccupied node $w \in C_i \setminus C_0$, this would imply the recontamination of u via w (which is adjacent to v). Hence, u must be in C_0. However, by the hypothesis, there is $0 < i \leq r$ with $u, v \in C_i$, and, moreover, there is an unoccupied node $w \in C_i \setminus C_0$. Again, u would be recontaminated by v via w. A contradiction. ■ ■

We now focus on 1-star-like graphs, also called *split graphs*. In other words, a connected graph $G = (V, E)$ is a split graph if V can be partitioned into C and I where C induces a clique and I is an independent set.

2.1.1 Structure of exclusive strategies in split graphs.

Let $G = (V, E)$ be a split graph with $V = C \cup I$. We say that G is *k-structured* if there exist three sets (possibly empty) $E_1, E_2, F \subseteq E$ with the following properties:

1. $E_1 = \{x_1u_1, \cdots, x_ru_r\}$ with $X = \{x_1, \cdots, x_r\} \subseteq I$ and $u_i \in C$ for any $i \leq r$,
   and $N(x_i) \cap \{u_{i+1}, \cdots, u_r\} = \emptyset$ for all $1 \leq i < r$;
2. $E_2 = \{y_1v_1, \cdots, y_rv_s\}$ with $Y = \{y_1, \cdots, y_s\} \subseteq I$ and $v_i \in C$ for any $i \leq s$,
   and $N(y_i) \cap \{v_1, \cdots, v_{i-1}\} = \emptyset$ for all $1 < i \leq s$;
3. $X \cap Y = \emptyset$;
4. $|F| \leq 1$ and, if $F = \{uv\}$, then $u, v \in C$ and u is adjacent with no node in Y and v is adjacent with no node in X;
5. and finally, $|V| - |F| - |X| - |Y| = |C| - |F| + |I \setminus (X \cup Y)| \leq k$.

Lemma 6 *Let G be a split graph. G is k-structured iff $mxs(G) \leq k$.*

Proof. First, let us assume that G is k-structured. We define the following monotone strategy using $|C| - |F| + |I \setminus (X \cup Y)|$ searchers. First, place searchers at any node in $I \setminus (X \cup Y)$ (they will never move), in X and in $C \setminus \{v, u_1, \cdots, u_r\}$ (if v is not defined, i.e., if $F = \emptyset$, then we do not consider it). Then, for $i = 1$ to r, slide the searcher at x_i to u_i. Then, if $F \neq \emptyset$, slide the searcher at u to v. Finally, for $i = 1$ to s, slide the searcher at v_i to y_i. It is easy to see that this strategy is exclusive and clears G in a monotone way. Hence, $mxs(G) \leq k$.

Second, let $k \geq mxs(G)$ and let $\mathcal{S}$ be any monotone exclusive strategy for G using k searchers.We show that there are monotone exclusive strategy with a particular shape.

We first prove that, while at most at most $|C| - 2$ nodes of C are occupied, the only possible move is to slide a searcher from I to C. Note that, by Claim 2 above, at least one edge of C, say $uv \in E(C)$, is contaminated with both its ends not occupied. If $\mathcal{S}$ slides a searcher from a node $x \in C$ to a node y, then the edge xy would be immediately recontaminated because of uv. Hence, the only possible action is to move a searcher occupying a node in I to a node in C.

Let $\{x_1u_1, \cdots, x_ru_r\}$ be the edges crossed by searchers until $|C| - 1$ nodes of C are occupied. By previous paragraph, $x_i \in I$ and $u_i \in C$ for all $i \leq r$ and, moreover, they are pairwise distinct since the strategy is exclusive. Finally, when a searcher moves along x_iu_i, all nodes in $N(x_i) \setminus u_i$ are occupied since otherwise there would be recontamination due to the contaminated edge of C. Hence, $N(x_i) \cap \{u_{i+1}, \cdots, u_r\} = \emptyset$.

