CMSO-Transducing Tree-Like Graph Decompositions

Rutger Campbell
Discrete Mathematics Group, Institute for Basic Science, Daejeon, South Korea

Bruno Guillon
Université Clermont Auvergne, Clermont Auvergne INP, LIMOS, CNRS, Clermont-Ferrand, France

Mamadou Moustapha Kanté
Université Clermont Auvergne, Clermont Auvergne INP, LIMOS, CNRS, Clermont-Ferrand, France

Eun Jung Kim
KAIST, Daejeon, South Korea
CNRS, France

Noleen Köhler
University of Leeds, UK

Abstract

We show that given a graph G we can CMSO-transduce its modular decomposition, its split decomposition and its bi-join decomposition. This improves results by Courcelle [Logical Methods in Computer Science, 2006] who gave such transductions using order-invariant MSO, a strictly more expressive logic than CMSO. Our methods more generally yield C_2MSO-transductions of the canonical decomposition of weakly-partitive set systems and weakly-bipartitive systems of bipartitions.

2012 ACM Subject Classification Theory of computation → Finite Model Theory

Keywords and phrases MSO-transduction, MSO-definability, graph decomposisions

Digital Object Identifier 10.4230/LIPIcs.STACS.2025.22

Related Version *Extended Version*: arXiv:2412.04970 [6]

Funding *Rutger Campbell*: Supported by the Institute for Basic Science (IBS-R029-C1).
Mamadou Moustapha Kanté: Supported by the French National Research Agency (ANR-18-CE40-0025-01 and ANR-20-CE48-0002).

1 Introduction

A decomposition of a graph, especially a tree-like decomposition, is a result of recursive separations of a graph and is extremely useful for investigating combinatorial properties such as colourability, and for algorithm design. Such a decomposition also plays a fundamental role when one wants to understand the relationship between logic and a graph class. Different notions of the complexity of a separation motivate different ways to decompose, such as tree-decomposition, branch-decomposition, rank-decomposition and carving-decomposition. Furthermore, some important graph classes can be defined through the tree-like decomposition they admit; cographs with cotrees and distance-hereditary graphs with split decompositions being prominent examples.

For a logic L, an L-transduction is a non-deterministic map from a class of relational structures to a new class of relational structures using L-formulas. Transducing a tree-like decomposition is of particular interest. Notably, transducing a decomposition of a graph implies that any property that is definable using a decomposition, is also definable on graphs that admit such a decomposition. Moreover, tree-like decompositions can be

often represented by labelled trees, for which the equivalence of recognisability by a tree automaton and definability in CMSO is well known [8]. Hence, it is an interesting question to consider what kind of graph decompositions can be transduced using L-transductions for some extension L of MSO.

In a series of papers [8, 9, 10, 12], Courcelle investigated the relationship between the graph properties that can be defined in an extension of MSO and the graph properties that can be recognized by a tree automaton. In particular, for graphs of bounded treewidth, Courcelle's theorem states that any property that is definable in the logic MSO with modulo counting predicate, denoted CMSO, can be recognized by a tree automaton [8]. Combining this result with the linear time algorithm for computing tree-decompositions [1], yields that CMSO model-checking can be done in linear time on graphs of bounded treewidth. The converse statement – whether recognisability by a tree automaton implies definability in CMSO on graphs of bounded treewidth – was conjectured by Courcelle in [8] and finally settled by Bojańczyk and Pilipczuk [4]. The key step to obtain this result is obtaining a tree-decomposition of a graph via an MSO-transduction, a strategy which was initially proposed in [9] and is now standard.

The obvious next question is whether an analogous result can be proved for graphs of bounded clique-width and for more general combinatorial objects, most notably, matroids representable over a fixed finite field and of bounded branch-width. Due to the above-mentioned strategy, the key challenge is to produce corresponding tree-like decompositions by MSO-transduction. It is known that clique-width decompositions can be MSO-transduced for graphs of bounded linear clique-width [3]. However, it is unknown if clique-width decompositions can be MSO-transduced in general. In fact, this question remains open even for distance-hereditary graphs, which are precisely graphs of rank-width 1 (thus, of constant clique-width).

Besides tree-decompositions, the problem of transducing cotrees, and in general hierarchical decompositions such as modular decompositions and split decompositions were considered in the literature [10, 11, 12, 14]. In [10], Courcelle provides transductions using order-invariant MSO for cographs and modular decompositions of graphs of bounded modular width. Order-invariant MSO allows the use of a linear order of the set of vertices and is more expressive than CMSO [20]. The applicability of these transductions was later generalized using the framework of weakly-partitive set systems[1] to obtain order-invariant MSO-transductions of modular and split decompositions [12]. It was left as an open question whether one can get rid of the order (see for instance [13] where an overview of the result on hierarchical decompositions was given).

1.1 Our results

In this paper, we consider decompositions given by separations that do not overlap with any other separations of the same type. We view separations of a given kind as a "set system". A set system consists of a set U, the universe, and a set $\mathcal{S}$ of subsets of U. Two sets overlap if they have non-empty intersection but neither of them is contained in the other. If no two elements in a set system $(U, \mathcal{S})$ overlap, i.e. the set system is *laminar*, then the subset relation in $(U, \mathcal{S})$ can be described by a rooted tree T, called the *laminar tree* of $(U, \mathcal{S})$. For any set system $(U, \mathcal{S})$ we can look at the subset of strong sets, i.e., sets that

[1] Weakly-partitive set systems are set systems enjoying some nice closure properties, which were then used to show that some set systems allow canonical tree representations, see for instance the thesis by Montgolfier and Rao [17, 24].

do not overlap with any other set, and the laminar tree T they induce. Additionally, we consider systems of bipartitions $\mathcal{B}$ of a universe U. Two bipartitions of U overlap if neither side of one of the bipartition is contained in either side of the other bipartition. In case $(U, \mathcal{B})$ has no overlapping bipartitions, then $(U, \mathcal{B})$ can be described by an undirected tree, also called *laminar tree*, in which bipartitions correspond to edge cuts. We can define the concept of strong bipartitions equivalently and consider the laminar tree induced by the strong bipartitions in $(U, \mathcal{B})$.

Given a graph G we can consider the set system $(V(G), \mathcal{M})$ where $\mathcal{M}$ is the set of all modules in G, or the system of bipartition $(V(G), \mathcal{S})$ where $\mathcal{S}$ contains all splits in G, or the system of bipartitions $(V(G), \mathcal{B})$ where $\mathcal{B}$ contains all bi-joins in G. We obtain the modular decomposition/cotree/split decomposition/bi-join decomposition by equipping the laminar tree of the suitable set system/system of bipartitions with some additional structure. The additional structure allows us to recover the graph from the respective decomposition.

Abstractly, the set systems/systems of bipartitions mentioned are instances of *weakly-partitive set systems/weakly-bipartitive systems of bipartitions*. Roughly speaking, if a set system $(U, \mathcal{S})$ is well behaved, i.e. $(U, \mathcal{S})$ is weakly-partitive (definition in Section 2), then there is a labelling λ of T and a partial order $<$ of its nodes such that $(U, \mathcal{S})$ is completely described by $(T, \lambda, <)$ [7, 24]. A similar statement holds for systems of bipartitions [17].

We show the following, wherein each item λ is a suitable labelling of the nodes of the laminar tree T, $<$ is a partial ordering of its nodes, and F is an additional edge relation defined only on pairs of siblings in T. A visualization how results depend on each other is given in Figure 1. Due to space constraints, some proofs are omitted, but they are available in the extended preprint [6].

▶ **Theorem 1.** *There are non-deterministic* C_2MSO*-transductions* $\tau_1, \ldots, \tau_7$ *such that:*

1. *For any laminar set system* $(U, \mathcal{S})$, $\tau_1(U, \mathcal{S})$ *is non-empty and every output in* $\tau_1(U, \mathcal{S})$ *is a laminar tree* T *of* $(U, \mathcal{S})$ *(Theorem 2);*
2. *For any graph* G, $\tau_2(G)$ *is non-empty and every output in* $\tau_2(G)$ *is a modular decomposition* (T, F) *of* G *(Theorem 18);*
3. *For any cograph* G, $\tau_3(G)$ *is non-empty and every output in* $\tau_3(G)$ *is a cotree* (T, λ) *of* G *(Corollary 20);*
4. *For any graph* G, $\tau_4(G)$ *is non-empty and every output in* $\tau_4(G)$ *is a split decomposition* (T, F) *of* G *[6, Theorem 29];*
5. *For any graph* G, $\tau_5(G)$ *is non-empty and every output in* $\tau_5(G)$ *is a bi-join decomposition* (T, F) *of* G *[6, Theorems 32 & 37];*
6. *For any weakly-partitive set system* $(U, \mathcal{S})$, $\tau_6(U, \mathcal{S})$ *is non-empty and every output in* $\tau_6(U, \mathcal{S})$ *is a weakly-partitive tree* $(T, \lambda, <)$ *of* $(U, \mathcal{S})$ *(Theorem 15);*
7. *For any weakly-bipartitive set system* $(U, \mathcal{B})$, $\tau_7(U, \mathcal{B})$ *is non-empty and every output in* $\tau_7(U, \mathcal{B})$ *is a weakly-bipartitive tree* $(T, \lambda, <)$ *of* $(U, \mathcal{B})$ *[6, Theorem 24].*

The key step in obtaining these transductions is to transduce the laminar tree T of a set system $(U, \mathcal{S})$, namely Theorem 2. The crux here is to find a suitable representative of each node of T amongst the elements of U and a non-deterministic coloring which allows the assignment of representatives to nodes by means of a C_2MSO-formula. It should be mentioned that a similar result is claimed in the preprint [2], where a proof sketch designing a C_3MSO-transduction is described. Once the laminar tree is obtained, the additional relations for each decomposition can be obtained using a deterministic MSO-transduction. Notice that for each of these transductions, there exists an inverse deterministic MSO-transduction, namely a transduction that from the tree-like decomposition outputs the original structure.

