Stratégies d'encerclement connexes dans un réseau

Pierre Fraigniaud[†] and Nicolas Nisse[‡]

CNRS
Laboratoire de Recherche en Informatique
Université Paris Sud
91405 ORSAY, France
{pierre,nisse}@lri.fr

Le problème de l'encerclement dans les réseaux a été introduit par Parson (1976) : étant donné un réseau "contaminé" (par exemple dans lequel un intrus s'est introduit), l'*encerclement* du réseau est le nombre minimum d'agents nécessaires pour "nettoyer" le réseau (c'est-à-dire capturer l'intrus). Une stratégie d'encerclement est dite connexe si à chaque étape de la stratégie, l'ensemble des liens nettoyés induit un sous-réseau connexe. Les stratégies d'encerclement connexes sont essentielles si l'on souhaite assurer des communications sûres entre les agents. Dans le cas des réseaux en arbres, Barrière *et al.* (2002, 2003) ont prouvé que le rapport entre l'encerclement connexe et l'encerclement est majoré par 2, et que cette borne est optimale. Dans cet article, nous donnons une borne pour ce rapport dans le cas des réseaux arbitraires. Pour cela nous utilisons une notion cruciale de théorie des graphes : la largeur arborescente. L'égalité entre la largeur arborescente connexe d'un graphe et sa largeur arborescente découle du théorème de Parra et Scheffler (1995). Nous donnons ici une preuve constructive de cette égalité. Plus précisemment, nous proposons un algorithme qui étant donnés un graphe G de n sommets et une décomposition arborescente de largeur k de G, calcule en temps $O(n\,k^3)$ une décomposition arborescente connexe de largeur $\leq k$ de G. Une conséquence importante de notre résultat est qu'il permet de borner par $\lceil \log n \rceil + 1$ le rapport entre encerclement connexe et encerclement d'un réseau de n nœuds.

Keywords: Décomposition arborescente, Décomposition en branches, Décomposition linéaire, Stratégie d'encerclement

1 Introduction

In *graph searching* (see, e.g.,[3, 4, 8, 12]), a fugitive is hidden in a graph G. A team of *searchers* is aiming at capturing this fugitive. The fugitive is assumed to be arbitrary fast, and permanently aware of the positions of the searchers. To capture the fugitive, three operations are allowed for the searchers. These basic operations, called *search steps*, are the following:

- **1:** place a searcher on a vertex,
- **2:** remove a searcher from a vertex,
- **3:** move a searcher along an edge.

There are several variants of the problem of graph searching. To clear an edge $e = \{u, v\}$ in the *edge-search* variant, a searcher must traverse this edge from one end u to v. The edge e is preserved from recontamination if either another searcher remains in u, or all edges incident to e in u are *cleared*. That is, a *cleared* edge e is recontaminated if there exists a path between e and a contaminated edge, with no searcher

[†]Additional supports from the INRIA Project "Grand Large", and from the Project PAIRAPAIR of the ACI "Masse de Données".
[‡]Additional supports from the Project FRAGILE of the ACI "Sécurité & Informatique".

on any vertex of this path. An edge-search strategy is a sequence of search steps that capture the fugitive (i.e. a strategy that completly clears the graph). The graph searching problem asks for the design of search strategies using a *minimum number of searchers*. The mimimal number of searchers for which there exists a strategy in a graph G is the *search number* $\mathbf{s}(G)$ of G. The search-number varies depending on the relative power of the fugitive and the searchers.

— If the searchers are permanently aware of the position of the fugitive, then the optimal size of the team is essentially the treewidth of the graph [3]. The *treewidth* of a graph is a central concept in the theory of Graph Minors developed by Robertson and Seymour (see, e.g., [10]). Roughly speaking, the treewidth, $\mathbf{tw}(G)$, of a graph G measures “how far” the graph G is from a tree. More formally, a *tree-decomposition* of a graph G is a pair (T,X) where T is a tree, and $X = \{X_v, v \in V(T)\}$ is a collection of subsets of $V(G)$ satisfying the following three conditions:

- **C1:** $V(G) = \cup_{v\in V(T)} X_v$;
- **C2:** For any edge e of G, there is a set X_v such that both end-points of e are in X_v;
- **C3:** For any triple u, v, w of nodes in $V(T)$, if v is on the path from u to w in T, then $X_u \cap X_w \subseteq X_v$.