After the move along x_ru_r, there is a single node $v \in C$ which is unoccupied and, by Claim 2, all its incident edges are contaminated. The next step of $\mathcal{S}$ cannot be to slide a searcher from C to I since otherwise there would be some recontamination due to v. Hence, there are two cases. Either the next step of $\mathcal{S}$ is to move a searcher from $x_{r+1} \in I$ to $u_{r+1} \in C$, in which case we have a set $\{x_1u_1, \cdots, x_ru_r, x_{r+1}u_{r+1}\}$ which satisfies the desired properties (for the same arguments as in previous paragraph). Or $\mathcal{S}$ slides a searcher along an edge $uv \in E(C)$ in which case $v \notin \bigcup_{i \leq r} N(x_i)$ since, otherwise, some node x_i would have been recontaminated. In the latter case, we set $F = \{uv\}$ (otherwise let $F = \emptyset$).

Let $Y = \{y_1, \cdots, y_s\}$ be the remaining unoccupied nodes in $I \setminus X$ in the order in which they are occupied by $\mathcal{S}$. Note that $k = |C| - |F| + |I \setminus (X \cup Y)|$. At this step of $\mathcal{S}$, the contaminated edges of G are exactly the edges incident to some node in Y (in particular, all edges in $E(C)$ are clear). Therefore, if $F = \{uv\}$, $N(u) \cap Y = \emptyset$ since, otherwise, edges incident to u would be recontaminated because nodes in Y.

Now, let $i \leq s$ and consider the first step when $\mathcal{S}$ occupies y_i by sliding a searcher from a node in C, call it v_i, to y_i. Since all edges incident to some node in $\{y_{i+1}, \cdots, y_s\}$ are still contaminated, we get that v_i is adjacent with none of these nodes (since otherwise there would be some recontamination). Hence, $N(y_i) \cap \{v_1, \cdots, v_{i-1}\} = \emptyset$ for all $1 < i \leq s$.

Thus, G is k-structured. ■ ■

The proof of Lemma 2 directly follows.

2.1.2 Maximum Augmenting cover is NP-hard.

In what follows, we prove that MAC is NP-hard by doing a reduction from MIN-SAT.

An instance of MIN-SAT in the Boolean variables $\{v_1, \cdots, v_n\}$ is composed of a collection of clauses $\{C_1, \cdots, C_m\}$. Each clause C_i is of the form $z_1 \vee z_2 \vee \cdots \vee z_{k_i}$ ($k_i \geq 1$) where each z_j is either a variable v_ℓ or its negation $\bar{v}_\ell$. Each such z_j is called a literal. The goal is to assign the variables $v_1, \cdots, v_n$ Boolean values 0 or 1 so that the total number of the satisfied clauses is minimized. The problem is known to be NP-hard even for $k_i \leq 2$ for any $i \leq m$ [AZ05].

Theorem 5 *MAC is NP-hard.*

Proof. Let $\Phi = \wedge_{1 \leq j \leq r} C_j$ be an instance of MIN-SAT, i.e., a boolean formula on variables $\{v_1, \cdots, v_n\}$ in Conjunctive Normal Form. From Φ, we define an instance $(A, \mathcal{F})$ of MAC as follows. First, let α and β be two integers whose values will be precised later.

The ground set A consists of the following $\alpha n + \beta m$ elements. For any $1 \leq j \leq m$, let $K_j = \{c_j^1, \cdots, c_j^\beta\}$ be a set of β elements corresponding to clause C_j. For any $1 \leq i \leq n$, let $X_i = \{x_i^1, \cdots, x_i^\alpha\}$ be a set of α elements corresponding to variable v_i. Let us set $K = \bigcup_{1 \leq j \leq m} K_j$, $X = \bigcup_{1 \leq i \leq n} X_i$ and $A = X \cup K$.

For any $1 \leq j \leq m$ and $1 \leq b \leq \beta$, let $Sc_j^b = \{c_j^b\} \cup X$.

For any $1 \leq i \leq n$, let J_i be the set of element c_j^b ($1 \leq b \leq \beta$) such that Variable v_i appears positively in Clause C_j. Similarly, let $\bar{J}_i$ be the set of element c_j^b ($1 \leq b \leq \beta$) such that Variable v_i appears negatively in Clause C_j.

