# Semantically Acyclic Conjunctive Queries under Functional Dependencies

Diego Figueira

# Semantically Acyclic Conjunctive Queries under Functional Dependencies

Diego Figueira
CNRS, LaBRI

## Abstract

The evaluation problem for Conjunctive Queries (CQ) is known to be NP-complete in combined complexity and W[1]-hard in parameterized complexity. However, acyclic CQs and CQs of bounded tree-width can be evaluated in polynomial time in combined complexity and they are fixed-parameter tractable.

We study the problem of whether a CQ can be rewritten into an equivalent CQ of bounded tree-width, in the presence of unary functional dependencies, assuming bounded arity signatures. We show that this problem is decidable in doubly exponential time, or in exponential time for a subclass of CQ's. When it exists, the algorithm also yields a witness query.

## 1. Introduction

The class of Conjunctive Queries (CQ) is one of the most studied database query languages. It corresponds to select-project-join expressions of the relational algebra, and it is widely used in practice. The evaluation problem for CQs is the problem of, given a relational database $D$, a tuple $\bar{a}$ and a conjunctive query $Q$, whether $\bar{a}$ is in the result set of $Q(D)$ (*i.e.*, of the query $Q$ evaluated in $D$).

However, the evaluation for CQs is NP-complete [7], it requires $|D|^{O(|Q|)}$ time. Notice that we consider both the database $D$ and the query $Q$ as part of the input (this is what is called *combined complexity*).[1] Further, this exponential dependence of the query in the database seems unavoidable since the problem is W[1]-complete in parameterized complexity [19]. When the database is very big, even with moderately small queries the evaluation may become infeasible. Ever since this result, there have been efforts towards finding well-behaved fragments that may lead to a tractable evaluation problem.

One such fragment is the class of *Acyclic* Conjunctive Queries, which corresponds to a syntactic restriction requiring that the hypergraph associated to the query be acyclic. Acyclic CQs can be evaluated in polynomial time, in fact in linear time both in the query and the database: $O(|D| \cdot |Q|)$ [22]. Polynomial-time tractability is further extended to queries of bounded *tree-width* [8, 14]. The tree-width of a query measures, intuitively, how close the query is to being acyclic (the smaller the tree-width the closer). What's more, testing whether a query has tree-width $k$ can be done very efficiently [5], which leads to a useful optimization technique.

The class of tree-width $k$ queries corresponds to a *syntactic* restrictions on the queries. A generalization of this result consists in considering CQs that, although they may not be of tree-width $k$, they are *equivalent* to a CQ of tree-width $k$. This is called *Semantic Acyclicity* [3] in the case of equivalence to acyclic CQs, and here we use the term *Semantic Tree-width-$k$* to denote equivalence to a CQ of tree-width $k$.[2] As pointed out in [3], semantically bounded tree-width CQs can be evaluated in polynomial time, and verifying whether a CQ is semantically of tree-width $k$ is NP-complete (for every $k$); this stems from results in CSP [9, 11]. Concretely, given a CQ $Q$, we can test in NP if $Q$ is equivalent to some query $Q'$ of tree-width $k$; if so, we can evaluate $Q$ in polynomial time. In the sentence before, the fact that $Q$ is "equivalent" to $Q'$ means that $Q(D) = Q'(D)$ for *every* database $D$. Our work is motivated by the following question: Can we extend this result for databases that verify some integrity constraints?

A very common integrity constraint on databases is the use of functional dependencies. These constraints capture the most prominent form of data dependency, which are fundamental in modern database models. A functional dependency states that an attribute of a relation functionally determines another attribute (*e.g.*, 'SSN' determines 'name' in the relation 'Employees'; in other words, every two rows with the same 'SSN' must have the same 'name').

This paper studies the semantic tree-width-$k$ problem under the presence of functional dependencies. Assuming relations have bounded arity, this problem generalizes the previous problems discussed, by making use of the information on data dependencies to produce a query that can be evaluated efficiently. As we will see, this makes a difference, as classes of queries which are not semantically of bounded tree-width may become of bounded tree-width when working under functional dependencies.

Simply put, the contribution of this paper is that the following problem is decidable:

[1] When the query is considered to be fixed (this is called *data complexity*), the evaluation of CQs are in the tractable class $AC^0$ [15] and thus, in particular, in LogSpace.

[2] We remark that if we assume that the arities of relations is bounded, with semantic tree-width-$k$ queries we are in a more general setup. Indeed, for any bound $b$ on the arity there exists $k$ so that the class of semantic acyclic queries over relations of arity $\leq b$ is also semantically of tree-width-$k$.

*Given a CQ $Q$ and a set of unary functional dependencies $\Sigma$ of bounded arity, is there a CQ $Q'$ of tree-width $\leq k$ so that $Q'$ and $Q$ are equivalent over databases satisfying $\Sigma$?*

We show that this problem can be decided in 2ExpTime for the full class of CQs, or in ExpTime for a fragment thereof; and that the witness query $Q'$ can also be provided. Thus, whenever the answer is positive the algorithm then yields a fixed-parameter tractable (FPT) evaluation algorithm of complexity $f(|Q|)\cdot|D|^c$ for a constant $c$ and a doubly-exponential function $f$. In [4], Barceló et al. show that this problem is undecidable as soon as we consider more general constraints, namely tgd's and egd's, instead of unary functional dependencies.

## 2. Preliminaries

Let $\mathbb{N} = \{0, 1, 2, \dots\}$. We use the bar notation $\bar{a}$ to denote a vector of elements, whose $i$-th element ($i > 0$) is denoted by $\bar{a}[i]$.

***Relational structures*** A **relational vocabulary** $\sigma$ consists of a collection of relation symbols, each with a specified **arity**. For a relation $R$ we write $\mathit{arity}(R) \in \mathbb{N} \setminus \{0\}$ to denote its arity.[3] A **$\sigma$-structure** $\mathbb{A}$ consists of a universe $A$, or **domain**, and an **interpretation** which associates to each relation symbol $R \in \sigma$, a relation $R^{\mathbb{A}} \subseteq A^{\mathit{arity}(R)}$. For any binary relation $R$, we say that $a \xrightarrow{R} b$ [resp. $a \xleftarrow{R} b$] is an **edge** of $\mathbb{A}$ if $(a, b) \in R^{\mathbb{A}}$ [resp. $(b, a) \in R^{\mathbb{A}}$]. Thus, whenever we say that we 'add'/'remove' an edge $a \xrightarrow{R} b$ to/from $\mathbb{A}$, we refer to the respective operation on $(a, b)$ and the set $R^{\mathbb{A}}$. We abuse notation writing $a \xrightarrow{R} b \xrightarrow{R'} c$ as short for $(a \xrightarrow{R} b)(b \xrightarrow{R'} c)$; $a \xrightarrow{R} b \xleftarrow{R'} c$ as short for $(a \xrightarrow{R} b)(b \xleftarrow{R'} c)$, etc. We use $\mathbb{A}, \mathbb{B}, \mathbb{C}, \mathbb{A}', \mathbb{B}', \dots$ to denote relational structures, and $A, B, C, A', B', \dots$ to denote their respective domains. We work here with *finite* structures, and henceforward by structure we mean a finite one. Further, we assume that all relations have *bounded arity*, that is, there is a fixed constant $n_0 \in \mathbb{N}$ so that all relations in the signature have arity bounded by $n_0$.

A **graph** is a structure $G = (V, E)$, where $E$ is a collection of subsets of $V$ of size 2. Thus, our graphs are undirected, loopless, and without parallel edges. The **Gaifman graph** of a $\sigma$-structure $\mathbb{A}$, denoted by $\mathcal{G}(\mathbb{A})$, is the graph whose set of nodes is the universe of $\mathbb{A}$, and whose set of edges consists of all pairs $\{a, a'\}$ of distinct elements of $A$ such that $a$ and $a'$ appear together in some tuple of a relation in $\mathbb{A}$.

Given two $\sigma$-structures $\mathbb{A}, \mathbb{B}$, we say that $\mathbb{A}$ is a **substructure** of $\mathbb{B}$ (noted $\mathbb{A} \subseteq \mathbb{B}$) if $A \subseteq B$, $R^{\mathbb{A}} \subseteq R^{\mathbb{B}}$ for all $R \in \sigma$. We say that $\mathbb{A}$ is an **induced substructure** of $\mathbb{B}$ if it is a substructure so that $R^{\mathbb{A}} = R^{\mathbb{B}} \cap A^{\mathit{arity}(R)}$ for all $R \in \sigma$. In this case we say that $\mathbb{A}$ is the **substructure induced by** $A$ and we denote it by $\mathbb{B}|_A$.

A **homomorphism** from a $\sigma$-structure $\mathbb{A}$ to a $\sigma$-structure $\mathbb{B}$ is a mapping $h : A \to B$ so that for each relation symbol $R \in \sigma$, if $(a_1, \dots, a_r) \in R^{\mathbb{A}}$, then $(h(a_1), \dots, h(a_r)) \in R^{\mathbb{B}}$. We will sometimes write $h(a_1, \dots, a_r)$ as short for $(h(a_1), \dots, h(a_r))$. An **onto homomorphism** is a surjective homomorphism. We write $\mathbb{A} \to \mathbb{B}$ to denote that there is a homomorphism from $\mathbb{A}$ to $\mathbb{B}$, and we write $h : \mathbb{A} \to \mathbb{B}$ to denote that $h$ is a homomorphism from $\mathbb{A}$ to $\mathbb{B}$. For $h : \mathbb{A} \to \mathbb{B}$ we write $h(\mathbb{A})$ to denote the structure resulting from identifying the elements of $\mathbb{A}$ with equal $h$-image (note that it is isomorphic to a substructure of $\mathbb{B}$). If $\mathbb{A} \subseteq \mathbb{B}$, we say that $h$ is **image-identity** if for every element $x$ of its image, $h(x) = x$. If $\mathbb{A} \to \mathbb{B}$ and $\mathbb{B} \to \mathbb{A}$ we say that $\mathbb{A}$ and $\mathbb{B}$ are **hom-equivalent**, and we write it $\mathbb{A} \leftrightarrow \mathbb{B}$. We use $\cong$ for the isomorphism relation. Given a $\sigma$-structure $\mathbb{A}$ there is (up to isomorphism) a unique structure $\mathbb{A}'$ so that

- it is hom-equivalent to $\mathbb{A}$, that is there are $h : \mathbb{A} \to \mathbb{A}'$ and $h' : \mathbb{A}' \to \mathbb{A}$,
- it has the minimal number of elements.

Such a structure $\mathbb{A}'$ is called the **core** of $\mathbb{A}$. We write $\mathit{core}(\mathbb{A})$ to denote the core of $\mathbb{A}$, and we say that $\mathbb{A}$ is **a core** if $\mathit{core}(\mathbb{A}) \cong \mathbb{A}$. It easy to see that the core of $\mathbb{A}$ is, up to isomorphism, a substructure of $\mathbb{A}$, and that there is always an image-identity $h : \mathbb{A} \to \mathit{core}(\mathbb{A})$ (see, *e.g.*, [18]).

***Conjunctive Queries*** One of the most studied fragments of First-Order logic (FO) in relation to database queries is the fragment of *Conjunctive Queries* (also known as Primitive Positive Logic, or Existential Positive FO). The class of Conjunctive Queries (CQ) is the fragment of FO corresponding to positive '*select-project-join*' queries of the Relational Algebra or to positive '*select-from-where*' queries of SQL, where by 'positive' we mean that there are no inequalities in the *select* [resp. *where*] conditions (we refer the reader to [1, §4] for more details). These are FO-formulas of the form

$$\varphi = \exists y_1, \dots, y_n\, \theta, \qquad (\dagger)$$

where $\theta$ is a conjunction of atomic formulas. For simplicity, we will work here with *boolean* CQs (*i.e.*, formulas with no free variables) without constants. Every conjunctive query of the form (†) over a relational vocabulary $\sigma$ gives rise to a **canonical structure** (sometimes called *tableau*) $\mathbb{C}_\varphi$ with $n$ elements, where the elements of $\mathbb{C}_\varphi$ are the variables $x_1, \dots, x_n$, and the relations of $\mathbb{C}_\varphi$ consist of the tuples of terms in the conjuncts of $\theta$. Given a CQ $\varphi$, we write $\mathbb{C}_\varphi$ for the canonical structure of $\varphi$. Likewise, any $\sigma$-structure $\mathbb{A}$ with domain $A = \{x_1, \dots, x_n\}$ gives rise to a **canonical conjunctive query** $\varphi_{\mathbb{A}}$ where $\varphi_{\mathbb{A}}$ has a conjunct $R(\bar{t})$ iff $\bar{t} \in R^{\mathbb{A}}$.

