Math. Struct. in Comp. Science (2002), *vol.* 12, *pp.* 203–235. © 2002 Cambridge University Press
DOI: 10.1017/S0960129501003565 Printed in the United Kingdom

Fusion in relational structures and the verification of monadic second-order properties

B. COURCELLE[†] and J. A. MAKOWSKY[‡]

[†]*CNRS, UMR 5800, Université Bordeaux I,*
351 cours de la Libération, 33405 Talence, France
Email: courcell@labri.u-bordeaux.fr
http://dept-info.labri.u-bordeaux.fr/~courcell/ActSci.html

[‡]*Department of Computer Science, Technion–Israel Institute of Technology,*
32000 Haifa, Israel
Email: janos@cs.technion.ac.il
http://cs.technion.ac.il/~janos

Received 4 July 2000; revised 2 January 2001

Relational structures offer a common framework for handling graphs and hypergraphs of various kinds. Operations like disjoint union, the creation of new relations by means of quantifier-free formulas, and relabellings of relations make it possible to denote them using algebraic expressions. It is known that every monadic second-order property of a structure is verifiable in time proportional to the size of such an algebraic expression defining it. We prove here that this result remains true if we also use in these algebraic expressions a *fusion* operation that fuses all elements of the domain satisfying some unary predicate. The value mapping from these algebraic expressions to the structures they denote is a *monadic second-order definable transduction*, which means that the structure is definable inside the tree representing the algebraic expression by monadic second-order formulas. It follows (by using results of other articles) that, with this fusion operation, we cannot generate more graph families, but we can generate them with less unary auxiliary predicates. We also obtain clear-cut characterizations of *Vertex Replacement* and *Hyperedge Replacement* context-free graph grammars in terms of four types of operations, amongst which is the fusion of vertices satisfying a specified predicate.

1. Introduction

The fundamental problem of *model checking* consists of deciding as efficiently as possible whether a given finite relational structure satisfies a fixed logical formula. Since graphs and hypergraphs can be handled as relational structures, the construction of efficient algorithms for verifying properties of finite graphs and hypergraphs is also relevant for

[†] Supported by the European Training and Mobility Research Network GetGrats
[‡] Supported by a Grant of the German Israeli Foundation (1995-1998) and by the Fund for Promotion of Research of the Technion–Israeli Institute of Technology

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

this approach, provided the properties of interest are expressible in tractable logical languages.

Monadic second-order logic (*MS logic* for short), that is, the extension of first-order logic with quantified variables denoting sets, is both expressive (because many graph properties such as connectivity, k-colorability, planarity and other forbidden configuration properties, and Hamiltonicity are expressible in this language) and tractable, at least for certain classes of hierarchically decomposable graphs.

A well-known paradigmatic result (Downey and Fellows 1997; Arnborg *et al.* 1991; Courcelle and Mosbah 1993) states that every graph property expressible in MS logic is verifiable in linear time on graphs of tree-width at most k (for fixed k). That a graph has *tree-width at most* k means that it has a *tree-decomposition* of width at most k. (This extends to relational structures.) Tree-decompositions form one particular type of hierarchical decomposition, and tree-width can be considered as a complexity measure for graphs. *Clique-width* (Courcelle and Olariu 2000; Courcelle *et al.* 2000; Corneil *et al.* 2000) is another complexity measure on graphs, which is based on another type of hierarchical decomposition and is relevant to the verification of MS properties: for every k and for every MS formula φ one can construct a linear time algorithm that takes as input a clique-width-k decomposition of a graph G and decides whether φ holds in G. Since every class of graphs of bounded tree-width has bounded clique-width and not *vice-versa* (Courcelle and Olariu 2000), clique-width is more powerful than tree-width for the verification of MS properties. Our starting questions are the following:

Question 1: Do there exist even more powerful types of hierarchical decomposition providing efficient verification algorithms for MS graph properties?

Question 2: Can one handle structures (equivalently, hypergraphs) as well as graphs in such a way?

We can reformulate these questions in terms of *operations* that build structures, just as concatenation builds words from letters. Typical examples of binary operations are the disjoint union of two structures, the series-composition (or concatenation) of two graphs given with an ordered pair of distinguished vertices, the parallel-composition of two such graphs, the extension of parallel-composition to graphs or hypergraphs equipped with a k-tuple of distinguished vertices.

Typical examples of unary graph operations are the edge-complement or the operation $add_{p,q}$, which adds edges between every vertex labelled by p and every vertex labelled by q (this last operation together with disjoint union is at the base of the definition of clique-width). See Bauderon and Courcelle (1987), Courcelle (1992), Courcelle (1997), Courcelle *et al.* (2000), Courcelle and Olariu (2000), and Engelfriet (1994) for details of these graph operations. In general, for every choice of a finite set $\mathcal{F}$ of operations on finite structures (including constant symbols denoting basic structures), one gets a mapping $val_{\mathcal{F}}$ from the set $T(\mathcal{F})$ of finite well-formed terms over $\mathcal{F}$ to the considered class of finite structures: the structure $val_{\mathcal{F}}(t)$ called the *value* of t, is defined by interpreting the symbols of $\mathcal{F}$ in the specified way. We will consider t as a *hierarchical decomposition* of $val_{\mathcal{F}}(t)$ and we will call $val_{\mathcal{F}}$ the *evaluation mapping relative to* $\mathcal{F}$. Our initial questions can be formulated in the following more precise way:

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Question 3: For which finite sets $\mathcal{F}$ of operations on structures does there exist an algorithm that associates with every closed MS formula φ a finite tree-automaton that defines $\{t \in T(\mathcal{F}) \ / \ val_{\mathcal{F}}(t) \models \varphi\}$?

Using the terminology of Courcelle and Walukiewicz (1998), an equivalent formulation is as follows:

Question 4: For which finite sets $\mathcal{F}$ of operations on structures is the mapping $val_{\mathcal{F}}$ MS-compatible?

The MS-*compatibility* of a mapping f on structures means that for every closed MS formula φ, one can construct a closed MS formula $f^{\#}(\varphi)$ such that for every structure S (of appropriate type), $S \models f^{\#}(\varphi)$ iff $f(S) \models \varphi$. Since MS-definability is equivalent to recognizability by finite tree-automata for subsets of $T(\mathcal{F})$ (for finite sets $\mathcal{F}$) a set $\mathcal{F}$ satisfies the requirement of Question 3 iff it satisfies that of Question 4.

Apart from model checking and construction of graph algorithms, there is another motivation, relevant to graph grammars, for being interested in sets of operations $\mathcal{F}$ such that $val_{\mathcal{F}}$ is MS-compatible.

In exactly the same way as context-free languages can be characterized as the least solutions of systems of recursive equations in languages, written with set union and concatenation extended to languages, context-free sets of graphs can be characterized (or defined) as the least solutions of systems of recursive equations in sets of graphs written with set union and graph operations extended to sets of graphs. Clearly this approach applies to hypergraphs and relational structures. The so-called *Hyperedge Replacement* (HR) (context-free) sets of hypergraphs, and the so-called *Vertex Replacement* (VR) (context-free) sets of graphs can be defined in this way in terms of appropriate operations. See Bauderon and Courcelle (1987), Engelfriet (1994), or Courcelle (1997). We say that a set of structures is $\mathcal{F}$*-context-free* if it is a component of the least solution of a system of recursive set equations over the operations in $\mathcal{F}$.

If a finite set $\mathcal{F}$ of operations on structures is such that $val_{\mathcal{F}}$ is MS-compatible, then, for every $\mathcal{F}$-context-free set L of structures, for every MS formula φ, the set $\{S \in L \ / \ S \models \varphi\}$ is also $\mathcal{F}$-context-free, and a system over $\mathcal{F}$ defining it can be obtained from φ and a system over $\mathcal{F}$ defining L. This generalizes the classical result stating that the intersection of a context-free language and a regular one is context-free.

In the present paper we consider the following operations on structures: disjoint union denoted by $\oplus$, quantifier-free definable operations and fusions. A *quantifier-free definable operation* transforms a structure S into a structure that has the same domain and the relations of which are defined by quantifier-free formulas over S. The *fusion* operation $fuse_p$ transforms a structure S into the structure obtained by fusing into a single element all elements of S labelled by p, or more formally satisfying a unary relation p.

Our main result states that for every finite set $\mathcal{F}$ consisting of $\oplus$, quantifier-free definable operations, fusions and constant symbols denoting finite structures, the mapping $val_{\mathcal{F}}$ is MS-compatible. (Some technical restrictions are made in the formal definitions.) We actually prove more than that. We prove that for such a set $\mathcal{F}$, the mapping $val_{\mathcal{F}}$ is a *non-copying MS-definable transduction*. This means that the object structure (here $val_{\mathcal{F}}(t)$)

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

is defined by MS formulas inside the input structure (here t). See Section 1.4 for details. And this gives two additional results.

The first is that the efficient algorithms for verification of MS properties extend to optimization and counting MS problems, as a consequence of the techniques of Courcelle *et al.* (2000). The second is that if $val_{\mathscr{F}}(T(\mathscr{F}))$ is a set of graphs, these graphs have bounded clique-width (by using a result of Courcelle (1992) and Engelfriet *et al.* (1997), which is stated below as Theorem 2.3). That $val_{\mathscr{F}}$ is an MS-definable transduction is known from Courcelle (1992) for sets $\mathscr{F}$ not containing fusions. The new result proves that fusions do not add expressive power (the corresponding context-free sets are the same) to the already studied operations on graphs, namely those defining clique-width and the quantifier-free definable ones. As another consequence, we get a unified characterization of VR and HR context-free graph grammars in terms of systems of equations over operations with the operations $add_{p,q}$ (for VR grammars) and without these operations (for HR grammars).

1.1. *Outline of the paper*

In Section 2 we recall the necessary background from previous work (Courcelle 1992; Courcelle 1997; Courcelle 1994; Courcelle and Mosbah 1993) on operations on graphs and relational structures. In Section 3 we introduce the new operations $fuse_p$. In Section 4 we show how certain inductive definitions on trees can be replaced by some explicit definitions by MS-formulas. In Section 5 we present the first part of the proof of Theorem 6.1 (the main theorem) in an abstract framework. In Section 6 we complete its proof. In Section 7, we give some consequences.

2. Operations on graphs, hypergraphs and relational structures

In this section we recall definitions and describe previous results from Courcelle (1992).

2.1. *Structures and monadic second-order logic*

We let R be a finite set of relation symbols, each of them, say A, being given with an *arity* $\rho(A)$ in $\mathbf{N}_+$. We use $\mathscr{S}(R)$ to denote the set of finite *R-structures*, that is, of tuples of the form $S =< D_S, (A_S)_{A\in R} >$ where $A_S \subseteq D_S^{\rho(A)}$ for $A \in R$. We recall that *monadic second-order logic* (MS logic for short) is first-order logic augmented with (uppercase) variables denoting subsets of the domain of the considered structure, and new atomic formulas of the form $x \in X$ expressing the membership of x in a set X. We use $MS(R, \mathscr{X})$ to denote the set of MS formulas over R with free variable in $\mathscr{X}$.

A property of structures (or of elements and/or of sets of elements of a structure) is MS-*definable* if it can be expressed by an MS formula.

We use $S \equiv S'$ to denote the existence of an isomorphism between two structures S and S'. If $L, L' \subseteq \mathscr{S}(R)$, we write $L \equiv L'$ if every structure of L is isomorphic to a structure in L' and *vice-versa.*

A *transduction* of structures is a multivalued mapping: $\mathscr{S}(R) \to \mathscr{S}(R')$, formally handled as a mapping $f \ : \ \mathscr{S}(R) \to \mathscr{P}(\mathscr{S}(R'))$ where $\mathscr{P}$ ( ) denotes power-set, such that $S \equiv S'$

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

implies $f(S) \equiv f(S')$. We say that f as above is *MS-compatible* (Courcelle and Walukiewicz 1998) if there exists a total recursive mapping $f^{\#} : MS(R', \emptyset) \to MS(R, \emptyset)$ such that $S \models f^{\#}(\varphi)$ iff $S' \models \varphi$ for some $S' \in f(S)$. We call $f^{\#}(\varphi)$ the *backwards translation* of φ relative to f.

Let $L \subseteq \mathscr{S}(R)$. We say that a transduction $f : \mathscr{S}(R) \to \mathscr{S}(R')$ is *deterministic on* L if for every $S \in L$, $f(S)$ is not empty and its elements are pairwise isomorphic.

We now consider transductions defined by MS formulas. *A parameterless MS-definable* transduction $f : \mathscr{S}(R) \to \mathscr{S}(R')$ is defined as follows, from $k \in \mathbf{N}_+$ and MS formulas α in $MS(R, \emptyset), \delta_1, ..., \delta_k$ in $MS(R, \{x\}), \theta_{A,i_1,...,i_n}$ in $MS(R, \{x_1, ..., x_n\})$, for $A \in R, n = \rho(A), 1 \leqslant i_1, ..., i_n \leqslant k$:

(1) $f(S)$ is non-empty iff $S \models \alpha$.

(2) Assuming $S \models \alpha$, then $f(S) = \{T\}$, where T is constructed as follows:

— $D_T = D_1 \times \{1\} \cup ... \cup D_k \times \{k\} \subseteq D_S \times \{1, ..., k\}$;

— each D_i is $\{x \in D_S \ / \ S \models \delta_i(x)\ \}$;

— each relation A_T is defined as the union of the sets of tuples of the form

$$\{((x_1, i_1), ..., (x_n, i_n)) \ / \ S \models \theta_{A,i_1,...,i_n}(x_1, ..., x_n)\},$$

for all $i_1, ..., i_n \in \{1, ..., k\}$.

It is *non-copying* if $k = 1$. In all cases, f is deterministic: $L \to \mathscr{S}(R')$ where $L = \{S \in \mathscr{S}(R) \ / \ S \models \alpha\}$. This set is thus MS-definable.

