Information and Computation 252 (2017) 243–260

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Exclusive graph searching vs. pathwidth

Euripides Markou [a,*,1], Nicolas Nisse [b,*], Stéphane Pérennes [c]

[a] University of Thessaly, Lamia, Greece
[b] Inria, France and Univ. Nice Sophia Antipolis, CNRS, I3S, Sophia Antipolis, France
[c] Univ. Nice Sophia Antipolis, CNRS, I3S, Sophia Antipolis, France

ARTICLE INFO

Article history:
Received 7 October 2014
Received in revised form 23 October 2016
Available online 2 December 2016

Keywords:
Graph searching
Pathwidth

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.

In this paper, we study the complexity of the *Exclusive Graph Searching* variant. We show that the problem is NP-hard in planar graphs 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.

1. Introduction

In *Graph Searching* [5,18], 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 [11]). 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.

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 [5]. Later on, Parsons formalized Graph Searching as a game to clear contaminated networks [18]. 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. A clear edge e is *recontaminated* as soon as there is a path from e to a contaminated edge without any searcher on it. As an example, to clear a path P, it is sufficient to place a searcher at an end of P and then to slide it until its other end.

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 [15]. In this variant, two searchers are required to clear a path $(v_1, \cdots, v_n)$, $n > 1$: place first a searcher at v_1, then, for $i = 2$ to n, place a searcher at v_i and

* Corresponding authors.
E-mail addresses: emarkou@ucg.gr (E. Markou), Nicolas.Nisse@inria.fr (N. Nisse), stephane.perennes@inria.fr (S. Pérennes).

[1] Part of this work was done while this author was visiting INRIA at Sophia-Antipolis. This research has been co-financed by the European Union (European Social Fund — ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) — Research Funding Program: THALIS-NTUA (MIS 379414).

http://dx.doi.org/10.1016/j.ic.2016.11.007
0890-5401/

remove the searcher at v_{i-1}. Note that, in this variant, it is not possible to clear a path with one searcher since, each time the searcher is removed, the whole path is recontaminated.

Then, Bienstock and Seymour defined the *mixed Graph Searching* [6], in which 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 edge-strategy described above for the path is also a mixed-strategy.

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. For instance, for any n-node path P_n, $n > 1$, $es(P_n) = s(P_n) = 1$ and $ns(P_n) = 2$.

One of the main motivations for studying Graph Searching arises from the fact that it provides an algorithmic interpretation of *path-decompositions* of graphs [20,15]. For the sake of completeness we mention below the definition of *path-decomposition* and *pathwidth* of a graph.

Definition 1. [20] Let $G(V, E)$ be a graph. A sequence $(X_1, \ldots X_r)$, of subsets of $V(G)$ is a *p*ath-decomposition of G if the following conditions hold:

- $\bigcup_{1 \le i \le r} X_i = V(G)$.
- For every edge e of G, there is a X_i, with $1 \le i \le r$, which contains both endpoints of e.
- For each i, j, k, where $1 \le i \le j \le k \le r$, $X_i \cap X_k \subseteq X_j$.

The *w*idth of a path decomposition $(X_1, \ldots X_r)$ is the maximum size of its subsets minus 1, i.e., $\max_{i \le r} |X_i| - 1$. The *p*athwidth of G is the minimum width of its path-decompositions.

The *node-search number* of any graph equals its *pathwidth* plus one [15,6] 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., [13]), 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.

An important property of Graph Searching is the *monotonicity*. A strategy is *monotone* if no edge is ever recontaminated. 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 [6]. 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.

Recently, Blin et al. introduced a new variant, namely *Exclusive Graph Searching*, that appears to be very different from the previous ones [2]. They show that the corresponding optimization problem is polynomial in trees and give some evidence that this new variant of the problem behaves differently than pathwidth. For instance, *Exclusive Graph Searching* is not *monotone* in trees. It is also shown that the search-number in *Exclusive Graph Searching* may differ exponentially from previous variants. However, it equals the pathwidth up to a constant ratio in bounded degree graphs. The complexity of this variant in arbitrary graphs is left open.

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* [2] consists of first placing k searchers at distinct nodes of a connected 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. In this paper, a node is said *clear* if it is occupied by a searcher or if all its incident edges are clear.

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 $n-1$ leafs) consists of: 1) first placing searchers at $n-2$ distinct leafs (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 $\le n-3$ 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. 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, $xs(G) \le mxs(G)$ for any graph G. Note that this inequality may be strict [2]. If $mxs(G) = xs(G)$ for any graph G in some class $\mathcal{C}$ of graphs, Exclusive Graph Searching is said *monotone* in $\mathcal{C}$.

In [2], 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.