***Tree-Width*** A **tree decomposition** of a graph $G = (V, E)$ is a tree (*i.e.*, an acyclic, connected graph) $T = (V', E')$ so that its vertices, also called **bags**, are subsets of $V$, $V' \subseteq 2^V$, and

- $\bigcup_{X \in V'} X = V$;
- for every edge $\{v, v'\} \in E$ there is some $X \in V'$ so that $\{v, v'\} \subseteq X$;
- for every $v \in V$ we have that $\{X \in V' \mid v \in X\}$ is a connected component of $T$.

The **width** of the tree decomposition $T$ is defined as

$$\max_{X \in V'} |X| - 1.$$

The **tree-width** of $G$ is defined as the minimum width over its tree decompositions. We denote the tree-width of $G$ as $\mathit{tw}(G)$. Note that $0 \leq \mathit{tw}(G) < |V|$. The notion of tree-width is generalized to structures and CQs via canonical structures and Gaifman graphs: the tree-width $\mathit{tw}(\mathbb{A})$ of a $\sigma$-structure $\mathbb{A}$ is defined as $\mathit{tw}(\mathcal{G}(\mathbb{A}))$, and that of a CQ $\varphi$ as $\mathit{tw}(\mathbb{C}_\varphi)$. Let $\mathrm{TW}_{\leq k}$ denote the set of all structures with tree-width $\leq k$, and let $\mathrm{CQ}_k$ be the set of all CQs of tree-width $\leq k$. We remind the reader that the main interest of tree-width for this paper stems from the fact that, although the evaluation of CQs is an NP-complete problem [7] (in combined complexity), the evaluation problem for $\mathrm{CQ}_k$ can be done in polynomial time, for every fixed $k$. Further, the problem is in the paralellizable class LogCFL [14].

***Functional dependencies*** A **unary functional dependency** (henceforward just 'FD') over a signature $\sigma$ is a triple $(R, i, j)$, that we will normally write '$R[i{\mapsto}j]$', where $R \in \sigma$, $i, j \in \{1, \dots, \mathit{arity}(R)\}$, and $i \neq j$. A $\sigma$-structure $\mathbb{A}$ is said to **satisfy** an FD $R[i{\mapsto}j]$ if for all $\bar{a}, \bar{b} \in R^{\mathbb{A}}$, if $\bar{a}[i] = \bar{b}[i]$ then $\bar{a}[j] = \bar{b}[j]$. We normally use the letter $\Sigma$ to denote a *set* of FDs. A structure satisfies $\Sigma$ if it satisfies

[3] In this work we do not consider constants (*i.e.*, 0-arity relations).

all of its FDs. We write $\mathcal{C}_\Sigma^\sigma$ for the class of all $\sigma$-structures satisfying $\Sigma$. We say that an edge $a \xrightarrow{R} b$ of $\mathbb{A}$ is a **$\Sigma$-edge**, if $R$ appears in $\Sigma$.

Given a structure $\mathbb{A}$ and a set of FDs $\Sigma$ we define the *Chase* relation [2, 16] between structures $\mathbb{A} \Rightarrow_\Sigma \mathbb{B}$, if there is $R[i{\mapsto}j] \in \Sigma$, $\bar{a}, \bar{b} \in R^{\mathbb{A}}$ with $\bar{a}[i] = \bar{b}[i]$ and $\bar{a}[j] \neq \bar{b}[j]$, and $\mathbb{B}$ is the result of replacing every $\bar{b}[j]$ with $\bar{a}[j]$ in every relation of $\mathbb{A}$ and deleting $\bar{b}[j]$ from the domain of $\mathbb{A}$. It can be seen that $\Rightarrow_\Sigma$ is terminating and Church-Rosser confluent, up to isomorphism [1]. Let us write $\Rightarrow_\Sigma^*$ to denote the reflexive-transitive closure of $\Rightarrow_\Sigma$. Let us call $\mathit{chase}_\Sigma(\mathbb{A})$ to the structure $\mathbb{B}$ so that $\mathbb{A} \Rightarrow_\Sigma^* \mathbb{B}$ and $\mathbb{B}$ satisfies $\Sigma$ (such $\mathbb{B}$ is unique, up to isomorphism). We say that $\mathbb{B}$ is **a chase**, if $\mathit{chase}_\Sigma(\mathbb{B}) \cong \mathbb{B}$. For $\mathbb{A} \Rightarrow_\Sigma \mathbb{B}$, where $\mathbb{B}$ is obtained by replacing $a, a' \in A$ with $a$ in $\mathbb{A}$, we define the **provenance homomorphism of** $\mathbb{A} \Rightarrow_\Sigma \mathbb{B}$ as the homomorphism $h : \mathbb{A} \to \mathbb{B}$ so that $h(a') = a$ and $h(b) = b$ for all other $b \in A \setminus \{a'\}$. The provenance homomorphism of $\mathbb{A} \Rightarrow_\Sigma^* \mathbb{B}$ is the homomorphism $\mathbb{A} \to \mathbb{B}$ resulting from the stepwise composition of provenance homomorphisms as just defined.

**Lemma 2.1.** *[2, 7, 16] A boolean CQ $\varphi$ is equivalent to $\psi$ over $\mathcal{C}_\Sigma^\sigma$ iff $\mathit{core}(\mathit{chase}_\Sigma(\mathbb{C}_\varphi)) \cong \mathit{core}(\mathit{chase}_\Sigma(\mathbb{C}_\psi))$.*

The following lemma is straightforward from the definition of $\mathit{chase}_\Sigma$ and the fact that the *core* is an induced substructure.

**Lemma 2.2.** *For every structure $\mathbb{A}$ and set of FDs $\Sigma$ we have $\mathit{chase}_\Sigma(\mathit{core}(\mathit{chase}_\Sigma(\mathbb{A}))) = \mathit{core}(\mathit{chase}_\Sigma(\mathbb{A}))$.*

***Semantic bounded tree-width queries*** The problem we study here is that of whether one can rewrite a CQ into a an equivalent one (for structures satisfying a set of FDs $\Sigma$) of treewidth at most $k$. We call this problem the **Semantic Tree-width-$k$**, noted STW$_k$, and it is formally defined as follows.

| | |
|---|---|
| Problem: | STW$_k$ |
| Input: | A CQ $\varphi$, a set of FDs $\Sigma$ |
| Output: | 'Yes' iff there exists $\psi \in$ CQ so that $\varphi \equiv_\Sigma \psi$ and $\mathit{tw}(\psi) \leq k$. |

## 3. Restriction to binary queries

Our study of the semantic tree-width problem will be focused on binary queries, that is, signatures whose relations are of arity at most 2. However, in this section we show that this restriction is without loss of generality (with the bounded arity assumption).

Given a $\sigma$-structure $\mathbb{A}$ and a set of FDs $\Sigma$, let $\mathbb{A}_\Sigma$ be a structure over a signature $\sigma'$ consisting of:

- a new binary relation $R_{S[i\mapsto j]}$ for every $S[i{\mapsto}j] \in \Sigma$,
- all the unary and binary relations of $\sigma$, and
- binary relations $S_1, \ldots, S_k$ for every $k$-ary relation $S \in \sigma$ with $k > 2$.

The universe of $\mathbb{A}_\Sigma$ consists of $A$ plus a new element '$key(\bar{a})$' for every $k$-tuple $\bar{a}$ appearing in some relation of $\mathbb{A}$ for some $k > 2$. The interpretation of unary and binary relations is as in $\mathbb{A}$. For each $k$-ary relation $S \in \sigma$ with $k > 2$ we define $(S_i)^{\mathbb{A}_\Sigma} = \{(key(\bar{a}), \bar{a}[i]) \mid \bar{a} \in S^{\mathbb{A}}\}$. Finally, we define $R^{\mathbb{A}_\Sigma}_{S[i\mapsto j]} = \{(\bar{a}[i], \bar{a}[j]) \mid \bar{a} \in S^{\mathbb{A}}\}$. Let $\Gamma_\Sigma = \{R_f[1{\mapsto}2] \mid f \in \Sigma\}$. Figure 1 shows an example.

**Lemma 3.1.** *$\mathbb{A}$ satisfies $\Sigma$ iff $\mathbb{A}_\Sigma$ satisfies $\Gamma_\Sigma$.*

*Proof.* If there are $\bar{a}, \bar{a}' \in S^{\mathbb{A}}$ and $S[i{\mapsto}j] \in \Sigma$ so that $\bar{a}[i] = \bar{a}'[i]$ and $\bar{a}[j] \neq \bar{a}'[j]$ (*i.e.*, $\mathbb{A}$ does not satisfy $\Sigma$), it follows that $(\bar{a}[i], \bar{a}[j]), (\bar{a}'[i], \bar{a}'[j]) \in R^{\mathbb{A}_\sigma}_{S[i\mapsto j]}$ and thus $\mathbb{A}_\Sigma$ does not satisfy

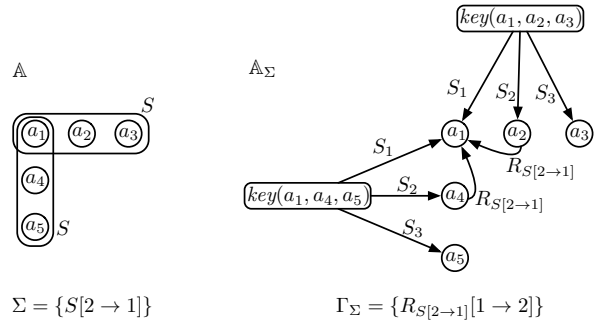



**Figure 1.** Example of construction of a binary structure $\mathbb{A}_\Sigma$ with FDs $\Gamma_\Sigma$ from a structure $\mathbb{A}$ with $S^{\mathbb{A}} = \{(a_1, a_2, a_3), (a_1, a_4, a_5)\}$.

$\Gamma_\Sigma$. Conversely, if $(a, b), (a, b') \in R^{\mathbb{A}_\sigma}_{S[i\mapsto j]}$ with $b \neq b'$ (*i.e.*, $\mathbb{A}_\Sigma$ does not satisfy $\Gamma_\Sigma$), there must be some $\bar{a}, \bar{a}' \in S^{\mathbb{A}}$ so that $\bar{a}[i] = a$, $\bar{a}[j] = b$, $\bar{a}'[i] = a$, $\bar{a}'[j] = b'$, which means that $\mathbb{A}$ does not satisfy $S[i{\mapsto}j]$. □

It is also easy to see the following.

**Lemma 3.2.** $\mathit{chase}_{\Gamma_\Sigma}(\mathbb{A}_\Sigma) \cong (\mathit{chase}_\Sigma(\mathbb{A}))_{\Gamma_\Sigma}$.

*Proof.* If $\mathbb{A}_\Sigma \Rightarrow_{\Gamma_\Sigma} \mathbb{A}'$ by collapsing two elements $a_1, a_2$, it is because there is some $a_0$ so that $(a_0, a_1)$ and $(a_0, a_2)$ are in the interpretation of $R_{S[i\mapsto j]}$ in $\mathbb{A}_\Sigma$, which means that there are tuples $\bar{a}_1, \bar{a}_2$ in $S^{\mathbb{A}}$ so that $\bar{a}_1[i] = \bar{a}_2[i] = a_0$, $\bar{a}_1[j] = a_1$ and $\bar{a}_2[j] = a_2$. Thus, we can apply $\Rightarrow_\Sigma$ on $\mathbb{A}$ for these two tuples obtaining $\mathbb{A}''$, the result of collapsing $a_1, a_2$. It is not hard to see that $(\mathbb{A}'')_{\Gamma_\Sigma} = \mathbb{A}'$.

In a similar way, one can show that if $\mathbb{A} \Rightarrow_\Sigma \mathbb{A}''$ by collapsing two elements, we can also collapse these elements in $\mathbb{A}_\Sigma \Rightarrow_{\Gamma_\Sigma} \mathbb{A}'$ obtaining $\mathbb{A}' = (\mathbb{A}'')_{\Gamma_\Sigma}$.

Since the Chase yields a unique structure up to isomorphism [1], by iterating the reasoning above the statement follows. □

For a CQ $\varphi$ we define $\varphi_\Sigma$ as the canonical conjunctive query corresponding to $(\mathbb{C}_\varphi)_\Sigma$.