Let f be a parameterless MS-definable transduction. It is MS-compatible (Courcelle 1994; 1997). One can even define a backwards translation $f^{\#}(\varphi)$ for MS formulas φ over R' with free variables. If φ has p free variables, then $f^{\#}(\varphi)$ has kp free variables. We refer the reader to Courcelle (1994) and Courcelle (1997) for details.

We now extend the previous definitions in order to define (by MS formulas) transductions : $\mathscr{S}(R) \to \mathscr{S}(R')$ that are not deterministic. Let $P_1, ..., P_n$ be n unary relation symbols, which we will call *parameters* ($P_1, ..., P_n \notin R \cup R'$). We let $\Pi_R : \mathscr{S}(R \cup \{P_1, ..., P_n\}) \to \mathscr{S}(R)$ be the mapping that 'forgets' the relations $P_1, ..., P_n$. (It is actually a non-copying MS-definable transduction.)

A transduction $f : \mathscr{S}(R) \to \mathscr{S}(R')$ is *MS-definable* (we also say that it is an *MS-transduction*) if there exists an MS-definable subset L of $\mathscr{S}(R \cup \{P_1, ..., P_n\})$ and a parameterless MS-definable transduction $g : L \to \mathscr{S}(R')$ such that:

$$f(S) = \{g(S') \ / \ S' \in L, \Pi_R(S') = S\}.$$

It is *non-copying* if g is. The set $\{S \in \mathscr{S}(R) \ / \ f(S) \neq \emptyset\}$ is MS-definable: it is defined by the formula $\exists X_1, ..., X_n.\alpha[X_1/P_1, ..., X_n/P_n]$ (where $\alpha \in MS(R \cup \{P_1, ..., P_n\}, \emptyset)$ defines L) and X_i/P_i denotes the substitution of X_i for P_i in α. (We replace $P_i(x)$ by $x \in X_i$ for every i and x.)

An MS-transduction f as above is MS-compatible: for every $\varphi \in MS(R', \emptyset)$ one takes $f^{\#}(\varphi)$ equal to

$$\exists X_1, ..., X_n.((\alpha \wedge g^{\#}(\varphi))[X_1/P_1, ..., X_n/P_n]).$$

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

If $f : \mathscr{S}(R) \to \mathscr{S}(R')$ and $g : \mathscr{S}(R') \to \mathscr{S}(R'')$ are two MS-transductions, then gof, the transduction $h : \mathscr{S}(R) \to \mathscr{S}(R'')$ defined by

$$h(S) = \cup\{g(S') \ / \ S' \in f(S),\ S \in \mathscr{S}(R)\}$$

is an MS-transduction. It is non-copying if f and g are. See Courcelle (1994) and Courcelle (1997).

We now consider as an example and for later use the mapping $S \longmapsto S/\approx$, which associates with a structure S its quotient by an MS-definable equivalence relation $\approx$. We will see that this mapping is a deterministic non-copying MS-transduction, and its definition uses one parameter.

Let $S =< D_S, (A_S)_{A \in R} >$ be a structure in $\mathscr{S}(R)$, and $\approx$ be an equivalence relation on D_S. We use $[d]$ to denote the equivalence class of $d \in D_S$. We let $S' = S/\approx$ be the structure in $\mathscr{S}(R)$ defined as follows:

— $D_{S'} = D_S/\approx$

— $A_{S'}([d_1], ..., [d_n])$ holds iff $A_S(d'_1, ..., d'_n)$ holds for some $d'_1 \approx d_1, ..., d'_n \approx d_n$.

We call it the *quotient structure of* S by $\approx$.

Lemma 2.1. Let η be a formula in $MS(R, \{x, y\})$ such that for every $S \in \mathscr{S}(R)$ the relation $\{(x, y) \in D_S^2 \ / \ S \models \varphi(x, y)\}$ is an equivalence relation $\approx$. The mapping $S \longmapsto S/\approx$ is a deterministic non-copying MS-transduction.

Proof. We let P be a parameter. We let $\alpha \in MS(R \cup \{P\})$ be such that $S \models \alpha$ iff every equivalence class of $\approx$ contains one and only one element of P_S.

The set P_S can be taken as domain for the structure $S/\approx$. Hence there exists a parameterless non-copying MS-transduction $g : L \to \mathscr{S}(R)$ such that $L = \{S' \in \mathscr{S}(R \cup \{P\}) \ / \ S' \models \alpha\}$, and $g(S') = \Pi_R(S')/\approx$ for $S' \in L$ where $\approx$ is the equivalence relation on $\Pi_R(S')$ defined by η.

It follows that the mapping $S \longmapsto S/\approx$ is an MS-transduction. □

For every S there are several possible sets P that satisfy α, hence several structures S' such that $\Pi_R(S') = S$. But the structures $f(S')$ are all isomorphic. Although the transduction $S \longmapsto S/\approx$ is deterministic, we need a parameter for defining it as an MS-transduction.

2.2. *Terms*

In this section we review notation concerning terms and their representation by relational structures.

Let $\mathscr{F}$ be a *finite signature* (a finite set of function symbols, given with an arity mapping $\rho : \mathscr{F} \to \mathbf{N}$). Let $K = Max\{\rho(f) \ / \ f \in \mathscr{F}\}$.

We let $R_{\mathscr{F}}$ be the set of relation symbols $\{root, lab_f, suc_i \ / \ f \in \mathscr{F}, 1 \leqslant i \leqslant K\}$. We define $root, lab_f$ as unary and suc_i as binary.

Let $T(\mathscr{F})$ be the set of finite well-formed terms over $\mathscr{F}$. Terms are considered, as usual, as finite rooted directed vertex- and edge-labelled trees.

The nodes of the tree representing $t \in T(\mathscr{F})$ can be identified with Dewey words in

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

$\{1,...,K\}^*$, where ε is the root and $u1,...,un$ are the n successors of a node u, labelled by some f with $\rho(f) = n$. A term $t \in T(\mathscr{F})$ can thus be represented by the relational structure in $\mathscr{S}(R_{\mathscr{F}})$:

$$\| t \| = < N(t), root_t, lab_{f,t}, suc_{i,t} >$$

where

— $N(t)$ is the set of nodes (a subset of $\{1,...,K\}^*$),
— $root_t = \{\varepsilon\}$,
— $lab_{f,t} = \{u \in N(t) \ / \ u \text{ has label } f\}$,
— $suc_{i,t} = \{(u, ui) \ / \ u, ui \in N(t), 1 \leqslant i \leqslant \rho(f),\ f \in \mathscr{F}, u \in lab_{f,t}\}$.

The *leaves of* t are the nodes labelled by nullary symbols. The set of leaves $L(t)$ is thus $\{u \in N(t) \ / \ lab_{f,t}(u) \text{ holds}, \rho(f) = 0\} = \{u \in N(t) \ / \ suc_{i,t}(u, v) \text{ holds for no } v \text{ and } i\}$.

If $t \in T(\mathscr{F})$ and $u \in N(t)$, we use t/u to denote the term corresponding to the subtree of t issued at node u. If $u, v \in N(t)$, we let $u \leqslant v$ iff v is a prefix of u, that is, v is on the path in t between its root and u (intuitively, u is below v).

We recall that a subset $L \subseteq T(\mathscr{F})$ is MS-definable (that is, $L = \{t \in T(\mathscr{F}) \ / \ \| t \| \models \varphi\}$ for some $\varphi \in MS(R_{\mathscr{F}}, \varnothing)$) iff it is recognizable, that is, is definable by a finite tree-automaton. See, for instance, Thomas (1990).

2.3. *Operations on structures*

Our intention is to denote structures by finite terms built over operations that typically, combine two structures by forming their union augmented with some edges between them, or augment one structure by adding new tuples in its relations, or define in a structure some new relations. The operations are given below.

Disjoint union Let S_1 and $S_2 \in \mathscr{S}(R)$. We define $S_1 \oplus S_2$ as the structure S_3 built as the union of S_1 and an isomorphic copy S_2' of S_2 such that $D_{S_1} \cap D_{S_2'} = \varnothing$. We let $D_{S_3} = D_{S_1} \cup D_{S_2'}$ and $A_{S_3} = A_{S_1} \cup A_{S_2'}$ for each $A \in R$. (See the end of this section for a discussion of structures considered up to isomorphism.)

Quantifier-free definable operations We use $QF(R, \{x_1, ..., x_n\})$ to denote the set of quantifier-free formulas over R with free-variables in $\{x_1, ..., x_n\}$. An example is

$$(A(x_1, x_2, x_3) \vee \neg B(x_1, x_1)) \wedge x_3 \neq x_4.$$

A *quantifier-free definition scheme* (a *qfd scheme* for short) is a tuple of formulas $(\varphi_A)_{A \in R}$ such that each φ_A belongs to $QF(R, \{x_1, ..., x_m\})$, where $m = \rho(A)$.

We associate with this tuple a mapping $f : \mathscr{S}(R) \to \mathscr{S}(R)$ defined as follows. For $S \in \mathscr{S}(R)$, we let $S' = f(S)$ iff

$$D_{S'} = D_S$$
$$A_{S'} = \{(d_1, ..., d_m) \in D_S^m \ / \ S \models \varphi_A(d_1, ..., d_m)\}$$

We call f a *quantifier-free definable* (*qfd*) operation on structures. For a fixed finite set R there are finitely many *qfd* operations.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Basic structure Finally, we use **1** to denote the structure with singleton domain and empty relations.

2.4. *Terms representing structures*

If $\mathscr{F}$ consists of $\mathscr{F}_0 = \{\mathbf{1}, \oplus\}$ and a set Q of *qfd* operations over a fixed finite set R, then every term $t \in T(\mathscr{F})$ denotes a structure $val_{\mathscr{F}}(t) \in \mathscr{S}(R)$. The domain of $val_{\mathscr{F}}(t)$ is the set $L(t)$ of leaves of t. (All leaves are labelled by the constant **1**). We call $val_{\mathscr{F}}(t)$ the *evaluation mapping* relative to $\mathscr{F}$.

The following theorem is proved in Courcelle (1992).

Theorem 2.2. The evaluation mapping relative to $\mathscr{F} = \{\mathbf{1}, \oplus\} \cup Q$, where Q is the (finite) set of *qfd* operations over R, is an MS-transduction : $T(\mathscr{F}) \to \mathscr{S}(R)$.

2.5. *Operations on labelled graphs*

We now consider simple undirected loop-free graphs. (The directed case is a straightforward variant.) A simple undirected loop-free graph is defined as an $\{edg\}$-structure $G =< V_G, edg_G >$, where V_G is the set of vertices of G and $edg_G \subseteq V_G \times V_G$ is a binary symmetric irreflexive relation. We use $\mathscr{G}$ to denote the class of finite structures like this.

We denote disjoint union by $\oplus$. It is a special case of the definition for structures. The *edge complement* for loop-free graphs is defined as a *qfd* operation by taking $\varphi_{edg}(x_1, x_2)$ to be the formula $\neg(x_1 = x_2) \wedge \neg\, edg(x_1, x_2)$.

We use $\overline{G}$ to denote this edge-complement (another edge-complement could be defined for graphs with loops by omitting $\neg(x_1 = x_2)$ in the above formula).

The *join* operation is defined by

$$G_1 \otimes G_2 = \overline{(\overline{G_1} \oplus \overline{G_2})}$$

as a combination of edge-complements and disjoint union. The *cographs* (Corneil *et al.* 1985) are the graphs generated from **1** by $\oplus$ and edge-complement, or, equivalently, that are generated from **1** by $\oplus$ and $\otimes$.

However, cographs form a very special class. We have introduced edge-complement and join as easy examples of operations. In order to generate larger families of graphs, it is convenient to use *vertex labelled graphs*. We let $P = \{p, q, r, ...\}$ be a finite set of *labels* handled below as unary relation symbols. We let $R = \{edg\} \cup P$. A *P-graph* is a structure $G =< V_G, edg_G, (p_G)_{p \in P} >$, where $< V_G, edg_G >$ is a (simple, undirected) graph and the sets $(p_G), p \in P$ form a partition of V_G (some sets p_G may be empty). A P-graph is thus a structure in $\mathscr{S}(R)$ satisfying some particular conditions. We will use the following operations on P-graphs:

(i) The disjoint union $\oplus$.

(ii) The operation $add_{p,q}$ (for p, q distinct) defined as a *qfd* operation by letting $\varphi_{edg}(x_1, x_2)$ be the formula

$$edg(x_1, x_2) \vee (p(x_1) \wedge q(x_2)) \vee (p(x_2) \wedge q(x_1)).$$

This operation adds a new edge between any vertex labelled by p and any vertex

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

labelled by q (unless such an edge already exists).

Since $p \neq q$, it does not create loops. In Courcelle and Olariu (2000) it is denoted $\eta_{p,q}$.

(iii) We also use the operation $ren_{p\to q}$ for $p \neq q$, which is defined as a qdf operation by letting $\varphi_{edg}(x_1, x_2)$ be $edg(x_1, x_2)$, $\varphi_p(x_1)$ be $false$, $\varphi_q(x_1)$ be $p(x_1) \vee q(x_1)$, and $\varphi_r(x_1) = r(x_1)$ if $r \in P - \{p, q\}$.

(iv) For each p in P, we let $\mathbf{p}$ be the constant denoting the singleton vertex labelled by p with no loop.

Clique width For each finite set P we let

$$\mathscr{C}_P = \{\mathbf{p}, \oplus, add_{p,q}, ren_{p\to q} \ / \ p, q \in P, p \neq q\} \quad \text{and} \quad \mathscr{C}_n = \mathscr{C}_{\{1,..,n\}}.$$

Every P-graph defined by $t \in T(\mathscr{C}_P)$ is simple, undirected and loop-free. Every simple, undirected, loop-free P-graph is the value of some $t \in T(\mathscr{C}_{P'})$ for some large enough set $P' \supseteq P$. We use $G(t)$ to denote the graph defined by t.

The *clique-width* of G (denoted $cwd(G)$) is defined as the smallest size of P such that G is $G(t)$ for some $t \in T(\mathscr{C}_P)$. See Courcelle and Olariu (2000) and Corneil *et al.* (2000) for more about clique-width. We only recall that trees have clique-width at most 3, and that the cographs are the graphs of clique-width at most 2.

