Theoretical Computer Science 412 (2011) 6340–6350

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Parameterizing cut sets in a graph by the number of their components[☆]

Takehiro Ito [a], Marcin Kamiński [b], Daniël Paulusma [c,*], Dimitrios M. Thilikos [d]

[a] Graduate School of Information Sciences, Tohoku University, Aoba-yama 6-6-05, Sendai, 980-8579, Japan

[b] Computer Science Department, Université Libre de Bruxelles, Boulevard du Triomphe CP212, B-1050 Brussels, Belgium

[c] Department of Computer Science, University of Durham, Science Laboratories, South Road, Durham DH1 3LE, England, United Kingdom

[d] Department of Mathematics, National and Kapodistrian University of Athens, Panepistimioupolis, GR15784 Athens, Greece

ARTICLE INFO

Article history:
Received 17 December 2010
Received in revised form 30 June 2011
Accepted 12 July 2011
Communicated by M. Kiwi

Keywords:
Cut set
2K2-partition
Graph contractibility

ABSTRACT

For a connected graph $G = (V, E)$, a subset $U \subseteq V$ is a disconnected cut if U disconnects G and the subgraph $G[U]$ induced by U is disconnected as well. A cut U is a k-cut if $G[U]$ contains exactly $k(\geq 1)$ components. More specifically, a k-cut U is a (k, ℓ)-cut if $V \setminus U$ induces a subgraph with exactly $\ell(\geq 2)$ components. The Disconnected Cut problem is to test whether a graph has a disconnected cut and is known to be NP-complete. The problems k-Cut and (k, ℓ)-Cut are to test whether a graph has a k-cut or (k, ℓ)-cut, respectively. By pinpointing a close relationship to graph contractibility problems we show that (k, ℓ)-Cut is in P for $k = 1$ and any fixed constant $\ell \geq 2$, while it is NP-complete for any fixed pair $k, \ell \geq 2$. We then prove that k-Cut is in P for $k = 1$ and NP-complete for any fixed $k \geq 2$. On the other hand, for every fixed integer $g \geq 0$, we present an FPT algorithm that solves (k, ℓ)-Cut on graphs of Euler genus at most g when parameterized by $k + \ell$. By modifying this algorithm we can also show that k-Cut is in FPT for this graph class when parameterized by k. Finally, we show that Disconnected Cut is solvable in polynomial time for minor-closed classes of graphs excluding some apex graph.

1. Introduction

Graph connectivity is a fundamental graph-theoretic property that is well-studied in the context of network robustness. In the literature several measures for graph connectivity are known, such as requiring hamiltonicity, edge-disjoint spanning trees, or edge- or vertex-cuts of sufficiently large size. Here, we study the problem of finding a vertex-cut, called a "disconnected cut" of a graph, such that the cut itself is disconnected. As we shall see in Section 3, this problem is strongly related to several other graph problems such as biclique vertex-covers. We give all further motivation later and first state our problem setting.

Let $G = (V, E)$ be a connected simple graph. For a subset $U \subseteq V$, we denote by $G[U]$ the subgraph of G induced by U. We say that U is a *cut* of G if U disconnects G, that is, $G[V \setminus U]$ contains at least two components. A cut U is *connected* if $G[U]$ contains exactly one component, and *disconnected* if $G[U]$ contains at least two components. We observe that $G[U]$ is a disconnected cut if and only if $G[V \setminus U]$ is a disconnected cut. In Fig. 1, the subset $V_1 \cup V_3$ is a disconnected cut of G, and hence its complement $V_2 \cup V_4$ $\left(= V \setminus (V_1 \cup V_3)\right)$ is also a disconnected cut of G. This leads to the decision problem Disconnected Cut which asks if a given connected graph has a disconnected cut.

☆ An extended abstract of this paper has been presented at the 20th International Symposium on Algorithms and Computation (ISAAC 2009) [14].

* Corresponding author. Tel.: +44 1913341723.
E-mail addresses: takehiro@ecei.tohoku.ac.jp (T. Ito), marcin.kaminski@ulb.ac.be (M. Kamiński), daniel.paulusma@durham.ac.uk (D. Paulusma), sedthilk@math.uoa.gr (D.M. Thilikos).

0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.07.005

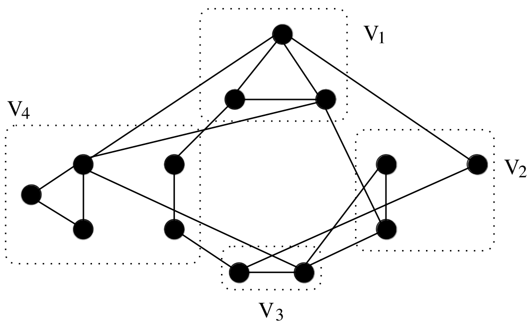

Fig. 1. A graph G with a disconnected cut $V_1 \cup V_3$ that is also a 2-cut and a (2, 4)-cut and a disconnected cut $V_2 \cup V_4$ that is also a 4-cut and a (4, 2)-cut.