**Lemma 3.3.** *For every pair of CQs $\varphi, \psi$, we have $\varphi \equiv_\Sigma \psi$ iff $\varphi_\Sigma \equiv_{\Gamma_\Sigma} \psi_\Sigma$.*

*Proof.* Let $\mathbb{A} = \mathit{chase}_{\Gamma_\Sigma}((\mathbb{C}_\varphi)_\Sigma)$, and $\mathbb{B} = \mathit{chase}_{\Gamma_\Sigma}((\mathbb{C}_\psi)_\Sigma)$.

For the left-to-right direction, in order to show $\varphi_\Sigma \equiv_{\Gamma_\Sigma} \psi_\Sigma$ it suffices to show that $\mathbb{A}$ and $\mathbb{B}$ are hom-equivalent. By Lemma 3.2 we have that $\mathbb{A} \cong (\mathit{chase}_\Sigma(\mathbb{C}_\varphi))_{\Gamma_\Sigma}$ and $\mathbb{B} \cong (\mathit{chase}_\Sigma(\mathbb{C}_\psi))_{\Gamma_\Sigma}$. Since $\varphi \equiv_\Sigma \psi$ we have that there is a homomorphism $f : \mathit{chase}_\Sigma(\mathbb{C}_\varphi) \to \mathit{chase}_\Sigma(\mathbb{C}_\psi)$. Then we simply extend $f$ with $key(\bar{a}) \mapsto key(f(\bar{a}))$ for every $key(\bar{a}) \in A$ obtaining a homomorphism $f' : \mathbb{A} \to \mathbb{B}$. The other homomorphism $\mathbb{B} \to \mathbb{A}$ is obtained in a similar way. Thus, $\varphi_\Sigma \equiv_{\Gamma_\Sigma} \psi_\Sigma$.

For the right-to-left direction, suppose we have $f : \mathbb{A} \to \mathbb{B}$ and $g : \mathbb{B} \to \mathbb{A}$. Due to Lemma 3.2, we can assume $f : (\mathit{chase}_\Sigma(\mathbb{C}_\varphi))_{\Gamma_\Sigma} \to (\mathit{chase}_\Sigma(\mathbb{C}_\psi))_{\Gamma_\Sigma}$. It is not hard to see that $f$ restricted to the universe of $\mathbb{C}_\varphi$ is a homomorphism from $\mathit{chase}_\Sigma(\mathbb{C}_\varphi)$ to $\mathit{chase}_\Sigma(\mathbb{C}_\psi)$. A similar reasoning applies to $g$ and we thus obtain $\varphi \equiv_\Sigma \psi$. □

We also have that these modifications of the structures can only increase the tree-width in 1. For a structure $\mathbb{A}$, let $\mathit{maxarity}(\mathbb{A})$ be defined as $\max\{|\{a_1, \ldots, a_n\}| : (a_1, \ldots, a_n) \in S^{\mathbb{A}} \text{ for some } S\}$. Observe that *maxarity* is a number between 1 and the maximum arity of the relations in the signature. Further, note that $\mathit{maxarity}(\mathbb{A}) \leq \mathit{tw}(\mathbb{A}) + 1$.

**Lemma 3.4.** *For every $\sigma$-structure $\mathbb{A}$ we have*

$$tw(\mathbb{A}_\Sigma) \leq tw(\mathbb{A}) + 1.$$

*Proof.* We show: $tw(\mathbb{A}_\Sigma) \leq \max(tw(\mathbb{A}), maxarity(\mathbb{A}))$. Given a tree decomposition of $\mathbb{A}$, it suffices to add, for each $key(a_1, \ldots, a_n)$ in the universe of $\mathbb{A}_\Sigma$ a new leaf with bag

$$\{key(a_1, \ldots, a_n), a_1, \ldots, a_n\}$$

of cardinality $\leq n + 1$ to the tree decomposition, hanging from any node containing $\{a_1, \ldots, a_n\}$ (note that there must be at least one). □

Since $maxarity(\mathbb{A}) - 1 \leq tw(\mathbb{A})$, the lemma above tells us that the $(\cdot)_\Sigma$ operation increases the tree-width in 1 at the most.

**Lemma 3.5.** *For every $\sigma$-structure $\mathbb{A}$ we have*

$$tw(\mathbb{A}) \leq tw(\mathbb{A}_\Sigma) + maxarity(\mathbb{A}) - 1.$$

*Proof.* Given a tree decomposition of $\mathbb{A}_\Sigma$, we obtain a decomposition of $\mathbb{A}$ by replacing, in every bag, $key(a_1, \ldots, a_n)$ with $a_1, \ldots, a_n$. The cardinality of the bags is then increased in at most $maxarity(\mathbb{A}) - 1$. □

In turn, the lemma above is simply stating that the tree-width of $\mathbb{A}_\Sigma$ cannot be much smaller than that of $\mathbb{A}$.

The previous two lemmas imply that we can focus on binary queries without much loss of generality. This, added to the fact that the technical contributions are greatly simplified when restricted to binary signatures (*i.e.*, to edge-labeled graphs), propels us to work on binary signatures and with sets of FD $\Sigma$ of the form:

$$\Sigma = \{R_1[1{\mapsto}2], \ldots, R_l[1{\mapsto}2]\} \qquad (\ddagger)$$

for binary relations $R_1, \ldots, R_l \in \sigma$. For these reasons, we will assume the simplified setup of binary signatures and a set of FDs as shown above for the remaining of this paper.

***Restatement in terms of structures*** To further simplify matters, we will work only with $\sigma$-structures, avoiding dealing with CQs and having to go back and forth in the CQ/structures duality. The $\text{STW}_k$ problem can be cast into the problem of whether, given a structure $\mathbb{A}$ and a set of FDs $\Sigma$ the following holds: $\{\mathbb{B} \mid core(chase_\Sigma(\mathbb{B})) \cong \mathbb{A}\} \cap \text{TW}_{\leq k} \neq \emptyset$. We denote this problem by "$(core\text{-}chase)^{-1} \cap \text{TW}_{\leq k}$".

**Lemma 3.6.** *For every fixed $k$, there is an NP reduction from* $\text{STW}_k$ *into* $(core\text{-}chase)^{-1} \cap \text{TW}_{\leq k}$.

*Proof.* Given a CQ $\varphi$, one can compute $core(chase_\Sigma(\varphi))$ in NP (the $chase_\Sigma$-computation is polynomial [1] and the *core*-computation is DP-complete [12]). For $\mathbb{A} = core(chase_\Sigma(\mathbb{C}_\varphi))$ we have that there is a structure $\mathbb{B} \in \text{TW}_{\leq k}$ so that $core(chase_\Sigma(\mathbb{B})) \cong \mathbb{A}$ iff $\varphi_\mathbb{B}$ is equivalent to $\varphi$ (by Lemma 2.1) and of tree-width $\leq k$. □

## 4. Tree-like queries

For queries of tree-width 1 the problem is trivial due to the fact that both the chase and core are monotone with respect to tree-width 1.

**Lemma 4.1.** *For every structure $\mathbb{A} \in \text{TW}_{\leq 1}$, we have*

- $tw(chase_\Sigma(\mathbb{A})) \leq 1$, *and*
- $tw(core(\mathbb{A})) \leq 1$.

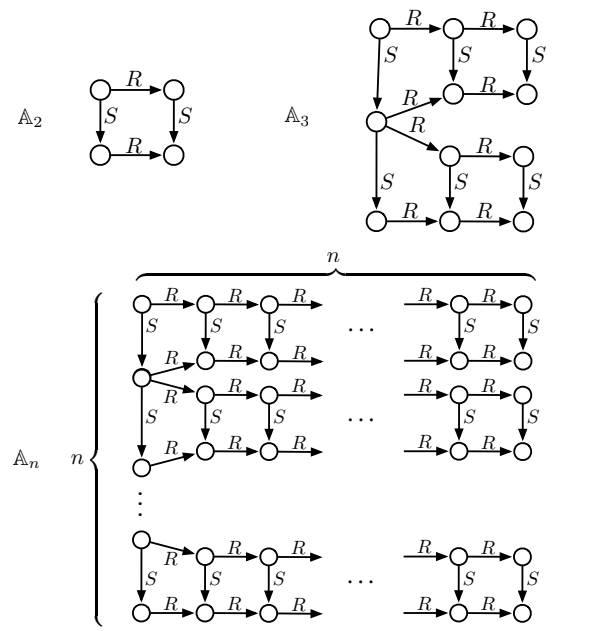



**Figure 2.** Depiction of $\mathbb{A}_2$, $\mathbb{A}_3$, $\mathbb{A}_n$.

*Proof.* Note that the structures of tree-width 1 are those whose underlying undirected graph is acyclic. Note that identifying any two nodes at distance 2 of an acyclic graph preserves acyclicity. Thus, $\Rightarrow_\Sigma$ preserves tree-width $\leq 1$. On the other hand, since the core of a structure is isomorphic to a substructure, and acyclicity is closed under substructures, it follows that the core of a tree-width 1 structure is tree-width $\leq 1$. □

In light of this, one can already answer the $(core\text{-}chase)^{-1} \cap \text{TW}_{\leq k}$ problem for $k = 1$ in polynomial time: the answer is positive iff the tree-width of the input is $\leq 1$. Since in general testing tree-width $\leq k$ (for any fixed $k$) is in linear time [5], this problem is linear. Through the reduction of Lemma 3.6 we obtain that $\text{STW}_1$ is decidable in NP.

**Theorem 4.2.** $\text{STW}_1$ *is in NP.*

If the previous lemma was true for every tree-width, this would imply that $\text{STW}_k$ is in NP for every $k$. However, the statement of Lemma 4.1 above fails for every $k > 1$ as the following lemma shows.

**Lemma 4.3.** *For every $n \in \mathbb{N}$ there is a structure $\mathbb{A}$ so that $tw(\mathbb{A}) = 2$, and $tw(chase_\Sigma(\mathbb{A})) = n$.*

*Proof.* Let $\sigma = \{R, S\}$ be binary relations and let $\Sigma = \{R[1{\mapsto}2]\}$. For every $n \geq 2$, let $\mathbb{A}_n$ be defined as in Figure 2. Let $\mathbb{B}_n$ be defined as an $n \times n$ grid:

$$B_n = \{(i, j) \mid 1 \leq i, j \leq n\},$$
$$S^{\mathbb{B}_n} = \{((i, j), (i + 1, j)) \mid 1 \leq j \leq n, 1 \leq i < n\},$$
$$R^{\mathbb{B}_n} = \{((i, j), (i, j + 1)) \mid 1 \leq i \leq n, 1 \leq j < n\}.$$

Note that $\mathbb{B}_2 \cong \mathbb{A}_2$. One can see that $chase_\Sigma(\mathbb{A}_n) = \mathbb{B}_n$, $tw(\mathbb{A}_n) = 2$ and $tw(\mathbb{B}_n) = n$ for every $n$. □

## 5. Cyclic queries

For the general case of CQs that can contain cycles, one obvious idea would be to describe the solutions to the problem with an

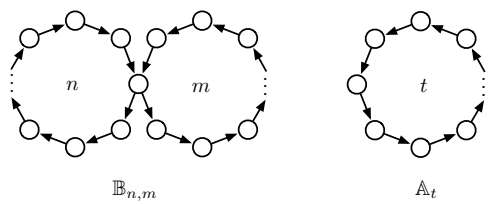



**Figure 3.** Counterexample of Lemma 5.1.

MSO formula. Since MSO is decidable on bounded tree-width structures [20], we would therefore obtain a decision procedure. That is, for a given structure $\mathbb{A}$ and FDs $\Sigma$, we produce an MSO formula $\varphi_{\mathbb{A}}$ whose models are $\{\mathbb{B} \mid \mathit{core}(\mathit{chase}_\Sigma(\mathbb{B})) = \mathbb{A}\}$, and we test whether $\varphi_{\mathbb{A}}$ has a model of tree-width $k$. This would yield a decision procedure for $(\mathit{core}\text{-}\mathit{chase})^{-1} \cap \mathrm{TW}_{\leq k}$ with input $\mathbb{A}, \Sigma$. However, this is in general not possible; the first problem we encounter is that the preimage of $\mathit{chase}_\Sigma$ is not MSO-definable, as the following lemma shows.

**Lemma 5.1.** *Given $\mathbb{A}$, $\Sigma$, the set $\{\mathbb{B} \mid \mathit{chase}_\Sigma(\mathbb{B}) \cong \mathbb{A}\}$ is not MSO definable in general.*

*Proof.* Let $\Sigma = \{R[1{\mapsto}2]\}$ and let $\mathbb{A}_t$ and $\mathbb{B}_{n,m}$ with $t, n, m \in \mathbb{N}$ be defined as in Figure 3. That is, $\mathbb{B}_{n,m}$ consists of two nested $R$-cycles of size $n$ and $m$ (where $n$ and $m$ refers to the number of edges), and $\mathbb{A}_t$ is an $R$-cycle of size $t$.