For algorithmic applications (Courcelle *et al.* 2000) it is useful to have efficient recognition algorithms for classes of graphs of clique-width at most k. For the moment we only know that this problem is NP. It is linear for $k = 2$ (Corneil *et al.* 1985) and polynomial for $k = 3$ (Corneil *et al.* 2000).

In addition to the consequences of Theorem 2.2, we have the following result, which is a kind of converse (Engelfriet *et al.* 1997).

Theorem 2.3. If $\mathscr{F}$ is a finite set of function symbols and τ is an MS-transduction, $T(\mathscr{F}) \to \mathscr{G}$, then the graphs in $\tau(T(\mathscr{F}))$ have bounded clique-width.

A final remark Properties of structures expressed by logical formulas do not distinguish isomorphic structures. Hence, they only specify sets of *abstract structures*, that is, of isomorphic classes of structures. However, for constructing a disjoint union, we 'copy' one of the structures to insure disjointness of domains. Any two such copies yield isomorphic structures. Hence a term over operations on structures actually specifies an isomorphism class of structures. However, for writing proofs it is more convenient to work with 'concrete' structures (not isomorphism classes of structures) without discussing selection of disjoint copies.

3. The fusion operation on labelled structures

Our objective is to extend Theorem 2.2 to finite sets of operations $\mathscr{F}$ on structures that contain $\oplus, qfd$ operations and certain fusion operations. A *fusion* is an operation of the form $fuse_p$ that fuses into a single element all elements of the domain of the given structure S that belong to p_S where p is a *predicate*, that is, a unary relation. We say that an element of D_S belonging to p_S is *labelled by* p.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

3.1. $fuse_p$ and restricted qfd operations

The predicates p, for $p \in P$, on which the fusion operations $fuse_p$ are based, will be handled exactly as the labels in the expressions defining graphs, recalled in Section 2.

We let R be a set of relation symbols as in Section 2.1. Each $A \in R$ has arity $\rho(A) \in \mathbf{N}_+$. We let P be a finite set of unary relation symbols $P = \{p, q, r, ...\}$, with $R \cap P = \emptyset$. We let $\mathscr{S}_P(R)$ denote the set of finite $(R \cup P)$-structures such that $\{p_S \ / \ p \in P\}$ forms a partition of D_S. The *type* of S is $\{p \in P \ / \ p_S \neq \emptyset\}$. It is denoted $\tau(S)$.

We use $\mathbf{p}$ to denote a structure with singleton domain $\{*\}$ such that $*$ satisfies p (hence $\tau(\mathbf{p}) = \{p\}$) and $A_{\mathbf{p}} = \emptyset$ for $A \in R$.

We will use $\oplus$ (disjoint union) and the operations $ren_{p \to q}$ for $p, q \in P$, $p \neq q$, defined as for graphs and structures in Section 2.3. We let

$$\mathscr{F}_P = \{\mathbf{p}, \oplus, ren_{p \to q} \ / \ p, q \in P, p \neq q\}.$$

The operation $fuse_p$ If $S \in \mathscr{S}_P(R)$ and $p \in P$, then $fuse_p(S)$ is the quotient structure $S' = S/\approx$ where $d \approx d'$ iff $d = d'$ or $p_S(d)$ and $p_S(d')$. See Lemma 2.1 on quotient structures.

We use $\mathscr{F}use_P$ to denote the set $\{fuse_p \ / \ p \in P\}$.

Restricted *qfd* operations In addition to the *qfd* operations $ren_{p \to q}$ that modify the predicates p_S for $p \in P$, (that is, the labellings of elements of structures) independently of the other relations, we will use *quantifier-free definable* (*qfd* operations that modify the relations A_S for A in R by means of quantifier-free formulas that may use predicates in P. We do not know whether our main theorem can be proved for more general *qfd* operations.

We will denote $QF(P \cup R, X)$ by $QF(P, R, X)$ in order to emphasize the roles of P and R.

A *restricted qfd operation* $f : \mathscr{S}_P(R) \to \mathscr{S}_P(R)$ is specified by a family of formulas $(\varphi_A)_{A \in R}$, where $\varphi_A \in QF(P, R, \{x_1, ..., x_{\rho(A)}\})$ for each A. Since $QF(P, R, X)$ is finite (up to equivalence) if X is finite, the set $2\mathscr{F}\mathscr{D}_{P,R}$ of restricted *qfd* operations over $P \cup R$ is finite. In order to simplify the terminology, we will write '*qfd* operation' for 'restricted *qfd* operation'.

3.2. The value mapping from terms to structures

Let $\mathscr{F} = \mathscr{F}_P \cup \mathscr{F}use_P \cup 2\mathscr{F}\mathscr{D}_{P,R}$. The value of a term $t \in T(\mathscr{F})$ is a structure $val_{\mathscr{F}}(t) \in \mathscr{S}_P(R)$, also denoted $S(t)$.

The domain of $S(t)$ is the quotient-set of the set of leaves of t (denoted by $L(t)$) by the equivalence relation $\approx_t$ defined inductively as follows:

(i) $\approx_t = \approx_{t_1} \cup \approx_{t_2}$ if $t = t_1 \oplus t_2$ (and, of course, $L(t_1) \cap L(t_2) = \emptyset$, because even if t_1 and t_2 are equal terms, they are considered as disjoint subtrees of t);
(ii) $\approx_t = \approx_{t_1} \cup (p_{S(t_1)} \times p_{S(t_1)})$ if $t = fuse_p(t_1)$;
(iii) $\approx_t = \approx_{t_1}$ if $t = ren_{p \to q}(t_1)$ or $t = f(t_1)$ and $f \in 2\mathscr{F}\mathscr{D}_{P,R}$;
(iv) $\approx_{\mathbf{p}}$ is the trivial equivalence on a singleton set.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

The type of $S(t)$ can be evaluated by the following induction:

$$\begin{aligned}
\tau(S(t_1 \oplus t_2)) &= \tau(S(t_1)) \cup \tau(S(t_2)) \\
\tau(S(f(t_1))) &= \tau(S(t_1)) \text{ if } \quad f \in \mathcal{2F}\mathcal{D}_{P,R} \cup \mathcal{F}use_P\,; \\
\tau(S(ren_{p\to q}(t_1))) &= \begin{cases} \tau(S(t_1)) & \text{if} \quad p \notin \tau(S(t_1)), \\ \tau(S(t_1)) \cup \{q\} - \{p\} & \text{if} \quad p \in \tau(S(t_1)). \end{cases}
\end{aligned}$$

We know from Courcelle (1992) that the mapping $val_{\mathcal{F}} : T(\mathcal{F}_P \cup \mathcal{2F}\mathcal{D}_{P,R}) \to \mathcal{S}(R \cup P)$ is a non-copying MS-transduction, and hence is MS-compatible.

Our main theorem (Theorem 6.1) states that the mapping val from $T(\mathcal{F}_P \cup \mathcal{F}use_P \cup \mathcal{2F}\mathcal{D}_{P,R})$ into $\mathcal{S}_P(R)$ is also a non-copying MS-transduction.

The proof will be divided into two parts, presented in Sections 5 and 6. We only present its general structure here. We let $S(t)$ denote $val(t)$. The term t being fixed, for a leaf x of t below u, we use $[x]_u$ to denote the corresponding element of the domain of $S(t/u)$, that is, the equivalence class of x under $\approx_{t/u}$, the equivalence $\approx$ on the subtree t/u (the subtree of t issued at u). Here are the main steps.

(i) In the structure $\| t \|$ representing $t \in T(\mathcal{F}_P \cup \mathcal{F}use_P \cup \mathcal{2F}\mathcal{D}_{P,R})$, we define for each A in R a $(\rho(A)+1)$-ary relation A^t by letting:

$$\begin{aligned}
(u, x_1, ..., x_n) \in A^t \quad \text{iff} \quad & u \text{ is a node of } t, \text{ and} \\
& x_1, ..., x_n \text{ are leaves of } t/u, \text{ and} \\
& ([x_1]_u, ..., [x_n]_u) \in A_{S(t/u)}.
\end{aligned}$$

Recall that $S(t)$ is a structure the domain of which is $L(t)/\approx_t$, and $\approx_t$ is an equivalence relation that handles the fusions of leaves arising from the occurrences of the operations $fuse_p$ in t.

(ii) Assuming each A^t defined in $\| t \|$ by an MS-formula, and by using Lemma 3.1 below, the proof is easy to complete:

(ii.a) $D_{S(t)} = L(t)/\approx_t$,

(ii.b) $L(t)$ is an MS-definable subset of $N(t)$, the domain of $\| t \|$,

(ii.c) $\approx_t$ is an MS-definable equivalence relation on $L(t)$ (by Lemma 3.1(iii)),

(ii.d) $p_{S(t)}([d])$ holds iff (u, d) satisfies $\pi_{2,p}$ of Lemma 3.1 (ii),

(ii.e) $A_{S(t)}([d_1], ..., [d_n])$ holds iff $(\varepsilon, d_1, ..., d_n) \in A^t$ (ε is the root of (t)).

(iii) Hence $S(t)$ is definable from $\| t \|$ by an MS-transduction. (We use Lemma 2.1 saying that taking the quotient of a structure by an MS-definable equivalence relation is an MS-transduction).

(iv) The proof that the above relations A^t for $A \in R$ are MS-definable is the difficult part of the proof, handled in Sections 5 and 6. It will use an argument presented in the next section.

We complete this section with an easy technical lemma.

Lemma 3.1. Let $\mathcal{2} = \mathcal{2F}\mathcal{D}_{P,R}$. The following properties of nodes $u, x, y, x_1, ...$ of t in $T(\mathcal{F}_P \cup \mathcal{F}use_P \cup \mathcal{2})$ are MS-definable in $\| t \|$:

(i) $\pi_1(u, x)$: x is a leaf of t and $x \leqslant u$, (that is, x is below u),

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

(ii) $\pi_{2,p}(u,x) : \pi_1(u,x)$ and $S(t/u) \models p([x]_u)$, for p in P,
(iii) $\pi_3(u,x,y) : \pi_1(u,x)$ and $\pi_1(u,y)$ and $S(t/u) \models [x]_u = [y]_u$,
(iv) $\pi_\theta(u,x_1,...,x_k) : \pi_1(u,x_1)$ and ... and $\pi_1(u,x_k)$ and $S(t/u) \models \theta([x_1]_u,\ldots,[x_k]_u)$, where $p \in P, \theta$ is a first-order formula over P.

The domain of $S(t/u)$ and the predicates $p_{S(t/u)}$ do not depend on the operations of $\mathcal{Q}$ because they are modified only by the operations $ren_{p\to q}$, which belong to $\mathcal{F}_P$. They depend only on the occurrences in t of $\mathbf{p}$, of $\oplus$, of $ren_{p\to q}$ and of $fuse_p$. The formulas θ do not use the relation symbols A of R. Hence the validity of π_θ does not depend on the operations from $\mathcal{Q}$ occurring in t. Examples of formulas θ are

$$\forall x.\neg p(x) \quad \text{and} \quad \forall x,x'.(p(x) \wedge p(x') \Rightarrow x = x')$$

expressing that $p_S = \emptyset$ and that p_S has at most one element for the considered structure $S = S(t/u)$, respectively.

Proof. Property π_1 is MS-definable as it only deals with paths in a tree.

We consider property $\pi_{2,p}$. The label of x in t is some p indicating that x is introduced as the singleton graph $\mathbf{p}$. If there is no occurrence of a renaming operation $ren_{p\to q}$ changing p into q on the path in t from x to u, (t is considered as a tree), then x also has label p in $S(t/u)$. Otherwise its label results from a sequence of renamings and can be defined by a finite-state automaton. These automata can be translated into MS-formulas, because, on words, MS logic is equivalent to finite automata, see Thomas (1990). Hence we get the desired result.

For π_3, let us consider leaves x, y, that are below u. From the definitions, we get that $[x]_u = [y]_u$ iff either $x = y$ or there is some $v \leqslant u$ labelled by $fuse_p$ for some p and such that $[x]_v$ and $[y]_v$ both have label p in $S(t/v)$. This is expressible by an MS-formula in $\| t \|$ (and by using $\pi_{2,p}$).

A formula θ over$P$ has $p(x)$ and $x = y$ as its only possible atomic formulas. The construction of π_θ is thus straightforward as soon as we have $\pi_{2,p}$ and π_3 available. □

4. Inductive definitions on trees

In this section we show that certain 'bottom-up' inductive definitions on trees can be solved by MS-formulas.

Let F be a finite signature, F_0 be the set of symbols of arity 0, and $F_+ = F - F_0$. We let $N = Max\{\rho(f) \,/\, f \in F\}$. Let $\mathcal{U}$ be a set of relation symbols, where $\rho(U) \in \mathbf{N}_+$ is the arity of $U \in \mathcal{U}$. A *tree-based relation on* $t \in T(F)$ is a relation $R \subseteq N(t) \times L(t) \times ... \times L(t)$ of arity $n+1$, $n \geqslant 0$, such that $(u,x_1,...,x_n) \in R$ implies $x_1 \leqslant u,...,x_n \leqslant u$ (this means that the leaves $x_1,..,x_n$ are below u). We recall from Section 2.2 that a term t in $T(F)$ is represented by a structure $\| t \|$ in $\mathcal{S}(R_F)$, where $R_F = \{lab_f, suc_i, root/f \in F\ , 1 \leqslant i \leqslant N\}$. We let $R'_F = \{lab_f, suc_i, rt_i/f \in F\ , 1 \leqslant i \leqslant N\}$.

A *tree-based $\mathcal{U}$-structure* is a structure $Z = (\| t \|, (U_Z)_{U\in\mathcal{U}})$ such that each U_Z is a tree-based relation on t.

Let $u \in N(t) - L(t)$. We let $Z' \in \mathcal{S}(\mathcal{U} \cup R'_F)$ be such that:

— $D_{Z'} = \{w \in N(t) \,/\, w < u\}$,

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

— $rt_{iZ'} = \{ui\}$,
— $suc_{iZ'} = suc_{iZ} \cap (D_{Z'} \times D_{Z'})$,
— $lab_{fZ'} = lab_{fZ} \cap D_{Z'}$,
— $U_{Z'} = U_Z \cap D^n_{Z'}$ for U of arity n.

