Discrete Applied Mathematics 187 (2015) 70–81

Contents lists available at ScienceDirect

Discrete Applied Mathematics

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

# A characterisation of clique-width through nested partitions

Bruno Courcelle [a], Pinar Heggernes [b], Daniel Meister [c], Charis Papadopoulos [d,*], Udi Rotics [e]

[a] LaBRI, CNRS and Bordeaux University, 351 Cours de la Libération, 33405 Talence, France

[b] Department of Informatics, University of Bergen, Norway

[c] Theoretical Computer Science, University of Trier, Germany

[d] Department of Mathematics, University of Ioannina, Greece

[e] Netanya Academic College, Netanya, Israel

ARTICLE INFO

*Article history:*
Received 9 July 2014
Received in revised form 6 February 2015
Accepted 17 February 2015
Available online 17 March 2015

*Keywords:*
Clique-width
Linear clique-width
Characterisation
Efficient representation

ABSTRACT

Clique-width of graphs is defined algebraically through operations on graphs with vertex labels. We characterise the clique-width in a combinatorial way by means of partitions of the vertex set, using trees of nested partitions where partitions are ordered bottom-up by refinement. We show that the correspondences in both directions, between combinatorial partition trees and algebraic terms, preserve the tree structures and that they are computable in polynomial time. We apply our characterisation to linear clique-width. And we relate our characterisation to a clique-width characterisation by Heule and Szeider that is used to reduce the clique-width decision problem to a satisfiability problem.



## 1. Introduction

Hierarchical graph decompositions are useful for the design of efficient graph algorithms. This usefulness is the foundation of the theory of fixed-parameter tractability, that emerged from the large body of graph algorithms working on tree decompositions [8]. There are many graph decompositions with interesting algorithmic applications, such as modular decomposition [12], tree decomposition [8,10], and rank decomposition [19]. In most cases, a *width* notion is associated with a graph decomposition, and the width of a graph is the minimum achievable width for the graph using this decomposition. Fixed-parameter tractability results are often of the form: a graph problem is tractable on graphs of bounded width, and the width of the input graph is a measure for the complexity of determining the solution.

Modular decomposition, tree decomposition and rank decomposition can be defined combinatorially. Tree decomposition and rank decomposition can be defined also algebraically [3,5]. An algebraic definition is built on operations from an appropriate graph algebra. Algebraic descriptions yield clean definitions of finite automata, that themselves can be seen as abstract descriptions of efficient parametrised algorithms (see [2] or Chapter 6 of [3] for a deeper treatment of the matter).

Clique-width is a graph complexity measure [3,4,7]. The original definition of clique-width is by means of algebraic terms. These terms use operations that create edgeless graphs, that add edges between specified vertices, and that combine graphs. Clique-width is a practically useful graph measure, because the results by Courcelle et al. in [6] in combination with the results by Oum and Seymour in [19] show that the model-checking problem for monadic second-order logic on graphs

* Corresponding author.
*E-mail addresses:* courcell@labri.fr (B. Courcelle), pinar.heggernes@ii.uib.no (P. Heggernes), daniel.meister@uni-trier.de (D. Meister), charis@cs.uoi.gr (C. Papadopoulos), rotics@netanya.ac.il (U. Rotics).

http://dx.doi.org/10.1016/j.dam.2015.02.016
0166-218X/

is fixed-parameter tractable when choosing the clique-width of the input graph as the parameter (see the monographs by Downey and Fellows [8] and by Flum and Grohe [10] for a deep and comprehensive consideration of fixed-parameter tractability).

The algebraic definition of clique-width, the same as similar characterisations of treewidth and rank-width, use vertex labels to specify classes of vertices. We can say that vertex labels express a "neighbourhood similarity" relation, since vertices with the same label have the same "one-sided" neighbourhood. The clique-width of a graph is the smallest number of necessary different labels in a clique-width expression. Computing the clique-width of a graph is hard [9], and the problem of choosing appropriate labels for vertices is a strong contribution to the hardness of the problem [18]. Combinatorial characterisations avoid labels and the necessity of choosing labels.

In this paper, we give a purely combinatorial characterisation of clique-width. Our characterisation uses rooted trees and nested partitions of sets of vertices. The partitions are ordered by the refinement relation, and they satisfy an adjacency condition. The resulting notion is *partition tree* for graphs. With partition trees, we associate a *width* notion, that measures the size of the nested partitions. In Section 3, we introduce partition trees and describe their relationship to clique-width. As the main result of this paper, we obtain a characterisation of clique-width by partition trees, that shows the equivalence between the clique-width of a graph and the width of its partition trees. Our results also show and provide efficient transformation algorithms between algebraic clique-width expressions and partition trees, and we provide a space-efficient representation of partition trees, that is of interest for efficient algorithms. Our combinatorial characterisation of clique-width is based on, combines, and extends combinatorial clique-width characterisations by Heggernes et al. [15] and Heule and Szeider [16].

