Theoretical Computer Science 580 (2015) 101–121

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Non-deterministic graph searching in trees ☆

Omid Amini [a], David Coudert [b,c,*], Nicolas Nisse [b,c]

[a] CNRS – DMA, École Normale Supérieure, Paris, France
[b] Inria, France
[c] Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, 06900 Sophia Antipolis, France

ARTICLE INFO

Article history:
Received 21 October 2013
Received in revised form 12 August 2014
Accepted 24 February 2015
Available online 3 March 2015
Communicated by R. Klasing

Keywords:
Graph searching
Treewidth
Pathwidth
Trees

ABSTRACT

Non-deterministic graph searching was introduced by Fomin et al. to provide a unified approach for pathwidth, treewidth, and their interpretations in terms of graph searching games. Given $q \geq 0$, the q-limited search number, $s_q(G)$, of a graph G is the smallest number of searchers required to capture an invisible fugitive in G, when the searchers are allowed to know the position of the fugitive at most q times. The search parameter $s_0(G)$ corresponds to the pathwidth of a graph G, and $s_\infty(G)$ to its treewidth. Determining $s_q(G)$ is NP-complete for any fixed $q \geq 0$ in general graphs and $s_0(T)$ can be computed in linear time in trees, however the complexity of the problem on trees has been unknown for any $q > 0$.

We introduce a new variant of graph searching called restricted non-deterministic. The corresponding parameter is denoted by rs_q and is shown to be equal to the non-deterministic graph searching parameter s_q for $q = 0, 1$, and at most twice s_q for any $q \geq 2$ (for any graph G).

Our main result is a polynomial time algorithm that computes $rs_q(T)$ for any tree T and any $q \geq 0$. This provides a 2-approximation of $s_q(T)$ for any tree T, and shows that the decision problem associated to s_1 is polynomial in the class of trees. Our proofs are based on a new decomposition technique for trees which might be of independent interest.

1. Introduction

Graph searching problems have been extensively studied for practical aspects such as pursuit-evasion problems [1], but also for their close relationship with fundamental structural parameters of graphs, namely pathwidth and treewidth, that serve as important tools in Robertson and Seymour's Graph Minor Theory [2]. In particular, many intractable problems can be solved in linear time when the input is restricted to graphs of bounded treewidth [3]. In this paper, $tw(G)$ and $pw(G)$ denote the treewidth and the pathwidth of a graph G, respectively.

Graph searching is a game in which a team of searchers is aiming at capturing a fugitive hidden in a graph. The searchers can be placed on or removed from the vertices of the graph. The fugitive stands at some vertex of the graph and can move

☆ Thanks to the anonymous referees for their constructive and helpful comments. This project has been partially supported by GDR ASR ResCom, by ANR project Stint under reference ANR-13-BS02-0007 and by ANR program "Investments for the Future" under reference ANR-11-LABX-0031-01.

* Corresponding author.
E-mail addresses: omid.amini@ens.fr (O. Amini), david.coudert@inria.fr (D. Coudert), nicolas.nisse@inria.fr (N. Nisse).
URLs: http://www.math.ens.fr/~amini/ (O. Amini), http://www-sop.inria.fr/members/David.Coudert/ (D. Coudert), http://www-sop.inria.fr/members/Nicolas.Nisse/ (N. Nisse).

http://dx.doi.org/10.1016/j.tcs.2015.02.038

arbitrary fast from its current vertex to another by following the paths in the graph as long as it does not cross any vertex occupied by a searcher. The fugitive has perfect knowledge about the position and future moves of searchers. The fugitive is caught when it occupies the same vertex as a searcher and has no way to escape. A vertex is *contaminated* if it may harbor the fugitive, and is *cleared* by placing a searcher on it. Once cleared, a vertex remains clear as long as every path from it to a contaminated vertex is guarded by at least one searcher. Otherwise, the vertex is *recontaminated*. The graph is clear as soon as all the vertices are simultaneously clear. Therefore, the fugitive is caught. A *node (search) strategy* is a sequence of searchers moves (place or remove), or *steps*, that guarantees the fugitive's capture. A strategy is *monotone* if no vertex is visited more than once by a searcher, i.e., if *recontamination* never occurs.

Two main variants of graph searching have been particularly studied: either the fugitive is *invisible*, meaning that the searchers do not know its position unless it is caught, or it is *visible*, i.e., at any step of the strategy, the searchers know the current position of the fugitive and they can thus adapt their strategy according to this knowledge. The *node search number* $s(G)$ (resp., the *visible search number* $vs(G)$) of a graph G is the minimum number of searchers for which a strategy capturing an invisible (resp., visible) fugitive exists for G [4,5]. One important result of the field is that *recontamination does not help*. That is, for any graph G, there is a monotone strategy using the optimal number of searchers to capture an invisible (resp., visible) fugitive in G [4,5]. In particular, it follows that the node search number and the visible search number of a graph are closely related to its pathwidth and treewidth, namely, for any graph G, $s(G) = pw(G) + 1$ and $vs(G) = tw(G) + 1$ (see [6] for a survey on graph searching).