We denote Z' by $Z[t//u]$. Hence $Z[t//u]$ is the restriction of Z to the subtrees of t issued at the successors of u. Furthermore, the unary relations $rt_{iZ'}$ retain the relative ordering of these successors. If u has a unique successor $u1$, then rt_1 is useless and $Z[t//u]$ is isomorphic to the restriction of Z to the subtree of t issued at $u1$.

Definition 4.1 (Inductively defined tree-based relations). Let $t \in T(F)$. A *system of definitions* Σ over t is a set of defining equivalences concerning tree-based relations U_Z, for $U \in \mathscr{U}$. These equivalences are intended to hold in the structure $Z = (\| t \|, (U_Z)_{U \in \mathscr{U}})$, and there is one and only one equivalence defining each U_Z. The equivalence defining U_Z is of the following form (with $n+1 = \rho(U)$):

$$(u, x_1, ..., x_n) \in U_Z$$
$$\Leftrightarrow$$
$$(x_1 \leqslant u \wedge ... \wedge x_n \leqslant u) \wedge$$
$$\Big[(\bigvee_{f \in F_0} (lab(u) = f \wedge B_{U,f}) \vee \bigvee_{f \in F_+} (lab(u) = f \wedge P_{U,f}(Z[t//u], x_1, .., x_n))\Big]$$

where each $B_{U,f}$ belongs to $\{true, false\}$ and each $P_{U,f}$ is a property of $(Z[t//u], x_1, ..., x_n)$ expressed by a fixed formula, say φ, in $MS(\mathscr{U} \cup R'_F, \{z_1, ..., z_n\})$ the free variables of which take the values $x_1, ..., x_n$ respectively. Furthermore, we require that in every atomic subformula of φ of the form $W(w, y_1, ..., y_m)$, (with $W \in \mathscr{U}$) the variables $y_1, \ldots, y_m$ are not quantified, and hence are free in φ and thus belong to $\{z_1, ..., z_n\}$; the other variables, which occur in atomic formulas like $x = x'$, $x = suc_i(x')$, $rt_i(x)$ or in first position in the above atomic formulas (like w), can be quantified.

Proposition 4.1. Let $F, \mathscr{U}, \Sigma$ be given as above.

(i) For every $t \in T(F)$ there exists a unique tree-based $\mathscr{U}$-structure Z over t satisfying Σ. In other words, Σ defines one and only one tuple $(U_Z)_{U \in \mathscr{U}}$.
(ii) For each U in $\mathscr{U}$ of arity $n+1$, $n \geqslant 0$, there is a formula $\alpha_U(w, z_1, ..., z_n)$ in $MS(R, \{z_1, ..., z_n\})$ that defines in $\| t \|$, for every $t \in T(F)$, the relation U_Z where Z is as in (i).

Proof. (1) The existence and unicity of the family $(U_Z)_{U \in \mathscr{U}}$ such that $Z = (\| t \|, (U_Z)_{U \in \mathscr{U}})$ is a tree-based structure satisfying Σ is clear from the definition of Σ since the membership of $(u, x_1, ..., x_n)$ in U_Z is defined in a unique way from that of tuples $(w, y_1, ..., y_m)$ in W_Z where w is strictly below u and $W \in \mathscr{U}$.

(2) Assume $N+1 = Max\{\rho(U)/U \in \mathscr{U}\}$. For every $\bar{x} = (x_1, x_2..., x_N) \in L(t)^N$, every $U \in \mathscr{U}$, we let $U[i_1, ..., i_n]_{Z,\bar{x}}$ denote $\{u \in N(t)/(u, x_{i_1}, ..., x_{i_n}) \in U_Z\}$ where $1 \leqslant i_1, ..., i_n \leqslant N$, $n+1 = \rho(U)$. We will prove that these sets are MS-definable in $\| t \|$ in terms of $x_1, ..., x_N$. From now on we will enumerate the elements of $\mathscr{U}$ as $U_1, ..., U_q$. We let $[U_j, i_1, ..., i_n]$ be new set variables for $1 \leqslant j \leqslant q$, $n+1 = \rho(U_j), 1 \leqslant i_1, ..., i_n \leqslant N$. The definitions in Σ of

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

the relations $U_j, j \geqslant 1$ can be collected into a single MS-formula

$$\Psi(x_1, ..., x_N, ..., [U_j, i_1, ..., i_n], ...)$$

(built with the relation symbols of R_F) such that for every $t \in T(F)$, if $x_1, ..., x_N \in L(t)$ and if $[U_j, i_1, ..., i_n]$ are sets of nodes of t, then

$$\| t \| \models \Psi(x_1, ..., x_N, ..., [U_j, i_1, ..., i_n], ...) \quad \text{iff} \quad [U_j, i_1, ..., i_n] = U_j[i_1, ..., i_n]_{Z,\bar{x}} \qquad (*)$$

for all relevant $[U_j, i_1, ..., i_n]$. It follows, in particular, from the definition of Σ that there is one and only one tuple of sets $[U_j, i_1, ..., i_n]$ satisfying Ψ (the variables $x_1, ..., x_N$ having values defined in $\bar{x}$). Hence one can construct formulas

$$\Psi_{j,i_1,...,i_n}(W, x_1, ..., x_N)$$

such that

$$\| t \| \models \Psi_{j,i_1,...,i_n}(W, x_1, ..., x_N) \quad \text{iff} \quad W = [U_j,\ i_1, ..., i_n] (= U_j[i_1, ..., i_n]_{Z,\bar{x}})$$

where $[U_j,\ i_1, ..., i_n]$ is a member of the unique family of sets satisfying (*). From this equivalence, it follows that in the formula $\Psi_{j,i_1,...,i_n}(W, x_1, ..., x_N)$, the variables x_k for k not in the set $\{i_1, ..., i_n\}$ play no role; their value does not influence the validity of the formula. Now the MS-formula with free variables $w, x_1, ..., x_n$:

$$\exists W(w \in W \wedge \Psi_{j,1,2,...,n}(W, x_1, ..., x_n, x_n, ..., x_n))$$

defines the relation $U_{j,Z}$. □

5. Abstract presentation of a proof method

In order to clarify the organization of the proof of our main Theorem, and to help perhaps in its application in other cases, we present it for a general set $\mathcal{Q}$ of *qfd* operations for which we assume the existence of a large enough but finite set Θ of auxiliary formulas. The application to $\mathcal{Q} = \mathcal{QFD}_{P,R}$ will be done in Section 6 by an explicit construction of the necessary set Θ.

5.1. *P-restricted formulas*

We say that two formulas to be interpreted in structures in $\mathcal{S}_P(R)$ are *P-equivalent* if they can be transformed into each other by use of Boolean laws, renamings of quantified variables, valid permutations of quantifications ($\exists$ and $\exists$, or $\forall$ and $\forall$), transformation of $\forall x.(A(x) \vee B)$ into $\forall x.A(x) \vee B$ if x is not free in B, and similar laws, together with simplifications based on the following *partition axioms*, which hold in all structures in $\mathcal{S}_P(R)$:

$$\begin{aligned} p(x) \wedge q(x) &\Leftrightarrow \mathit{false} \ (\text{if } p \neq q), \\ \bigvee_{p \in P} p(x) &\Leftrightarrow \mathit{true}. \end{aligned}$$

In other words, if two formulas are P-equivalent they are logically equivalent in all structures in $\mathcal{S}(R)$ satisfying the partition axioms. The P-equivalence of two formulas is

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

decidable. We use $FO(P, R, X)$ to denote the set of first-order formulas over $P \cup R$ with free variables in X, or rather of P-equivalence classes of such formulas.

Definition 5.1. A formula φ with free variables $x_1, \ldots, x_n$ is *P-restricted* if it is P-equivalent to a formula of the form

$$p_1(x_1) \wedge p_2(x_2) \wedge \ldots \wedge p_n(x_n) \wedge \psi,$$

where $p_1, \ldots, p_n \in P$, and all quantifications in ψ are *restricted by predicates in P*, that is, are of the form $\forall y : p.\theta$ (which stands for $\forall y.(p(y) \Rightarrow \theta)$) or $\exists y : p.\theta$ (which stands for $\exists y.(p(y) \wedge \theta)$).
Such variables y in θ are said to be in the *scope* of p. A free variable x_i is in the *scope* of p_i.

Since predicate symbols in P represent disjoint sets, a formula $x = y$ where x and y are in the scopes of different predicates can be replaced by $false$. Similarly, if x is in the scope of p, then an atomic formula $q(x)$ for $q \neq p$, $q \in P$, can be replaced by $false$. We will assume that all such simplifications are done, and that a formula is always replaced by a shortest P-equivalent one.

We use $RFO(P, R, X)$ to denote the set of P-restricted formulas over $R \cup P$ with free variables in X. (We will omit P, R, when they are clear from the context.)

Lemma 5.1. Every first-order formula $\varphi \in FO(P, R, X)$ is P-equivalent to a disjunction of P-restricted formulas.

Proof. Let φ be the first-order formula with free variables $x_1, \ldots, x_n$. First we note that it is P-equivalent to the disjunction of the formulas

$$p_1(x_1) \wedge \ldots \wedge p_n(x_n) \wedge \varphi \qquad (+)$$

where the disjunction extends to all sequences $(p_1, \ldots, p_n) \in P^n$.
We have to deal with the quantified variables in φ. Let us enumerate P as $\{q_1, \ldots, q_m\}$. We can replace recursively in φ each subformula:

— $\exists y.\theta$ by $\exists y : q_1.\theta' \vee \exists y : q_2.\theta' \vee \ldots \vee \exists y : q_m.\theta'$
and:
— $\forall y.\theta$ by $\forall y : q_1.\theta' \wedge \forall y : q_1.\theta' \wedge \ldots \wedge \forall y : q_m.\theta'$

where θ' is obtained similarly from θ. Each formula $(+)$ can thus be transformed into a P-equivalent P-restricted formula. □

We now prove a 'backwards translation' lemma for the operation $fuse_p$.

Lemma 5.2. For all $n \geqslant 0$ and $p \in P$, and every formula $\theta \in RFO(P, R, \{x_1, \ldots, x_n\})$ one can build a formula θ_{fuse_p} in $RFO(P, R, \{x_1, \ldots, x_n\})$ such that, for every structure $S \in \mathscr{S}_P(R)$ such that $p_S \neq \emptyset$, for every $d_1, \ldots, d_n \in D_S$ we have

$$fuse_p(S) \models \theta([d_1], \ldots, [d_n]) \text{ iff } S \models \theta_{fuse_p}(d_1, \ldots, d_n),$$

where we use $[d_i]$ to denote the element of the domain of $fuse_p(S)$ corresponding to d_i, that is, the equivalence class of d_i under the relation $d \approx d'$ iff $d = d' \vee (p_S(d) \wedge (p_S(d'))$.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Proof. If p does not occur in θ, we let $\theta_{fuse_p} = \theta$. Otherwise, we assume that θ is of the form

$$p_1(x_1) \wedge p_2(x_2) \wedge \ldots \wedge p_n(x_n) \wedge \psi$$

We let θ_{fuse_p} be

$$p_1(x_1) \wedge p_2(x_2) \wedge \ldots \wedge p_n(x_n) \wedge \psi_p,$$

where ψ_p is defined inductively on subformulas φ of ψ as follows:

(i) φ is an atomic formula.
Consider the case where φ is $z_1 = z_2$. If z_1, z_2 are in the scope of $q \neq p$, then φ_p is φ. If they are in the scope of p, we let φ_p be *true*.
We can assume that ψ is simplified so as not to contain atomic formulas $z_1 = z_2$ with z_1, z_2 in the scopes of different predicates because such formulas are always false in the structures in $\mathscr{S}_P(R)$.
Similarly, we can assume that there are no atomic formulas of the form $q(y)$ for q in P except those occurring in the restrictions of quantifications in $\forall y : q$ and $\exists y : q$.
The last case is that of ϕ with the form $A(z_1, \ldots, z_m)$ where the variables z_i are in $\{x_1, \ldots, x_n\}$ or are quantified. We let $i_1, \ldots, i_k$ be the increasing list of indices $i \in \{1, \ldots, m\}$ such that z_i is in the scope of p. Then we let

$$\phi_p = \exists y_1, y_2, \ldots, y_k : p.A(z'_1, \ldots, z'_m)$$

where $y_1, \ldots, y_k$ are new variables and

$$z'_j = \begin{cases} z_j & \text{if } z_j \text{ is not in the scope of } p \\ y_\ell & \text{if } j = i_\ell \text{ for some } \ell \in \{1, \ldots, k\}. \end{cases}$$

(ii) If φ is $\neg\varphi'$, $\varphi' \wedge \varphi''$ or $\varphi' \vee \varphi''$, we let φ_p be $\neg\varphi'_p$, $\varphi'_p \wedge \varphi''_p$ or $\varphi'_p \vee \varphi''_p$, respectively.
(iii) If φ is $Qy : q.\varphi'$ with $Q \in \{\exists, \forall\}, q \neq p$, we let φ_p be $Qy : q.\varphi'_p$.
(iv) If φ is $Qy : p.\varphi'$, then φ_p is φ'_p.

Note that in case (iv), y has no free occurrence in φ'_p. Since we consider structures S such that $p_S \neq \emptyset$, the property $\exists y : p.true$ is valid and $\exists y : p.\varphi'_p$ is equivalent to φ'_p. Similarly, $\forall y : p.\varphi'_p$ is equivalent to φ'_p.
Next we outline the proof of validity of this construction by induction on subformulas φ of ψ.

We consider a subformula φ with free variables among $x_1, \ldots, x_n, y_1, \ldots, y_m$ in the scope of $p_1, \ldots, p_n, q_1, \ldots, q_m$, respectively. Now we want to prove that we have for $d_1, \ldots, d_n, d'_1, \ldots, d'_m$

$$fuse_p(S) \models \varphi([d_1], \ldots, [d'_1], \ldots) \text{ iff } S \models \varphi_p(d_1, \ldots, d'_1, \ldots).$$

We review the various cases of the inductive definition above.