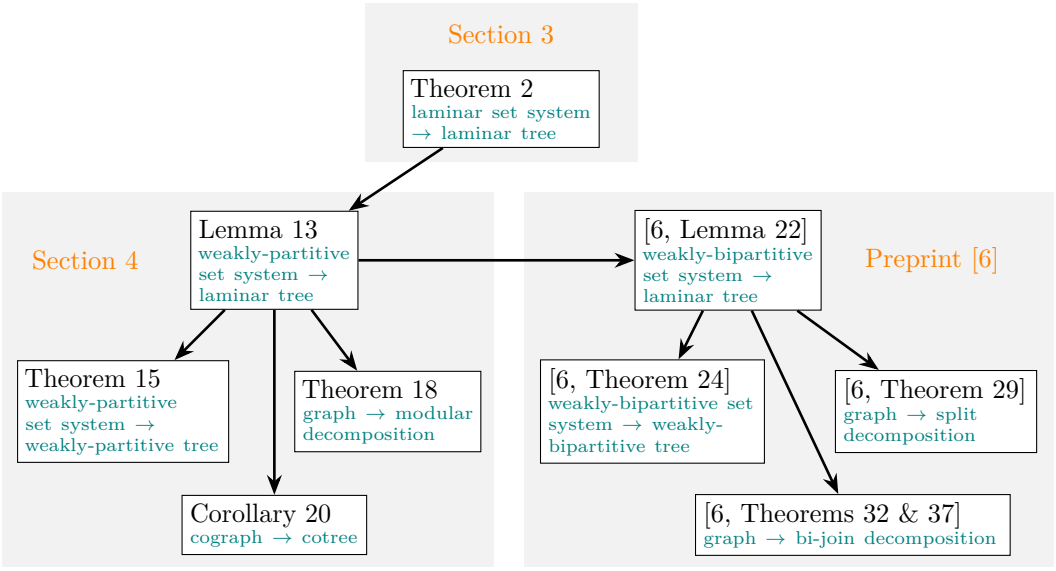

Figure 1 Overview of the various transductions in the paper. An arrow from x to y indicates that result x is used in the proof of result y.

1.2 Organization

The paper is organized as follows. In Section 2 we introduce terminology and notation needed. In Section 3 we prove Theorem 2. In Section 4 we provide the proof of Theorems 15 and 18 and obtain Corollary 20.

2 Preliminaries

2.1 Graphs, trees, set systems

Graphs. We use standard terminology of graph theory, and we fix some notations. Given a *directed graph* G, its sets of *vertices* and *edges* are denoted by $V(G)$ and $E(G)$, respectively. We denote by uv an edge $(u, v) \in E(G)$. An undirected graph is no more than a directed graph for which $E(G)$ is symmetric (*i.e.*, $uv \in E(G) \iff vu \in E(G)$). The notions of *paths*, *connected components*, *etc...* are defined as usual. Given a subset Z of $V(G)$, we denote by $G[Z]$ the sub-graph of G induced by Z.

Trees. A *tree* is a connected undirected graph without cycles. In the context of trees, we use a slightly different terminology than for graphs. In particular, vertices are called *nodes*, nodes of degree at most 1 are called *leaves*, and nodes of degree greater than 1 are called *inner*. The set of leaves is denoted $L(T)$; thus the set of inner nodes is $V(T) \setminus L(T)$. For a node t of a tree T and a neighbor s of t, we denote by T_s^t the connected component of $T - t$ containing s. We sometimes consider *rooted trees*, namely trees with a distinguished node, called the *root*. Rooted trees enjoy a natural orientation of their edges toward the root, which induces the usual notions of *parent*, *child*, *sibling*, *ancestor* and *descendant*. Hence, we represent a rooted tree by a set of nodes with an ancestor/descendant relationship (instead of specifying the root). We use the convention that every node is one of its own ancestors and descendants. We refer to ancestors (resp. descendants) of a node that are not the node itself as *proper ancestors* (resp. *proper descendants*). For a node t of a rooted tree T, we denote by T_t the subtree of T rooted in t (*i.e.*, the restriction of T to the set of descendants of t).

Set systems and laminar trees. A *set system* is a pair $(U, \mathcal{S})$ where U is a finite set, called the *universe*, and $\mathcal{S}$ is a family of subsets of U where $\emptyset \notin \mathcal{S}$, $U \in \mathcal{S}$, and $\{a\} \in \mathcal{S}$ for every $a \in U$.[2] Two sets X and Y *overlap* if they are neither disjoint nor related by containment. A set system $(U, \mathcal{S})$ is said to be *laminar* (aka *overlap-free*) when no two sets from $\mathcal{S}$ overlap. By extension, a set family $\mathcal{S}$ is *laminar* if $(\bigcup \mathcal{S}, \mathcal{S})$ is a laminar set system (note this also requires that $\emptyset \notin \mathcal{S}$, $\bigcup \mathcal{S} \in \mathcal{S}$, and $\{a\} \in \mathcal{S}$ for every $a \in \bigcup \mathcal{S}$).

A laminar family $\mathcal{S}$ of subsets of U naturally defines a rooted tree where the nodes are the sets from $\mathcal{S}$, the root is U, and the ancestor relation corresponds to set inclusion. We call this rooted tree the *laminar tree of U induced by $\mathcal{S}$* (or *laminar tree of $(U, \mathcal{S})$*). In this rooted tree, the leaves are the singletons $\{x\}$ for $x \in U$, which we identify with the elements themselves. That is to say, $L(T) = U$. Laminar trees have the property that each inner node has at least two children. Observe also that the size of a laminar tree is linearly bounded in the size of the universe.

2.2 Logic and transductions

We use *relational structures* to model both graphs and the various tree-like decompositions used in this paper. In order to concisely model set systems we use the more general notion of *extended relational structures*, namely *relational structure* extended with set predicate names. Such structures also naturally arise as outputs of MSO-*transductions* defined below.

Define an *(extended) vocabulary* to be a set of symbols, each being either a *relation* name R with associated arity $\mathrm{ar}(R) \in \mathbb{N}$, or a *set predicate* name P with associated arity $\mathrm{ar}(P) \in \mathbb{N}$. Set predicate names are aimed to describe relations between sets, *e.g.*, one may have a unary set predicate for selecting finite sets of even size, or a binary set predicate for selecting pairs of disjoint sets. We use capital R or names starting with a lowercase letter (*e.g.*, edge, ancestor, t-edge) for relation names, and capital P or uppercase names (*e.g.*, SET, C_2) for set predicate names. To refer to an arbitrary symbol of undetermined kind, we use capital Q. A *relational vocabulary* is an extended vocabulary in which every symbol is a relation name.

Let Σ be a vocabulary. An *extended relational structure over Σ* (Σ*-structure*) is a structure $\mathbb{A} = \langle U_\mathbb{A}, (Q_\mathbb{A})_{Q \in \Sigma} \rangle$ consisting on the one hand of a set $U_\mathbb{A}$ called *universe*, and on the other hand, for each symbol Q in Σ, an *interpretation $Q_\mathbb{A}$ of Q*, which is a relation of arity $\mathrm{ar}(Q)$ either over the universe if Q is a relation name, or over the family of subsets of the universe if Q is a set predicate name. When Σ is not extended, $\mathbb{A}$ is simply a *relational structure*.

Given a Σ-structure $\mathbb{A}$ and, for some vocabulary Γ, a Γ-structure $\mathbb{B}$, we write $\mathbb{A} \sqsubseteq \mathbb{B}$ if $\Sigma \subseteq \Gamma$, $U_\mathbb{A} \subseteq U_\mathbb{B}$ and for each symbol Q in Σ, $Q_\mathbb{A} = Q_\mathbb{B}$.[3] We write $\mathbb{A} \sqcup \mathbb{B}$ to denote the $(\Sigma \cup \Gamma)$-structure consisting of the universe $U_\mathbb{A} \cup U_\mathbb{B}$ and, for each symbol $Q \in \Sigma \cup \Gamma$, the interpretation $Q_{\mathbb{A} \sqcup \mathbb{B}}$ which is $Q_\mathbb{A}$, $Q_\mathbb{B}$, or $Q_\mathbb{A} \cup Q_\mathbb{B}$ according to whether Q belongs to $\Sigma \setminus \Gamma$, to $\Gamma \setminus \Sigma$, or to $\Sigma \cap \Gamma$.[4]

To describe properties of (extended) relational structures, we use *monadic second order logic* (MSO) and refer for instance to [15, 19, 22, 25] for the definition of MSO on extended relational structures such as matroids or set systems in general. This logic allows quantification

[2] Though these restrictions on $\mathcal{S}$ are not usual for set systems, it is convenient for our contribution and it does not significantly impact the generality of set systems: every family $\mathcal{F}$ of subsets of U can be associated with a set system $(U, \mathcal{S})$ where $\mathcal{S} = \big(\mathcal{F} \setminus \{\emptyset\}\big) \cup \big\{U\big\} \cup \big\{\{a\} \mid a \in U\big\}$.

[3] We require equality here (in particular only elements or subsets of $U_\mathbb{A}$ are related in $Q_\mathbb{B}$). This differs from classical notions of inclusions of relational structures which typically require equality only on the restriction of the universe to $U_\mathbb{A}$, *i.e.*, $Q_\mathbb{A} = Q_{\mathbb{B}/U_\mathbb{A}}$, *e.g.*, in order to correspond to induced graphs.

