Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in DISCRETE MATHEMATICS

www.elsevier.com/locate/endm

Recognizability Equals Definability for Graphs of Bounded Treewidth and Bounded Chordality

Hans L. Bodlaender

Utrecht University and Eindhoven University of Technology, The Netherlands

Pinar Heggernes, Jan Arne Telle

University of Bergen, Norway

Abstract

We prove that every recognizable family of graphs of bounded treewidth and bounded chordality is definable in counting monadic second-order logic.

Keywords: Automata, recognizability, logic, definability, treewidth, chordality

1 Introduction

The technique of translating between monadic second-order logic (MSOL) formulae and equivalent automata has a long history. An early result is a theorem of Büchi from 1960 [2] showing that the languages accepted by finite automata are exactly the MSOL-definable sets of strings. Viewed as a result on families of graphs this can be seen as establishing that recognizability equals definability for labeled paths. In a series of seminal papers starting in 1990 Courcelle [3] introduced the concept of a recognizable set of graphs and began an investigation of the monadic second-order logic of graphs and of sets of graphs. He established that any MSOL-definable family of graphs is recognizable, but showed that for graphs in general the converse cannot hold. However, for unordered unbounded trees Courcelle [3] did establish that recognizability equals definability, using a counting monadic second-order logic (CMSOL). The following quote from a later paper in the series illustrates the situation at the time:

It is not clear at all how an automaton should traverse a graph. A "general" graph has no evident structure, whereas a word or a tree is (roughly speaking) its own algebraic structure. [4]

The proposal for how to deal with this, using tree-decompositions of graphs, has nowadays become standard, see the recent book of Courcelle and Engelfriet [6], and will be the main tool that we use also in this paper. Courcelle proceeded to show that recognizability equals CMSOL-definability for graphs of treewidth at most two and conjectured that recognizability equals CMSOL-definability for graphs of bounded treewidth [4]. In this paper we establish that recognizability equals CMSOL-definability for graphs of bounded treewidth and bounded chordality (no chordless cycles of length larger than a constant c), thereby proving a special case of Courcelle's conjecture. Let us mention related work on the conjecture. In 1995 Kaller [10] established the special case of graphs of treewidth at most 3 and k-connected graphs of treewidth at most k. Two conference papers from 1997 [9] and 1998 [11] claimed to be able to prove the conjecture for, respectively, graphs of bounded pathwidth and graphs of bounded treewidth. However, full versions have not appeared of any of these two papers, and people in the field do not consider either of them satisfactory. Very recently, Jaffke and Bodlaender showed that recognizability implies MSOL-definability for Halin graphs, which are of treewidth 3, and for some related graph classes [8].

The main difficulty in proving Courcelle's conjecture for a class of graphs is to define in CMSOL, for any graph in the class, concrete minimum-width tree-decompositions, and that is what we do in this paper. We end the paper by arguing that to prove the full conjecture an extension of the techniques used in this paper will be needed.

2 Definitions

A *tree decomposition* of a graph $G = (V, E)$ is a pair $(\{X_i \mid i \in I\}, T = (I, F))$ with $\{X_i \mid i \in I\}$ a family of subsets (called *bags*) of V, and T a rooted tree, such that 1) $\bigcup_{i \in I} X_i = V$, 2) for each edge $\{v, w\} \in E$, there exists an $i \in I$ with $\{v, w\} \subseteq X_i$, and 3) for each vertex $v \in V$, the set $I_v = \{i \in I \mid v \in X_i\}$ induces a (connected) subtree of T. The *width* of a tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ equals $|\max_{i \in I} |X_i| - 1$, and the *treewidth* of a graph G is the minimum width of a tree decomposition of G.

The chordality of a graph is the length of a longest induced cycle. A graph is chordal if it has chordality 3. A triangulation of a graph $G = (V, E)$ is a supergraph $H = (V, E \cup F)$ which is chordal. The edges in F are called the fill edges. The triangulation is minimal if the graph $H' = (V, E \cup F')$ is not chordal for any strict subset $F' \subset F$. It is a well-known fact that a graph has treewidth at most k if and only if it has a triangulation with maximum clique size at most $k+1$, and also that graphs of treewidth at most k are k-colorable. See Figure 1 for an example of a chordal graph and a tree decomposition.