(i) If φ is $z_1 = z_2$, and both variables are in the scope of $q \neq p$, then $[f_1] = [f_2]$ iff $f_1 = f_2$, since $[f_i] = \{f_i\}$ for $i = 1, 2$ (We use $f_i, i = 1, 2$ to denote the elements of $\{d_1, \ldots, d'_1, \ldots\}$ taken as value of z_i).
If φ is $z_1 = z_2$ with both variables in the scope of p, then $[f_1] = [f_2]$ is always true in $fuse_p(S)$ since $[f_1] = [f_2] = p_S$.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Next comes the case of φ equal to $A(z_1, \dots, z_m)$, with f_i the value of z_i:
$A([f_1], \dots, [f_m])$ holds in $fuse_p(S)$ iff $A(f'_1, \dots, f'_m)$ holds for some $f'_i \in [f_i], i = 1, \dots, m$; this is true by (i) above, since if z_i is in the scope of p, then $[f_i] = p_S$, otherwise $[f_i] = \{f_i\}$. This concludes the proof of the validity of the translation of $A(z_1, \dots, z_m)$.

(ii) This is straightforward.

(iii) This is straightforward.

(iv) If φ is $Qy : p.\varphi'_p$, we want to prove that

$$fuse_p(S) \models Qy : p.\varphi'_p([d_1], \dots, [d'_m]) \text{ iff } S \models \varphi'_p([d_1], \dots, [d'_m]). \tag{1}$$

where, by clause (iv), φ_p is φ'_p.
Since p_S is non-empty and y is not free in φ'_p (by the transformations of atomic formulas), (1) is equivalent to

$$fuse_p(S) \models \varphi'([d_1], ..., [d'_1], ...)$$

which is equivalent (by the induction hypothesis) to

$$S \models \varphi'_p([d_1], ..., [d'_1], ...)$$

Hence, we have the desired equivalence. □

Example. If θ is

$$p(x_1) \wedge q(x_2) \wedge \forall z : p.\exists z' : p.((\neg A(x_1, x_2, z, z') \wedge (B(z, z') \vee x_1 = z)),$$

then θ_{fuse_p} is

$$p(x_1) \wedge q(x_2) \wedge (\neg(\exists y, y', y'' : p.A(y, x_2, y', y'')) \wedge (\exists y, y' : p.B(y, y') \vee true))$$

The latter can be simplified to

$$p(x_1) \wedge q(x_2) \wedge \forall y, y', y'' : p.\neg A(y, x_2, y', y'').$$

We can now state a 'backwards translation' lemma for *qfd* operations.

Lemma 5.3. Let

$$f \in \mathcal{2F D}_{P,R} \cup \{ren_{p \to q} / p \neq q, p, q \in P\}$$

and

$$\theta \in RFO(P, R, \{x_1, ..., x_n\}).$$

One can build a formula $\theta_f \in FO(P, R, \{x_1, ..., x_n\})$ such that for every $S \in \mathcal{S}_P(R)$ and every $d_1, ..., d_n \in D_S$ we have

$$f(S) \models \theta(d_1, ..., d_n) \Leftrightarrow S \models \theta_f(d_1, ..., d_n).$$

If $f \in \mathcal{2F D}_{P,R}$, then $\theta_f \in RFO(P, R, \{x_1, ..., x_n\})$.

If $f = ren_{p \to q}$, then $\theta_f = \theta_{f,1} \vee .. \vee \theta_{f,k}$, where we have that each formula $\theta_{f,i}$ is in $RFO(P, R, \{x_1, ..., x_n\})$.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Proof. If f is in $\mathcal{2FD}_{P,R}$ and it is defined by the *qfd* formulas $(\varphi_A)_{A\in R}$, $\varphi_A \in QF(P,R,\{u_1,...,u_{\rho(A)}\})$, then θ_f is obtained from θ by the replacement of every atomic formula $A(w_1,...,w_{\rho(A)})$ by $\varphi_A[w_1/u_1,...,w_{\rho(A)}/u_{\rho(A)}]$. It is clear that θ_f is P-restricted if θ is.

For $f = ren_{p\to q}$, one obtains θ_f by replacing subformulas:

(i) $p(x)$ by *false*,
(ii) $\exists y : p.\theta'$ by *false*,
(iii) $\forall y : p.\theta'$ by *true*,
(iv) $q(x)$ by $p(x) \vee q(x)$,
(v) $\exists y : q.\theta'$ by $(\exists y : p.\theta'_f) \vee (\exists y : q.\theta'_f)$
(vi) $\forall y : q.\theta'$ by $(\forall y : p.\theta'_f) \wedge (\forall y : q.\theta'_f)$.

Because of case (iv), the formula θ_f is not always P-restricted. However, it can be transformed into a disjunction of P-restricted formulas. □

For $f = ren_{p\to q}$, we will denote θ_f by $\theta_{p\to q}$ and the components of this disjunction by $\theta_{p\to q,1},...,\theta_{p\to q,k}$.

Example 5.1. Let $f = ren_{p\to q}$ and θ be the formula

$$r(x) \wedge q(y) \wedge \forall u : q.((A(x,y,u) \wedge \forall w : p.A(x,u,w)) \vee \exists w : p.A(y,u,w)).$$

We obtain for θ_f

$$r(x) \wedge (p(y) \vee q(y)) \wedge \forall u : p.((A(x,y,u) \wedge true) \vee false) \\ \wedge \forall u : q.((A(x,y,u) \wedge true) \vee false)\ ,$$

which is P-equivalent to the disjunction of the following formulas:

$$r(x) \wedge p(y) \wedge \forall u : p.A(x,y,u) \wedge \forall u : qA(x,y,u)$$
$$r(x) \wedge q(y) \wedge \forall u : p.A(x,y,u) \wedge \forall u : qA(x,y,u)\ .$$

Definition 5.2. Let φ be a formula in $FO(P,R,X)$ such that all quantifications are P-restricted. We define a finite set of words over P, denoted by $Nest(\varphi)$, by the following induction on the structure of φ:

— $Nest(\varphi) = \{\varepsilon\}$ (the empty word) if φ is quantifier-free;
— $Nest(\varphi_1 \wedge \varphi_2\ ) = Nest(\varphi_1 \vee \varphi_2\ ) = Nest(\varphi_1) \cup Nest(\varphi_2)$;
— $Nest(\neg\varphi) = Nest(\varphi)$;
— $Nest(\forall u : q.\varphi) = Nest(\exists u : q.\varphi) = Nest(\varphi) \cup q.Nest(\varphi)$.

Every nesting of quantifications in φ is represented by a word in $Nest(\varphi)$. From the last clause of the definition it follows that every subword of a word in $Nest(\varphi)$ is also in this set. The words of maximal length correspond to nestings of quantifications.

Using the terminology of Courcelle (1997), the following lemma says that first-order properties are inductive with respect to disjoint union. (This result is known from Feferman and Vaught; see Courcelle (1997) for its extension and use in MS logic.)

If $\overline{d} = (d_1,...,d_N) \in D^N$ and $E \subseteq D$, we use $\overline{d} \cap E$ to denote the sequence obtained from $\overline{d}$ by deleting the elements not in E.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Lemma 5.4. Let $\theta \in RFO(P, R, \{x_1, ..., x_N\})$. Let P_1 be a non-empty proper subset of P, and $P_2 = P - P_1$.

Let $V_1 \cup V_2$ be the partition of $\{x_1, ..., x_N\}$ such that if x_i is free in θ and is in the scope of some $p \in P_1$, then $x_i \in V_1$, and otherwise $x_i \in V_2$.

One can determine $k \in \mathbf{N}$ and construct formulas $\theta_1, ..., \theta_k \in RFO(P_1, R, V_1)$ and formulas $\theta'_1, ..., \theta'_k \in RFO(P_2, R, V_2)$ such that $Nest(\theta_i)$, $Nest(\theta'_i) \subseteq Nest(\theta)$ for each i, and for every $S_1 \in \mathscr{S}_{P_1}(R)$ and every $S_2 \in \mathscr{S}_{P_2}(R)$ with disjoint domains, and for every $\overline{d} = (d_1, .., d_N) \in (D_{S_1} \cup D_{S_2})^N$ such that $d_i \in D_{S_1}$ iff $x_i \in V_1$, we have $S_1 \oplus S_2 \models \theta(\overline{d})$ iff $S_1 \models \theta_i(\overline{d} \cap D_{S_1})$ and $S_2 \models \theta'_i(\overline{d} \cap D_{S_2})$ for some $i = 1, ..., k$.

Proof. The proof is by induction on the structure of θ in a standard way as in Courcelle (1987, Lemma 4.5). We do not repeat the routine details. See also Lemma 3.3 of Courcelle (1992). These lemmas state similar results for MS logic. □

5.2. *A sufficient condition for the main result*

Definition 5.3 (Separated terms). We let $\mathscr{F} = \mathscr{F}_P \cup \mathscr{F}use_P \cup 2\mathscr{F}\mathscr{D}_{P,R}$. We say that a term $t \in T(\mathscr{F})$ is *separated* if for every subterm of t of the form $t_1 \oplus t_2$, we have $\tau(val_{\mathscr{F}}(t_1)) \cap \tau(val_{\mathscr{F}}(t_2)) = \varnothing$. (We recall that for $S \in \mathscr{S}_P(R), \tau(S) = \{p \in P \ / \ p_S \neq \varnothing\}$.) We use $T_{sep}(\mathscr{F})$ to denote the set of separated terms.

Definition 5.4 ((P, R, Q)-closed sets of formulas). Let Q be a subset of $2\mathscr{F}\mathscr{D}_{P,R}$. For each $A \in R$ and $w = (p_1, ..., p_n) \in P^n, n = \rho(A)$, we use $\beta_{A,w}$ to denote the P-restricted formula $p_1(x_1) \wedge ... \wedge p_n(x_n) \wedge A(x_1, ..., x_n)$.

We let $N = Max\{\rho(A)/A \in R\}$. We say that a subset Θ of $RFO(P, R, \{x_1, ..., x_N\})$ is (P, R, Q)-closed if the following conditions hold:

(C1) It contains the formulas $\beta_{A,w}$ for all $A \in R$ and all $w \in P^{\rho(A)}$.

(C2) For every $\theta \in \Theta$ and every $p \in P$, the formula θ_{fuse_p} constructed in the proof of Lemma 5.2 is P-equivalent to a formula in Θ.

(C3) For every $\theta \in \Theta$, every $f \in Q$ and every $p, q \in P$ with $p \neq q$, the formulas θ_f and $\theta_{p \to q,i}$ constructed in the proof of Lemma 5.3 are P-equivalent to formulas in Θ.

(C4) For every $\theta \in \Theta$ and for every partition of P into two non-empty sets P_1, P_2, the formulas $\theta_1, ..., \theta_k, \theta'_1, ..., \theta'_k$ constructed in the proof of Lemma 5.4 are P-equivalent to formulas in Θ.

Proposition 5.5. Let Q be a subset of $2\mathscr{F}\mathscr{D}_{P,R}$ such that there exists a finite (P, R, Q)-closed set Θ. The evaluation mapping $T_{sep}(\mathscr{F}_P \cup \mathscr{F}use_P \cup Q) \to \mathscr{S}_P(R)$ is a non-copying MS-transduction.

Proof. We let $\mathscr{F} = \mathscr{F}_P \cup \mathscr{F}use_P \cup Q$. We let Θ be a finite (P, R, Q)-closed subset of $RFO(P, R, \{x_1, ..., x_N\})$. We consider $t \in T_{sep}(\mathscr{F})$.

Let $\theta \in \Theta$ and n be the smallest integer such that θ is in $RFO(P, R, \{x_1, ..., x_n\})$. For $d_1, ..., d_n \in L(t)$ (the set of leaves of t) we let $Val(\theta, t, d_1, ..., d_n)$ denote the set of nodes u of t such that

$$d_1 \leqslant u, ..., d_n \leqslant u \text{ and } val_{\mathscr{F}}(t/u) \models \theta([d_1]_u, ..., [d_n]_u), \tag{1}$$

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

where $[d_i]_u$ is the equivalence class of d_i with respect to the equivalence relation $\approx_{t/u}$. This equivalence relation originates from the fusion operations in t/u that make different leaves of t/u into a same element of $val_{\mathcal{F}}(t/u)$.

Goal Our objective is to prove that the $(n+1)$-ary relations on the set of nodes of t defined by

$$(u, d_1, ..., d_n) \in U_\theta(t) \text{ iff } u \in Val(\theta, t, d_1, ..., d_n)$$

(where n depends on θ) are MS-definable in $\| t \|$ for each $\theta \in \Theta$ by MS-formulas constructed from Θ and θ. For this it suffices to prove that the relations $U_\theta(t)$ satisfy a system of inductive definitions as considered in Proposition 4.1.

Considering t as fixed, we let Z be the tree-based $\mathcal{U}$-structure $(\| t \|, (U_\theta(t))_{\theta \in \Theta})$, where $\mathcal{U} = \{U_\theta / \theta \in \Theta\}$.

We will express properties of the form $(u, d_1, ..., d_n) \in U_\theta(t)$ in terms of conditions of the form $(u', d_{i_1}, ..., d_{i_m}) \in U_{\theta'}(t)$ such that u' is a successor of u, and $(d_{i_1}, ..., d_{i_m})$ is the subsequence of $(d_1, ..., d_n)$ consisting of its elements below u', and $\theta' \in \Theta$.

By the last assertion of Lemma 3.1, we get that for every $\theta \in RFO(P, \varnothing, \{x_1, ..., x_n\})$, the $(n+1)$-ary relation $U_\theta(t)$ is MS-definable in $\| t \|$. Hence we need only introduce as unknown relations into Σ the relations $U_\theta(t)$ for θ containing at least one occurrence of a symbol in R.