Note that for $n > m$, we have that $\mathbb{B}_{n,m} \Rightarrow^*_\Sigma \mathbb{B}_{n-m,m}$ and that $\mathit{chase}_\Sigma(\mathbb{B}_{n,n}) = \mathbb{A}_n$. Thus, $\mathit{chase}_\Sigma(\mathbb{B}_{n,m})$ basically computes $\mathrm{GCD}(n, m)$ through the Euclidean algorithm,

$$\mathit{chase}_\Sigma(\mathbb{B}_{n,m}) = \mathbb{A}_{\mathrm{GCD}(n,m)}.$$

Suppose, by means of contradiction, that there exists an MSO sentence $\varphi$ of quantifier rank $k$ so that $\mathbb{B} \models \varphi$ iff $\mathit{chase}_\Sigma(\mathbb{B}) = \mathbb{A}_1$ (note that $\mathbb{A}_1$ consists of one element in a reflexive $R$ relation). Note that the MSO type of rank $k$ of $\mathbb{B}_{n,m}$ is determined by the MSO type of rank $k$ of $\mathbb{A}_n$ and the MSO type of rank $k$ of $\mathbb{A}_m$. Let $p_i$ be the $i$-th smallest positive prime number, and let $S = \{(p_i, (p_i{-}1)!) \mid i \in \mathbb{N}\}$. Since there is a finite number of rank $k$ MSO types, there must be $i < j$ so that the type of $\mathbb{A}_{p_i}$ is equal to that of $\mathbb{A}_{p_j}$ and the type of $\mathbb{A}_{(p_i-1)!}$ is equal to that of $\mathbb{A}_{(p_j-1)!}$. Therefore, $\mathbb{B}_{p_j,(p_j-1)!} \models \varphi \quad \Leftrightarrow \quad \mathbb{B}_{p_i,(p_j-1)!} \models \varphi$,, which is in contradiction with our assumption since $\mathrm{GCD}(p_j, (p_j - 1)!) = 1$ but $\mathrm{GCD}(p_i, (p_j - 1)!) \neq 1$. □

Since the chased structures in the proof above are cores, we also have the following.

**Corollary 5.2.** *Given $\mathbb{A}$, $\Sigma$, the set $\{\mathbb{B} \mid \mathit{core}(\mathit{chase}_\Sigma(\mathbb{B})) \cong \mathbb{A}\}$ is not MSO definable.*

Instead of attempting to describe *all* the structures from

$$\{\mathbb{B} \mid \mathit{core}(\mathit{chase}_\Sigma(\mathbb{B})) = \mathbb{A}\}$$

with MSO, we will describe some necessary and sufficient properties that at least *one* structure from

$$\{\mathbb{B} \mid \mathit{core}(\mathit{chase}_\Sigma(\mathbb{B})) = \mathbb{A}\} \cap \mathrm{TW}_{\leq k}$$

must have, should there be any. These properties can be informally described as the existence of some paths whose labels form words from a regular language, and that can be described with MSO.

***Structure of the proof***

- In Section 6 we show that there is always a tree-width 2 structure in the $\mathit{chase}_\Sigma$-preimage of any rooted structure (*i.e.*, a structure with a 'least' element from which every other element can be reached) containing only edges from $\Sigma$.
- In Section 7 we define, given $h : \mathbb{A} \to \mathbb{C}$, the *$h$-regular complex paths* of $\mathbb{A}$, as those paths whose $h$-image belongs to a regular language $\mathcal{L}_\mathbb{C}$ which depends on $\mathbb{C}$. The idea is that every such path of $\mathbb{A}$ becomes a path of $\mathbb{C}$ once we apply the chase procedure. We exhibit necessary and sufficient conditions for $\mathbb{A}$ to verify $\mathit{core}(\mathit{chase}_\Sigma(\mathbb{A})) = \mathbb{C}$ in terms of the existence of $h$ and some $h$-regular complex paths in $\mathbb{A}$. These conditions ask for a homomorphism $h : \mathbb{A} \to \mathbb{C}$ and the existence of a representative element $a_i$ in $\mathbb{A}$ for every least strongly connected component $X_i$ of $\mathbb{C}|_\Sigma$ (*i.e.*, $\mathbb{C}$ restricted to relations of $\Sigma$), and the existence of $h$-regular complex paths from $a_i$ to an element $a$ in $\mathbb{A}$ whenever there is a path from $h(a_i)$ to $h(a)$ in $\mathbb{C}$. This result uses the decomposition of the previous section. Since these conditions can be encoded in MSO, decidability for $(\mathit{core}\text{-}\mathit{chase})^{-1} \cap \mathrm{TW}_{\leq k}$ follows.
- Finally, in Section 8, we show that the aforementioned conditions can be encoded in a tree-walking automaton (TWA) of exponential size, running on a tree-width $k$ decomposition of the input structure $\mathbb{A}$. In this way, we reduce the $(\mathit{core}\text{-}\mathit{chase})^{-1} \cap \mathrm{TW}_{\leq k}$ problem to the emptiness problem for some TWA of exponential size. Since the latter problem is in ExpTime, we obtain a 2ExpTime procedure for $(\mathit{core}\text{-}\mathit{chase})^{-1} \cap \mathrm{TW}_{\leq k}$, and thus also for $\mathrm{STW}_k$. We also identify a class of CQs for which $\mathrm{STW}_k$ can be solved in single exponential time.

## 6. Decomposition of Σ-components

In this section we show how to decompose any rooted structure $\mathbb{A}$ (*i.e.*, one so that there is an element that can reach any other element) containing only $\Sigma$-edges into a structure $\mathbb{A}'$ so that $\mathit{tw}(\mathbb{A}') = 2$ and $\mathbb{A}' \Rightarrow^*_\Sigma \mathbb{A}$. To prove this, we show that all simple cycles in the underlying undirected graph of $\mathbb{A}$ can be rearranged in a cactus shape of tree-width 2. The idea is that structures that look like the left structure of Figure 5 are rearranged to look like the one on the right. Every such simple cycle is called either a $\Sigma$-*cycle* or $\Sigma$-*confluence* depending on the shape of the path it induces in $\mathbb{A}$.

***Cycles and confluences*** As before, let us assume $\Sigma$ of the form (‡). The **$\Sigma$-substructure** of a $\sigma$-structure $\mathbb{A}$, noted $\mathbb{A}|_\Sigma$, is the substructure induced by the restriction to the relations of $\Sigma$. In a similar way, $\mathbb{A}|_{\sigma\setminus\Sigma}$ denotes the substructure restricted to the relations which are *not* in $\Sigma$. A **$\Sigma$-cycle** of a $\sigma$-structure $\mathbb{A}$ is a substructure $\mathbb{B} \subseteq \mathbb{A}$ consisting of a cycle on the relations of $\Sigma$. That is, $\mathbb{B}$ is a connected substructure of $\mathbb{A}$, it contains only $\Sigma$-edges, and every element of $\mathbb{B}$ has in-degree and out-degree equal to 1. For $a \in A$, a **$\Sigma$-confluence rooted at** $a$ of $\mathbb{A}$ is the union of two paths of $\Sigma$-edges

$$\begin{array}{c} a_1 \xrightarrow{R_1} \cdots \xrightarrow{R_n} a_{n+1} \quad \text{and} \\ a'_1 \xrightarrow{R'_1} \cdots \xrightarrow{R'_m} a'_{m+1} \end{array} \qquad (\star)$$

so that $a = a_1 = a'_1$, $a_{n+1} = a'_{m+1}$ and $(a_n, R_n, a_{n+1}) \neq (a'_m, R'_m, a'_{m+1})$. See Figure 4 for an example.

***$\Sigma$-reachability order*** For a given structure $\mathbb{C}$, we define the partial order relation $\preceq_\mathbb{C}$, where $a \preceq_\mathbb{C} b$ iff there is a (possibly empty) directed path from $a$ to $b$ in $\mathbb{C}|_\Sigma$. In particular $a \preceq_\mathbb{C} a$ for every $a \in C$. If $a \preceq_\mathbb{C} b$ and $b \preceq_\mathbb{C} a$ we write $a \equiv_\mathbb{C} b$, which means that $a$, $b$ belong to the same strongly connected component (SCC) in $\mathbb{C}|_\Sigma$. If $a \preceq_\mathbb{C} b$ but $b \not\preceq_\mathbb{C} a$, we write $a \prec_\mathbb{C} b$. The **$\Sigma$-rank** of an element $c \in C$ is the maximum number $n \geq 0$ so that there are $c_0, \ldots, c_n$ verifying $c_0 \prec_\mathbb{C} c_1 \prec_\mathbb{C} \cdots \prec_\mathbb{C} c_n = c$. The $\Sigma$-rank of a structure $\mathbb{C}$ is the maximum among the $\Sigma$-ranks of its elements.

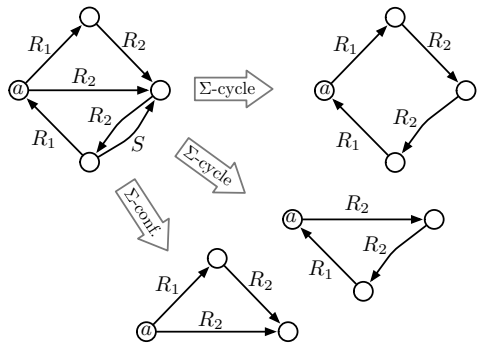



**Figure 4.** A $\sigma$-structure, its two $\Sigma$-cycles, and a $\Sigma$-confluence rooted at $a$; for $\Sigma = \{R_1[1{\mapsto}2], R_2[1{\mapsto}2]\}$.

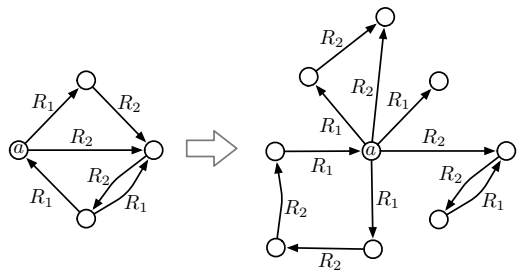


**Figure 5.** A cactus decomposition of a structure.

For a given SCC $X$ of $\mathbb{C}|_\Sigma$, we say that $X$ is a **least** SCC if all its elements are of $\Sigma$-rank 0.

The **substructure generated by** $a$ of $\mathbb{A}$, noted $\mathbb{A}{\restriction}a$, is the substructure of $\mathbb{A}$ induced by $\{b \in A \mid a \preceq_{\mathbb{C}} b\}$. The **$\Sigma$-substructure generated by** $a$ of $\mathbb{A}$, noted $\mathbb{A}{\restriction}_\Sigma a$, is $(\mathbb{A}{\restriction}a)_\Sigma$ (or, equivalently, $(\mathbb{A}|_\Sigma){\restriction}a$).

***Cactus decomposition*** We are now in conditions to show the main result of this section, namely, that for every structure $\mathbb{A}$ and $a \in A$, the *chase*$_\Sigma$-preimage of $\mathbb{A}{\restriction}_\Sigma a$ contains a structure of tree-width $\leq 2$.

**Lemma 6.1.** *For every $\sigma$-structure $\mathbb{A}$, set of FDs $\Sigma$, and element $a \in A$ there exists a structure $\mathbb{B}$ so that tw$(\mathbb{B}) \leq 2$, and $\mathbb{B} \Rightarrow^*_\Sigma \mathbb{A}{\restriction}_\Sigma a$.*

To prove this, we show how to decompose $\Sigma$-substructures into a equivalent structures (modulo $\Rightarrow^*_\Sigma$) whose underlying undirected graph is a *cactus* (*i.e.*, whose every edge belongs to at most one simple cycle), as in Figure 5. Since cacti have tree-width $\leq 2$, the lemma follows.

*Proof of Lemma 6.1.* Let $\mathbb{A}$ be a $\sigma$-structure, and $a \in A$. Let $\mathbb{B} = \mathbb{A}{\restriction}_\Sigma a$. Note that every simple cycle in the underlying undirected graph of $\mathbb{B}$ induces a

(a) $\Sigma$-cycle; or

(b) the presence of $b \xrightarrow{R} c \xleftarrow{R'} b'$ in $\mathbb{B}$, for some relations $R, R'$ and elements $b, b', c$ so that $b \neq b'$ or $R \neq R'$.