[4] We do not require $\mathbb{A}$ and $\mathbb{B}$ to be disjoint structures whence we may have $\mathbb{A} \not\sqsubseteq \mathbb{A} \sqcup \mathbb{B}$.

both over single elements of the universe and over subsets of the universe. We also use *counting* MSO (CMSO), which is the extension of MSO with, for every positive integer p, a unary set predicate C_p that checks whether the size of a subset is divisible by p or not. We only use C_2. As usual, lowercase variables indicates first-order-quantified variables, while uppercase variables indicates monadic-quantified variables. For a formula ϕ, we write, *e.g.*, $\phi(x, y, X)$ to indicate that the variables x, y, and X belong to the set of *free variables of* ϕ, namely, the set of variables occurring in ϕ that are not bound to a quantifier within ϕ. A *sentence* is a formula without free variables.

We now fix some (extended) vocabularies that we will use.

Graphs. To model both graphs, unrooted trees, and directed graphs, we use the relational vocabulary $\{\mathsf{edge}\}$ where edge is a relation name of arity 2. A (directed) graph $G = (V, E)$ is modeled as the $\{\mathsf{edge}\}$-structure $\mathbb{G}$ with universe $U_{\mathbb{G}} = V$ and interpretation $\mathsf{edge}_{\mathbb{G}} = E$. In particular if G is undirected, then $\mathsf{edge}_{\mathbb{G}}$ is symmetric.

Rooted trees. We use the relational vocabulary $\{\mathsf{ancestor}\}$ to model rooted trees where $\mathsf{ancestor}$ is a relation name of arity 2. A rooted tree T is modeled as the $\{\mathsf{ancestor}\}$-structure $\mathbb{T}$ with universe $U_{\mathbb{T}} = V(T)$ and the interpretation $\mathsf{ancestor}_{\mathbb{T}}$ being the set of pairs (u, v) such that u is an ancestor of v in T. It is routine to define FO-formulas over this vocabulary to express the binary relations parent, child, proper ancestor, (proper) descendant, as well as the unary relations leaf and root.

Set systems. To model set systems, we use the extended vocabulary $\{\mathsf{SET}\}$ where SET is a set predicate name of arity 1. A set system $S = (U, \mathcal{S})$ is thus naturally modeled as the $\{\mathsf{SET}\}$-structure $\mathbb{S}$ with universe $U_{\mathbb{S}} = U$ and interpretation $\mathsf{SET}_{\mathbb{S}} = \mathcal{S}$.

Transductions

Let Σ and Γ be two extended vocabularies. A *Σ-to-Γ transduction* is a set τ of pairs formed by a Σ-structure, call the *input*, and a Γ-structure, called the *output*. We write $\mathbb{B} \in \tau(\mathbb{A})$ when $(\mathbb{A}, \mathbb{B}) \in \tau$. When for every pair $(\mathbb{A}, \mathbb{B}) \in \tau$ we have $\mathbb{A} \sqsubseteq \mathbb{B}$, we call τ an *overlay transduction*. Some transductions can be defined by means of MSO- or C_2MSO-formulas. This leads to the notion of MSO- and C_2MSO-*transductions*. Following the presentation of [3], for L denoting MSO or C_2MSO, define an L-*transduction* to be a transduction obtained by composing a finite number of *atomic* L-*transductions* of the following kinds.

Filtering. An overlay transduction specified by an L-sentence ϕ over the input vocabulary Σ, which discards the inputs that do not satisfy ϕ and keeps the other unchanged. Hence, it defines a partial function (actually, a partial identity) from Σ-structures to Σ-structures.

Universe restriction. A transduction specified by a L-formula ϕ over the input vocabulary Σ, with one free first-order variable, which restricts the universe to those elements that satisfy ϕ. The output vocabulary is Σ and the interpretation of every relation (resp. every predicate) in the output structure is defined as the restriction of its interpretation in the input structure to those tuples of elements satisfying ϕ (resp. tuples of sets of elements that satisfy ϕ). This defines a total function from Σ-structures to Σ-structures.

Interpretation. A transduction specified by a family $(\phi_Q)_{Q \in \Gamma}$ over the input vocabulary Σ where Γ is the output vocabulary and each ϕ_Q has $\mathrm{ar}(Q)$ free variables which are first-order if Q is a relation name and monadic if it is a set predicate name. The transduction outputs the Γ-structure that has the same universe as the input structure and in which each relation or predicate Q is interpreted as those set of tuples that satisfy ϕ_Q. This defines a total function from Σ-structures to Γ-structures.

Copying. An overlay transduction parametrized by a positive integer k that adds k copies of each element to the universe. The output vocabulary consists in the input vocabulary Σ extended with k binary relational symbols $(\mathsf{copy}_i)_{i \in [k]}$ interpreted as pairs of elements (x, y)

saying that "y is the i-th copy of x". The interpretation of the relations (resp. predicates) of the input structure are preserved, on original elements. This defines a total function from Σ-structures to Γ-structures, where $\Gamma = \Sigma \cup \{\mathsf{copy}_i \mid i \in [k]\}$.

Colouring. An overlay transduction that adds a new unary relation $\mathsf{color} \notin \Sigma$ to the structure. Any possible interpretation yields an output; indeed the interpretation is chosen non-deterministically. The interpretation of the relations (resp. predicates) of the input structure are preserved. Hence, it defines a total (non-functional) relation from Σ-structures to Γ-structures where $\Gamma = \Sigma \cup \{\mathsf{color}\}$ (providing $\mathsf{color} \notin \Sigma$).

We say an L-transduction is *deterministic* if it does not use colouring, it is *non-deterministic* otherwise. By definition, deterministic L-transductions define functions.

3 Transducing the laminar-tree

In this section, we present an overlay C_2MSO-transduction that takes as input a laminar set system and outputs the laminar tree it induces.

▶ **Theorem 2.** *Let Σ be an extended vocabulary, including a unary set predicate name* SET *and not including the binary relational symbol* ancestor*. There exists a non-deterministic C_2MSO-transduction τ such that, for each laminar set system $(U, \mathcal{S})$ represented as the $\{\mathsf{SET}\}$-structure $\mathbb{S}$ and inducing the laminar tree T with $L(T) = U$, and for each Σ-structure $\mathbb{A}$ with $\mathbb{S} \sqsubseteq \mathbb{A}$, $\tau(\mathbb{A})$ is non-empty and every output in $\tau(\mathbb{A})$ is equal to some $\{\mathsf{ancestor}\}$-structure $\mathbb{T}$ representing T.*

Since the sets from $\mathcal{S}$ are precisely the sets of leaves of the subtrees of T, there is an MSO-transduction which is the inverse of the above C_2MSO-transduction; that is to say, given an $\{\mathsf{ancestor}\}$-structure $\mathbb{T}$ representing the laminar tree, it outputs the original set system $\mathbb{S}$. Namely,

1. An interpretation $\Phi_{\mathsf{SET}}(S)$ that is true for a set S when there exists an element a such that an element x is in S if and only if a is an ancestor of x.
2. The filtering $\phi(x)$ keeping leaves only.

We prove Theorem 2 in Section 3.2, using the key tools developed in Section 3.1, that allow us to represent each inner node of T with a pair of leaves from its subtree, while keeping the number of inner nodes represented by a given leaf bounded.

3.1 Inner node representatives

In this section, the root of any rooted tree is always an inner node (unless the tree is a unique node). We fix a rooted tree T, in which every inner node has at least two children (a necessary assumption that is satisfied by laminar trees), and we let V denote the set of its nodes ($V = V(T)$) and $L \subseteq V$ denote the subset of its leaves ($L = L(T)$).

Let $S \subseteq V \setminus L$, and let (π, σ) be a pair of injective mappings from S to L. We say that the pair (π, σ) *identifies* S if for each $s \in S$, s is the least common ancestor of $\pi(s)$ and $\sigma(s)$. For $s \in S$ and $x \in V$, we say that x is *s-requested in* (π, σ) if x lies on the path from $\pi(s)$ to $\sigma(s)$ (namely, on either of the paths from $\pi(s)$ to s and from $\sigma(s)$ to s). We say that x is *requested in* (π, σ) if it is s-requested for some s. The pair (π, σ) has *unique request* if every node x of T is requested at most once in (π, σ), *i.e.*, there exists at most one $s \in S$ such that x is s-requested in (π, σ). Note that if (π, σ) has unique request then the paths from $\pi(s)$ to $\sigma(s)$ are pairwise disjoint for all the $s \in S$. We now state a few basic observations.

▶ Remark 3. Let (π, σ) identifying some subset S of $V \setminus L$.

1. The reversed pair (σ, π) also identifies S and has unique request whenever (π, σ) does.
2. If $S' \subset S$, then $(\pi_{|S'}, \sigma_{|S'})$ identifies S', and has unique request if (π, σ) does.
3. For each $s \in S$, s is s-requested in (π, σ).
4. If (σ, π) has unique request, then $\pi(S)$ and $\sigma(S)$ are disjoint subsets of L.

▶ **Lemma 4.** *Let (π, σ) identifying some subset S of $V \setminus L$ with unique request. Then for each $a \in \pi(S)$ the node $\pi^{-1}(a)$ is the least ancestor of a which belongs to S.*

Proof. Let $a \in \pi(S)$ and let $s = \pi^{-1}(a)$. By definition, s is an ancestor of a which belongs to S. Let y be the least ancestor of a that is contained in S. As s is an ancestor of a belonging to S, y must be a descendant of s. Hence, y is s-requested in (π, σ). Additionally, by Item 3 of Remark 3, y is y-requested in (π, σ). Since (π, σ) has unique request, $y = s$. Thus, s is the least ancestor of a which belongs to S. ◀

Notice that, by Item 1 of Remark 3, a similar result holds for each $b \in \sigma(S)$. It follows that the sets $\pi(S)$ and $\sigma(S)$ characterize (π, σ).