Tree-based graph decompositions have linear variants: pathwidth is the linear variant of treewidth, linear rank-width is the linear variant of rank-width, and linear clique-width is the linear variant of clique-width. The linear variants are obtained when restricting the decomposition tree to a path-like tree, a *caterpillar*, more precisely. In Section 4, we apply our clique-width characterisation result to linear clique-width, obtaining an analogue combinatorial characterisation of linear clique-width. We also study combinatorial properties of our partition trees, that provide an alternative approach to corresponding known results.

Finally, in Section 5, we review the clique-width characterisation by Heule and Szeider [16]. This characterisation is based on a sequence of pairs of partitions of the vertex set of graphs. We relate this partition-sequence characterisation to our partition-tree characterisation, show the equivalence of the two characterisations, and also provide efficient transformation algorithms.

## 2. Graph preliminaries and clique-width

*Graph preliminaries*. The studied graphs in this paper are simple, finite, undirected. In addition, we use directed graphs and rooted trees for the representation of information about undirected graphs. All definitions, if not otherwise said, are for undirected graphs. For rooted trees and directed graphs, we use only standard terminology, that we do not always define explicitly.

A graph $G$ is an ordered pair $(V, E)$ where $V = V(G)$ is the *vertex set* and $E = E(G)$ is the *edge set* of $G$. Edges are denoted as $uv$. Let $u, v$ be a vertex pair of $G$ with $u \neq v$. If $uv$ is an edge of $G$ then $u$ and $v$ are *adjacent* in $G$, and $u$ is a *neighbour* of $v$ in $G$; if $uv$ is not an edge of $G$ then $u$ and $v$ are *non-adjacent* in $G$. The *neighbourhood* of $u$ in $G$, $N_G(u)$, is the set of neighbours of $u$ in $G$. A graph $H$ is a *subgraph* of $G$ if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. For $X \subseteq V(G)$, the subgraph of $G$ *induced* by $X$, $G[X]$, has $X$ as its vertex set, and for each vertex pair $u, v$ from $X$, $uv$ is an edge of $G[X]$ if and only if $u \neq v$ and $uv$ is an edge of $G$. For $R \subseteq E(G)$, $G \setminus R$ denotes the graph $(V(G), E(G) \setminus R)$, that is a subgraph of $G$. For $X, Y \subseteq V(G)$, where $X \cap Y = \emptyset$, we denote by $X \times Y$ the set of all possible edges between vertices in $X$ and vertices in $Y$, that is $X \times Y = \{xy : x \in X \text{ and } y \in Y\}$.

An important graph operation throughout the paper is the disjoint union of graphs. Let $H_1, \ldots, H_p$, where $p \geq 2$, be pairwise vertex-disjoint graphs, which means $V(H_i) \cap V(H_j) = \emptyset$ for every $1 \leq i < j \leq p$. The *disjoint union* of $H_1, \ldots, H_p$, denoted as $\bigoplus(H_1, \ldots, H_p)$, is the graph $\big(V(H_1) \cup \cdots \cup V(H_p), E(H_1) \cup \cdots \cup E(H_p)\big)$.

*Clique-width and linear clique-width.* Let $k \geq 1$. A *k-labelled graph* is an ordered pair $(G, \ell)$ where $G$ is a graph and $\ell : V(G) \to \{1, \ldots, k\}$ is a mapping, that assigns a label from $\{1, \ldots, k\}$ to every vertex of $G$. The *vertices* and *edges* of $(G, \ell)$ are the vertices and edges of $G$. For $(G, \ell)$ a $k$-labelled graph, $(G, \ell)^\circ$ denotes the underlying graph $G$ without the labels, and $\ell$ is the *label function* of $(G, \ell)$.

Consider the following inductive definition of $k$-expressions and linear $k$-expressions, slightly generalising the original definition in [4]:

- for $o \in \{1, \ldots, k\}$ and $u$ a vertex name, $o(u)$ is a $k$-expression and a linear $k$-expression
- for $\delta$ a $k$-expression and $s, o \in \{1, \ldots, k\}$ with $s \neq o$, $\eta_{s,o}(\delta)$ and $\rho_{s\to o}(\delta)$ are $k$-expressions; if $\delta$ is a linear $k$-expression then $\eta_{s,o}(\delta)$ and $\rho_{s\to o}(\delta)$ are linear $k$-expressions
- for $p \geq 2$, and for $\delta_1, \ldots, \delta_p$ pairwise vertex-disjoint $k$-expressions, $\bigoplus(\delta_1, \ldots, \delta_p)$ is a $k$-expression; if $\delta_1$ is a linear $k$-expression and $\delta_2 = o_2(u_2), \ldots, \delta_p = o_p(u_p)$ then $\bigoplus(\delta_1, \ldots, \delta_p)$ is a linear $k$-expression.

We call two $k$-expressions *vertex-disjoint* if the vertex names occurring in the two $k$-expressions are pairwise different. Note that a vertex name occurring in a $k$-expression occurs exactly once.