For any $1 \leq i \leq n$ and any $1 \leq a \leq \alpha$, let $Sv_i^a = \{x_i^a\} \cup J_i$ and $S\bar{v}_i^a = \{x_i^a\} \cup \bar{J}_i$.

Finally, let $\mathcal{F} = \{Sc_j^b \mid 1 \leq j \leq m, 1 \leq b \leq \beta\} \cup \{Sv_i^a, S\bar{v}_i^a \mid 1 \leq i \leq n, 1 \leq a \leq \alpha\}$.

Claim 5 *If there is a Boolean assignment of $\{v_1, \cdots, v_n\}$ that satisfies at most k clauses of Φ then there is an augmenting sequence $\mathcal{S}$ of $(A, \mathcal{F})$ of length at least $\ell = \alpha n + (m-k)\beta$.*

W.l.o.g., let us assume that there is $r \leq n$ such that assigning 1 to $\{v_1, \cdots, v_r\}$ and 0 to $\{v_{r+1}, \cdots, v_n\}$ does not satisfy the clauses $C_1, \cdots, C_{m-k}$ in Φ.

Let us consider the sequence $\mathcal{S} = (S'_1, \cdots, S'_\ell) = (Sv_1^1, \cdots, Sv_1^\alpha, Sv_2^1,$
$\cdots, Sv_2^\alpha, \cdots, Sv_r^1, \cdots, Sv_r^\alpha, S\bar{v}_{r+1}^1, \cdots, S\bar{v}_{r+1}^\alpha, \cdots, S\bar{v}_m^1, \cdots, S\bar{v}_m^\alpha,$
$Sc_1^1, \cdots, Sc_1^\beta, \cdots, Sc_{m-k}^1, \cdots, Sc_{m-k}^\beta)$.

For any $j = \alpha i + a$, $0 \leq i < n$ and $1 \leq a \leq \alpha$, $x_{i+1}^a \in S'_j \setminus \bigcup_{p<j} S'_p$. Moreover, for any $j = \alpha n + \beta j + b$, $0 \leq j < m-k$ and $1 \leq b \leq \beta$, $c_{j+1}^b \in S'_j \setminus \bigcup_{p<j} S'_p$. Hence, $\mathcal{S}$ is an augmenting sequence.

Claim 6 *If there is an augmenting sequence $\mathcal{S} = (S'_1, \cdots, S'_\ell)$ of $(A, \mathcal{F})$ of lengtht $\ell \geq \alpha n + (m-k)\beta$, then there is a Boolean assignment of $\{v_1, \cdots, v_n\}$ that satisfies at most k clauses of Φ.*

Let $\mathcal{S}$ be an augmenting sequence with maximum length. We first need to prove that we can restrict ourselves to sequences $\mathcal{S}$ with a particular form.

- Let $r \leq \ell$ be the smallest integer such that $S'_r = Sc^b_j$ for some $b \leq \beta$ and $j \leq m$. We first prove that we may assume that, for any $r' \geq r$, $S'_{r'} = Sc^{b'}_{j'}$ for some $b' \leq \beta$ and $j' \leq m$ (i.e., $S'_{r'} \notin \{Sv^a_i, S\bar{v}^a_i \mid 1 \leq i \leq n, 1 \leq a \leq \alpha\}$). Indeed, let us assume that there is $r' \geq r$ such that $S'_{r'} \in \{Sv^a_i, S\bar{v}^a_i\}$ for some $1 \leq i \leq n, 1 \leq a \leq \alpha$. Since $X \subseteq S'_r$ and $\mathcal{S}$ is an augmenting sequence, there must be c^b_j (for some $b \leq \beta$ and $j \leq m$) such that $c^b_j \in S'_{r'} \setminus \bigcup_{p<r'} S'_p$. Hence, replacing S'_r by Sc^b_j in $\mathcal{S}$ leads to another augmenting sequence with length at least ℓ.