▶ **Lemma 5.** *Let $S \subseteq V \setminus L$, and (π, σ) and (π', σ') be two pairs of injections from S to L identifying S with unique request. If $\pi(S) = \pi'(S)$ and $\sigma(S) = \sigma'(S)$ then $\pi = \pi'$ and $\sigma = \sigma'$.*

Proof. Let $s \in S$, $a = \pi(s)$, and $s' = \pi'^{-1}(a)$. Both s and s' are the least ancestor of a which belongs to S, hence $s = s'$ and $\pi'(s) = a$. Thus $\pi = \pi'$. By Item 1 of Remark 3, we also obtain that $\sigma = \sigma'$. ◀

Let A and B be two disjoint subsets of L. We call such a pair (A, B) a *bi-colouring*. We say that (A, B) *identifies* S if there exists a pair (π, σ) identifying S with unique request such that $\pi(S) = A$ and $\sigma(S) = B$. By the previous lemma, for a fixed set $S \subseteq V \setminus L$ the pair (π, σ) identifying S is unique when it exists. We will also prove that S is actually uniquely determined from (A, B). Before that, we state the following technical lemma.

▶ **Lemma 6.** *Let (A, B) identify some subset S of $V \setminus L$ through a pair (π, σ) of injections having unique request. Then, for each inner node x, exactly one of the three following cases holds:*

1. *$x \notin S$, x is not requested in (π, σ), and for each child c of x, $|A \cap V(T_c)| = |B \cap V(T_c)|$;*
2. *$x \notin S$, x is requested in (π, σ), and there exists one leaf $z \in (A \cup B) \cap V(T_x)$ such that, for each child c of x, $|(A \setminus \{z\}) \cap V(T_c)| = |(B \setminus \{z\}) \cap V(T_c)|$;*
3. *$x \in S$, x is requested in (π, σ), and there exists two leaves $a \in A \cap V(T_x)$ and $b \in B \cap V(T_x)$ such that, for each child c of x, $|(A \setminus \{a\}) \cap V(T_c)| = |(B \setminus \{b\} \cap V(T_c))|$ and $\{a, b\} \nsubseteq V(T_c)$.*

Proof. Let x be an inner node. We consider the set $S' = S \setminus (V(T_x) \setminus \{x\})$ of all nodes which are not proper descendants of x and the restrictions π' and σ' of, respectively, π and σ to S'. By Item 2 of Remark 3 (π', σ') identify S' with unique request. We denote $A' = \pi'(S')$ and $B' = \sigma'(S')$, thus (A', B') identify S'. Let $A'_x = A' \cap V(T_x)$ and $B'_x = B' \cap V(T_x)$ be the sets of representatives contained in T_x of nodes in S'. Let a be an element of A'_x. The element $s_a = \pi^{-1}(a)$ is an ancestor of x because it is an ancestor of $a \in V(T_x)$ and belongs to S' whence not to $V(T_x) \setminus \{x\}$. Therefore, x is s_a-requested. As (π', σ') has unique request, there exists at most one element in A'_x. Similarly, B'_x has size at most 1. Moreover, $A'_x \cap B'_x = \emptyset$ by Item 4 of Remark 3. We thus we have three cases:

Case $A'_x \cup B'_x = \emptyset$. Then there is no $s \in S'$ such that $\pi(s) \in V(T_x)$ or $\sigma(s) \in V(T_x)$. Hence, x is not requested in (π, σ), in particular, $x \notin S$.
Let c be a child of x. If $|A \cap V(T_c)| \neq |B \cap V(T_c)|$, then there exists $a \in (A \cup B) \cap V(T_c)$ such that, assuming without loss of generality that $a \in A$ and denoting $s_a = \pi^{-1}(a)$, $\sigma(s_a) \notin V(T_c)$. Hence, s_a is a proper ancestor of c, thus, equivalently, an ancestor of x, implying that x is s_a-requested in (π, σ). This contradicts the above argument. So, $|A \cap V(T_c)| = |B \cap V(T_c)|$ for each child c of x.

Case $A'_x \cup B'_x = \{a\}$. Assume, without loss of generality, that $a \in A$, and denote $s_a = \pi^{-1}(a)$ and $b = \sigma(s_a)$. We have that s_a is a proper ancestor of x, since it is the least common ancestor of $a \in V(T_x)$ and $b \notin V(T_x)$. Thus, x is s_a-requested and $s_a \neq x$ so $x \notin S$.
Let c be a child of x. If $|A \cap V(T_c)| \neq |B \cap V(T_c)|$, then it means that there exists $z \in (A \cup B) \cap V(T_c)$, such that either $z \in A$ and $s_z = \pi^{-1}(z) \notin V(T_c)$, or $z \in B$ and $s_z = \sigma^{-1}(z) \notin V(T_c)$. In both cases, s_z is a proper ancestor of c, whence an ancestor of x. Thus, x is s_z-requested in (π, σ). However, x is s_a-requested in (π, σ) which has unique request, hence $s_z = s_a$. If $z \in B$, it follows that $z = b$, which contradicts the fact $b \notin V(T_x)$. Hence, $z \in A$ and thus, $z = \pi(s_a) = a$. Therefore, $|(A \setminus \{a\}) \cap V(T_c)| = |B \cap V(T_c)|$ for each child c of x.

Case $A'_x = \{a\}$ and $B'_x = \{b\}$. Let $s_a = \pi^{-1}(a)$ and $s_b = \sigma^{-1}(b)$. The node x is s_a-requested and s_b-requested in (π, σ), so, by the unique request property, $s_a = s_b$. Since s_a is the least common ancestor of a and b, it belongs to $V(T_x)$ whence $s_a = x$ implying $x \in X$.
Let c be a child of x. If $|A \cap V(T_c)| \neq |B \cap V(T_c)|$, then there exists $z \in (A \cup B) \cap V(T_c)$ such that either $z \in A$ and $s_z = \pi^{-1}(z) \notin V(T_c)$, or $z \in B$ and $s_z = \sigma^{-1}(z) \notin V(T_c)$. In both cases, s_z is a proper ancestor of c, whence an ancestor of x, and thus x is s_z-requested in (π, σ). However, x is x-requested in (π, σ) which has unique request, hence $s_z = x$ and thus $z \in \{a, b\}$. Therefore, $|(A \setminus \{a\}) \cap V(T_c)| = |(B \setminus \{b\}) \cap V(T_c)|$ for each child c of x. Because the least common ancestor of a and b is x, there is no child c of x containing both a and b as leaves, *i.e.*, $\{a, b\} \not\subseteq V(T_c)$.

This concludes the proof of the statement. ◀

It follows that for each bi-colouring (A, B), there exists at most one set S of inner nodes identified by (A, B).

▶ **Lemma 7.** *Let (A, B) be two disjoint subsets of L and let S and S' be two subsets of $V \setminus L$. If (A, B) identify both S and S', then $S = S'$.*

Proof. We proceed by contradiction and thus assume $S \neq S'$. Let $s \in S \setminus S'$. By Lemma 6, the number of children c of s for which the set $(A \cup B) \cap V(T_c)$ has odd size is 2 since $s \in S$, while it should also be less than 2 since $s \notin S'$. A contradiction. ◀

When (A, B) identifies a set S, we call *A-representative* (resp. *B-representative*) of $s \in S$ the leaf $\pi(s) \in A$ (resp. $\sigma(s) \in B$), where (π, σ) witnessing that (A, B) identifies S. An example of bi-colouring identifying a subset of inner nodes is given in Figure 2.

Not every set of inner nodes has a bi-colouring identifying it. To ensure that such a pair exists, we consider *thin sets of inner nodes*. While thin sets always have bi-colourings identifying them, it is also guaranteed that the set of inner nodes can be partitioned into only 4 thin sets. A subset $X \subseteq V \setminus L$ is *thin* when, for each $x \in X$ not being the root, on the one hand, the parent p_x of x does not belong to X, and on the other hand, x admits at least one sibling (including possible leaves) that does not belong to X. Having a thin set X allows to find branches avoiding it.

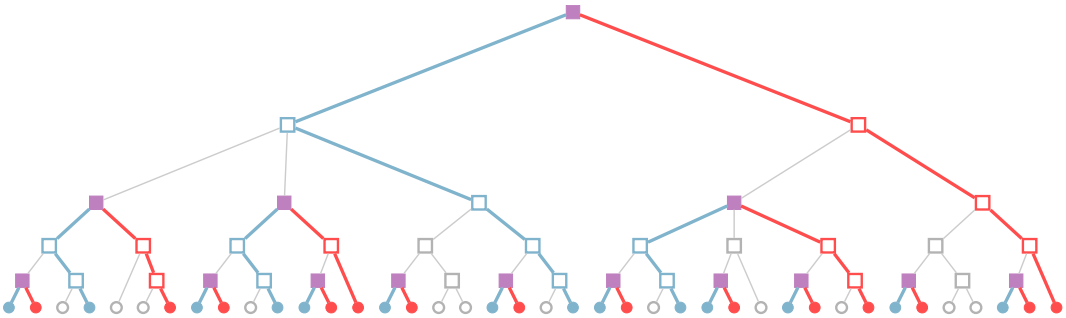

Figure 2 Illustration of a bi-colouring (A, B) identifying some set $S \subseteq V \setminus L$ in a binary tree T. Leaves from A are coloured blue, leaves from B are coloured red, and inner nodes from S are marked purple. Furthermore, the two paths connecting an inner node $s \in S$ to its A- and B-representatives are coloured blue and red, respectively.