Condition **C3** can be rephrased as: for any node x of G, $\{v \in V(T) \mid x \in X_v\}$ is a subtree of T. The sets X_v's are often called *bags*. The *width*, $\omega(T,X)$, of a tree-decomposition (T,X) is defined as $\max_{v\in V(T)} |X_v| - 1$, i.e., the width of (T,X) is roughly the maximum size of its bags. The treewidth $\mathbf{tw}(G)$ is defined as $\min \omega(T,X)$ where the minimum is taken over all tree-decompositions (T,X) of G. Beside its own interest, the treewidth has several important applications. In particular, it is known that several NP-hard problems can be solved in polynomial time if instances are restricted to graphs of bounded treewidth. Actually, on graphs of treewidth at most k, where k is fixed, every decision or optimization problem expressible in monadic second-order logic has a linear algorithm [5].

— If the searchers are unaware of the position of the fugitive, then the optimal size of the team, $\mathbf{s}(G)$, is essentially the *pathwidth* of G [3]. A path-decomposition of G is a tree-decomposition (T,X) of G, where T is a path. The pathwidth $\mathbf{pw}(G)$ is defined as $\min \omega(T,X)$ where the minimum is taken over all path-decompositions (T,X) of G. For any graph G, we have [3]: $\mathbf{pw}(G) \leq \mathbf{s}(G) \leq \mathbf{pw}(G) + 2$.

2 Our Objective

It has been argued (cf., e.g., [1, 2] and the references therein) that several practical applications (e.g., network security, speleological rescue, etc.) requires the search strategy be *connected*, i.e., at any time of the search strategy, the portion of the searched graph is a connected subgraph. In particular, this property insures safe and secure communications among the searchers across the cleared area.

Formally, a search strategy is *connected* if the set of *cleared* edges induces a connected subgraph at every step of the search. Another way to define such strategies is not to allow operation **2**, and to force all searchers to start from the same vertex. Assuming that the searchers are unaware of the position of the fugitive, the *connected* search number $\mathbf{cs}(G)$ of the graph G is the minimum number of searchers for which a connected search strategy exists for G. In [2], Barrière *et al.* showed that the non-connectness helps. In other words, there exists graphs for which optimal connected search strategy requires more searchers than optimal search strategy. Nevertheless, [2] proved that, for any tree T,

$$\mathbf{cs}(T) \leq 2\,\mathbf{s}(T) - 2,$$

and that this bound is tight.

Using the concept of connected branchwidth, Fomin *et al.* [6] have shown that, for any connected m-edge graph G, the *connected* search number, $\mathbf{cs}(G)$, satisfies

$$\mathbf{cs}(G)/\mathbf{s}(G) \leq \lceil \log m \rceil + 1. \tag{1}$$

However, this bound has not been proved to be tight, and it is conjectured that

$$\mathbf{cs}(G)/\mathbf{s}(G) \leq 2$$

for any connected graph G. Our objective is therefore to improve the bound of Eq. 1. For that purpose, we have revisited the several notions of connectedness for graph decompositions.

3 Connectedness of graph decomposition

Connectedness has been defined directly on many graph decompositions. In particular, a *branch-decomposition* of a graph G is a tree T whose all internal nodes have degree 3, with a one-to-one correspondence between the leaves of T and the edges of G. Given an edge e of T, removing e from T results in two trees $T_1^{(e)}$ and $T_2^{(e)}$, and an *e-cut* is defined as the pair $\{E_1^{(e)}, E_2^{(e)}\}$, where $E_i^{(e)} \subset E(G)$ is the set of leaves of $T_i^{(e)}$ for $i = 1,2$. The width of T is defined as $\omega(T) = \max_e |\delta(E_1^{(e)})|$ where the maximum is taken over all e-cuts in T, and where, for any edge-set E, $\delta(E)$ denotes the set of nodes with one extremity in E and the other in $E(G) \setminus E$. The branchwidth $\mathbf{bw}(G)$ of G is then $\min \omega(T)$ where the minimum is taken over all branch-decompositions T of G. A branch-decomposition is *connected* if, for any of its e-cut, the two subgraphs of G induced by $E_1^{(e)}$ and $E_2^{(e)}$ are connected. A major result about branchwidth is that if a 2-edge-connected G has a branch-decomposition of width k, then it has a connected branch-decomposition of width $\leq k$ [11]. Therefore, for any 2-edge-connected graph G, there is equality between its connected branchwidth, $\mathbf{cbw}(G)$, and its branchwidth, $\mathbf{bw}(G)$. This equality is the argument used to derive Eq. 1.