Recently, Disconnected Cut has been shown to be NP-complete [15]. However, the problem can be solved in polynomial time for some restricted graph classes, as in the following theorem, which we will use in the proofs of some of our results. In particular, we mention that every graph of diameter at least three has a disconnected cut [11].

Theorem 1 ([11]). *The* Disconnected Cut *problem is solvable in polynomial time for the following classes of connected graphs*:

(i) *graphs of diameter not equal to two*;
(ii) *graphs with bounded maximum vertex degree*;
(iii) *graphs that are not locally connected*;
(iv) *triangle-free graphs*; *and*
(v) *graphs with a dominating edge* (*including cographs*).

Besides Disconnected Cut, we study two closely related problems in which we wish to find a cut having a prespecified number of components. For a fixed constant $k \geq 1$, a cut U of a connected graph G is called a k-*cut* of G if $G[U]$ contains exactly k components. Furthermore, for a pair (k, ℓ) of fixed constants $k \geq 1$ and $\ell \geq 2$, a k-cut U is called a (k, ℓ)-*cut* of G if $G[V \backslash U]$ consists of exactly ℓ components. Note that a k-cut and a (k, ℓ)-cut are connected cuts if $k = 1$; otherwise (when $k \geq 2$) they are disconnected cuts. It is obvious that, for a fixed pair $k, \ell \geq 2$, a (k, ℓ)-cut U of G corresponds to an (ℓ, k)-cut $V \setminus U$ of G. For example, the disconnected cut $V_1 \cup V_3$ in Fig. 1 is a 2-cut and a (2, 4)-cut, while its complement $V_2 \cup V_4$ is a 4-cut and a (4, 2)-cut. In this paper, we study the following two decision problems, where k and ℓ are fixed, *i.e.*, not part of the input. The k-Cut problem asks if a given connected graph has a k-cut. The (k, ℓ)-Cut problem asks if a given connected graph has a (k, ℓ)-cut.

Our results and the paper organization. Our three main results are as follows. First, we show that Disconnected Cut is strongly related to several other graph problems. In this way we determine the computational complexity of (k, ℓ)-Cut. Second, we determine the computational complexity of k-Cut. Third, for every fixed integer $g \geq 0$, we give an FPT algorithm that solves (k, ℓ)-cut for graphs of Euler genus at most g when parameterized by $k + \ell$. In the following, we explain our results in detail.

In Section 2 we define our terminology. Section 3 contains our first result. We state our motivation for studying these three types of cut problems. We then pinpoint relationships to other cut problems, and to graph homomorphism, biclique vertex-cover and vertex coloring problems. We show a strong connection to graph contractibility problems. In this way we prove that (k, ℓ)-Cut is solvable in polynomial time for $k = 1$, $\ell \geq 2$, and is NP-complete otherwise.

Section 4 gives our second result: we classify the computational complexity of k-Cut. We show that k-Cut is solvable in polynomial time for $k = 1$, while it becomes NP-complete for every fixed constant $k \geq 2$. Note that the NP-completeness of (k, ℓ)-Cut, shown in Section 3, does not imply this result, because ℓ is fixed and the subgraph obtained after removing a (k, ℓ)-cut must consist of *exactly* ℓ components.

In Section 5 we present our third result: an FPT algorithm that solves (k, ℓ)-Cut for graphs on surfaces when parameterized by $k + \ell$. We also show that k-Cut is FPT in k for graphs on surfaces and that Disconnected Cut is solvable in polynomial time for this class of graphs.

In Section 6 we state some further results and mention a number of open problems that are related to some other well-known graph classes, namely chordal, claw-free and line graphs.

2. Preliminaries

Without loss of generality, the graphs we consider are undirected and without multiple edges. Unless explicitly stated otherwise, they do not contain loops either. For undefined (standard) graph terminology we refer to Diestel [8].

Let $G = (V, E)$ be a graph. The vertex set V and the edge set E of G are often denoted by V_G and E_G, respectively. Each maximal connected subgraph of G is called a *component* of G. Let $N(u)$ denote the *neighborhood* of a vertex $u \in V$, that is, $N(u) = \{v \mid uv \in E\}$. Two disjoint nonempty subsets $U, U' \subset V$ are *adjacent* if there exist vertices $u \in U$ and $u' \in U'$ with