In the case (a), suppose $\mathbb{B}$ has a $\Sigma$-cycle $\mathbb{B}'$ consisting of $a_1 \xrightarrow{R_1} \cdots \xrightarrow{R_n} a_{n+1} = a_1$. Let $\hat{\mathbb{B}}$ be the result of removing the edge $a_n \xrightarrow{R_n} a_{n+1}$ from $\mathbb{B}$, and let $\hat{\mathbb{B}}'$ be the result of renaming every element $a_i$ of $\mathbb{B}'$ with a fresh element $b_i$, for all $2 \leq i \leq n$ (*i.e.*, so that $\hat{\mathbb{B}}' \cong \mathbb{B}'$ and the domain of $\hat{\mathbb{B}}'$ is $\{a_1, b_2, \ldots, b_n\}$). Note that

(i) $\hat{\mathbb{B}}$ and $\hat{\mathbb{B}}'$ have only $a_1$ in common,
(ii) $\hat{\mathbb{B}} \cup \hat{\mathbb{B}}' \Rightarrow^*_\Sigma \mathbb{B}$,
(iii) $(\hat{\mathbb{B}} \cup \hat{\mathbb{B}}'){\restriction}_\Sigma a = \hat{\mathbb{B}} \cup \hat{\mathbb{B}}'$.

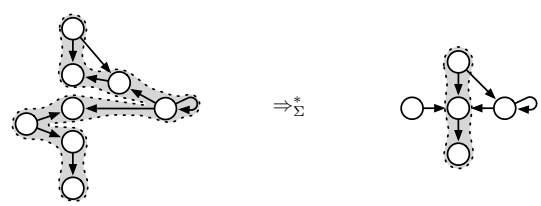
In the second case (b), this implies that there is a $\Sigma$-confluence rooted at $a$ with some paths as in ($\star$) so that $a_n = b$, $a'_m = b'$, $a_{n+1} = a'_{m+1} = c$, $R_n = R$ and $R'_m = R'$. We can assume, without any loss of generality, that $(a'_i, R'_i, a'_{i+1}) \neq (b', R', c)$ for all $i$. Let $\mathbb{B}'$ be such $\Sigma$-confluence. Let $\hat{\mathbb{B}}$ be the result of removing the edge $b \xrightarrow{R} c$ from $\mathbb{B}$; and let $\hat{\mathbb{B}}'$ be the result of renaming every element except $a$ with a fresh element. Note that the properties (i)–(iii) above continue to hold also in this case.

It is easy to see that by applying iteratively these two operations eventually we obtain a structure whose underlying undirected graph is a cactus. □

In the light of the lemma above, we call such structure $\mathbb{B}$ the **cactus decomposition** of $\mathbb{A}, a$.

## 7. Complex paths

We define a type of paths between vertices of a structure that we call *complex paths*. A complex path corresponds, intuitively, to the path in a structure $\mathbb{A}$ induced by a directed path in *chase*$_\Sigma(\mathbb{A})$. For example, in the figure below, the directed path on the right becomes the *complex* path on the left.



These paths are of prime importance to our result. In later developments we show that if a structure $\mathbb{A}$ contains elements connected in a certain way (depending on a structure $\mathbb{C}$) through complex paths, this implies that *chase*$_\Sigma(\hat{\mathbb{A}})$ contains $\mathbb{C}$ as substructure—where $\hat{\mathbb{A}}$ is $\mathbb{A}$ extended with the cactus decompositions as defined in Section 6. Concretely, we give an MSO-definable property $\varphi$ so that

- if $\mathbb{A} \models \varphi$, then *core*(*chase*$_\Sigma(\mathbb{A}')$) $\cong \mathbb{C}$ for some $\mathbb{A}'$ so that *tw*$(\mathbb{A}')$ = *tw*$(\mathbb{A})$, and
- if *core*(*chase*$_\Sigma(\mathbb{A})$) $\cong \mathbb{C}$, then $\mathbb{A} \models \varphi$.

Hence, by testing whether the property has a tree-width $k$ model (which is decidable for MSO [20]) we obtain a decision procedure for the semantic tree-width problem.

For defining complex paths, we also need to define what we will call *moving* and *static* paths.

A **moving path** from $a$ to $a'$ of $\mathbb{A}$ is simply an edge $a \xrightarrow{R} a'$ of $\mathbb{A}$, for some $R$ in $\Sigma$. A **static path** of $\mathbb{A}$ from $a$ to $a'$ is a path of the form

- $(a \xleftarrow{R} b)(b \xrightarrow{R} a')$, for $b \xrightarrow{R} a$, $b \xrightarrow{R} a'$ in $\mathbb{A}$; or
- $(a \xleftarrow{R} b)\ p\ (b' \xrightarrow{R} a')$ for $b \xrightarrow{R} a$, $b' \xrightarrow{R} a'$ in $\mathbb{A}$, and $p$ a static path from $b$ to $b'$; or
- $p\ p'$ for $p$ a static path from $a$ to $b$, and $p'$ a static path from $b$ to $a'$, for some $b$,

where $R$ is in $\Sigma$. A **complex path** from $a$ to $a'$ is either a moving or static path from $a$ to $a'$, or the composition of a complex path from

$a$ to $b$ with a complex path from $b$ to $a'$ for some $b$. The **moving length** of a complex path is the number of moving paths it contains.

**Lemma 7.1.** *Given $\mathbb{A} \Rightarrow^*_\Sigma \mathbb{A}'$, the provenance homomorphism $h : \mathbb{A} \to \mathbb{A}'$, and $a, a' \in A$, the following statements are equivalent:*

*i. there is a complex path from $a$ to $a'$ in $\mathbb{A}$ of moving length $m$;*
*ii. there is a complex path from $h(a)$ to $h(a')$ in $\mathbb{A}'$ of moving length $m$.*

*Proof.* The (i) $\Rightarrow$ (ii) part is straightforward since the homomorphic image of a complex path is a complex path of equal moving length. For the (ii) $\Rightarrow$ (i) part, it is not hard to prove the statement for $\mathbb{A} \Rightarrow_\Sigma \mathbb{A}'$. By iterating the argument we obtain it for $\mathbb{A} \Rightarrow^*_\Sigma \mathbb{A}'$. □

Note that the set of complex paths of a structure $\mathbb{A}$ is not a regular language but a context-free one. Since our ultimate objective is to encode the existence of these paths into MSO, this supposes a problem. However, we will show that for every structure $\mathbb{A}$ we can *expand* it to some superstructure $\mathbb{A} \cup \mathbb{A}'$ so that $\mathit{chase}_\Sigma(\mathbb{A} \cup \mathbb{A}') = \mathit{chase}_\Sigma(\mathbb{A})$, $\mathit{tw}(\mathbb{A} \cup \mathbb{A}') = \mathit{tw}(\mathbb{A})$, and the same statement as in Lemma 7.1 holds for some simpler "not so complex" paths, which in particular are regular.

***Expansion*** Given $\sigma$-structures $\mathbb{A}, \mathbb{C}$ and a homomorphism $h : \mathbb{A} \to \mathbb{C}$ we define the **expansion** of $\mathbb{A}$, as the superstructure of $\mathbb{A}$ resulting from adding, for each $a \in A$, a disjoint copy of the cactus decomposition of $\mathbb{C}, h(a)$ from our previous Section 6, identifying the cactus element $h(a)$ with $a$ (resulting in the union of two structures intersecting in one vertex). Note that the expansion of $\mathbb{A}$ has the same tree-width as $\mathbb{A}$ (assuming that $\mathit{tw}(\mathbb{A}) \geq 2$).

***Regular Complex Paths*** Let $h : \mathbb{A} \to \mathbb{C}$. A **regular complex path** of $\mathbb{C}$ is just like a complex path but now a *static path* is redefined as a **regular static path** from $a$ to $a'$, which is a path of the form

- an empty path, starting and ending in the same node; or
- $(a \xleftarrow{R} b)\ p\ (b \xrightarrow{R} a)$, for $b \xrightarrow{R} a$, $b' \xrightarrow{R} a'$ in $\mathbb{C}|_\Sigma$, $p$ a regular static path from $b$ to $b$, and $b \prec_\mathbb{C} a$; or
- $(a_1 \overset{R_1}{\leftrightsquigarrow}_1 b_1)\ p_1 \cdots p_{n-1}\ (a_n \overset{R_n}{\leftrightsquigarrow}_n b_n)$, where, for every $i$, $a_i \overset{R_i}{\leftrightsquigarrow}_i b_i$ is either an edge $a_i \xrightarrow{R_i} b_i$ or $a_i \xleftarrow{R_i} b_i$ from $\mathbb{C}|_\Sigma$, $p_i$ is a regular static path from $b_i$ to $a_{i+1}$, and $a_i \equiv_\mathbb{C} b_i \equiv_\mathbb{C} a_1$, $a = a_1 = a' = a_n$.
- $p\,p'$ for $p$ a regular static path from $a$ to $b$, and $p'$ a regular static path from $b$ to $a'$, for some $b$.

Given $h : \mathbb{A} \to \mathbb{C}$, an $h$**-regular complex path** of $\mathbb{A}$ is a path $p$ so that $h(p)$ is a regular complex path of $\mathbb{C}$. In this definition, note that the rule $(a \xleftarrow{R} b)\ p\ (b \xrightarrow{R} a)$ can be nested only a bounded amount of times (bounded in the size of $\mathbb{C}$). This is, in fact, a generalization of complex paths.

**Lemma 7.2.** *For $\mathit{core}(\mathit{chase}_\Sigma(\mathbb{A})) = \mathbb{C}$ and $h : \mathbb{A} \to \mathbb{C}$, complex paths of $\mathbb{C}$ are in particular regular complex paths; and complex paths of $\mathbb{A}$ are in particular $h$-regular complex paths.*

*Proof.* We show this by induction. Note that, since $\mathbb{C}$ is a chase, For any static path of $\mathbb{C}$ with the form $(a \xleftarrow{R} b)p(b \xrightarrow{R} a)$ so that $b \prec_\mathbb{C} a$, we can apply the inductive hypothesis on $p$, obtaining that $p$ is a regular static path, and by one of the rules of regular static paths we obtain that $(a \xleftarrow{R} b)p(b \xrightarrow{R} a)$ is a regular static path. For a path $(a \xleftarrow{R} b)(b \xrightarrow{R} a)$ with $b \prec_\mathbb{C} a$, the reasoning is the same. On the other hand, for a static path of the form $(a \xleftarrow{R} b)p(b \xrightarrow{R} a)$ with $b \equiv_\mathbb{C} a$, we can apply the inductive hypothesis on $p$, and we have that both $(a \xleftarrow{R} b)$ and $(b \xrightarrow{R} a)$ are regular static paths. Thus, by composition $(a \xleftarrow{R} b)p(b \xrightarrow{R} a)$ is a regular static path. Moving paths are the same kind of objects, and for general complex paths we simply apply the inductive hypothesis on the composition.

Any complex path $p$ of $\mathbb{A}$ is mapped through $h$ to a complex path $h(p)$ of $\mathbb{C}$ by Lemma 7.1. Applying the previous part of this lemma we have that $h(p)$ is also a regular complex path, and thus that $p$ is an $h$-regular complex path. □

The main difference implied by the new definition is that regular complex paths of $\mathbb{A}$ form now a *regular* language. The size required by an NFA to describe this language depends on what we call the **tree unravelling** of $\mathbb{C}$. The tree unravelling of $\mathbb{C}$ is the result of applying recursively the following rule until it can be no longer applied. Given an SCC $X$ of $\mathbb{C}|_\Sigma$ and two distinct edges $a \xrightarrow{R} b$, $a' \xrightarrow{R'} b'$ of $\mathbb{C}|_\Sigma$, so that $a, a' \in X$ and $b, b' \notin X$:

(a) remove $a' \xrightarrow{R'} b'$ from $\mathbb{C}$;
(b) add a fresh copy of $(\mathbb{C}|_\Sigma)|_{\downarrow X}$ with $\downarrow X = \{c \in C \mid c \preceq_\mathbb{C} a'\}$; and
(c) add an edge $a'' \xrightarrow{R'} b'$, where $a''$ is the fresh copy of $a'$ just inserted.

Note that the tree unravelling $\mathbb{C}'$ of $\mathbb{C}$ contains only $\Sigma$-edges, and that there is a canonical homomorphism $h_{\mathit{tree}} : \mathbb{C}' \to \mathbb{C}$ associating a an element of $\mathbb{C}'$ with the element that originated it. Figure 6 contains an example.