We examine the property $(u, d_1, ..., d_n) \in U_\theta(t)$ for those $\theta's$. There are several cases depending on the label of u. We represent the nodes of the tree as Dewey words over the alphabet $\{1,2\}$, hence $u1, u2$ are the first and second successors of u. We let $S(t') = val_{\mathcal{F}}(t')$ for t' a subterm of t.

Case 1 $u = \mathbf{p}$.

We let $B_{U_\theta, \mathbf{p}}$ be the truth value '$\mathbf{p} \models \theta(s, ..., s)$', where s is the unique element of $\mathbf{p}$.

Hence $(u, d_1, ..., d_n) \in U_\theta(t)$ iff $d_1 \leqslant u, ..., d_n \leqslant u$ and $B_{U_\theta, \mathbf{p}}$ is *true*.

Note that since u is a leaf, $d_1 \leqslant u \wedge ... \wedge d_n \leqslant u$ is equivalent to $d_1 = d_2 = ... = d_n = u$.

The notation $B_{U_\theta, \mathbf{p}}$ is as in Definition 4.1

Case 2 The node u is labelled by $fuse_p$.

The node u has the unique successor $u1$ and $S(t/u) = fuse_p(S(t/u1))$.

By Condition C2 of Definition 5.4 the formula θ_{fuse_p} constructed in Lemma 5.2 is (P-equivalent to a formula) in Θ. We have by Lemma 5.2 that

$$u \in Val(\theta, t, d_1, ..., d_n)$$

iff either

$$p_{S(t/u1)} = \varnothing \text{ and } u1 \in Val(\theta, t, d_1, ..., d_n)$$

or

$$p_{S(t/u1)} \neq \varnothing \text{ and } u1 \in Val(\theta_{fuse_p}, t, d_1, ..., d_n).$$

The property $P_{U_\theta, fuse_p}(Z[t//u], x_1, ..., x_n)$ of Definition 4.1 can be written as follows:

$$\exists u'.[rt_1(u') \wedge \{(A \wedge B) \vee (\neg A \wedge C)\}]$$

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

where A is the formula

$$\forall x.(\pi_1(u', x) \Rightarrow \neg\pi_{2,p}(u', x))$$

(it expresses that $p_{S(t/u1)} = \varnothing$ because u' in $Z[t//u]$ satisfies rt_1 iff it is equal to $u1$); the formulas π_1 and $\pi_{2,p}$ are defined in Lemma 3.1. The formula B is $U_\theta(u', x_1, ..., x_n)$, and C is the formula

$$U_{\theta_{fuse_p}}(u', x_1, ..., x_n).$$

We have thus a formula $P_{U,f}$ of the desired form since the formulas π_1 and $\pi_{2,p}$ are monadic second-order (by Lemma 3.1) and without occurrences of symbols in $\mathscr{U}$.

Case 3 The node u is labelled by $f \in Q \cup \{ren_{p\to q}/p \neq q\}$. We consider first the case of $f = ren_{p\to q}$. The formulas $\theta_{p\to q,i}$ $(1 \leqslant i \leqslant k)$ constructed in Lemma 5.3 are (P-equivalent to formulas) in Θ. We thus have

$u \in Val(\theta, t, d_1, ..., d_n)$ iff

$u1 \in Val(\theta_{p\to q,1}, t, d_1, ..., d_n) \vee ... \vee u_1 \in Val(\theta_{p\to q,k}, t, d_1, ..., d_n)$.

Hence $P_{U_\theta, ren_{p\to q}}$ is

$$\exists u'.[rt_1(u') \wedge (U_{\theta_{p\to q,1}}(u', x_1, ..., x_n) \vee ... \vee U_{\theta_{p\to q,k}}(u', x_1, ..., x_n))].$$

The case of $f \in Q$ is similar and even simpler using the single formula θ_f instead of a disjunction.

Case 4 The node u is labelled by $\oplus$. Its two successors are $u1$ and $u2$. As elements of $Z[t//u]$ they are the unique elements u' of its domain satisfying rt_1 and rt_2, respectively. We have $S(t/u) = S(t/u1) \oplus S(t/u2)$.

Since t is separated, we have $\tau(S(t/u1)) \cap \tau(S(t/u2)) = \varnothing$.

We first observe that for $Q \subseteq P$, for each $i = 1, 2$, the property '$\tau(S(t/ui)) \subseteq Q$' is expressible in $Z[t//u]$ by the formula $A_{i,Q}$ defined as follows:

$$\exists u'.[rt_i(u') \wedge \forall x(\pi_1(u', x) \Rightarrow \bigvee_{p\in Q} \pi_{2,p}(u', x))].$$

The formula $P_{U_\theta,\oplus}$ we want to construct will be taken to have the form

$$\bigvee A_{1,Q} \wedge A_{2,P-Q} \wedge P_{U_{\theta,}\oplus,Q},$$

where the disjunction extends to all non-empty proper subsets Q of P, and $P_{U_{\theta,}\oplus,Q}$ says that u belongs to $Val(\theta, t, d_1, ..., d_n)$ under the condition that $\tau(S(t/u1)) \subseteq Q$ and $\tau(S(t/u2)) \subseteq P - Q$.

Let $\theta_1, ..., \theta'_k$ be the formulas associated with θ by Lemma (5.4) (used with $P_1 = Q$). They are (P-equivalent to formulas) in Θ. Hence, by this lemma, $P_{U_{\theta,}\oplus,Q}$ can be taken to be equal to

$$\exists u', u''[rt_1(u') \wedge rt_2(u'') \wedge \bigvee_{1\leqslant i\leqslant k} U_{\theta_i}(u', x_1, ..., x_n) \wedge U_{\theta'_i}(u'', x_1, ..., x_n)].$$

From these formulas we get $P_{U_\theta,\oplus}$ as desired.

Since we have defined all necessary formulas $B_{U_\theta,\mathbf{p}}$ and $P_{U_\theta,f}$ for $\theta \in \Theta$, $p \in P$, $f \in \mathscr{F}_+$, we have a system Σ satisfying the conditions of Definition 4.1.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Hence by Proposition 4.1, it has a unique solution, and the corresponding relations U_Z for $U \in \mathcal{U}$ are MS-definable in $\|t\|$. Hence this proves that the transduction

$$\|t\| \longmapsto Z = (\|t\|, (U_\theta(t))_{\theta \in \Theta})$$

is MS-definable and non-copying.

Since the composition of two non-copying MS-transductions is a non-copying MS-transduction, it is enough to prove that the transduction $Z \longmapsto S(t)$ is also MS-definable and non-copying.

We recall that $D_{S(t)}$ is $L(t)/\approx$, where by Lemma 3.1, $\approx$ is an equivalence relation that is MS-definable in $\|t\|$. It also follows from Lemma 3.1.ii that each unary relation $p_{S(t)}$ is the quotient by $\approx$ of an MS-definable unary relation on $L(t)$. Finally, we recall that

$$A_{S(t)} = \{([d_1], ..., [d_n]) \ / \ (d_1, ..., d_n) \in A_t\},$$

where $A_t \subseteq L(t)^n$ is defined as the set of tuples $(d_1, ..., d_n)$ such that $U_{\beta_{A,w}}(u, d_1, ..., d_n)$ holds for some $u \in N(t)$ satisfying $root_t$ and some $w \in P^n$. Hence this relation is also MS-definable in Z since each $\beta_{A,w}$ belongs to Θ. The relation $A_{S(t)}$ is its quotient by $\approx$. The transduction $Z \longmapsto S(t)$ is thus MS-definable and non-copying since Lemma 2.1 ensures this for quotients by MS-definable equivalences. This concludes the proof. □

Let $\mathcal{F}$ and $\mathcal{F}'$ be two sets of operations on $\mathcal{S}_P(R)$. We say that $t \in T(\mathcal{F})$ is *equivalent* to $t' \in T(\mathcal{F}')$ if $val_{\mathcal{F}}(t)$ is isomorphic to $val_{\mathcal{F}'}(t')$.

The following lemma will help to lift the restriction to separated terms of Proposition 5.1.

Lemma 5.6. Every term $t \in T(\mathcal{F}_P \cup \mathcal{F}use_P \cup \mathcal{2F}\mathcal{D}_{P,R})$ can be transformed into an equivalent separated term t' in $T(\mathcal{F}_{P'} \cup \mathcal{F}use_P \cup \mathcal{2F}\mathcal{D}_{P,R})$, where $P' \supseteq P$ and $Card(P') = 2 \cdot Card(P)$. Moreover, this transformation can be formalized by a non-copying MS-transduction mapping $\| t \|$ to $\| t' \|$.

Proof. Let us observe that if $\tau(S(t_1)) \cap \tau(S(t_2)) = \{p_1, \ldots, p_n\}$, then $t_1 \oplus t_2$ is equivalent to

$$ren_{q_1 \to p_1, \ldots, q_n \to p_n}(t_1 \oplus ren_{p_1 \to q_1, p_2 \to q_2, \ldots, p_n \to q_n}(t_2)),$$

where $q_1, \ldots, q_n$ are pairwise distinct and not in $\tau(S(t_1)) \cup \tau(S(t_2))$. Hence, using this fact whenever necessary, we can obtain a separated term equivalent to the given t. The replacement of $\oplus$ by a combination of $\oplus$ and renamings can be performed by a linear deterministic bottom-up tree transducer. This follows because $\tau(S(t'))$, for t' a subterm of t, can be determined by the computation on t of a finite bottom-up automaton. By Courcelle (1997, Theorem 5.5.3), this transformation is MS-definable and non-copying. □

6. The main theorem

We will apply Proposition 5.5 to the set $\mathcal{2F}\mathcal{D}_{P,R}$ for each pair P, R and obtain our main theorem.

Theorem 6.1 (Main theorem). For all finite sets R, P of relation symbols such that those of P are unary, the evaluation mapping : $T(\mathcal{F}_P \cup \mathcal{F}use_P \cup \mathcal{2F}\mathcal{D}_{P,R}) \to \mathcal{S}_P(R)$ is a non-copying MS-definable transduction.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

In order to apply Proposition 5.5, we have to determine Θ. By a predicate we mean a relation symbol in P.

Definition 6.1 (M-limited formulas). Let M be a finite multiset of elements of P. An *M-limited first-order formula* is a formula φ in $FO(P \cup R, X)$ such that all quantifications are restricted by predicates belonging to M (as in Definition 5.1) and, furthermore, such that for each $p \in M$, the number of occurrences of p in M is not smaller than the number of nested quantifications restricted by p, that is, no smaller than the maximum number of occurrences of p in a word in $Nest(\varphi)$.

Example. Let $M = \{p, p, p, p, q\}$, and let $r \in P, r \notin M$. The formula φ defined as

$$r(x) \wedge \exists y, y', y'' : p.(y \neq y' \wedge y' \neq y'' \wedge y'' \neq y \wedge \neg A(x, y))$$

is M-limited. It is not M'-limited for $M' = \{p, p, q, r\}$. For to be Q-limited, φ needs a 'resource' multiset Q with at least three occurrences of p. The formula

$$r(x) \wedge \exists y : p.(A(x, y) \wedge \forall y' : p.\neg B(x, y'))$$

is not $\{p\}$-limited as written. However, it is P-equivalent to the formula

$$r(x) \wedge \exists y : p.A(x, y) \wedge \forall y' : p.\neg B(x, y'),$$

which is $\{p\}-$limited.

We use $LFO_M(X)$ to denote the set of M-limited formulas with free variables in X. We assume that P and R are finite and fixed; we will use the notation $LFO_M(P, R, X)$ if we need to specify them. We use $B(\Phi)$ to denote the set of Boolean combinations of formulas in a set Φ.

Lemma 6.2. The set $LFO_M(X)$ can be defined by induction on the size of M with the help of auxilliary sets $LFO'_M(X)$ as follows:

$$LFO_\varnothing(X) = QF(P, R, X) \tag{2}$$

$$LFO_M(X) = B(LFO'_M(X) \cup \bigcup\{LFO_N(X) \ / \ N \subsetneq M\}) \tag{3}$$

$$LFO'_M(X) = \{Qy : p.\theta \ / \ \theta \in LFO_{M-\{p\}}(X \cup \{y\}),\ p \in M\} \tag{4}$$

where $M \neq \varnothing$, $M - \{p\}$ denotes M minus one occurrence of p, and $Q \in \{\exists, \forall\}$.

Lemma 6.3. For every finite set of variables X and for every finite multiset M of elements of P, the set $LFO_M(X)$ is finite up to P-equivalence. If φ is M-limited and ψ is a formula such that $Nest(\psi) \subseteq Nest(\varphi)$, then ψ is also M-limited.

Proof. If Φ is finite, so is $B(\Phi)$. In particular, $QF(P, R, X)$ is finite (up to P-equivalence in both cases). The proof is then straightforward from the above definitions by induction on the cardinality of M simultaneously for all finite sets of variables.
The second assertion is clear from the definitions. □

Lemma 6.4. Every formula φ in $LFO_M(X)$ is P-equivalent to a disjunction of P-restricted formulas from the same set.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Proof. We use a similar technique as in Lemma 5.1. All quantifications are of the appropriate form. We need only consider the formulas

$$p_1(x_1) \wedge \ldots \wedge p_n(x_n) \wedge \varphi$$

where $\varphi \in LFO_M(\{x_1, \ldots, x_n\})$, for all $p_1, \ldots, p_n \in P$, and we take their disjunction. □

Remark 6.1. Taking the conjunction of φ with atomic formulas like $p_1(x_1) \wedge p_2(x_2)$ may cause inconsistencies. The corresponding formula may be equivalent to *false*. Here is an example. Let $P = \{p, q, r\}$ and φ be the formula

$$\neg r(x) \wedge \exists y : p.(A(x, y) \wedge (\neg B(x, y) \vee q(x)))$$

Then:

— $p(x) \wedge \varphi$ reduces to φ_1 defined as : $p(x) \wedge \exists y : p.(A(x, y) \wedge \neg B(x, y))$,
— $q(x) \wedge \varphi$ reduces to φ_2 defined as : $q(x) \wedge \exists y : p.A(x, y)$,
— $r(x) \wedge \varphi$ reduces to $false$.