  From now on, let us assume that $\mathcal{S}$ satisfies this property, i.e., there is $r \leq \ell$ such that $S'_{r'} = Sc^{b'}_{j'}$ for some $b' \leq \beta$ and $j' \leq m$ if and only if $r' \geq r$. In particular, note that $\ell - r \leq \beta m$.

- We then prove that, for any $1 \leq i \leq n$, there is $a \leq \alpha$ such that Sv^a_i or $S\bar{v}^a_i$ belongs to $\mathcal{S}$. Indeed, if it not the case, then $x^a_i \notin \bigcup_{p<r} S'_p$ for any $a \leq \alpha$. Therefore, $\mathcal{T} = (S'_1, \cdots, S'_{r-1}, Sv^1_i, \cdots, Sv^\alpha_i)$ is an augmenting sequence of length $\alpha + r - 1$. Since $r \geq \ell - \beta m$, we get that $|\mathcal{T}| \geq \ell + \alpha - \beta m - 1$. Taking $\alpha > \beta m + 1$, we get that $\mathcal{T}$ is a larger augmenting sequence, contradicting the maximality of $\mathcal{S}$.

- We now prove that, for any $1 \leq i \leq n$, we may assume that either $Z_i = \{Sv^a_i \mid 1 \leq a \leq \alpha\}$ or $\bar{Z}_i = \{S\bar{v}^a_i \mid 1 \leq a \leq \alpha\}$ is a subset of $\mathcal{S}$. Moreover, if Z_i (resp., $\bar{Z}_i$) is a subset of $\mathcal{S}$, then $\mathcal{S}$ contains at most one element in $\bar{Z}_i$ (resp., in Z_i).

  Let $1 \leq i \leq n$ and let D be the first element of $\mathcal{S}$ in $\{Sv^a_i, S\bar{v}^a_i \mid 1 \leq a \leq \alpha\}$ (D exists by previous item). Let us assume that $D \in Z_i$, then we prove that we may assume that $Z_i \subseteq \mathcal{S}$ (by a similar proof, if $D \in \bar{Z}_i$ then we may assume that $\bar{Z}_i \subseteq \mathcal{S}$). Indeed, assume that there is $a \leq \alpha$ and $Sv^a_i \notin \mathcal{S}$. There are two cases to be considered.

  - if $S\bar{v}^a_i \in \mathcal{S}$, simply consider the sequence obtained from $\mathcal{S}$ by replacing $S\bar{v}^a_i$ with Sv^a_i. It is easy to show that it is still augmenting and with same length.
  - otherwise, it can be checked that either $(S'_1, \cdots, S'_{r-1}, Sv^a_i, S_r, \cdots, S_\ell)$ or $(S'_1, \cdots, S'_{r-1}, Sv^a_i, S_{r+1}, \cdots, S_\ell)$ is an augmenting sequence.

  We now prove the second statement of this item, that is, if $Z_i \subseteq \mathcal{S}$ then $|\mathcal{S} \cap \bar{Z}_i| \leq 1$ (the other case can be proved in a similar way). For purpose of contradiction, let us assume that $|\mathcal{S} \cap \bar{Z}_i| > 1$. Let S'_u be the first element of Z_i appearing in $\mathcal{S}$ and let S'_v and S'_w be the first two elements of $\bar{Z}_i$ that appear in $\mathcal{S}$. In particular, $u < v < w$. W.l.o.g., let $S'_w = S\bar{v}^1_i$. Since $S'_w \setminus S'_v = \{x^1_i\}$ and $\mathcal{S}$ is augmenting, $x^1_i \notin \bigcup_{p<w} S'_p$. Hence, $Sv^1_i \notin \{S'_1, \cdots, S'_w\}$. However, because $Sv^1_i \setminus S'_u = \{x^1_i\}$, we get that $Sv^1_i \subseteq \bigcup_{p \leq w} S'_p$. Therefore, Sv^1_i cannot belong to $\mathcal{S}$ (otherwise $\mathcal{S}$ would not be augmenting) which contradicts the fact that $Z_i \subseteq \mathcal{S}$.