**Lemma 7.3.** *There is a regular language $\mathcal{L}_\mathbb{C}$ over the alphabet of edges of $\mathbb{C}$, consisting in the set of all regular complex paths of $\mathbb{C}$. Further, an NFA recognizing $\mathcal{L}_\mathbb{C}$ can be built in polynomial time in the size of the tree unravelling of $\mathbb{C}$.*

*Proof.* The NFA accepting $\mathcal{L}_\mathbb{C}$ works over the alphabet $\{a \xrightarrow{R} b, b \xleftarrow{R} a \mid a \xrightarrow{R} b \text{ in } \mathbb{C}\}$. It is a polynomial union of languages, each of these being basically described by the tree unravelling $\mathbb{C}'$ of $\mathbb{C}$ and the canonical homomorphism $h_{\mathit{tree}} : \mathbb{C}' \to \mathbb{C}$. We build one automaton $\mathcal{A}_a$ for each element $a$ of $\mathbb{C}'$. The language $L(\mathcal{A}_a)$ of $\mathcal{A}_a$ consists in all regular static paths of $\mathbb{C}$ beginning and ending in $h_{\mathit{tree}}(a)$. The automaton $\mathcal{A}_a$ for element $a$ is built as having the elements $X = \{a' \mid a' \preceq_{\mathbb{C}'} a\}$ of $\mathbb{C}'$ as state space; $a$ as initial and final state; and a transition $(a, h_{\mathit{tree}}(a) \xrightarrow{R} h_{\mathit{tree}}(b), b)$ and $(b, h_{\mathit{tree}}(b) \xleftarrow{R} h_{\mathit{tree}}(a), a)$ for every edge $a \xrightarrow{R} b$ in $\mathbb{C}'|_X$. It follows that one can build an NFA for $\mathcal{L}_\mathbb{C}$ in polynomial time in $\{\mathcal{A}_a \mid a \text{ in } \mathbb{C}'\}$. □

Note that the tree unravelling of $\mathbb{C}$ can be exponential, and in this case the exponential size description of $\mathcal{L}_\mathbb{C}$ seems unavoidable, since the description of regular static paths for structures such as the one of Figure 6 is related to the language $\mathcal{L}_{\mathbb{C}_n} = \{w\,w^r \mid w \in A^{\leq n}, \text{ and } w^r \text{ is the reverse of } w\}$ for some alphabet $A$. Notice also that if the $\Sigma$-rank of $\mathbb{C}$ is bounded by a constant, the tree unravelling of $\mathbb{C}$ is polynomial, and so is the NFA describing $\mathcal{L}_\mathbb{C}$.

The interest of these paths is that they allow us to have a result in the same spirit as Lemma 7.1.

Given structures $\mathbb{A}, \mathbb{C}$ consider the following conditions:

1. There is an onto homomorphism $h : \mathbb{A} \to \mathbb{C}$ and $\mathbb{C}$ is a core and a chase (*i.e.*, $\mathit{core}(\mathbb{C}) = \mathbb{C}$, $\mathit{chase}_\Sigma(\mathbb{C}) = \mathbb{C}$);
2. $h(\mathbb{A})|_{\sigma \setminus \Sigma} \cong \mathbb{C}|_{\sigma \setminus \Sigma}$;

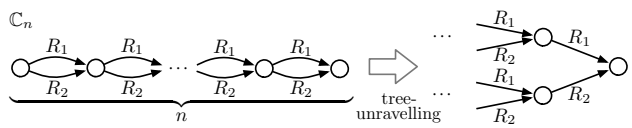



**Figure 6.** Example of a structures $\mathbb{C}_n$, $\Sigma = \{R_1[1{\mapsto}2], R_2[1{\mapsto}2]\}$, with its tree unravelling (a complete, height-$n$ binary tree). For the class of structures $\{\mathbb{C}_n\}_n$ we have that the NFA description of $\mathcal{L}_{\mathbb{C}_n}$ takes exponential space in $n$.

3. for $X_1, \ldots, X_n$ the least SCCs of $\mathbb{C}|_\Sigma$, there are $a_i, c_i$ so that $c_i \in X_i$ and $h(a_i) = c_i$ for every $i$ where the following holds: For every $a \in A$ so that $c_i \prec_{\mathbb{C}} h(a)$ there is an $h$-regular complex path from $a_i$ to $a$ in $\mathbb{A}$.

**Lemma 7.4.** *For $\mathbb{A}, \mathbb{C}$ verifying the conditions 1–3 we have that core(chase$_\Sigma(\hat{\mathbb{A}})) \cong \mathbb{C}$, where $\hat{\mathbb{A}}$ is the expansion of $\mathbb{A}$.*

*Proof.* It is not hard to see that every time we apply one step of $\Rightarrow_\Sigma$ we maintain the invariant of points 1–3. That is, if $\hat{\mathbb{A}} \Rightarrow_\Sigma \mathbb{A}'$ by a provenance homomorphism $f : \hat{\mathbb{A}} \to \mathbb{A}'$, there must be $a \xrightarrow{R} b$ and $a \xrightarrow{R} b'$ in $\hat{\mathbb{A}}$ so that $b, b'$ are identified in $\mathbb{A}'$ (that is, $f(b) = b'$ and the identity otherwise). Then it must be that $h(b) = h(b')$, as otherwise we would have $h(a) \xrightarrow{R} h(b)$ and $h(a) \xrightarrow{R} h(b')$ in $\mathbb{C}$ where $h(b) \neq h(b')$ which would mean that $\mathbb{C}$ is not a chase structure. Thus, $h$ is still a homomorphism from $\mathbb{A}'$ to $\mathbb{C}$, where $h(\mathbb{A}')|_{\sigma\setminus\Sigma} = \mathbb{C}|_{\sigma\setminus\Sigma}$. Finally, every $h$-regular complex path $p$ present in $\hat{\mathbb{A}}$ appears also in $\mathbb{A}'$ as $f(p)$.

Using the properties of the cactus decomposition of the previous section (Lemma 6.1), one can show by induction that for any $h$-regular complex path departing from $a_i$ leading to some $a$ in $\hat{\mathbb{A}}$, and the provenance homomorphism $f : \hat{\mathbb{A}} \to$ *chase*$_\Sigma(\hat{\mathbb{A}})$ one obtains: *chase*$_\Sigma(\hat{\mathbb{A}})\restriction_\Sigma f(a_i) \cong \mathbb{C}\restriction_\Sigma c_i$ and $f(a) = h(a)$ is in *chase*$_\Sigma(\hat{\mathbb{A}})\restriction_\Sigma f(a_i)$. In plain words, after some applications of $\Rightarrow_\Sigma$ we obtain precisely the structure $\mathbb{C}\restriction_\Sigma c_i$, plus perhaps something else that can be homomorphically mapped to $\mathbb{C}$.

Repeating this argument for each complex path of 1–3, we obtain that $\bigcup_i$ *chase*$_\Sigma(\hat{\mathbb{A}})\restriction_\Sigma f(a_i) = \mathbb{C}|_\Sigma$. This, together with point 2, implies that $\bigcup_i$ *chase*$_\Sigma(\hat{\mathbb{A}})\restriction f(a_i) = \mathbb{C}$, and that therefore there is a homomorphism $\mathbb{C} \to$ *chase*$_\Sigma(\hat{\mathbb{A}})$. Since there is also a homomorphism *chase*$_\Sigma(\hat{\mathbb{A}}) \to \mathbb{C}$ by the $\Rightarrow_\Sigma$-invariance of 1–3, and since $\mathbb{C}$ is a chase and core structure, we have that *core*(*chase*$_\Sigma(\hat{\mathbb{A}})) = \mathbb{C}$. □

It is not hard to see that the converse of the previous property holds without the need of expanded structures, as in the following lemma.

**Lemma 7.5.** *If core(chase$_\Sigma(\mathbb{A})) = \mathbb{C}$, conditions 1–3 hold.*

*Proof.* We show that each of the conditions is verified.

1. For the provenance (onto) homomorphism $f : \mathbb{A} \to$ *chase*$_\Sigma(\mathbb{A})$ and the image-identity (onto as well) homomorphism $g :$ *chase*$_\Sigma(\mathbb{A}) \to \mathbb{C}$, it follows that $h = f \circ g$ is an onto homomorphism $\mathbb{A} \to \mathbb{C}$. Further, $\mathbb{C}$ is a core and a chase. The fact that it is a core is straightforward, the fact that it is a chase is a consequence of Lemma 2.2.
2. On the one hand, it is clear that $h(\mathbb{A})|_{\sigma\setminus\Sigma} \subseteq \mathbb{C}_{\sigma\setminus\Sigma}$. On the other hand, for every $\mathbb{A}_1 \Rightarrow_\Sigma \mathbb{A}_2$ with its provenance homomorphism $f' : \mathbb{A}_1 \to \mathbb{A}_2$, it is easy to see that for every $a_2 \xrightarrow{S} a_2'$ in $\mathbb{A}_2$ with $S \notin \Sigma$, there are $a_1, a_1'$ so that $h(a_1) = a_2$, $h(a_1') = a_2'$ and $a_1 \xrightarrow{S} a_1'$ in $\mathbb{A}_1$. By induction we can show the same for $\mathbb{A}_1 \Rightarrow_\Sigma^* \mathbb{A}_2$. In a similar way, for a homomorphism $f' : \mathbb{A}' \to$ *core*$(\mathbb{A}')$ we have that $c \xrightarrow{S} c'$ in *core*$(\mathbb{A}')$ implies $a \xrightarrow{S} a'$ in $\mathbb{A}'$ for some $a, a'$ so that $f'(a) = c$, $f'(a') = c'$ (remember that the core is isomorphic to a substructure). Putting these two properties together, we have that $c \xrightarrow{S} c'$ in $\mathbb{C}$, for $S \notin \Sigma$, implies $a \xrightarrow{S} a'$ in $\mathbb{A}$ for $a, a'$ so that $h(a) = c$, $h(a') = c'$. Thus, $h(\mathbb{A})|_{\sigma\setminus\Sigma} = \mathbb{C}_{\sigma\setminus\Sigma}$.
3. Notice that for every $c_i \preceq_{\mathbb{C}} h(a)$ there is a complex path (composed of only of moving paths) from $c_i$ to $h(a)$. By Lemma 7.1, there is a complex path from $a_i$ to $a$. By Lemma 7.2, complex paths of $\mathbb{A}$ are in particular $h$-regular complex paths, thus the third condition follows. □

We therefore have, as a consequence of Lemmas 7.5 and 7.4, that 1–3 are both sufficient and necessary conditions for $\mathbb{A}$ being a witness of a positive outcome of the $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ problem on input $\mathbb{C}, \Sigma$.

**Lemma 7.6.** *The $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ problem holds for a structure $\mathbb{C}$ and set of FDs $\Sigma$ iff there is a tree-width $k$ structure $\mathbb{A}$ verifying conditions 1–3.*

Since the existence of homomorphisms and the existence of paths from a given regular language are all MSO-definable, 1–3 can be expressed in MSO.

**Lemma 7.7.** *Given $\Sigma, \mathbb{C}$, the set $\{\mathbb{A} \mid \mathbb{A}, \mathbb{C}$ verify 1–3$\}$ is MSO-definable. A formula recognizing it can be computed.*

*Proof.* The first two conditions are very easy to encode by guessing the homomorphism by partitioning the domain with monadic predicates $\{X_c\}_{c\in C}$, where $a \in X_c$ codes $h(a) = c$. For the third condition, note that once $h$ and $\mathbb{C}$ is fixed, the $h$-regular complex paths become a regular language depending on $\mathbb{C}$ and the monadic predicates $\{X_c\}_{c\in C}$. One can then test the existence of an $h$-regular complex path from $x$ to $y$ with an MSO formula using Lemma 7.3. □

We can therefore conclude that the Semantic Tree-width problem is decidable.

**Theorem 7.8.** *The $\mathrm{STW}_k$ problem is decidable, for every $k$.*

*Proof.* By Lemma 3.6 we can reduce $\mathrm{STW}_k$ to the $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ problem. Given an input $\mathbb{C}$ of the latter, by Lemma 7.7 there is an MSO formula $\varphi_{\mathbb{C}}$ whose models are $\{\mathbb{A} \mid \mathbb{A}, \mathbb{C}$ verify 1–3$\}$. Since MSO is decidable on $\mathrm{TW}_{\leq k}$ [20], we can decide whether $\{\mathbb{A} \mid \mathbb{A}, \mathbb{C}$ verify 1–3$\} \cap \mathrm{TW}_{\leq k}$ is empty, and thus, by Lemma 7.6, we can decide whether the $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ problem holds for $\mathbb{C}$. □

## 8. Complexity

In this section we explain how to build a tree-walking automaton (TWA) of exponential size in a structure $\mathbb{C}$ and set of FDs $\Sigma$, so that the automaton is non-empty if, and only if, the $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ problem yields a positive answer on $\mathbb{C}, \Sigma$. Since the emptiness problem for TWA is decidable in exponential time [10, 21], and there is an NP reduction from $\mathrm{STW}_k$ to $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$, we obtain that the semantic tree-width problem is in 2ExpTime.