In [7], Fomin et al. introduced a parametric variant called *non-deterministic graph searching*, and proved that the corresponding parameter establishes a link between invisible and visible search numbers, i.e., between pathwidth and treewidth. They proved that computing this parameter is NP-hard in general and asked whether it can be computed in polynomial time when the input is restricted to be a tree. In this paper, we study this latter problem.

In non-deterministic graph searching, the fugitive is invisible but the searchers have the possibility to query an oracle that knows the current position of the fugitive (a limited number of times). That is, given the set W of clear vertices, *performing a query* returns a connected component C of $G \setminus W$. The vertices of C remain contaminated and those of $G \setminus C$ become clear. Obviously, the number of searchers required to catch the fugitive cannot increase when the number of permitted performing-a-query steps increases.

A *non-deterministic (search) strategy* is a sequence of the three basic operations:

- Placing a searcher on a vertex,
- Removing a searcher from a vertex, and
- Performing a query.

Note that such a strategy corresponds to a decision tree so that the performing-a-query steps correspond to the forks in the decision-tree. A possible execution of this strategy is a sequence of such operations following a path of the decision-tree from its root to a leaf, corresponding to some choice for any query step, i.e., depending on the behavior of the fugitive. The strategy must result in catching the fugitive whatever it does. The number of query-steps, denoted by $q \geq 0$, is however fixed. The *q-limited search number* of a graph G, $s_q(G)$, is the smallest number of searchers required to catch a fugitive performing at most q query-steps. Mazoit and Nisse [8] generalized the monotonicity results of [4] and [5]. They proved that *recontamination does not help* neither in non-deterministic case: for any $q \geq 0$ and any graph G, there is a monotone strategy performing at most q queries that uses at most $s_q(G)$ searchers [8]. Hence, throughout this paper, we consider only monotone strategies. We moreover assume that useless moves such as placing a searcher on a clear or occupied node never occur.

The monotonicity result is also important because monotone non-deterministic graph searching realizes a link between treewidth and pathwidth through the notion of *q-branched tree decompositions* [7]. The definition of q-branched treewidth and its relationship with the q-limited search number are as follows.

Given a rooted tree $\mathfrak{T}$, with root $\mathfrak{r}$, a *branching node* of $\mathfrak{T}$ is a node with at least two children. Let $q \geq 0$. A *q-branched tree* $\mathfrak{T}$ is a rooted tree such that every path in $\mathfrak{T}$ from (root) $\mathfrak{r}$ to a leaf contains at most q branching nodes.

Let $G = (V, E)$ be a connected graph and let $q \geq 0$. A *q-branched tree decomposition* [7] of a graph G is a pair $(\mathfrak{T}, \mathcal{X})$ where $\mathfrak{T}$ is a q-branched tree on a set of nodes $\mathfrak{I}$, and $\mathcal{X} = \{X_i : i \in \mathfrak{I}\}$ is a collection of subsets of V, subject to the following three conditions:

1. $V = \cup_{i \in \mathfrak{I}} X_i$,
2. for any edge e in G, there is a set $X_i \in \mathcal{X}$ which contains both end-points of e,
3. for any triple i_1, i_2, i_3 of nodes of $\mathfrak{T}$, if i_2 is on the path from i_1 to i_3 in $\mathfrak{T}$, then $X_{i_1} \cap X_{i_3} \subseteq X_{i_2}$.

The *width* of $(\mathfrak{T}, \mathcal{X})$ is defined as $w(\mathfrak{T}, \mathcal{X}) = \max_{i \in \mathfrak{I}} |X_i| - 1$. The *q-branched treewidth* of a graph G, denoted by $tw_q(G)$, is the minimum width of any q-branched tree decomposition of G. Note that $tw_{q'}(G) \leq tw_q(G)$ for any $q \leq q'$. Obviously, for q large enough, $tw_q(G) = tw(G)$, where $tw(G)$ denotes the treewidth of G. In other word, $tw(G) = \min_{q \geq 0} tw_q(G) =: tw_\infty(G)$. Moreover, $tw_0(G) = pw(G)$, where $pw(G)$ denotes the pathwidth of G. In this way, the family of parameters $tw_q(G)$ can be regarded as an interpolating family of parameters between the pathwidth and the treewidth a graph G. The main theorem of [7] and the monotonicity result of [8] establish the link between q-limited search number and q-branched treewidth.