We address the connectivity constraint directly on tree-decompositions (cf. Section 1 for the definition). An e-cut of a tree-decomposition (T,X) of a graph G is defined as the pair $\{X_1^{(e)}, X_2^{(e)}\}$, where $X_i^{(e)} \subseteq V(G)$ is the set of nodes in $\{X_v, v \in V(T_i^{(e)})\}$ for $i = 1,2$. A tree-decomposition is connected if, for any of its e-cuts, the two subgraphs $G[T_1^{(e)}]$ and $G[T_2^{(e)}]$ of G, induced by $X_1^{(e)}$ and $X_2^{(e)}$, respectively, are connected. The connected treewidth, $\mathbf{ctw}(G)$, of a connected graph G, is defined as the minimum width of any connected tree-decomposition of G. It is well known [7] that a clique tree (T,X) of a minimal triangulation H of a connected graphe G is a tree decomposition with width $\mathbf{tw}(G)$ of G. Moreover, Parra and Scheffler proved in [9], that the set Δ_H of minimal separators of H is exactly the set of pairwise parallel minimal separators in G and that for any $S \in \Delta_H$, S induces the same connected components in H and G. This implies that (T,X) is *connected*, and thus we get that

$$\mathbf{tw}(G) = \mathbf{ctw}(G) \tag{2}$$

for any connected graph G. This result extends to treewidth the result on carvings (and therefore branchwidth) in [11]. Importantly, the equality $\mathbf{ctw} = \mathbf{tw}$ holds for any connected graph, whereas the equality $\mathbf{cbw} = \mathbf{bw}$ holds for 2-edge-connected graphs only.

4 Our Results

We first give a constructive proof for the equality $\mathbf{ctw}(G) = \mathbf{tw}(G)$, for any connected graph G. This equality between treewidth and connected treewidth is obtained via the design of a polynomial-time algorithm transforming a tree-decomposition into a connected tree-decomposition of same width. More precisely, we prove the following:

Theorem 1 *There exists a $O(Nk^3)$-time algorithm that, given any N-node tree-decomposition of a connected graph G, of width k, returns a connected tree-decomposition of G, of width $\leq k$.*

The algorithm of theorem 1 is linear in the case of graphs with bounded treewidth. A consequence of the equality between treewidth and connected treewidth is that we propose a polynomial-time algorithm which, given a connected graph G and a tree decomposition with width $\mathbf{tw}(G)$ of G, computes a connected search strategy using at most $\mathbf{s}(G)$ $(\lceil \log n \rceil + 1)$ searchers, thus improving [6]. More precisely, we proved the following:

Theorem 2 *For any connected n-node graph G, $\frac{\mathbf{cs}(G)}{\mathbf{s}(G)} \leq \lceil \log n \rceil + 1$.*

Sketch of the Proof. Let G be an n-node connected graph, and let (T,X) be an optimal connected tree-decomposition of G, i.e., of width $\mathbf{tw}(G)$. We use a result in [10] to show that, for any tree-decomposition (T,X) of an n-node graph G, there exists a vertex $v \in V(T)$ (or an edge $e = (u,v) \in E(T)$) such that removing v (or u and v) from T results in a forest $T_1, \cdots, T_r$, such that for every $i = 1 \cdots r$, the subgraph G_i induced by $\bigcup_{v \in T_i} X_v$ has at most $n/2$ vertices. Using this result, we prove, by induction on n, that $\mathbf{cs}(G) \leq 1 + \mathbf{tw}(G)\lceil \log n \rceil$. Then, since $\mathbf{s}(G) \geq \mathbf{pw}(G) \geq \mathbf{tw}(G) = \mathbf{ctw}(G)$, we get $\mathbf{cs}(G) \leq 1 + \mathbf{s}(G)\lceil \log n \rceil$, and thus $\mathbf{cs}(G)/\mathbf{s}(G) \leq 1 + \lceil \log n \rceil$. □