Unfortunately, we don't know how to code condition 3 in TWA without adding an extra exponential blowup, as it would seem to require some type of alternation. To sort out this problem, we must first remark that conditions 1–3 can be weakened while preserving a similar result to that of Lemma 7.4. Here, condition 3 is replaced with the following:

$3'$. For $X_1, \ldots, X_n$ the least SCCs of $\mathbb{C}|_\Sigma$, there are $a_i, c_i$ so that $c_i \in X_i$ and $h(a_i) = c_i$ for every $i$ where the following holds:

- For every $c \in C$ there is some $a \in h^{-1}(c)$ so that for every $c_i \preceq_{\mathbb{C}} c$ there is an $h$-regular complex path from $a_i$ to $a$ in $\mathbb{A}$.
- For every $c \xrightarrow{S} c'$ in $\mathbb{C}$ with $S \in \sigma \setminus \Sigma$ there is $a \xrightarrow{S} a'$ in $\mathbb{A}$ so that $h(a) = c$, $h(a') = c'$ and for every $c_i \preceq_{\mathbb{C}} c$ [resp. $c_i \preceq_{\mathbb{C}} c'$] there is an $h$-regular complex path from $a_i$ to $a$ [resp. from $a_i$ to $a'$] in $\mathbb{A}$.

Notice that the condition above only asks for the existence of a *polynomial* number of paths (although the paths involved have an unbounded number of vertices). It is not hard to see that these conditions are still sufficient for the positive solution of a $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ instance.

**Lemma 8.1.** *For every $\mathbb{A}, \mathbb{C}$ verifying the conditions $1, 2, 3'$ we have that core(chase$_\Sigma(\mathbb{B})$) $\cong \mathbb{C}$, for some $\mathbb{B}$ with tw($\mathbb{B}$) $\leq$ tw($\mathbb{A}$).*

*Proof.* The proof is just as the one of Lemma 7.4, but now we consider the substructure $\mathbb{A}'$ of $\mathbb{A}$ obtained by taking only the elements and edges from the (polynomially many) witness vertices described in $3'$ to the $a_i$'s. Applying Lemma 7.4 to $\mathbb{A}'$ we obtain that $\mathbb{C}$ is isomorphic to *core*(*chase*$_\Sigma(\hat{\mathbb{A}}')$), where $\hat{\mathbb{A}}'$ is the expansion of $\mathbb{A}'$. □

The TWA verifies conditions $1, 2, 3'$ on a width-$(k-1)$ tree decomposition of the structure $\mathbb{A}$, which we assume to be binary for simplicity (and without any loss of generality).

The alphabet of the tree consists in pairs $(\mathbb{S}, f)$ where, $\mathbb{S}$ is a $\sigma$-structure of at most $k$ elements $S$, with names taken from the set $S \subseteq \{1, \ldots, 2k\} \cup C$ as well as a mapping $f : S \to C$ so that $f$ restricted to $C$ is the identity (remember that $C$ is the domain of $\mathbb{C}$). The mapping $f$ will represent the homomorphism to the structure $\mathbb{C}$, and the $C$ elements will be special representatives for each element of $\mathbb{C}$. Since $k$ and $\sigma$ are fixed, the alphabet is of polynomial size. Between parent and child nodes, the elements of the substructure in the alphabet that they share represent which ones are the elements in common. An example is given in Figure 7.

A **tree walking automaton** (TWA) is a sequential device that can recognize properties of paths of labeled trees. The automaton is located at a node of a tree, it can perform tests of the form "is this node a leaf / root / right-child / left-child?", or "is the current label $a$?". Based on the result of these tests it can accept or move to a parent or a child with a given state. More formally, a TWA on a binary finite tree over an alphabet $A$ is given as a tuple $\mathcal{A} = \langle Q, A, q_0, F, \delta \rangle$, where $Q$ is the state space, $q_0 \in Q$ is the initial state, $F \subseteq Q$ the set of final states, and $\delta \subseteq Q \times \textit{Types} \times A \times Q \times \{\text{parent, left child, right child}\}$ the set of transitions. Transitions of the form $(q, t, a, p, c)$ are interpreted as: "if the current state is $q$, the type of the current node is $t$, and its label is $a$, continue the computation in node $c$ with state $p$", where the possible types *Types* indicate whether the current node has a parent, a left child or a right child. An accepting run corresponds to a traversal in the tree, which starts with $q_0$ and ends with a final state from $F$. Notice that, in particular, TWA can make DFS traversals of the tree. We refer the reader to [6, 17] for a formal definition and more details on this model.

**Lemma 8.2.** *There is a TWA $\mathcal{A}$ so that $\mathcal{A}$ is non-empty iff there exists a structure $\mathbb{A}$ and a homomorphism $h : \mathbb{A} \to \mathbb{C}$ verifying conditions $1, 2, 3'$. Further, $\mathcal{A}$ can be built in polynomial time in the NFA description of $\mathcal{L}_\mathbb{C}$.*

*Proof.* The TWA $\mathcal{A}$ runs on the tree-width-$k$ decomposition of $\mathbb{A}$ labeled with the alleged homomorphism as in Figure 7. Let

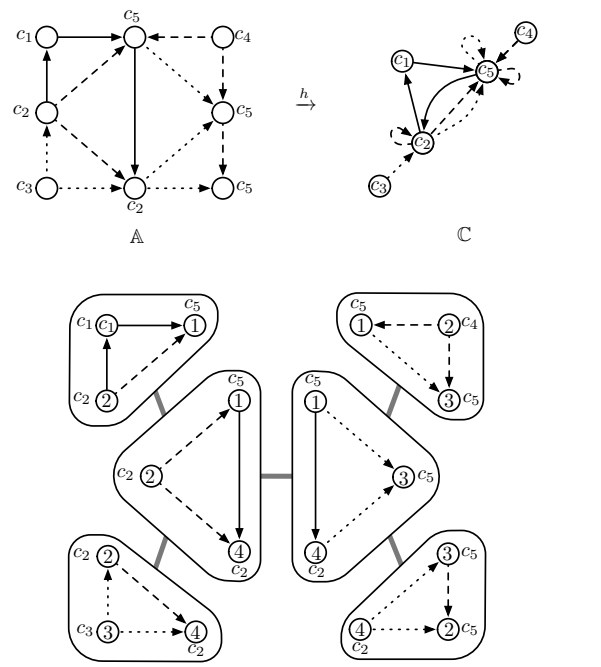



**Figure 7.** A structure $\mathbb{A}$—where different shapes of arrow correspond to different relation symbols—, together with a homomorphism $h$ to a structure $\mathbb{C}$. The homomorphism $\mathbb{A} \to \mathbb{C}$ is represented by a label attached to the nodes of $\mathbb{A}$: For example, for $x_1, x_2, x_3$ the top elements of $\mathbb{A}$, we have $h(x_1) = c_1$, $h(x_2) = c_5$, $h(x_3) = c_4$. On the bottom, a tree-width 2 decomposition of $\mathbb{A}$ in our chosen representation. Note that each vertex contains a substructure of $\mathbb{A}$, and the mapping is depicted via the labels attached to the substructure. Thus, the upper-left vertex is $(\mathbb{S}, f)$, where $S = \{1, 2, c_1\}$ and $f(1) = c_5$, $f(2) = c_2$, $f(c_1) = c_1$.

$c_1, \ldots, c_n$ be elements from the $n$ least SCC of $\mathbb{C}|_\Sigma$ as described in $1, 2, 3'$. We now list the properties that our automaton $\mathcal{A}$ must verify.

(a) There is a homomorphism $\mathbb{A} \to \mathbb{C}$. On the one hand, $\mathcal{A}$ verifies that the mapping is consistent: for every two neighboring nodes of the tree with labels $(\mathbb{S}_1, f_1), (\mathbb{S}_2, f_2)$ and for every two vertices of its structures $v_1 \in S_1, v_2 \in S_2$ we have that if $v_1 = v_2 \in \{1, \ldots, 2k\}$ then $f_1(v_1) = f_2(v_2)$. Besides, $\mathcal{A}$ verifies that every label $(\mathbb{S}, f)$ in the tree is so that $f$ is a homomorphism from $\mathbb{S}$ to $\mathbb{C}$. These two verifications imply that the functions in the vertices can be merged to form a homomorphism $h : \mathbb{A} \to \mathbb{C}$ from the original structure to $\mathbb{C}$. Since the alphabet is polynomial, $\mathcal{A}$ can perform a tree traversal making sure that these conditions are met through a polynomial number of transitions.

(b) For every edge $a \xrightarrow{S} b$ in $\mathbb{C}$ with $S$ not in $\Sigma$, there exists some $a' \xrightarrow{S} b'$ in $\mathbb{A}$ so that the homomorphism above sends $a$ to $a'$ and $b$ to $b'$. This is translated as $\mathcal{A}$ guessing and finding the pair of elements inside a label of the tree for each such edge, which amounts to a polynomial number of transitions.

(c) The $C$ elements are special representatives. For every $c \in C$: There is a node with a label $(\mathbb{S}, f)$ containing $c$, so that $f(c) = c$, and the substructure of the tree that uses the name $c$ forms a connected component (in other words, $c$ is not "reused", as other names from $\{1, \ldots, 2k\}$ may be). Thus, for every $c \in C$ there is an element $a_c$ of $\mathbb{A}$ that represents $c$ given by the decomposition, where $h(a_c) = c$.

(d) For every $c_i \preceq_{\mathbb{C}} c$ there is a $h$-regular complex path from the element $a_{c_i}$ representing $c_i$ to the element $a_c$ representing $c$ in $\mathbb{A}$. Notice that this amounts to testing the existence of a path in the graph encoded in the tree, whose homomorphic image is in $\mathcal{L}_{\mathbb{C}}$ as described in Lemma 7.3, which is easy to express using a TWA. Also, note that there are only a polynomial number of tests of this kind to be performed.

The automaton $\mathcal{A}$ verifying this can be built in polynomial time in the NFA recognizing $\mathcal{L}_{\mathbb{C}}$ which can be built in exponential time due to Lemma 7.3. It is not hard to see that it enforces conditions $1, 2, 3'$ in $\mathbb{A}$. Thus, it is non-empty iff the $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ problem on $\mathbb{C}, \Sigma$ yields a positive answer. Further, the witnessing tree for its non-emptiness yields a structure $\mathbb{A}$ whose expansion $\hat{\mathbb{A}}$ is so that $\textit{core}(\textit{chase}_{\Sigma}(\hat{\mathbb{A}})) = \mathbb{C}$ and $\textit{tw}(\hat{\mathbb{A}}) \leq k$. □

Since the emptiness problem for TWA is in ExpTime [10, 21], a doubly exponential time procedure follows.

**Theorem 8.3.** *The* $\mathrm{STW}_k$ *problem is decidable in 2ExpTime, for every* $k$.

*Proof.* By Lemma 3.6 we can reduce, in NP, the $\mathrm{STW}_k$ into $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$. By Lemma 8.2, we can build a TWA testing conditions $1, 2, 3'$ in exponential time which, by Lemma 8.1, yields a non-empty language iff the $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ problem has a positive answer. Since the emptiness problem for TWA is ExpTime-complete, it follows that the $\mathrm{STW}_k$ problem is decidable in doubly exponential time. □

**Corollary 8.4.** *Given a CQ* $\varphi$ *and a set of FDs* $\Sigma$ *one can produce, in doubly exponential time, a CQ* $\psi$ *so that* $\textit{tw}(\psi) \leq k$ *and* $\varphi \equiv_{\Sigma} \psi$*, if such query exists.*

***$\Sigma$-rank bounded queries*** For any fixed $r$, consider the queries of **semantic $\Sigma$-rank** $r$, defined as those $\varphi$ so that $\mathbb{C} = \textit{core}(\textit{chase}_{\Sigma}(\mathbb{C}_{\varphi}))$ has $\Sigma$-rank $\leq r$. Since this implies that the tree unravelling of $\mathbb{C}$ is polynomial, by Lemma 7.3 a NFA for $\mathcal{L}_{\mathbb{C}}$ can be produced in polynomial time in $\mathbb{C}$, and by Lemma 8.2 a TWA testing $(\textit{core-chase})^{-1} \cap \mathrm{TW}_{\leq k}$ for $\mathbb{C}$ can be built in polynomial time, yielding an exponential-time procedure for the $\mathrm{STW}_k$ problem.