In the monadic second-order graph logic known as MSOL, sometimes called MSOL2, the graph is described by a set of vertices, a set of edges, and an incidence relation between vertices and edges, and the restriction to monadic logic means that the graph property in question may be defined using quantification over sets of vertices or edges, but not over more complex relations on tuples of vertices or edges. For a full introduction, see [6]. In CMSOL, Counting MSOL, we allow the unary predicate symbols $mod_{p,q}$ for non-negative integers $p < q$ with the interpretation that $mod_{p,q}(V) = True$ iff $|S| = p \mod q$, where S is the set denoted by the set variable V. A graph property P is called CMSOL-definable over a class of graphs F if there is a CMSOL-formula Φ such that for each $G \in F$, G satisfies P iff Φ is true on G. Using $mod_{0,2}$ we can express in CMSOL the property that a graph has an even number of vertices, something which cannot be done in MSOL alone [3]. Refer to e.g. [1] for a further discussion of encoding expressions in MSOL and CMSOL.

A tree automaton will process a tree decomposition by assigning each of its nodes to one of a finite number of states based on the label of the node and the states of its children. The tree decomposition is accepted iff its root is thus assigned to a designated accepting state. In order for a tree automaton to be a decision algorithm over graphs of treewidth bounded by k, it must accept either all of the width k tree decompositions of a given graph, or none of them. A family of graphs of treewidth bounded by k is said to be recognizable if there exists a tree automaton that accepts exactly the tree decompositions of graphs in the family.

3 Chordal graphs of bounded treewidth

To prove that a recognizable family of chordal graphs of bounded treewidth k is CMSOL-definable, the main task is to define in CMSOL, for every such graph G, some tree decomposition of width k of G. A perfect elimination ordering (peo) of a graph is a linear ordering of its vertices such that the higher-numbered neighbors of any vertex form a clique. We say that an orientation of the edges of a graph has the *adjacent out-neighbors* property, if for each pair of edges $\{u, v\}$ and $\{u, w\}$, if $\{u, v\}$ is directed from u to v and $\{u, w\}$ is directed from $\{u\}$ to $\{w\}$, then v and w are adjacent in G.

Theorem 3.1 *Let $G = (V, E)$ be a graph. The following are equivalent.*

(i) *G is chordal.*

(ii) *G has a perfect elimination ordering.*

(iii) *G has a tree decomposition of optimal width where each bag X_i induces a clique in G.*

(iv) *There is an acyclic orientation of the edges of G that has the adjacent out-neighbors property.*

Proof. (1) $\Leftrightarrow$ (2) $\Leftrightarrow$ (3) is well known, see e.g. [7]. (2) $\Rightarrow$ (3): Orient the edges by the order in which vertices appear in the peo. (3) $\Rightarrow$ (2): Take an arbitrary topological ordering of the acyclic orientation with the adjacent out-neighbors property. One easily verifies that this is a peo. □

We call an acyclic orientation with the adjacent out-neighbors property an aon-ordering. Using properties of peo's and chordal graphs, one can show that if G is connected, then an aon-ordering has exactly one vertex with out-degree zero. Suppose we have a connected chordal graph G of treewidth at most k. Take an acyclic orientation of G with the adjacent out-neighbors property. We now first define a spanning tree T of G, as follows. For each vertex v with outdegree at least one, its out-neighbors form a clique, and hence v has a neighbor w such that for each other neighbor $x \neq w$ of v, the edge $\{w, x\}$ is directed from w to x. Add the edge $\{v, w\}$ to T, i.e., for each v with outdegree at least one, we add the edge to the neighbor of v that is first in the peo. This forms a spanning tree, with the last vertex in the peo as root. Call T the aon-defined spanning tree.