- Finally, we prove that, for any $j \leq m$, if there is $b \leq \beta$ such that Sc^b_j belongs to $\mathcal{S}$ then $Sc^{b'}_j$ belongs to $\mathcal{S}$ for any $b' \leq \beta$.

  Note first that the first statement of previous item implies that $X = \bigcup_{1 \leq i \leq n} X_i \subseteq \bigcup_{p<r} S'_p$. hence, if $Sc^b_j = \{c^b_j\} \cup X$ belongs to $\mathcal{S}$, it means that $c^b_j \notin \bigcup_{p<r} S'_p$. By construction, it implies that $c^{b'}_j \notin \bigcup_{p<r} S'_p$ for any $b' \leq \beta$. Therefore, if $Sc^{b'}_j \notin \mathcal{S}$ for some $b' \leq \beta$, it implies that $c^{b'}_j \notin \bigcup_{p \leq \ell} S'_p$. Therefore, we could add $Sc^{b'}_j$ at the end of $\mathcal{S}$, contradicting the maximality of $\mathcal{S}$.

We are now ready to prove the claim. We have just proved that we may assume that $\mathcal{S}$ consists of the following sets: for any $1 \leq i \leq n$, either all elements in Z_i and at most one element in $\bar{Z}_i$ belong to $\mathcal{S}$, or all elements in $\bar{Z}_i$ and at most one element in Z_i belong to $\mathcal{S}$; for any $1 \leq j \leq m$, either all elements in $\{Sc_j^b \mid 1 \leq b \leq \beta\}$ belong to $\mathcal{S}$ or none. Let Q be the set of integers $j \leq m$ such that all elements of $\{Sc_j^b \mid 1 \leq b \leq \beta\}$ belong to $\mathcal{S}$. We have $\ell = \alpha n + \beta|Q| + q$ with $q \leq n$. Moreover, $\ell \geq \alpha n + (m-k)\beta$. Hence, taking $\beta > n$, we get that $|Q| \geq (m-k)$.

To conclude, it is sufficient to consider the following Boolean assignment of the variables. For any $1 \leq i \leq n$, let the variable v_i be assigned to 1 if all elements of Z_i belong to $\mathcal{S}$ and to 0 otherwise. For any $j \in Q$, the clause C_j is not satisfied by such an assignment. Indeed, for purpose of contradiction, let us assume C_j is satisfied by such an assignment. Let us assume C_j contains a positive occurence of some variable v_i assigned to 1 (the case when C_j contains the negation of a variable v_i assigned to 0 is similar). Then, it means that all elements of Z_i appear in $\mathcal{S}$. In particular, it implies that, for any $b \leq \beta$, $c_j^b \in \bigcup_{p<r} S'_p$. This implies that, for any $b \leq \beta$, Sc_j^b cannot belong to $\mathcal{S}$, contradicting the fact that $j \in Q$. ■ ■

3 Exclusive Graph Searching in Cographs

Lemma 3 *Let $G = G_1 \otimes G_2$ with G_1 and G_2 two cographs. For $i \in \{1,2\}$, let G'_i be equal to (1) G_i if G_i is connected or has no trivial component (2) $G_i \setminus v$ if G_i is not connected and has a unique trivial component $\{v\}$, or (3) $G_i \setminus \{v,w\}$ if G_i is not connected and has at least two trivial components $\{v\}$ and $\{w\}$.*

$$xs(G) = \min\{xs(G'_1) + |V(G_2)|; xs(G'_2) + |V(G_1)|\}$$

Proof. First we show that $xs(G) \leq xs(G'_1) + |V(G_2)|$ by describing a strategy. If $G'_1 = G_1$, simply place $|V(G_2)|$ searchers on the nodes of $V(G_2)$ (these searchers will never move) and use the remaining $xs(G_1)$ searchers to clear the remaining graph. Otherwise, let v be a trivial component of G_1 and w be another trivial component of G_1 (if w exists). The strategy first places searchers at $|V(G_2)| - 1$ nodes of G_2 and at v. The searcher at v then moves to the unoccupied node of G_2. Then, $xs(G'_1)$ searchers are used to clear G'_1. Finally, if w exists, one searcher at some node of G_2 moves to w.