▶ **Lemma 8.** *Let $X \subseteq V \setminus L$ and $s \in V \setminus X$. If X is thin, then there exists a leaf $t \in V(T_s)$ such that the path from t to s avoids X (*i.e.*, none of the nodes along this path belong to X).*

Proof. If T_s has height 0, then s is a leaf and taking $t = s$ trivially gives the expected path. Otherwise, s is an inner node and, because X is thin, s has at least one child c_s not belonging to X. By induction, there is a path from some leaf $t \in V(T_{c_s})$ to c_s avoiding X and, since $s \notin X$, this path could be extended into a path from t to s avoiding X. ◀

The previous lemma allows to identify every thin set.

▶ **Lemma 9.** *If X is a thin set, then there exists a pair (π, σ) of injections from X to L that has unique request and that identifies X.*

Proof. We proceed by induction on the size of X. If $X = \emptyset$, the result is trivial. Let $n \in \mathbb{N}$ and suppose that for every thin set of size n there exists a pair of injections identifying it with unique request. Let X be a thin set of size $n+1$, and let $s \in X$ be of minimal depth. Clearly, $X \setminus \{s\}$ is thin and thus there exists, by induction, a pair (π, σ) of injections from $X \setminus \{s\}$ to L identifying $X \setminus \{s\}$ with unique request. Since X is thin and $s \in X$, we can find two distinct children c_a and c_b of s not belonging to X.[5] Then, by Lemma 8, there exists a leaf $a \in V(T_{c_a})$ (resp. a leaf $b \in V(T_{c_b})$) such that the path from a to c_a (resp. from b to c_b) avoids X. In particular, for each node y along these paths, since y has no ancestor that belongs to X but s, y is not requested in (π, σ). Hence, extending π (resp. σ) in such a way that, besides mapping each $x \in X \setminus \{s\}$ to $\pi(x)$ (resp. to $\sigma(x)$), it maps s to a (resp. to b), we obtain a pair $(\hat{\pi}, \hat{\sigma})$ of injections from X to L that identifies X with unique request. ◀

A family $F = (A_1, B_1), \ldots, (A_n, B_n)$ of bi-colourings *identifies* a set $S \subseteq V \setminus L$, if there exists a partition $(S_1, \ldots, S_n)$ of S such that, for each $i \in [n]$, (A_i, B_i) identifies S_i. Whenever $S = V \setminus L$ we say that F *identifies* T. A collection of subsets of $V \setminus L$ is *thin* if each of its subsets is thin. We now show that there exists a thin 4-partition.

▶ **Lemma 10.** *There exists a thin 4-partition of $V \setminus L$.*

[5] Remember that every inner node of T has at least two children (including possible leaves).

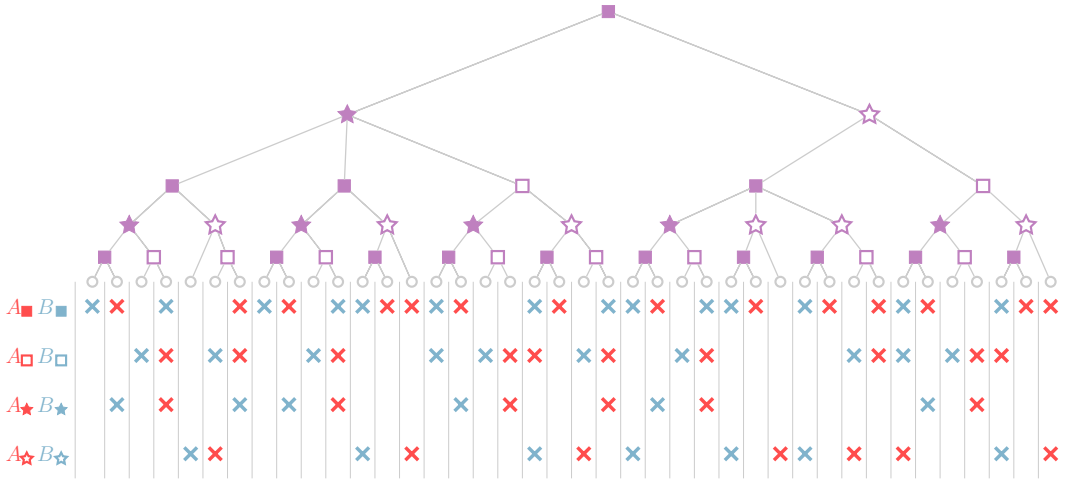

Figure 3 Illustration of a partition of the inner nodes into 4 thin sets indicated by different shapes/fillings. The four colourings identifying each thin set are indicated in the table below the leaves.

Proof. We build such a thin 4-partition as follows. First, consider the partition (D_e, D_o) of $V \setminus L$ in which D_e (resp. $D_o = V \setminus (D_e \cup L)$) is the set of all inner nodes of even (resp. odd) depth. Second, arbitrarily fix one child c_x of x for each inner node x, and consider the set $C = \{c_x \mid x \in V \setminus L\} \setminus L$, inducing a partition $(C, \overline{C})$ of $V \setminus L$ where $\overline{C} = V \setminus (L \cup C)$. By refining these two bi-partitions, we obtain a 4-partition which is thin by construction.[5] ◀

Hence, four bi-colourings are enough to identify T. For an example see Figure 3.

▶ **Corollary 11.** *There exists a family of four bi-colourings identifying T.*

Proof. The result immediately follows from Lemmas 9 and 10. ◀

3.2 The transduction

The goal of this section is to prove Theorem 2, that is, to design a C_2MSO-transduction that produces the laminar tree induced by an input laminar set system. We fix a laminar set system $(U, \mathcal{S})$, represented by a $\{\mathsf{SET}\}$-structure $\mathbb{S}$. Before proving the theorem, we make the following basic observation. On $\mathbb{S}$, we can define two MSO-formulas $\mathsf{desc}(X, Y)$ and $\mathsf{child}(X, Y)$ expressing that in the laminar tree induced by $\mathcal{S}$, X and Y are nodes and X is a descendant or a child of Y, respectively:

$$\mathsf{desc}(X, Y) := \mathsf{SET}(X) \wedge \mathsf{SET}(Y) \wedge X \subseteq Y;$$

$$\mathsf{child}(X, Y) := \mathsf{desc}(X, Y) \wedge X \neq Y \wedge \forall Z \left(\big(\mathsf{desc}(Z, Y) \wedge Z \neq Y\big) \rightarrow \mathsf{desc}(Z, X) \right).$$

The key point of our construction consists in defining a C_2MSO-formula $\mathsf{repr}_{A,B}(a, X)$ which, assuming a bi-colouring (A, B) of the universe modeled as disjoint unary relations and identifying a subset S of inner nodes of T, is satisfied exactly when $X \in S$ and a is its A-representative.

▶ **Lemma 12.** *Let (A, B) be a bi-colouring of $L(T)$ identifying a subset S of inner nodes of T. There exists a C_2MSO-formula $\mathsf{repr}_{A,B}(a, X)$ that is satisfied exactly when X is an inner node of T that belongs to S and is A-represented by a.*

Proof. First, we define a formula $\phi_{A,B}(X)$ that, under the above assumption, is satisfied exactly when X belongs to S. According to Lemma 6, this happens if and only if X is a node of T (*i.e.*, $\mathsf{SET}(X)$ is satisfied) and there exists $a \in X \cap A$ and $b \in X \cap B$ such that for each child Z of X, $\{a,b\} \not\subseteq Z$ and the set $(Z \setminus \{a,b\}) \cap (A \cup B)$ has even size. This property is easily expressed in C_2MSO, using the MSO-formula $\mathsf{child}(X,Y)$ defined previously, as well as the predicate SET:

$$\phi_{A,B}(X) := \mathsf{SET}(X) \wedge \exists a \exists b \Big[a \in (X \cap A) \wedge b \in (X \cap B) \wedge \forall Z \left(\mathsf{child}(Z,X) \rightarrow \left(\{a,b\} \not\subseteq Z \wedge \mathrm{C}_2 \left((Z \setminus \{a,b\}) \cap (A \cup B) \right) \right) \right) \Big].$$

Now, we can easily define $\mathsf{repr}_{A,B}(a,X)$ based on Lemma 4:

$$\mathsf{repr}_{A,B}(a,X) := \phi_{A,B}(X) \wedge a \in (X \cap A) \wedge \forall Z \subsetneq X \left(a \in Z \rightarrow \neg \phi_{A,B}(Z) \right).$$

This concludes the proof. ◀

We are now ready to prove the theorem.

Proof of Theorem 2. The C_2MSO-transduction is obtained by composing the following atomic C_2MSO-transductions. The transduction makes use of the formulas $\mathsf{repr}_{A,B}(a,X)$ given by Lemma 12.

1. Guess a family of four bi-colourings $(A_i, B_i)_{i \in [4]}$ identifying T (which exists by Corollary 11).
2. Copy the input graph four times, thus introducing four binary relations $(\mathsf{copy}_i)_{i \in [4]}$ where $\mathsf{copy}_i(x,y)$ indicates that x is the i-th copy of the original element y.
3. Filter the universe keeping only the original elements as well as the i-th copy of each vertex a for which there exists X such that $\mathsf{repr}_{A_i,B_i}(a,X)$.
4. Define the relation $\mathsf{ancestor}(x,y)$ so that it is satisfied exactly when there exist x', X, i, and Y such that, on the one hand $\mathsf{desc}(Y,X)$ and $\mathsf{copy}_i(x,x') \wedge \mathsf{repr}_{A_i,B_i}(x',X)$, and, on the other hand, either y is an original element and $Y = \{y\}$, or there exists y' and j such that $\mathsf{copy}_j(y,y') \wedge \mathsf{repr}_{A_j,B_j}(y',Y)$. ◀

4 Transducing modular decompositions

The set of modules of a directed graph is a specific example of a particular type of set system, a "weakly-partitive set system" (and a "partitive set system" in the case of an undirected graph). In this section, we first give a general C_2MSO-transduction to obtain the canonical tree-like decomposition of a weakly-partitive set system from the set system itself. We then show how to obtain the modular decomposition of a graph via a C_2MSO-transduction as an application.