This aon-defined spanning tree fulfils an important role in our proof. It can be defined in Monadic Second Order Logic; we can use it to build a tree decomposition (of width at most k) of G, and on this tree, we can follow Courcelle's proof for trees [3] to obtain the main result of this section. Given an aon-defined spanning tree T, we build a tree decomposition $(\{B_v \mid v \in V\}, T)$ as follows. The tree used in the tree decomposition is T; to each node of the tree, i.e., vertex v of the graph, we associate the bag B_v consisting of v and its out-neighbors. Following standard graph theory, it follows that for each aon-ordering, the associated tree decomposition is a tree decomposition of G of width at most k. A useful technical point is the following: the acyclic orientation also defines a total ordering on each bag of the tree decomposition (as each bag is a clique). As k is a constant, and the formula length can depend on k, we can thus express what is the ith vertex in the bag of a vertex

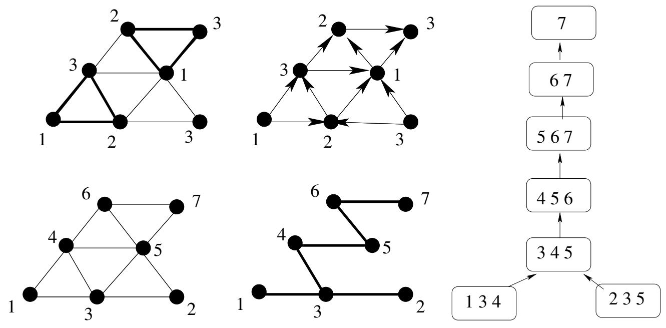

Fig. 1. On top left a 3-coloring V_1, V_2, V_3 and a chosen subset S of edges in bold, of a chordal graph of treewidth 2. On top right the orientation defined by (V_1, V_2, V_3, S), which we note is acyclic with 'adjacent out-neighbors'. On bottom left one of the peo's that is therefore defined by (V_1, V_2, V_3, S). On bottom right the spanning tree defined by the acyclic aon-orientation. On the right the tree-decomposition thus defined by (V_1, V_2, V_3, S), with vertices identified by the chosen peo number.

v. Our proof now uses the following sequence of steps:

- Define a $(k+1)$-vertex coloring of G in MSOL.
- Use this vertex coloring to implicitly define an orientation of the edges of G. This technique is due to Courcelle [5, page 120].
- Express in MSOL that the orientation is acyclic and fulfills the adjacent out-neighbors (aon) property.
- Choose a set of edges, and verify in MSOL that it is the spanning tree T defined by the aon orientation.
- Mimic on T the proof by Courcelle for trees [3], keeping in mind the corresponding tree decomposition.

A vertex coloring can be defined easily in MSOL: for $k+1$ vertex sets, we can verify that these sets partition V and have no edge between vertices in the same set. Then, any orientation can be represented by a subset S of the edges, as follows: for each edge of the graph its two endpoints have different colors, if the edge is oriented from the lower color endpoint to the higher color, then put this edge in the subset S. Note that an orientation will be defined regardless of whether G is chordal or not, and this will be of use to us in the next section when we consider non-chordal graphs. Thus, to define a tree-decomposition of a chordal graph $G = (V, E)$ of treewidth at most k we first use an MSOL formula stating

$\exists V_1, V_2, ..., V_{k+1}$ forming a partition of V where adjacent vertices are in different classes, and $\exists S \subseteq E$, that together defines an orientation that is checked to be acyclic with the 'adjacent out-neighbors' property.

The details of how to define this in MSOL is not complicated. Let us give an example of a macro that will tell us if there is an arc from vertex v to vertex w, for $k = 2$:
$arc(v, w) \equiv (v, w) \in E \wedge (((v \in V_1 \wedge w \in V_2) \vee (v \in V_1 \wedge w \in V_3) \vee (v \in V_2 \wedge w \in V_3)) \wedge (v, w) \in S) \vee$
$((w \in V_1 \wedge v \in V_2) \vee (w \in V_1 \wedge v \in V_3) \vee (w \in V_2 \wedge v \in V_3)) \wedge (v, w) \notin S))$
This has to be coupled with macro MSOL formulae which will allow us to handle the resulting tree-decomposition. For an example, to express that in the tree decomposition the parent of bag B_v is bag B_w:
$par(v, w) \equiv arc(v, w) \wedge \forall x \neq w : arc(v, x) \Rightarrow arc(w, x)$ and when we need to access the parent bag of B_v we state $\exists w : par(v, w)$. There are several such macros. Now, suppose we have a recognizable family of edge-labelled chordal graphs of bounded treewidth. This gives us a (deterministic) finite state tree automaton that runs on any tree decomposition of an input graph. Let $Q = \{s_1, \ldots, s_r\}$ be the set of states of this automaton. The automaton gives for every bag in the tree decomposition a state from Q; the automaton precisely defines the function that gives the state of a bag i, given the labels of the edges between its vertices, the states of the child bags, and the information which vertices of X_i appear in which child bags.