By symmetry, $xs(G) \leq xs(G'_2) + |V(G_1)|$ and, thus, $xs(G) \leq \min\{xs(G'_1) + |V(G_2)|; xs(G'_2) + |V(G_1)|\}$.

It remains to prove that $xs(G) \geq \min\{xs(G'_1) + |V(G_2)|; xs(G'_2) + |V(G_1)|\}$. Let us consider any exclusive strategy $\mathcal{S}$ for G that uses $xs(G)$ searchers.

Consider the first move of $\mathcal{S}$ to be the sliding of a searcher from some node u to some node v. After this step, the node u must not be contaminated since otherwise we could have shorten the strategy by removing the first move (the searcher at u would rather have started at v). Let $i \in \{1,2\}$ such that $u \in V(G_i)$ and let $j \in \{1,2\} \setminus \{i\}$. There are two cases to be considered:

- Either $v \in V(G_i)$. In this case, all nodes of $V(G_j)$ must be initially occupied since otherwise u would have been recontaminated.
- or $v \in V(G_j)$. In that case, all nodes of $V(G_j) \setminus \{v\}$ must be initially occupied since otherwise u would have been recontaminated.

  Moreover, we may assume that u is an isolated node of $V(G_i)$. Indeed, otherwise, let C_u be the connected component of G_i that contains u. Then, all nodes of C_u that are adjacent to u must be initially occupied since otherwise u is recontaminated. Let $x \in V(C_i)$ be such a neighbor of u (it exists since u is not isolated). We could modify $\mathcal{S}$ as follows: instead of occupying initially the node x, then occupy the node v, and replace the first move of $\mathcal{S}$ by the sliding of the searcher at u to x. It is easy to check that the strategy can continue as $\mathcal{S}$ (and that we have not increased the number of searchers).

Therefore, after the first step, all nodes of $V(G_j)$ are occupied.

We claim that, while at least two nodes of $V(G_i)$ are contaminated, no searcher occupying a node in $V(G_j)$ can move. Indeed, otherwise let $x, y \in V(G_i)$ that are contaminated and assume that a searcher leaves $z \in V(G_j)$. Then, z is contaminated by x or y and all unoccupied nodes of $V(G_j)$ are contaminated (because of x or y) and all unoccupied nodes of $V(G_i)$ are contaminated because of z. Therefore, there is a shorter strategy which clears the graph starting from this configuration (the only clear nodes are the occupied ones).

Let $v \in V(G_i)$ to be the last node of G_i to be occupied. By previous paragraph, just before a searcher slides to v, all other nodes of G are clear. If v is not an isolated node of G_i, then, just before being occupied, all its neighbors are occupied (since otherwise v would have recontaminated one of its neighbors). Therefore, we may assume that the last move of $\mathcal{S}$ is to move a searcher from one neighbor of v in G_i to v, while all nodes in $V(G_j)$ are occupied.

To summarize, we have shown that there is an optimal exclusive search strategy $\mathcal{S}$ for G that satisfies the following properties. There is $i \in \{1,2\}$ (let $j \in \{1,2\} \setminus \{i\}$) such that either all nodes of $V(G_j)$ are initially occupied, or the first move of $\mathcal{S}$ is to slide a searcher from a node $u \in V(G_i)$ (isolated in G_i) to the single node of $V(G_j)$ that is initially unoccupied. Then, all nodes of $V(G_j)$ remain occupied either until the end, or until the last step. In the latter case, the last step of $\mathcal{S}$ consists in moving a searcher from some node in $V(G_j)$ to a node v that is isolated in G_i.

Therefore, for any connected component C of G_i (excepted the isolated nodes u and v if they exist), the number of searchers in C remains constant during the whole strategy. Hence, for each such a component C, there must be at least $xs(C)$ searchers used by $\mathcal{S}$ in C during the whole strategy.

All together, we get that $xs(G) \geq \min\{xs(G'_1) + |V(G_2)|; xs(G'_2) + |V(G_1)|\}$. ■ ■

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399