4.1 Transducing weakly-partitive trees

A set system $(U, \mathcal{S})$ is *weakly-partitive* if for every two overlapping sets $X, Y \in \mathcal{S}$, the sets $X \cup Y$, $X \cap Y$, $X \setminus Y$, and $Y \setminus X$ belong to $\mathcal{S}$. It is *partitive* if, moreover, for every two overlapping sets $X, Y \in \mathcal{S}$, their symmetric difference, denoted $X \triangle Y$, also belongs to $\mathcal{S}$. By extension, a set family $\mathcal{S}$ is called *weakly-partitive* or *partitive* whenever $(\bigcup \mathcal{S}, \mathcal{S})$ is a set system which is weakly-partitive or partitive, respectively (note these also requires that $\emptyset \notin \mathcal{S}$, $\bigcup \mathcal{S} \in \mathcal{S}$, and $\{a\} \in \mathcal{S}$ for every $a \in \bigcup \mathcal{S}$).

A member of a set system $(U, \mathcal{S})$ is said to be *strong* if it does not overlap any other set from $\mathcal{S}$. The sub-family $\mathcal{S}_!$ of strong sets of $\mathcal{S}$ is thus laminar by definition. Hence, it induces a laminar tree T. By extension, we say that T is *induced by the set system* $(U, \mathcal{S})$ (or simply by $\mathcal{S}$). The next result extends Theorem 2, by showing that T can be C_2MSO-transduced from $\mathcal{S}$.

▶ **Lemma 13.** *Let Σ be an extended vocabulary, including a set unary predicate name* SET *and not including the binary relational symbol* ancestor*. There exists a non-deterministic C_2MSO-transduction τ such that, for each set system $(U, \mathcal{S})$ represented as the $\{\mathsf{SET}\}$-structure $\mathbb{S}$ and inducing the laminar tree T with $L(T) = U$, and for each Σ-structure $\mathbb{A}$ with $\mathbb{S} \sqsubseteq \mathbb{A}$, $\tau(\mathbb{A})$ is non-empty and every output in $\tau(\mathbb{A})$ is equal to some $\{\mathsf{ancestor}\}$-structure $\mathbb{T}$ representing T.*

Proof. On the $\{\mathsf{SET}\}$-structure $\mathbb{S}$, it is routine to define an MSO-formula $\phi_{\mathsf{SET!}}(Z)$ that identifies those subsets $Z \subseteq U$ that are strong members of $\mathcal{S}$:

$$\phi_{\mathsf{SET!}}(Z) := \mathsf{SET}(Z) \wedge \forall X \Big(\big(\mathsf{SET}(X) \wedge X \cap Z \neq \emptyset\big) \to \big(X \subseteq Z \vee Z \subseteq X\big) \Big).$$

Hence, we can design an MSO-interpretation that outputs the $\Sigma \cup \{\mathsf{SET!}\}$-structure corresponding to $\mathbb{A}$ equipped with the set unary predicate SET! that selects strong members of $\mathcal{S}$. Thus, up to renaming the predicates SET! and SET, by Theorem 2, we can produce, through a C_2MSO-transduction, the $\Sigma \cup \{\mathsf{ancestor}\}$-structure $\mathbb{A} \sqcup \mathbb{T}$ where $\mathbb{T}$ is the tree-structure modeling the laminar tree T induced by $\mathcal{S}_!$ with $L(T) = U$ (once the output obtained, the interpretation drops the set predicate SET! which is no longer needed). ◀

Clearly, the laminar tree T of a weakly-partitive set system $(U, \mathcal{S})$ does not characterize $(U, \mathcal{S})$. However, as shown by the below theorem, a labeling of its inner nodes and a controlled partial ordering of its nodes are sufficient to characterize all the sets of $\mathcal{S}$. For Z a set equipped with a partial order $<$ and X a subset of Z, we say that X is a *$<$-interval* whenever $<$ defines a total order on X and for every $a, b \in X$ and every $c \in Z$, $a < c < b$ implies $c \in X$.

▶ **Theorem 14** ([7, 24])**.** *Let $\mathcal{S}$ be a weakly-partitive family, $\mathcal{S}_!$ be its subfamily of strong sets, and T be the laminar tree it induces. There exists a total labeling function λ from the set $V(T) \setminus L(T)$ of inner nodes of T to the set $\{\mathsf{degenerate}, \mathsf{prime}, \mathsf{linear}\}$, and, for each inner node $t \in \lambda^{-1}(\mathsf{linear})$, a linear ordering $<_t$ of its children, such that every inner node having exactly two children is labeled by* degenerate *and the following two conditions are satisfied:*

- *for each $X \in \mathcal{S} \setminus \mathcal{S}_!$, there exists $t \in V(T)$ and a subset $\mathcal{C}$ of children of t such that $X = \bigcup_{c \in \mathcal{C}} L(T_c)$ and either $\lambda(t) =$ linear and $\mathcal{C}$ is a $<_t$-interval, or $\lambda(t) =$ degenerate;*
- *conversely, for each inner node t and each non-empty subset $\mathcal{C}$ of children of t, if either $\lambda(t) =$ linear and $\mathcal{C}$ is a $<_t$-interval, or $\lambda(t) =$ degenerate, then $\bigcup_{c \in \mathcal{C}} L(T_c) \in \mathcal{S}$.*

Furthermore, T and λ are uniquely determined from $\mathcal{S}$, and, for each inner node t of T labeled by linear, only two orders $<_t$ are possible, one being the inverse of the other (indeed, inverting an order $<$ does preserve the property of being a $<$-interval). Hence, every weakly-partitive family $\mathcal{S}$ is characterized by a labeled and partially-ordered tree $(T, \lambda, <)$ where T is the laminar tree induced by the subfamily $\mathcal{S}_!$ of strong sets of $\mathcal{S}$, $\lambda : V(T) \setminus L(T) \to \{\mathsf{degenerate}, \mathsf{prime}, \mathsf{linear}\}$ is the labeling function, and $<$ is the partial order $\bigcup_{t \in \lambda^{-1}(\mathsf{linear})} <_t$ over $V(T)$. As, up-to inverting some of the $<_t$ orders, $(T, \lambda, <)$ is unique, we abusively call it ***the*** *weakly-partitive tree induced by $\mathcal{S}$*. Conversely, a weakly-partitive tree characterizes the unique weakly-partitive set system which induced it.

We naturally model a weakly-partitive tree $(T, \lambda, <)$ of a partitive set system $(U, \mathcal{S})$ by the $\{\mathsf{ancestor}, \mathsf{degenerate}, \mathsf{betweeness}\}$-structure $\mathbb{T}$ of universe $U_{\mathbb{T}} = V(T)$ such that $\langle V(T), \mathsf{ancestor}_{\mathbb{T}} \rangle \sqsubseteq \mathbb{T}$ models T with $L(T) = U$, $\mathsf{degenerate}_{\mathbb{T}}$ is a unary relation which selects

the inner nodes of T of label degenerate, *i.e.*, $\mathsf{degenerate}_{\mathbb{T}} = \lambda^{-1}(\mathsf{degenerate})$, and $\mathsf{betweeness}_{\mathbb{T}}$ is a ternary relation selecting triples (x, y, z) satisfying $x < y < z$ or $z < y < x$. (Although it is possible, through a non-deterministic MSO-transduction, to define $<$ from $\mathsf{betweeness}_{\mathbb{T}}$, the use of $\mathsf{betweeness}_{\mathbb{T}}$ rather than $<_{\mathbb{T}}$ ensures unicity of the output weakly-partitive tree.) The inner nodes of T that are labeled by linear can be recovered through an MSO-formula as those inner nodes whose children are related by betweeness. The inner nodes labeled by prime can be recovered through an MSO-formula as those inner nodes that are labeled neither by degenerate nor by linear. Using Theorem 14, it is routine to design an MSO-transduction which takes as input a weakly-partitive tree and outputs the weakly-partitive set system which induced it. The inverse C_2MSO-transduction is the purpose of the next result. The proof is omitted due to space constraints.

▶ **Theorem 15.** *There exists a non-deterministic* C_2MSO*-transduction* τ *such that, for every weakly-partitive set system* $(U, \mathcal{S})$ *represented as the* $\{\mathsf{SET}\}$*-structure* $\mathbb{S}$ *and inducing the weakly-partitive tree* $(T, \lambda, <)$ *represented as the* $\{\mathsf{ancestor}, \mathsf{degenerate}, \mathsf{betweeness}\}$*-structure* $\mathbb{T}$*, we have* $\mathbb{T} \in \tau(\mathbb{S})$ *and every output in* $\tau(\mathbb{S})$ *is a weakly-partitive tree of* $(U, \mathcal{S})$*.*

If $\mathcal{S}$ is partitive then the weakly-partitive tree it induces enjoys a simple form, and is truly unique. Indeed, the label linear and, thus, the partial order $<$, are not needed.

▶ **Theorem 16** ([7])**.** *Let* $\mathcal{S}$ *be a weakly-partitive family and* $(T, \lambda, <)$ *be the weakly-partitive tree it induces. If* $\mathcal{S}$ *is partitive, then* $\lambda^{-1}(\mathsf{linear}) = \emptyset$ *and* $<$ *is empty.*

Hence, in case of a partitive set systems $(U, \mathcal{S})$, we can consider the simpler object (T, λ), called *the partitive tree induced by* $\mathcal{S}$ (or *the partitive tree of* $(U, \mathcal{S})$) in which λ maps $V(T) \setminus L(T)$ to $\{\mathsf{degenerate}, \mathsf{prime}\}$. As a direct consequence of Theorems 15 and 16, we can produce, through a C_2MSO-transduction, the partitive tree induced by a partitive set system and naturally modeled by an $\{\mathsf{ancestor}, \mathsf{degenerate}\}$-structure.