We now label the nodes of the tree decomposition as follows. Each bag i has a label, consisting of three parts: first, the size $|X_i|$, second, for each pair (a, b), $1 \leq a < b \leq |X_i|$, the label of the edge between the ath and the bth vertex in X_i, and third: either the information that i is the root of T, or a subset $Z \subseteq \{1, \ldots, |X_i|\}$, with $a \in Z$, if and only if the ath vertex in X_i also belongs to the bag of the parent of i in T. Note that this information is all that the automaton on the tree decomposition uses, and hence, we can view the automaton on the tree decomposition as a deterministic finite state tree automaton on this labelled tree.

By Courcelle's result [4], we have that there is a CMSOL sentence ϕ that expresses whether this latter automaton accepts on this labelled tree. This automaton uses quantification over vertices and edges in T, and checks for vertices of their label. We want to translate this sentence to a CMSOL sentence expressing a property of G. As above, we define the aon-ordering and the corresponding spanning tree $T = (V, E_T)$. Then, the quantifications over elements of T directly translate to quantifications over vertex and edge sets in G and a check for edges if they belong to T (which just translates to $e \in E_T$.) For a check on the label of a vertex in T, we can observe that each part of the label follows from the aon-ordering in a way that can be expressed in MSOL. Thus, ϕ can be translated to a sentence ϕ', such that ϕ holds for the labeled tree T, if and only if ϕ' holds for G.

Lemma 3.2 *Any recognizable family of connected edge-labelled chordal graphs of bounded treewidth is CMSOL-definable.*

4 Bounded treewidth graphs of bounded chordality

In this section, let G be a graph of treewidth at most k and chordality c. As in the previous section, our goal is to define in CMSOL, for every such G, a tree-decomposition of width at most k. Graphs of chordality $c > 3$ are not chordal and thus do not have a tree decomposition where every bag induces a clique. Instead we will for these graphs define a set of fill edges, with the property that when the fill edges are added to the graph it becomes chordal. Let $\mathcal{F}$ be a family of graphs of treewidth at most k. We define $\mathcal{F}_{\mathcal{C}}$ to be the family of edge-labelled chordal graphs (labels Edge or Fill) of treewidth at most k, such that $G \in \mathcal{F}_{\mathcal{C}}$ iff G is chordal and of treewidth at most k and the graph G restricted to edges with labels Edge is in $\mathcal{F}$. We have the following easy but important lemma.

Lemma 4.1 *If $\mathcal{F}$ is recognizable then $\mathcal{F}_{\mathcal{C}}$ is recognizable.*

Proof. Use the same automaton ignoring edges with label Fill. □

Fill edges are defined by taking a linear order π of the vertices, called an elimination order, and removing vertices from the graph in this order, while also adding fill edges between any two remaining neighbors of the removed vertex that have not already been made adjacent. We also give a label to every fill edge at the moment it is introduced, and use this label to partition the fill edges into a number of levels $Fill = F_1 \cup F_2 \cup \ldots \cup F_q$. The original edges of the graph $G = (V, E)$ are assigned the label 0 and belong to F_0. The graph $G_\pi = (V, F \cup Fill)$ is called the filled graph. In general, when removing vertex v, if v has two remaining non-adjacent neighbors u and w, with edge $\{v, u\} \in F_i$ and $\{v, w\} \in F_j$, then a fill edge $\{u, w\}$ belonging to $F_{1+\max\{i,j\}}$ is added. The maximum length of an induced cycle gives an upper bound on the number of fill levels.