Hence φ is P-equivalent to $\varphi_1 \vee \varphi_2$. These simplifications keep the formulas in $LFO_M(X)$ and they can only reduce the sets of words on P representing nestings of quantifications.

We now consider the mapping $\theta \longmapsto \theta_f$ associated with $f \in \mathscr{QFD}_{P,R}$ defined in Lemma 5.3.

Lemma 6.5. For every *qfd* operation f and for every formula θ in $LFO_M(X)$ the formula θ_f belongs to $LFO_M(X)$.

Proof. Recall that the mapping $\theta \longmapsto \theta_f$ consists in substituting $\varphi_A[z_1/x_1, \ldots, z_n/x_n]$ for $A(z_1, \ldots, z_n)$ simultaneously for all $A \in R$ (where $n = \rho(A)$, and $z_1, \ldots, z_n$ are variables). Since the formulas φ_A are quantifier-free, this mapping preserves all sets $LFO_M(X)$, $LFO'_M(X)$. This can be proved by induction on the cardinality of M. □

We let N denote $Max\{\rho(A)/A \in R\}$. The mapping $\theta \longmapsto \theta_{fuse_p}$ was defined in Lemma 5.2. For $p \in P$ and $n \in \mathbf{N}$, we use $n\{p\}$ to denote the multiset $\{p, p, \ldots, p\}$ with n occurrences of p.

Lemma 6.6. Let $p \in P$ and let M contain at least N occurrences of each $q \in P$. If θ is a P-restricted formula in $LFO_M(X)$, so is θ_{fuse_p}.

Proof. Let θ be written as

$$\theta = p_1(x_1) \wedge \ldots \wedge p_n(x_n) \wedge \varphi,$$

where $X = \{x_1, \ldots, x_n\}$. We can assume that φ belongs to $LFO_M(\{x_1, \ldots, x_n\})$ and has no occurrence of atomic formulas $q(x_i)$, because any such occurrence is redundant with the 'external' constraints $p_i(x_i)$ or inconsistent, and hence can be removed. Similarily, there are no atomic formulas of the form $q(y)$, since each quantified variable y is introduced as restricted by some $r, r \in M$.
The transformation of Lemma 5.2 can be described in terms of several steps:

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

(i) Replacement by *true* of certain atomic formulas.
(ii) Deletion of all quantifications restricted by p.
(iii) Replacement of atomic formulas of the form $A(z_1, \ldots, z_m)$ by

$$\exists y_1, \ldots, y_{m'} : p.A(z'_1, \ldots, z'_m)$$

which is in $LFO'_{m'\{p\}}(X \cup \{y_1, \ldots, y_{m'}\})$, where $y_1, \ldots, y_{m'}$ replace in the list $z_1, \ldots, z_m$ the variables that are in the scope of p ($m' \leqslant m \leqslant N$).

It follows that θ_{fuse_p} is built from $N\{p\}$-limited formulas by Boolean combinations and quantifications restricted by predicates $q \neq p$, which are limited by M. Since M contains at least N occurrences of p, the formula θ_{fuse_p} is P-restricted and belongs to $LFO_M(X)$, as desired. □

We now consider similarly the transformation $\theta \mapsto \theta_{p \to q}$ of Lemma 5.3.

Lemma 6.7. Let $p, q \in P, p \neq q$. Let M contain the same number h of occurrences of p and q. If θ is a P-restricted formula in $LFO_M(X)$, then so are the formulas $\theta_{p \to q,i}$.

Proof. The transformation of θ uses two steps.

Step 1 'Killing p'.

All subformulas $\exists y : p.\theta'$ and $p(x)$ are replaced by *false*, and all subformulas $\forall y : p.\theta'$ are replaced by *true*. One obtains a formula in $LFO_{M'}(X)$ where M' consists of M minus the h occurrences of p.

Step 2 Replacement of q by $p \vee q$.

This applies to a formula of the form, say, without loss of generality,

$$q(x_1) \wedge \ldots \wedge q(x_{n'}) \wedge p_{n'+1}(x_{n'+1}) \wedge \ldots \wedge p_n(x_n) \wedge \varphi,$$

where φ by Step 1, has no occurrence of p ($p_s \neq q$ for $n' < s \leqslant n$).

We first consider φ. This step transforms φ into a formula φ' defined as follows by induction on the structure of φ:

(i) φ' is φ if φ is atomic.
(ii) φ' is $\varphi'_1 \wedge \varphi'_2$ (or, respectively, $\varphi'_1 \vee \varphi'_2$, $\neg\varphi'_1$, $Qy : r.\varphi'_1$) if φ is $\varphi_1 \wedge \varphi_2$ (or, respectively, $\varphi_1 \vee \varphi_2$, $\neg\varphi_1$, $Qy : r.\varphi_1$ with $r \notin \{p, q\}$).
(iii) φ' is $\exists y : p.\varphi'_1 \vee \exists y : q.\varphi'_1$ if φ is $\exists y : q.\varphi_1$.
(iv) φ' is $\forall y : p.\varphi'_1 \wedge \forall y : q.\varphi'_1$ if φ is $\forall y : q.\varphi_1$.

Claim For every multiset M_1 with no occurrence of p and h occurrences of q, if φ belongs to $LFO_{M_1}(X)$, the formula φ' belongs to $LFO_{M'_1}(X)$ where $M'_1 = M_1 \cup h\{p\}$.

Proof of claim The proof is by induction on the structure of φ. □

Going back to the main proof, we get that φ' associated with φ belongs to $LFO_M(X)$ since M is M' augmented with h occurrences of p. The replacement of $q(x_i)$ by $p(x_i) \vee q(x_i)$ in θ for $i = 1, \ldots, n'$ yields a formula that is equivalent to a disjunction of P-restricted formulas, all in $LFO_M(X)$, as desired. □

We illustrate Lemma 6.7 with an example.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Example. We let θ be

$$q(x_1) \wedge r(x_2) \wedge \forall y : p.A(x_1, x_2, y) \wedge \forall y : q.\exists z : q.B(x_1, x_2, y, z).$$

Clearly, θ is P-restricted and $\{p, p, q, q\}$-limited. After the first step we get

$$q(x_1) \wedge r(x_2) \wedge \varphi,$$

where φ is

$$\forall y : q.\exists z : q.B(x_1, x_2, y, z).$$

The replacement of q by $p \vee q$ yields for φ' the formula

$$\forall y : p.(\exists z : p.B(x_1, x_2, y, z) \vee \exists z : q.B(x_1, x_2, y, z)) \wedge$$
$$\forall y : q.(\exists z : p.B(x_1, x_2, y, z) \vee \exists z : q.B(x_1, x_2, y, z)),$$

which is clearly $\{p, p, q, q\}$-limited. Now $\theta_{p \to q}$ is

$$(p(x_1) \wedge r(x_2) \wedge \varphi') \vee (q(x_1) \wedge r(x_2) \wedge \varphi'),$$

and hence is the disjunction of two P-restricted $\{p, p, q, q\}$-limited formulas.

Lemma 6.8. Let P, R be finite, let $N = Max\{\rho(A)/A \in R\}$ and let M be the multiset containing N occurrences of each $p \in P$. The set of P-restricted formulas in $LFO_M(\{x_1, ..., x_N\})$ is $(P, R, \mathcal{2F}\mathcal{D}_{P,R})$-closed and finite.

Proof. Let Θ be this set. It is finite by Lemma 6.3. We check conditions (C1)–(C4) of Definition 5.4.

The set Θ contains the formulas $\beta_{A,W}$ for all $A \in R$ and $w \in P^{\rho(A)}$ (hence C1 holds). Condition C2 holds by Lemma 6.6. Condition C3 holds by Lemmas (6.5) and (6.7).

For condition C4 we use Lemma (5.4) and the second part of Lemma 6.3. From the observation about *Nest* in Lemma (5.4) we obtain that the formulas θ_i, θ_i' constructed in this lemma belong to Θ if θ belongs to Θ. □

Proof of Theorem 6.1. We let $\mathcal{F} = \mathcal{F}_P \cup \mathcal{F}use_P \cup \mathcal{2F}\mathcal{D}_{P,R}$. We let $P' \supseteq P$ with $Card(P') = 2.Card(P)$ and $\mathcal{F}' = \mathcal{F}_{P'} \cup \mathcal{F}use_P \cup \mathcal{2F}\mathcal{D}_{P,R}$. The mapping $val_{\mathcal{F}} : T(\mathcal{F}) \to \mathcal{S}_P(R)$ is the composition of the non-copying MS-transduction of Lemma 5.5 that transforms $t \in T(\mathcal{F})$ into an equivalent separated term $t' \in T_{sep}(\mathcal{F}')$ and the non-copying MS-transduction

$$val_{\mathcal{F}'} : T_{sep}(\mathcal{F}') \to \mathcal{S}_{P'}(R).$$

This later transduction is MS-definable and non-copying because we can apply Proposition 5.1 by choosing for Θ the set of P'-restricted formulas in $LFO_M(\{x_1, ..., x_N\})$, where M is the multiset containing N occurrences of each $p \in P'$. It is $(P, R, \mathcal{2F}\mathcal{D}_{P,R})$-closed by Lemma 6.8. Hence $val_{\mathcal{F}}$ is MS-definable and non-copying as it is the composition of two non-copying MS-transductions. □

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

7. Consequences of the main theorem

In this section we review some consequences and applications of the main theorem. They do not concern MS logic but graph grammars and clique-width. However, although MS logic is an essential tool in our proofs, one may certainly find alternative graph theoretical proofs that avoid using logic.

7.1. *Graphs without loops*

We have discussed P-graphs in Section 2.5, and considered them as structures G in $\mathscr{S}_P(\{edg\})$ such that edg_G is irreflexive ($edg_G(x,x) = false$ for all x) and symmetric. The relevant operations are $\mathbf{p}$, $\oplus$, $ren_{p\to q}$, $add_{p,q}$ for $p,q \in P, p \neq q$. They form the set $\mathscr{C}_P$ upon which the definition of clique-width is based.

Theorem 6.1 is applicable because $add_{p,q} \in 2\mathscr{F}\mathscr{D}_{P,\{edg\}}$. However, there is a difficulty: the operation $fuse_p$ creates a loop if it applies to a graph with two neighbour vertices with label p. We can define a loop-free fusion denoted $fuse_p^{n\ell}$ and defined by

$$fuse_p^{n\ell}(G) = fuse_p(del - edge_p(G))$$

where $del - edge_p(G)$ eliminates in G all edges between two vertices labelled by p. This is a *qfd* operation. The formula is

$$\varphi_{edg}(x_1,x_2) \text{ defined as } edg(x_1,x_2) \wedge \neg(p(x_1) \wedge p(x_2)).$$

Let $\mathscr{F}use_P^{n\ell} = \{fuse_p^{n\ell}/p \in P\}$. Consider a term t in $T(\mathscr{C}_P \cup \mathscr{F}use_P^{n\ell})$. By replacing in it every occurrence of $fuse_p^{n\ell}(\ldots)$ by $fuse_p(del - edge_p(\ldots))$, we get a term t' in $T(\mathscr{C}_P \cup \mathscr{F}use_P \cup 2\mathscr{F}\mathscr{D}_{P,\{edg\}})$ that is equivalent to t. The transformation of t into t' is MS-definable, and that of t' into $G(t')$ is also MS-definable by Theorem 6.1, hence, so is that of t into $G(t)$, since MS-transductions are closed under composition.

We thus have a richer set of graph operations, namely $\mathscr{C}_P \cup \mathscr{F}use_P^{n\ell}$, to denote graphs without loops. An alternative notion of clique-width follows immediately:

$$cwd'(G) = Min\{Card(P)/G = G(t), t \in T(\mathscr{C}_P \cup \mathscr{F}use_P^{n\ell})\}.$$

Proposition 7.1.

(i) There exists a function f such that, for every loop-free graph G, we have

$$cwd'(G) \leqslant cwd(G) \leqslant f(cwd'(G)).$$

(ii) For every loop-free graph G,

$$cwd(G) \leqslant 2 \text{ iff } cwd'(G) \leqslant 2 \text{ iff } G \text{ is a cograph.}$$

(iii) There exists a graph G such that $cwd(G) = 4$ but $cwd'(G) = 3$.

Proof. (i) The first inequality is trivial. The graphs having cwd' at most k are the images under an MS-transduction (by Theorem 6.1) of a set of trees. Hence they have bounded cwd (Courcelle 1992; Courcelle 1997; Engelfriet and van Oostrom 1997). Let us denote this bound by $f(k)$. This gives the desired function (which we do not try to evaluate explicitly).

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

(ii) We leave this as an exercise.
(iii) Let G consist of the path $1-2-3-4-5-6-7$ with additional edges $1-3$, $2-6$, $2-7$ and $4-7$. Its *cwd* is more than 3 (which can be checked by hand and verified by the algorithm of Corneil *et al.* (2000)). However, it can be constructed as follows with 3 labels p, q, r:

$$G = add_{r,q}(\mathbf{r} \oplus ren_{r\to p}(fuse_p(H \oplus K)))$$

where the constant $\mathbf{r}$ defines vertex 4.

Without fusion and with p, q, r only, it is easy to find expressions for

— $H = p(2) - q(7) - r(6) - q(5)$ with additional edge $2-6$,
— $K = p(2') - r(1) - q(3)$ with additional edge $2'-3$.

The effect of $fuse_p$ is to fuse vertices 2 and $2'$, which are distinct in $H \oplus K$. (The notation $p(2)$ means that vertex 2 has label p.) □

Proposition 7.2. For every finite set R of relation symbols, there exists an integer function g such that $cwd(G) < g(Card(P))$ for every graph G defined by a term in $T(\mathscr{F}_P \cup \mathscr{F}use_P \cup \mathscr{2F}\mathscr{D}_{P,R})$.

Proof. This is an immediate consequence of Theorem 6.1, since by the results of Courcelle (1992), and Engelfriet and van Oostrom (1997), a set of graphs that is the image of a set of terms over a finite signature by an MS-transduction has bounded clique-width. □

Relation symbols in R that are not binary might be useful at intermediate stages for the construction of graphs. Proposition 7.2 shows that they can be eliminated, but at the cost of enlarging the set P.