▶ **Corollary 17.** *There exists a non-deterministic* C_2MSO*-transduction* τ *such that, for each partitive set system* $(U, \mathcal{S})$ *represented as the* $\{\mathsf{SET}\}$*-structure* $\mathbb{S}$ *and inducing the partitive tree* (T, λ) *represented as the* $\{\mathsf{ancestor}, \mathsf{degenerate}\}$*-structure* $\mathbb{T}$*, we have* $\mathbb{T} \in \tau(\mathbb{S})$ *and every output in* $\tau(\mathbb{S})$ *is a partitive tree of* $(U, \mathcal{S})$*.*

4.2 Application to modular decomposition

Let G be a directed graph and let $M \subseteq V(G)$. We say that M is a *module (of G)* if for every $u \notin M$ and every $v, w \in M$, $uv \in E(G) \iff uw \in E(G)$ and $vu \in E(G) \iff wu \in E(G)$. Clearly, the empty set, $V(G)$, and all the singletons $\{x\}$ for $x \in V(G)$ are modules; they are called *the trivial modules* of G. We say a non-empty module M is *maximal* if it is not properly contained in any non-trivial module. Let M and M' be two disjoint non-empty modules of G. Considering the edges that go from M to M', namely edges from the set $(M \times M') \cap E(G)$, we have two possibilities: either it is empty, or it is equal to $M \times M'$. We write $M \not\to M'$ in the former case and $M \to M'$ in the latter. (It is of course possible to have both $M \to M'$ and $M' \to M$.) A *modular partition* of G is a partition $\mathcal{P} = \{M_1, \ldots, M_\ell\}$ of $V(G)$ such that every M_i is a non-empty module. A modular partition $\mathcal{P} = \{M_1, \ldots, M_\ell\}$ is called *maximal* if it is non-trivial and every M_i is strong and maximal. Note that every graph has exactly one maximal modular partition.

A *modular decomposition* of G is a rooted tree T in which the leaves are the vertices of G, and for each inner node $t \in T$, t has at least two children and the set $L(T_t)$ is a module of G. In a modular decomposition T of G, for each inner node $t \in V(T)$ with children $c_1, \ldots, c_r$,

the family $\mathcal{P}_t = \{L(T_{c_1}), \dots, L(T_{c_r})\}$ is a modular partition of $G[V(T_t)]$. When each such partition is maximal, the decomposition is unique and it is called *the maximal modular decomposition* of G. The maximal modular decomposition T of G alone is not sufficient to characterize G. However, enriching T with, for each inner node t with children $c_1, \dots, c_j$, the information of which pair of modules $\big(L(T_{c_i}), L(T_{c_j})\big)$ is such that $L(T_{c_i}) \to L(T_{c_j})$, yields a unique canonical representation of G. Formally, the *enriched modular decomposition* of G is the pair (T, F) where T is the maximal modular decomposition of G (with $L(T) = V(G)$) and $F \subset V(T) \times V(T)$ is a binary relation, that relates a pair (s, t) of nodes of T, denoted $st \in F$, exactly when s and t are siblings and $L(T_s) \to L(T_t)$. The elements of F are called m-*edges*.

It should be mentioned that the family of all non-empty modules of G is known to be weakly-partitive (or even partitive when G is undirected). In particular, the maximal modular decomposition T of G is the laminar tree induced by the family of strong modules. Hence Theorem 15 could be used to produce a partially-ordered and labeled tree which displays all the modules of G. However, this weakly-partitive tree is not sufficient for being able to recover the graph G from it. We now prove how to obtain the enriched modular decomposition of G through a C_2MSO-transduction.

To model enriched modular decompositions as relational structures we use the relational vocabulary $\{\mathsf{ancestor}, \mathsf{m\text{-}edge}\}$ where $\mathsf{ancestor}$ and $\mathsf{m\text{-}edge}$ are two binary relation names. An enriched modular decomposition (T, F) of a graph G is modeled by the $\{\mathsf{ancestor}, \mathsf{m\text{-}edge}\}$-structure $\mathbb{M}$ with universe $U_{\mathbb{M}} = V(T)$, $\mathsf{ancestor}_{\mathbb{M}}$ being the set of pairs (s, t) for which s is an ancestor of t in T, and $\mathsf{m\text{-}edge}_{\mathbb{M}}$ being the set of all pairs (s, t) such that $st \in F$ (in particular, s and t are siblings in T). We use Lemma 13 in order to transduce the maximal modular decomposition of a graph.

▶ **Theorem 18.** *There exists a non-deterministic* C_2MSO*-transduction* τ *such that for every directed graph* G *represented as the* $\{\mathsf{edge}\}$*-structure* $\mathbb{G}$*,* $\tau(\mathbb{G})$ *is non-empty and every output in* $\tau(\mathbb{G})$ *is equal to some* $\{\mathsf{ancestor}, \mathit{m}\text{-}\mathsf{edge}\}$*-structure* $\mathbb{M}$ *representing the enriched modular decomposition* (T, F) *of* G*.*

Proof. Let G be a graph represented by the $\{\mathsf{edge}\}$-structure $\mathbb{G}$. Several objects are associated to G, and each of them can be described by a structure:

- let $\mathcal{M}$ be the family of non-empty modules of G and let $\mathbb{S}$ be the $\{\mathsf{SET}\}$-structure modeling the weakly-partitive set system $(V(G), \mathcal{M})$ with $U_{\mathbb{S}} = V(G)$;
- let T be the laminar tree induced by the weakly-partitive family $\mathcal{M}$ and let $\mathbb{T}$ be the $\{\mathsf{ancestor}\}$-structure modeling it with $U_{\mathbb{T}} = V(T)$ and $L(T) = V(G) \subset U_{\mathbb{T}}$;
- let F be the m-edge relation, namely the subset of $V(T) \times V(T)$ such that (T, F) is the maximal modular decomposition of G, and let $\mathbb{M}$ be the $\{\mathsf{ancestor}, \mathsf{m\text{-}edge}\}$-structure modeling it with $\mathbb{T} \sqsubset \mathbb{M}$ and $U_{\mathbb{T}} = U_{\mathbb{M}}$.

Our C_2MSO-transduction is obtained by composing the three following transductions:

- τ_1: an MSO-interpretation which outputs the $\{\mathsf{edge}, \mathsf{SET}\}$-structure $\mathbb{G} \sqcup \mathbb{S}$ from $\mathbb{G}$;
- τ_2: the non-deterministic C_2MSO-transduction given by Lemma 13 which produces the $\{\mathsf{edge}, \mathsf{SET}, \mathsf{ancestor}\}$-structure $\mathbb{G} \sqcup \mathbb{S} \sqcup \mathbb{T}$ from $\mathbb{G} \sqcup \mathbb{S}$;
- τ_3: an MSO-interpretation which outputs the $\{\mathsf{ancestor}, \mathsf{m\text{-}edge}\}$-structure $\mathbb{M}$ from $\mathbb{G} \sqcup \mathbb{S} \sqcup \mathbb{T}$.

In order to define τ_1 it is sufficient to observe that there exists an MSO-formula $\phi_{\mathsf{SET}}(Z)$ with one monadic free-variable, which is satisfied exactly when Z is a non-empty module of G. Then, since τ_2 is given by Lemma 13, it only remains to define τ_3. Given an inner node t, we can select, within MSO, the set $L(T_t)$ of leaves of the subtree rooted in t. We

thus assume a function leafset, with one first-order free-variable which returns the set of leaves of the subtree rooted at the given node. Equipped with this function, we can define the MSO-formula $\phi_{\text{m-edge}}$ which selects pairs (s, r) of siblings such that $sr \in F$. Remember that this happen exactly when there exist $u \in L(T_s)$ and $v \in L(T_r)$ such that $uv \in E(G)$. Hence, $\phi_{\text{m-edge}}$ could be defined as:

$$\phi_{\text{m-edge}}(s,r) := s \neq r \wedge \exists t \left(\text{parent}(t,s) \wedge \text{parent}(t,r)\right) \wedge \\ \exists x \exists y \big(x \in \text{leafset}(s) \wedge y \in \text{leafset}(r) \wedge \text{edge}(x,y)\big).$$

Once defined, the MSO-interpretation τ_3 simply drops all non-necessary relations and predicates (namely edge and SET) and keeps only the ancestor and m-edge relations. ◀

Notice that it is routine to design a deterministic MSO-transduction which, given an $\{\text{ancestor}, \text{m-edge}\}$-structure $\mathbb{M}$ representing an enriched modular decomposition of some directed graph G, produces the $\{\text{edge}\}$-structure $\mathbb{G}$ representing G.

Cographs

Let G be a graph, (T, F) be its modular decomposition, and $(T, \lambda, <)$ be the weakly-partitive tree induced by the (weakly-partitive) family of its modules. Let t be an inner node of T, let $\mathcal{C}$ be its set of children, and let C be the graph $\big(V(T) \setminus L(T), F\big)[\mathcal{C}]$ induced by F on the set of children of t. It can be checked that, if $\lambda(t) = \text{degenerate}$ then C is either a clique or an independant, and if $\lambda(t) = \text{linear}$ then C is a *tournament consistent with* $<_t$ (*i.e.*, for every $x, y \in \mathcal{C}$, xy is an edge of C if and only if $x <_t y$) or with the inverse of $<_t$. In the former case, we can refine the degenerate label into series and parallel labels, thus expressing that C is a clique or an indenpendant, respectively. In the latter case, up-to reversing $<_t$, we can ensure that C is a tournament consistent with $<_t$. This yields a refined weakly-partitive tree $(T, \gamma, <)$, where γ maps inner nodes to $\{\text{series}, \text{parallel}, \text{prime}, \text{linear}\}$ and $<$ is the order $\bigcup_{t \in \gamma^{-1}(\text{linear})} <_t$ which ensures that tournaments are consistent with the corresponding $<_t$. Notice that this labeled and partially-ordered tree is now uniquely determined from G. Moreover, edges from F that connect children of a node not labeled by prime can be recovered from the so-refined weakly-partitive tree. In particular, if no nodes of T is labeled by prime, (T, F) and thus G is fully characterized by $(T, \gamma, <)$. Graphs for which this property holds are known as *directed cographs*, and can be described by the refined weakly-partitive tree, called *cotree*, explained above and formalized in the following statement.