Lemma 4.2 *If G has treewidth k and chordality $c \geq 3$ then G has an elimination order π giving $Fill = F_1 \cup F_2 \cup \ldots \cup F_q$ with $q \leq c - 3$ and with the filled graph G_π being chordal and having treewidth k.*

Since the filled graph is chordal we can apply the techniques from the previous section to define a tree decomposition of it, which will be also a tree decomposition of G, of width k. However, applying the techniques from the previous section is not straightforward, and involves several considerations. We need macros that handle quantification over $F_0 \cup Fill$, and also macros that count modulo the size of a subset of $F_0 \cup Fill$. For a graph G of treewidth at most k and chordality c we start as follows:

- Guess a vertex coloring $V_1, ..., V_{k+1}$ and $S \subseteq E = F_0$
- Check that this defines an acyclic orientation of the edges F_0 of G

Consider how this acyclic orientation gives an elimination order: start by removing a source vertex, add a fill edge belonging to F_1 between any two of its non-adjacent neighbors, add an orientation to these fill edges that maintains acyclicity. Then iterate this procedure on the remaining partially filled graph. Note that the guessed orientation of $F_0 \cup F_1 \cup ... \cup F_i$ defines the fill F_{i+1}. This procedure is inherently iterative but it can be defined in CMSOL, by macros for level i based on macros of levels lower than i, as the number of levels is a constant depending on the chordality of G. We continue with the following steps to express this in CMSOL:

- Guess the set of fill edges $Fill = F_1 \cup F_2 \cup ... \cup F_q$, for $q \leq c - 3$
- Check that fill edges obey the vertex coloring and that the resulting graph is chordal of treewidth at most k
- Guess the orientation of the fill edges and check that it is acyclic with the aon-property
- Check that the the guessed orientation of $F_0 \cup F_1 \cup ... \cup F_i$ defines the fill F_{i+1}, for each $0 \leq i \leq q$

Let us give some details. We represent a fill edge by the vertex creating it and the pair of colors of the endpoints of the fill edge. Thus, to guess the fill edges we actually guess $k(k+1)/2$ vertex subsets for each fill level, and use the vertex coloring $V_1, ..., V_{k+1}$. For example, to guess F_1 for $k = 2$ we state: $\exists Z_{12}, Z_{13}, Z_{23}$ subsets of V, which gives a fill edge $vw \in F_1$ for each $vw \notin E$ such that either Z_{12} defines $vw \in F_1$ by $v \in V_1 \wedge w \in V_2 \wedge \exists x \in Z_{12} \wedge arc(x, v) \wedge arc(x, w)$ or $Z_{1,3}$ or Z_{23} defines, in a similar way, $vw \in F_1$.

We are now able to translate a quantification, say existential, over edge sets of the filled graph, for simplicity say over $F_0 \cup F_1$, by $\exists S_0 \subseteq E \wedge \exists S_{12} \subseteq Z_{12}, S_{13} \subseteq Z_{13}, S_{23} \subseteq Z_{23}$.

We guess the orientation of the fill edges by the same technique as before, i.e. guess subset of edges of the filled graph and orient from low to high color for edges in this subset, and vice versa for those not in the subset.

To check that the guessed fill edges $Fill = F_1 \cup F_2 \cup \ldots \cup F_q$ actually follows from the guessed orientations we follow the explanation given above. We need macros $fill_i(u, w)$ and $arc_i(u, w)$, for each level $0 \leq i \leq q$ such that $fill_i(u, w)$ is True if and only if $fill_j(u, w)$ is false for all $j < i$ and there is a vertex x with $fill_{j'}(x, u), arc_{j'}(x, u)$ and $fill_{j''}(x, w), arc_{j''}(x, w)$ all True for some $j', j'' \leq i$ and either $j' = i - 1$ or $j'' = i - 1$.