5 Conclusion and further works

Barrière *et al.* proved in [2], that for any tree T, $\mathbf{cs}(T) \leq 2\ \mathbf{s}(T) - 2$. In this paper, we proved that $\mathbf{cs}(G)/\mathbf{s}(G) = O(\log n)$ for any n-nodes connected graph G. However, we conjecture that there exists a constant k such that, for any connected graph G, $\mathbf{cs}(G)/\mathbf{s}(G) \leq k$. In fact, we even conjecture that $k = 2$:

Conjecture: For any connected graph G, $\mathbf{cs}(G)/\mathbf{s}(G) \leq 2$.

Obviously, one can generalize the definition of connected tree-decomposition to k-connected tree-decomposition, for any $k \geq 1$. In this case, it is required that every e-cut of the decomposition induces a k-connected graph. The k-connected treewidth $\mathbf{ctw}_k(G)$ of a k-connected graph G is the smallest w for which there is a k-connected tree-decomposition of G, of width w. One can show that, for any $k \geq 2$, there exists a k-connected n-node graph G such that $\mathbf{ctw}_k(G)/\mathbf{tw}(G) \geq \Omega(n)$. Hence Eq. 2 cannot be trivially generalized to connectivities greater than 1. We however ask the following question:

Open problem: Is there a function f such that $\mathbf{ctw}_k(G) = \mathbf{tw}(G)$ for any $f(k)$-connected graph G?

References

[1] L. Barrière, P. Flocchini, P. Fraigniaud, and N. Santoro. Capture of an intruder by mobile agents. In 14th ACM Symp. on Parallel Algorithms and Architectures (SPAA), pages 200-209, 2002.

[2] L. Barrière, P. Fraigniaud, N. Santoro, and D. Thilikos. Connected and Internal Graph Searching. In 29th Workshop on Graph Theoretic Concepts in Computer Science (WG), Springer-Verlag, LNCS 2880, pages 34–45, 2003.

[3] D. Bienstock. Graph searching, path-width, tree-width and related problems (a survey). DIMACS Series in Discrete Mathematics and Theoretical Computer Science 5, pages 33-49, 1991.

[4] D. Bienstock and P. Seymour. Monotonicity in graph searching. Journal of Algorithms 12, pages 239-245, 1991.

[5] B. Courcelle, J. Makowsky, and U. Rotics. Linear-time solvable optimization problems on graphs of bounded cliquewidth. In 24th Workshop on Graph-Theoretic Concepts in Computer Science (WG), LNCS 1517, pages 1-16, 1998.

[6] F. Fomin, P. Fraigniaud, D. Thilikos. The Price of Connectedness in Expansions. Technical Report LSI-04-28-R, UPC Barcelona, 2004.

[7] M. C. Golumbic. Algorithmic graph theory and perfect graphs. Computer Science and Applied Mathematics, 1980.

[8] T. Parson. Pursuit-evasion in a graph. Theory and Applications of Graphs, Lecture Notes in Mathematics, Springer-Verlag, pages 426-441, 1976.

[9] A. Parra and P. Scheffler. Characterizations and algorithmic applications of chordal graphe embeddings. Discrete Applied Mathematics 79, pages 171-188, 1997.

[10] N. Robertson and P. D. Seymour. Graph minors II, Algorithmic Aspects of Tree-Width. Journal of Algorithms 7, pages 309-322, 1986.

[11] P. Seymour and R. Thomas. Call routing and the rat-catcher. Combinatorica 14(2), pages 217–241, 1994.

[12] K. Skodinis. Computing optimal linear layouts of trees in linear time. In 8th European Symp. on Algorithms (ESA), Springer-Verlag, LNCS 1879, pages 403-414, 2000.