▶ **Theorem 19.** *Let G be a directed cograph and let T be the laminar tree induced by the family of its strong modules. There exists a unique total labeling λ from the set $V(T) \setminus L(T)$ of inner nodes of T to the set $\{\text{series}, \text{parallel}, \text{linear}\}$ of labels, and, for each inner node $t \in \lambda^{-1}(\text{linear})$, a unique linear ordering $<_t$ of its children, such that every inner node having exactly two children is labeled* series *or* parallel*, and the following condition is satisfied:*

- *for every two leaves x and y of T, denoting by t their least common ancestor, xy is an edge of G if and only if either t is labeled by* linear *and $x <_t y$, or t is labeled by* series*.*

We naturally model cotrees as $\{\text{ancestor}, \text{series}, \varepsilon\text{-ord}\}$-structures as follow. A cotree $(T, \gamma, <)$ is modeled by $\mathbb{C}$ where $U_{\mathbb{C}} = V(T)$, $\text{series}_{\mathbb{C}} = \gamma^{-1}(\text{series})$, and $\varepsilon\text{-ord}_{\mathbb{C}} = \{(x, y) \mid x < y\}$. The nodes that are labeled by linear could be recovered as those inner nodes whose children are related by ε-ord, while the nodes that are labeled by parallel could be recovered as those inner nodes which are labeled neither by series nor by linear. Based on Theorem 19 and as a consequence of Theorem 18, we can design a C_2MSO-transduction which produces the cotree of a cograph G from G.

▶ **Corollary 20.** *There exists a non-deterministic* C_2MSO*-transduction* τ *such that, for each cograph* G *modeled by the* $\{\mathsf{edge}\}$*-structure* $\mathbb{G}$*,* $\tau(\mathbb{G})$ *is non-empty and every output in* $\tau(\mathbb{G})$ *is equal to some* $\{\mathsf{ancestor}, \mathsf{series}, \varepsilon\text{-}\mathsf{ord}\}$*-structure* $\mathbb{C}$ *representing the cotree* $(T, \gamma, <)$ *of* G*.*

5 Conclusion

We provide transductions for obtaining tree-like graph decompositions such as modular decompositions, cotrees, split decompositions and bi-join decompositions from a graph using CMSO. This improves upon results of Courcelle [10, 12] who gave such transductions for ordered graphs. In a more general settings, we also obtain CMSO-transductions outputing weakly-partitive and weakly-bipartitive trees of weakly-partitive and weakly-bipartitive systems (Items 6 and 7 of Theorem 1). It is worth mentionning that the latter transduction can be also used to CMSO-transduce canonical decompositions of other structures such as Tutte's decomposition of matroids or generally split-decompositions of submodular functions [16] or modular decompositions of 2-structures [18] or of hypergraphs [21]. As shown by the application given in [12] for transducing Whitney's isomorphism class of a graph, a line of research is to more investigate which structures can be CMSO-transduced from a graph or a set system by using the transductions from Theorem 1. Also, naturally, the question arises whether counting is necessary or whether MSO is sufficient to transduce such decompositions. Furthermore, our results include that transducing rank decompositions for graphs of rank-width 1 is possible using CMSO. But it is not known whether rank-decompositions can be transduced in general using some suitable extension of MSO. Nevertheless, a corollary of our results is that CMSO-transducing rank-decompositions can be now reduced to consider CMSO-transducing rank-decompositions of prime graphs wrt either modular decomposition or split-decomposition as if a graph has rank-width at least 2, then its rank-width is equal to the rank-width of its prime induced graphs wrt either modular or split decomposition. Thus, our results imply that, for any fixed k, there is a CMSO-transduction that computes a clique-decomposition of small width for any graph belonging to a graph class whose prime graphs have sizes bounded by k or prime graphs have linear clique-width bounded by k, *e.g.*, many H-free graphs have small prime graphs or have small linear clique-width (see for instance [5] or [23] for prominent such examples).

References

1 Hans L Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 226–234, 1993. doi:10.1145/167088.167161.

2 Mikołaj Bojańczyk. The category of mso transductions. *CoRR*, May 2023. arXiv:2305.18039v1.

3 Mikołaj Bojańczyk, Martin Grohe, and Michał Pilipczuk. Definable decompositions for graphs of bounded linear cliquewidth. *Log. Methods Comput. Sci.*, 17(1), 2021. URL: https://lmcs.episciences.org/7125.

4 Mikołaj Bojańczyk and Michał Pilipczuk. Definability equals recognizability for graphs of bounded treewidth. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 407–416. ACM, 2016. doi:10.1145/2933575.2934508.

5 Andreas Brandstädt, Feodor F. Dragan, Hoàng-Oanh Le, and Raffaele Mosca. New graph classes of bounded clique-width. *Theory Comput. Syst.*, 38(5):623–645, 2005. doi:10.1007/S00224-004-1154-6.

6 Rutger Campbell, Bruno Guillon, Mamadou Moustapha Kanté, Eun Jung Kim, and Noleen Köhler. CMSO-transducing tree-like graph decompositions, 2024. doi:10.48550/arXiv.2412.04970.

7 Michel Chein, Michel Habib, and Marie-Catherine Maurer. Partitive hypergraphs. *Discrete mathematics*, 37(1):35–50, 1981. doi:10.1016/0012-365X(81)90138-2.

8 Bruno Courcelle. The monadic second-order logic of graphs. I. recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

9 Bruno Courcelle. The monadic second-order logic of graphs V: On closing the gap between definability and recognizability. *Theoretical Computer Science*, 80(2):153–202, 1991. doi:10.1016/0304-3975(91)90387-H.

10 Bruno Courcelle. The monadic second-order logic of graphs X: linear orderings. *Theor. Comput. Sci.*, 160(1&2):87–143, 1996. doi:10.1016/0304-3975(95)00083-6.

11 Bruno Courcelle. The monadic second-order logic of graphs XI: hierarchical decompositions of connected graphs. *Theor. Comput. Sci.*, 224(1-2):35–58, 1999. doi:10.1016/S0304-3975(98)00306-5.

12 Bruno Courcelle. The monadic second-order logic of graphs XVI: Canonical graph decompositions. *Logical Methods in Computer Science*, 2, 2006. doi:10.2168/LMCS-2(2:2)2006.

13 Bruno Courcelle. Canonical graph decompositions. Talk, 2012. Available at https://www.labri.fr/perso/courcell/Conferences/ExpoCanDecsJuin2012.pdf.

14 Bruno Courcelle. The atomic decomposition of strongly connected graphs. Technical report, Université de Bordeaux, 2013. Available at https://www.labri.fr/perso/courcell/ArticlesEnCours/AtomicDecSubmitted.pdf.

15 Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic*. Cambridge University Press, 2009. doi:10.1017/cbo9780511977619.

16 William H Cunningham. Decomposition of directed graphs. *SIAM Journal on Algebraic Discrete Methods*, 3(2):214–228, 1982.

17 Fabien de Montgolfier. *Décomposition modulaire des graphes. Théorie, extension et algorithmes*. Phd thesis, Université Montpellier II, LIRMM, 2003.

18 Andrzej Ehrenfeucht, Tero Harju, and Grzegorz Rozenberg. *The Theory of 2-Structures – A Framework for Decomposition and Transformation of Graphs*. World Scientific, 1999. URL: http://www.worldscibooks.com/mathematics/4197.html.

19 Daryl Funk, Dillon Mayhew, and Mike Newman. Tree automata and pigeonhole classes of matroids: I. *Algorithmica*, 84(7):1795–1834, 2022. doi:10.1007/S00453-022-00939-7.

20 Tobias Ganzow and Sasha Rubin. Order-invariant MSO is stronger than counting MSO in the finite. In Susanne Albers and Pascal Weil, editors, *STACS 2008, 25th Annual Symposium on Theoretical Aspects of Computer Science, Bordeaux, France, February 21-23, 2008, Proceedings*, volume 1 of *LIPIcs*, pages 313–324. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Germany, 2008. doi:10.4230/LIPICS.STACS.2008.1353.

21 Michel Habib, Fabien de Montgolfier, Lalla Mouatadid, and Mengchuan Zou. A general algorithmic scheme for combinatorial decompositions with application to modular decompositions of hypergraphs. *Theor. Comput. Sci.*, 923:56–73, 2022. doi:10.1016/J.TCS.2022.04.052.

22 Petr Hlinený. Branch-width, parse trees, and monadic second-order logic for matroids. *J. Comb. Theory B*, 96(3):325–351, 2006. doi:10.1016/J.JCTB.2005.08.005.

23 Michaël Rao. MSOL partitioning problems on graphs of bounded treewidth and clique-width. *Theor. Comput. Sci.*, 377(1-3):260–267, 2007. doi:10.1016/J.TCS.2007.03.043.

24 Michaël Rao. *Décompositions de graphes et algorithmes efficaces*. Phd thesis, Université Paul Verlaine – Metz, 2006.

25 Yann Strozecki. Monadic second-order model-checking on decomposable matroids. *Discret. Appl. Math.*, 159(10):1022–1039, 2011. doi:10.1016/J.DAM.2011.02.005.