Proposition 7.3. Let g be obtained from the previous proposition for $R = \{edg\}$. If G is a graph defined as the union of two graphs H and K with n common vertices, its clique-width is at most $g((n+1) \cdot \max\{cwd(H), cwd(K)\})$.

Proof. We consider H and K as 1-graphs, that is, with all vertices labelled by 1. Let k be the maximum of $cwd(H)$ and $cwd(K)$, and t be a term in $T(\mathscr{C}_k)$ defining H. The labels in this expression are among the integers from 1 to k. Let $s_1, \ldots, s_n$ be the vertices of H belonging to K.

By using $(n+1) \cdot k$ labels (that is, the original labels of t and new labels in bijection with pairs (i, j) for $i = 1, \ldots, n$ and $j = 1, \ldots, k$), we can construct from t an expression t' for the graph H' consisting of H with each vertex s_i labelled by $(i, 1)$ instead of by 1. All other vertices bear the label 1.

We do the same using an expression w for K getting w' that defines K' where the vertices $s_1, \ldots, s_n$ that also belong to H' have the same labels $(1,1), \ldots, (n,1)$. We take the disjoint union of t' and w', and then apply the operations $fuse_{(1,1)}, \ldots, fuse_{(n,1)}$. The resulting graph is G with all vertices labelled by 1 except those common to H and K. We then transform their labels $(i, 1)$ into 1. This gives an expression for G built with fusion operations and using $(n+1) \cdot k$ labels. We then get the desired upperbound by Proposition 7.2. □

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

7.2. *Graphs with loops*

If we want to generate graphs with loops, we may add to $\mathscr{C}_P$ the new constants $\mathbf{p}^\ell$ denoting a single vertex labelled by p incident with a loop. One can then observe that $\mathbf{p}^\ell$ is equivalent to

$$fuse_p(ren_{q\to p}(add_{p,q}(\mathbf{p} \oplus \mathbf{q}))).$$

Hence graphs with loops are also generated by $\mathscr{C}_P \cup \mathscr{F}use_P$.

If we admit loops, it is reasonable to admit also the operations $add_{p,p}$ that create edges between vertices with label p and loops on each of these vertices. We can also remove all loops by using the *qfd* operation $rem - loop$ specified by

$$\varphi_{edg}(x_1, x_2) \text{ defined as } edg(x_1, x_2) \wedge (x_1 \neq x_2).$$

These examples show the flexibility of the formalism.

7.3. *Context-free graph grammars*

The two main classes of context-free graph grammars are:

VR: the class of VR grammars, where VR stands for *Vertex Replacement*, surveyed in Engelfriet and Rozenberg (1997);

HR: the class of HR grammars (HR stands for *Hyperedge Replacement*), surveyed in Drewes *et al.* (1997).

Both classes can be charaterized in terms of systems of recursive equations in sets of finite graphs.

In comparing these classes there is a small technical problem: the HR grammars can generate graphs with multiple edges, whereas VR grammars, by definition, only generate simple graphs. However, the set of simple graphs underlying the graphs of an HR set is also an HR set. By comparing sets of simple graphs (without loops), we get that every HR set is VR and not *vice-versa*. One can decide whether a VR set is HR or not (Courcelle 1995).

Here is another technical difficulty: systems of equations defining HR sets are based on *graphs with sources*, that is, on graphs that have, say n, distinguished vertices labelled by $1, \ldots, n$, respectively, all other vertices being unlabelled, or more conveniently here, labelled by 0. Hence, a graph with n sources is a P-graph for P of cardinality $n+1$. This makes it possible to combine graphs with sources by operations on P-graphs. However, one must restrict the use of these operations (in particular disjoint union and renaming) so that they only produce graphs with sources.

HR and VR sets also have logical characterizations. To recall them, we will use the notion of incidence graph. The *incidence graph* $I(G)$ of a graph G is a bipartite graph having for vertices those of G, on the one hand, and its edges, on the other. It has an edge directed from any vertex representing an edge of G to each of its ends. If L is a set of graphs, $I(L)$ is the set of its incidence graphs.

A set of graphs L is VR iff it is the image of a recognizable set of finite terms under an MS-definable transduction (Courcelle 1992; Engelfriet and van Oostrom 1997). A set

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

of graphs is HR iff $I(L)$ is the image of a recognizable set of finite terms under an MS-definable transduction (Courcelle and Engelfriet 1995), hence, iff $I(L)$ is VR.

Theorem 7.4. Let L be a set of simple loop-free undirected graphs.

(i) L is VR iff it is a component of the least solution of a system of recursive equations written with the operations

$$\oplus, ren_{p\to q}, fuse_p, add_{p,q}$$

and finitely many constants denoting graphs.

(ii) L is HR iff it is a component of the least solution of a system of recusive equations written with the operations

$$\oplus, ren_{p\to q}, fuse_p$$

and finitely many constants denoting graphs.

Hence, only the presence of $add_{p,q}$ distinguishes VR from HR. The operations $\oplus$, $ren_{p\to q}$, $fuse_p$, $add_{p,q}$ make it possible to investigate both kinds of systems in the same algebraic framework.

Proof. (i) Every system written with the considered operations and finitely many 'basic' graphs defines a VR set by application of Theorem 6.1 and the characterization of VR sets given by Engelfriet and van Oostrom (1997). Conversely, a VR set is definable by a system using only the operations $\oplus$, $ren_{p\to q}$, $add_{p,q}$ and one basic graph (Courcelle *et al.* 1993; Engelfriet 1994). Hence, it is also definable by a system of equations using the considered operations and constants.

(ii) By Bauderon and Courcelle (1987), every HR set of graphs is definable by a system of equations using disjoint union combined with a renaming of source vertices, and certain fusions of source vertices. All these operations can be defined in terms of the operations $\oplus$, $ren_{p\to q}$, $fuse_p$.

For the converse, there is a technical difficulty (already mentioned): if an HR set of graphs is defined by a system using the operations $\oplus, ren_{p\to q}, fuse_p$, it is possible that some components of the least solution do not contain only graphs with sources (because one may have several vertices with the same label).

However, we can do the proof as follows. We first observe that the transformation I is a homomorphism for the operations $\oplus$, $ren_{p\to q}$ and $fuse_p$ (that is, $I(G\oplus H) = I(G)\oplus I(H)$, *etc.*). Hence, from a system of equations using operations of $\oplus$, $ren_{p\to q}$ and $fuse_p$, one can construct another one, using the same operations and different basic graphs (the images under I of those of the original system), such that the new system defines the incidence graphs of the graphs of the considered set L. Hence, the set $I(L)$ is VR. It follows that L is HR, which completes the proof. □

The conference paper, Engelfriet (1994), contains some (less clear-cut) characterizations of VR and HR context-free sets of graphs in terms of graph operations.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

7.4. *Extensions that fail*

We may ask for an extension of Theorem 6.1 to more powerful fusion operations. A popular operation in graph theory is *edge contraction.*

For P-labelled graphs, let us introduce the following *contraction.* For $p, q \in P, p \neq q$, we let $contr_{p,q}(G)$ be the graph obtained from G by the simultaneous contraction of all edges between a vertex x labelled by p and a vertex y labelled by q, and such that the vertex resulting from the fusion of such x and y has label p.

Remark 7.1. It is important to contract simultaneously all these edges, that is, contraction is performed in parallel rather than sequentially. As an example, consider the graph $x - y - z$ with x, z labelled p and y labelled q. The result of the simultaneous contraction is one single vertex. If we first contract $x - y$, we get an edge with two ends labelled p, and no further contraction is possible.

Proposition 7.5. If $Card(P) \geqslant 3$ and $\mathscr{F} = \mathscr{C}_P \cup \{contr_{p,q}\}$ where $p, q \in P$, then the mapping $val_{\mathscr{F}}$ is not MS-compatible, hence is not an MS-transduction.

Proof. The proof is by contradiction. Assume that $val_{\mathscr{F}}$ is MS-compatible, where $P \supseteq \{p, q, r\}$. Let $\mathscr{F} = \mathscr{C}_P \cup \{contr_{p,q}\}$. The mapping $t \longmapsto val_{\mathscr{F}}(t)$ from $T(\mathscr{F})$ to the set of P-graphs is an MS-transduction. Consider e_n defined inductively as follows:

$$e_0 = \mathbf{p}, \qquad e_n = ren_{r \to p}(ren_{p \to q}(add_{p,r}(e_{n-1} \oplus \mathbf{r}))).$$

This defines a path of length n, *viz.*

$$\underbrace{q - q - q - q - \ldots - q}_{n-1} - p$$

with one end labelled by p, the other vertices labelled by q.

Let us now define for every $m \geqslant 0$ the term

$$f_{n,m} = contr_{p,q}(\ldots contr_{p,q}(e_n))) \ldots))$$

with m times $contr_{p,q}$. Consider the property of $val_{\mathscr{F}}(t)$, for $t \in T(\mathscr{F})$ saying 'the graph has a unique vertex'. By backwards translation, this is equivalent to an MS-property of t. But for t of the form $f_{n,m}$, this property is equivalent to $m \geqslant n$. Hence we get a contradiction because $\{f_{n,m} / m \geqslant n\}$ is not MS-definable in $T(\mathscr{F})$ using classical arguments on finite-state tree-automata and their equivalence to MS-formulas (Thomas 1990). □

8. Summary and open questions

We have considered sets of operations on structures that are compatible with MS logic in the sense that the corresponding evaluation mappings are MS-transductions, so that they generate graphs of bounded clique-width. Hence, we have new tools to prove that certain graph families have bounded clique-width.

It is still not known whether one can verify in polynomial time that a given graph has clique-width k, for fixed k bigger than 3. Even if these problems are not polynomial for large values of k, it might be the case that a variant in the definition of clique-width,

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

based on *qfd* operations and fusions, could yield a hierarchy of graphs that is equivalent to the one based on clique-width ('equivalent' here meaning that the same sets of graphs have bounded clique-width and bounded 'width in the new sense'), and such that each level has a polynomial decision problem.

Our operations can also generate structures, that is, simple, directed, ranked hypergraphs. In order to generate hypergraphs, some operations have been defined in Courcelle (1992). It remains to define a complexity measure that would behave nicely with respect to operations such as taking an induced subhypergraph, and, hopefully, with a polynomial decision algorithm at each level of the hierarchy.

References

Arnborg, S., Lagergren, J. and Seese, D. (1991) Easy problems for tree decomposable graphs. *Journal of Algorithms* **12** 308–340.

Bauderon, M. and Courcelle, B. (1987) Graph expressions and Graph rewriting. *Mathematical Systems Theory* **20** 83–127.

Corneil, D., Perl, Y. and Stewart, L. K. (1985) A linear recognition algorithm for cographs. *SIAM Journal on Computation* **14** 926–934.

Corneil, D. G., Habib, M., Lanlignel, J.-M., Reed, B. and Rotics, U. (2000) Polynomial time recognition of clique-width $\leqslant 3$ graphs. Proceedings LATIN'2000. *Springer-Verlag Lecture Notes in Computer Science* **1776** 126–134.

Courcelle, B. (1987) An axiomatic definition of context-free rewriting and its application to NLC grammars. *Theoretical Computer Science* **55** 141–181.

Courcelle, B. (1992) Monadic second-order logic of graphs VII: Graphs as relational structures. *Theoretical Computer Science* **101** 3–33.

Courcelle, B. (1994) Monadic second order graph transductions: A survey. *Theoretical Computer Science* **126** 53–75,

Courcelle, B. (1995) Structural properties of context-free sets of graphs generated by vertex replacement. *Information and Computation* **116** 275–293.

Courcelle, B. (1997) The expression of graph properties and graph transformations in monadic second-order logic. In: Rozenberg, G. (ed.) *Handbook of graph grammars and computing by graph transformations, Vol. 1: Foundations*, World Scientific 313–400.

Courcelle, B., Engelfriet, J. and Rozenberg, G. (1993) Handle rewriting hypergraph grammars. *J. Comput. System Sci.* **46** 218–246.

Courcelle, B. and Engelfriet, J. (1995) A logical characterization of the sets of hypergraphs defined by hyperedge replacement grammars. *Mathematical Systems Theory* **28** 515–552.

Courcelle, B., Makowsky, J. A. and Rotics, U. (2000) Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems* **33** 125–150.

Courcelle, B. and Mosbah, M. (1993) Monadic second–order evaluations on tree–decomposable graphs. *Theoretical Computer Science* **109** 49–82.

Courcelle, B. and Olariu, S. (2000) Upper bounds to the clique-width of graphs. *Discrete Applied Mathematics* **101** 77–114.

Courcelle, B. and Walukiewicz, I. (1998) Monadic Second-order logic, graph coverings and unfoldings of transition systems. *Annals of Pure and Applied Logic* **92** 35–62.

Downey, R. and Fellows, M. (1997) *Parametrized complexity*, Springer-Verlag.

Drewes, F., Kreowski, H.-J. and Habel, H. (1997) Hyperedge replacement. In: Rozenberg, G. (ed.) *Handbook of graph grammars and computing by graph transformations, Vol. 1: Foundations*, World Scientific 95–162.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press

Engelfriet, J. (1994) Graph grammars and tree transducers. Proceedings CAAP'94. *Springer-Verlag Lecture Notes in Computer Science* **787** 15–36.

Engelfriet, J. and van Oostrom, V. (1997) Logical description of context-free graph-languages. *Journal of Computer and System Sciences* **55** 489–503.

Engelfriet, J. and Rozenberg, G. (1997) Node replacement graph grammars. In: Rozenberg, G. (ed.) *Handbook of graph grammars and computing by graph transformations, Vol. 1: Foundations*, World Scientific 1–94.

Thomas, W. (1990) Automata on infinite objects. *Handbook of Theoretical Computer Science, vol. B*, Elsevier 133–192.

https://doi.org/10.1017/S0960129501003565 Published online by Cambridge University Press