Theorem 4.3 *Any recognizable family $\mathcal{F}$ of graphs of chordality c and treewidth at most k is CMSOL-definable.*

Proof. By Lemma 4.1 we know that the corresponding family of edge-labelled chordal graphs (with edge labels Edge and Fill) $\mathcal{F}_{\mathcal{C}}$ is recognizable. By Lemma 3.2 we know that $\mathcal{F}_{\mathcal{C}}$ is CMSOL-definable by some formula Φ. Above, we have described a CMSOL-formula which defines tree decompositions of any c-chordal graph, which in fact is an edge-labelled chordal graph, and macros that are used to translate the formula Φ into a formula that will explicitly handle graphs of chordality c and only implicitly edge-labelled chordal graphs. Hence $\mathcal{F}$ is CMSOL-definable just as $\mathcal{F}_{\mathcal{C}}$ was. $\square$

5 Obstacles to proving the full Courcelle conjecture

On a high level view, our proof consists of a CMSOL formulation of the sentence *There exists a minimal triangulation of G such that the automaton accepts on the tree decomposition, corresponding to this minimal triangulation.* Unfortunately, this approach cannot work in general. A simple example can be found in the following class of graphs $\mathcal{G}$. Take a cycle with even length n, and add two pendant vertices to a chosen pair of vertices at distance $n/2$ on the cycle. The graph on the left in Figure 2 shows an example of an element of the class and the graph next to it is a triangulation. We can find such a triangulation to all even $n \geq 4$. If we view the triangulation as an edge labelled graph, we obtain a class of labelled chordal graphs of treewidth two; it is not hard to see that this class is regular, i.e. recognizable. However, $\mathcal{G}$ is not regular. This is an easy consequence of well known theory. Thus, we have a non-regular class of graphs of treewidth two, with a regular set of minimal triangulations, also of treewidth two. Thus, it is insufficient to guess a minimal triangulation in a proof of Courcelle's conjecture: a different approach would be necessary, e.g., guess some special triangulation that can be expressed in (C)MSOL.

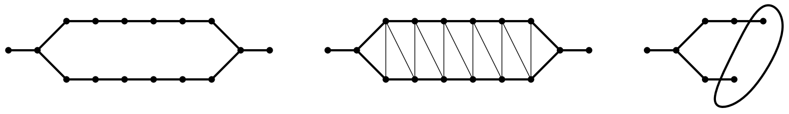

Fig. 2. A non-regular class of graphs with a regular class of triangulations

References

[1] Borie, R. B., R. G. Parker and C. A. Tovey, *Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families*, Algorithmica **7** (1992), pp. 555–581.

[2] Büchi, J. R., *Weak second-order arithmetic and finite automata*, Mathematical Logic Quarterly **6**.

[3] Courcelle, B., *The monadic second-order logic of graphs I: recognizable sets of finite graphs*, Inf. Comput. **85** (1990), pp. 12–75.

[4] Courcelle, B., *The monadic second-order logic of graphs V: on closing the gap between definability and recognizability*, Theor. Comput. Sci. **80** (1991), pp. 153–202.

[5] Courcelle, B., *The monadic second-order logic of graphs VIII: Orientations*, Annals of Pure and Applied Logic **72** (1995), pp. 103–143.

[6] Courcelle, B. and J. Engelfriet, "Graph Structure and Monadic Second-Order Logic - A Language-Theoretic Approach," Encyclopedia of mathematics and its applications **138**, Cambridge University Press, 2012.

[7] Golumbic, M. C., "Algorithmic graph theory and perfect graphs," Annals of Discrete Mathematics, 2004.

[8] Jaffke, L. and H. Bodlaender, *"MSOL-definability equals recognizability for Halin graphs and bounded degree k-outerplanar graphs"*, CoRR, http://arXiv:1503.01604, 2015 (2015).

[9] Kabanets, V., *Recognizability equals definability for partial k-paths*, in: *Automata, Languages and Programming, 24th International Colloquium, ICALP'97, Bologna, Italy, 7-11 July 1997, Proceedings*, 1997, pp. 805–815.

[10] Kaller, D., *Definability equals recognizability of partial 3-trees and k-connected partial k-trees*, Algorithmica **27** (2000), pp. 348–381.

[11] Lapoire, D., *Recognizability equals monadic second-order definability for sets of graphs of bounded tree-width*, in: *STACS 98, 15th Annual Symposium on Theoretical Aspects of Computer Science, Paris, France, February 25-27, 1998, Proceedings*, 1998, pp. 618–628.