**Corollary 8.5.** *The* $\mathrm{STW}_k$ *problem on semantic* $\Sigma$*-rank* $r$ *CQs is decidable in ExpTime, for every* $k, r$.

Note that semantic $\Sigma$-rank $r$ does not impose any restrictions on the substructure of the edges which are not in $\Sigma$. Thus, in particular, it is still a generalization of the Semantic Tree-width-k problem in the absence of dependencies.

## 9. Final remarks

We have shown that the Semantic tree-width $k$ problem is decidable, and that we can also produce an equivalent query of tree-width $k$ when it exists. Although in principle the bounded tree-width CQ $Q'$ yielded by the algorithm could be doubly exponential in the input query $Q$, we couldn't produce an example witnessing a double-exponential blowup (in fact, not even for a single-exponential). Whether our result is amenable to an optimization procedure—reducing the complexity for the evaluation from $|D|^{O(|Q|)}$ (W[1]-complete) to $f(|Q|) \cdot |D|^k$ (FPT)—will depend, to a large extent, on this blowup.

We believe that these results can be extended with constants and free variables, at the expense of slightly more involved definitions.

As mentioned in the introduction, Barceló et al. show that this problem is undecidable for egd's [4], which generalizes functional dependencies. We leave open the question of whether decidability still holds for arbitrary functional dependencies.

Finally, when the arity of the signature is not fixed, a larger class of tractable queries can be found by considering classes of CQs of bounded hypertree-width [13]. It would be interesting to generalize our result to this setup.

***Acknowledgements*** I am grateful to Pablo Barceló and Miguel Romero for having introduced me to this subject, and to anonymous reviewers for helpful comments.

## References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of databases*. Addison-Wesley Reading, 1995.

[2] A. V. Aho, C. Beeri, and J. D. Ullman. The theory of joins in relational databases. *ACM Transactions on Database Systems*, 4(3):297–314, 1979. doi: 10.1145/320083.320091.

[3] P. Barceló, M. Romero, and M. Y. Vardi. Semantic acyclicity on graph databases. In *Proceedings of the 32nd symposium on Principles of database systems*, pages 237–248. ACM, 2013.

[4] P. Barceló, G. Gottlob, and A. Pieris. Semantic acyclicity under constraints. In *ACM Symposium on Principles of Database Systems (PODS'16)*, 2016.

[5] H. 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. ACM, 1993.

[6] M. Bojańczyk. Tree-walking automata. In *Language and Automata Theory and Applications*, pages 1–2. Springer, 2008.

[7] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proceedings of the ninth annual ACM symposium on Theory of computing*, pages 77–90. ACM, 1977.

[8] C. Chekuri and A. Rajaraman. Conjunctive query containment revisited. In *International Conference on Database Theory*, pages 56–70. Springer, 1997.

[9] H. Chen and V. Dalmau. Beyond hypertree width: Decomposition methods without decompositions. In *Principles and Practice of Constraint Programming-CP 2005*, pages 167–181. Springer, 2005.

[10] S. Cosmadakis, H. Gaifman, P. Kanellakis, and M. Vardi. Decidable optimization problems for database logic programs. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 477–490. ACM, 1988.

[11] V. Dalmau, P. G. Kolaitis, and M. Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *Principles and Practice of Constraint Programming-CP 2002*, pages 310–326. Springer, 2002.

[12] R. Fagin, P. G. Kolaitis, and L. Popa. Data exchange: getting to the core. *ACM Transactions on Database Systems (TODS)*, 30(1):174–210, 2005.

[13] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. In *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 21–32. ACM, 1999.

[14] G. Gottlob, N. Leone, and F. Scarcello. The complexity of acyclic conjunctive queries. *Journal of the ACM*, 48(3):431–498, 2001.

[15] N. Immerman. Expressibility and parallel complexity. *SIAM Journal on Computing*, 18(3):625–638, 1989.

[16] D. Maier, A. O. Mendelzon, and Y. Sagiv. Testing implications of data dependencies. *ACM Transactions on Database Systems*, 4(4): 455–469, 1979. doi: 10.1145/320107.320115.

[17] A. Muscholl, M. Samuelides, and L. Segoufin. Complementing deterministic tree-walking automata. *Information processing letters*, 99(1): 33–39, 2006.

[18] J. Nešetřil and P. O. de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012. ISBN 978-3-642-27874-7. doi: 10.1007/978-3-642-27875-4.

[19] C. H. Papadimitriou and M. Yannakakis. On the complexity of database queries. In *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 12–19. ACM, 1997.

[20] D. Seese. The structure of the models of decidable monadic theories of graphs. *Annals of pure and applied logic*, 53(2):169–195, 1991.

[21] M. Y. Vardi. A note on the reduction of two-way automata to one-way automata. *Information Processing Letters*, 30(5):261–264, 1989.

[22] M. Yannakakis. Algorithms for acyclic database schemes. In *Proceedings of the seventh international conference on Very Large Data Bases-Volume 7*, pages 82–94. VLDB Endowment, 1981.

# Appendix: Extended proofs

*Detailed proof of Lemma 3.3.* Let $\mathbb{A} = \mathit{chase}_{\Gamma_\Sigma}((\mathbb{C}_\varphi)_\Sigma)$, and $\mathbb{B} = \mathit{chase}_{\Gamma_\Sigma}((\mathbb{C}_\psi)_\Sigma)$, for $\mathbb{C}_\varphi$, $\mathbb{C}_\psi$ the canonical structures for $\varphi, \psi$.

For the left-to-right direction, in order to show $\varphi_\Sigma \equiv_{\Gamma_\Sigma} \psi_\Sigma$ it suffices to show, due to Lemma 2.1, that $\mathbb{A}$ and $\mathbb{B}$ are hom-equivalent: $\mathbb{A} \to \mathbb{B}$ and $\mathbb{B} \to \mathbb{A}$ (having isomorphic cores is the same as being hom-equivalent). By Lemma 3.2 we have that $\mathbb{A} \cong (\mathit{chase}_\Sigma(\mathbb{C}_\varphi))_{\Gamma_\Sigma}$ and $\mathbb{B} \cong (\mathit{chase}_\Sigma(\mathbb{C}_\psi))_{\Gamma_\Sigma}$. Since $\varphi \equiv_\Sigma \psi$ we also have, again by Lemma Lemma 2.1, that there are homomorphisms $f : \mathit{chase}_\Sigma(\mathbb{C}_\varphi) \to \mathit{chase}_\Sigma(\mathbb{C}_\psi)$, and $g : \mathit{chase}_\Sigma(\mathbb{C}_\psi) \to \mathit{chase}_\Sigma(\mathbb{C}_\varphi)$, due to hom-equivalence. Then we simply extend $f$ with $key(\bar{a}) \mapsto key(f(\bar{a}))$ for every $key(\bar{a}) \in A$ obtaining a homomorphism $f' : \mathbb{A} \to \mathbb{B}$ or, equivalently, $f' : (\mathit{chase}_\Sigma(\mathbb{C}_\varphi))_{\Gamma_\Sigma} \to (\mathit{chase}_\Sigma(\mathbb{C}_\psi))_{\Gamma_\Sigma}$. Indeed, note that for $key(\bar{a}), b) \in S_i^{(\mathit{chase}_\Sigma(\mathbb{C}_\varphi))_{\Gamma_\Sigma}}$ we have $\bar{a} \in S^{(\mathit{chase}_\Sigma(\mathbb{C}_\varphi))}$, thus $f(\bar{a}) \in S^{(\mathit{chase}_\Sigma(\mathbb{C}_\psi))}$ hence $(key(f(\bar{a})), f(\bar{a}[i])) \in S_i^{(\mathit{chase}_\Sigma(\mathbb{C}_\psi))_{\Gamma_\Sigma}}$. The other homomorphism $\mathbb{B} \to \mathbb{A}$ is obtained in a similar way, this time using $g$. Thus, $\varphi_\Sigma \equiv_{\Gamma_\Sigma} \psi_\Sigma$.

For the right-to-left direction, suppose we have $f : \mathbb{A} \to \mathbb{B}$ and $g : \mathbb{B} \to \mathbb{A}$. Due to Lemma 3.2, we can assume $f : (\mathit{chase}_\Sigma(\mathbb{C}_\varphi))_{\Gamma_\Sigma} \to (\mathit{chase}_\Sigma(\mathbb{C}_\psi))_{\Gamma_\Sigma}$. It is not hard to see that $f$ restricted to the universe of $\mathbb{C}_\varphi$ is a homomorphism from $\mathit{chase}_\Sigma(\mathbb{C}_\varphi)$ to $\mathit{chase}_\Sigma(\mathbb{C}_\psi)$. Indeed, for every $\bar{a} \in S^{\mathit{chase}_\Sigma(\mathbb{C}_\varphi)}$ there are $(key(\bar{a}), \bar{a}[i]) \in S_i^{(\mathit{chase}_\Sigma(\mathbb{C}_\varphi))_\Sigma}$ for every $1 \leq i \leq \mathit{arity}(S)$, and thus $(f(key(\bar{a})), f(\bar{a}[i])) \in S_i^{(\mathit{chase}_\Sigma(\mathbb{C}_\psi))_\Sigma}$ for every $i$, which in turn implies that $f(key(\bar{a})) = key(f(\bar{a}))$ by definition of $(\cdot)_\Sigma$ (because in any structure $\mathbb{C}_\Sigma$, if an element $c$ is $S_i$-related to $c_i$ for every $1 \leq i \leq \mathit{arity}(S)$, it is because $c = key(c_1, \ldots, c_{\mathit{arity}(S)})$. A similar reasoning applies to $g$ and we thus obtain $\varphi \equiv_\Sigma \psi$. □

*Detailed Proof of Lemma 3.4.* We can actually show:

$$\mathit{tw}(\mathbb{A}_\Sigma) \leq \max(\mathit{tw}(\mathbb{A}), \mathit{maxarity}(\mathbb{A})).$$

Given a tree decomposition $T = (V, E)$ of $\mathbb{A}$, it suffices to add, for each $key(a_1, \ldots, a_n)$ in the universe of $\mathbb{A}_\Sigma$ and for some bag $X \in V$ so that $\{a_1, \ldots, a_n\} \subseteq X$ a new leaf with bag $X' =$

$$\{key(a_1, \ldots, a_n), a_1, \ldots, a_n\}$$

of cardinality $\leq n+1$ to $T$, so that $X'$ is a child of $X$. Note that $X$ always exists because one of the conditions the tree decomposition imposes is that every hyper-edge must appear in a bag. It is not hard to see that the resulting tree is a tree decomposition for $\mathbb{A}_\Sigma$. Since the added bags are of size bounded by $\mathit{maxarity}(\mathbb{A}) + 1$, in the worst case we are increasing the width of the tree from $\mathit{maxarity}(\mathbb{A}) - 1$ to $\mathit{maxarity}(\mathbb{A})$. Otherwise, if the tree had already width $\geq \mathit{maxarity}(\mathbb{A})$, notice that the width is not incremented. □

*Detailed proof of Lemma 3.5.* Given a tree decomposition $T = (V, E)$ of $\mathbb{A}_\Sigma$, we obtain a decomposition of $\mathbb{A}$ by replacing, in every bag, $key(a_1, \ldots, a_n)$ with $a_1, \ldots, a_n$. This is because, for every $1 \leq i \leq n$, the subtree $T|_{X \ni key(a_1, \ldots, a_n)}$ induced by the bags containing $key(a_1, \ldots, a_n)$ must have non-empty intersection with the subtree induced by $a_i$. Thus, replacing $key(a_1, \ldots, a_n)$ with $a_1, \ldots, a_n$ does not break the connectivity condition of the decomposition for no $a_i$. The difference is that the resulting decomposition verifies that every $\bar{a} \in S^{\mathbb{A}}$ is in some bag, and therefore it is a tree decomposition of $\mathbb{A}$. The cardinality of the bags is then increased in at most $\mathit{maxarity}(\mathbb{A}) - 1$, as well as the width of the resulting decomposition. □

*Detailed proof of Lemma 6.1.* Let $\mathbb{A}$ be a $\sigma$-structure, and $a \in A$. Let $\mathbb{B} = \mathbb{A}{\restriction}_\Sigma a$. Note that every simple cycle in the underlying undirected graph of $\mathbb{B}$ induces a

(a) $\Sigma$-cycle; or

(b) the presence of $b \xrightarrow{R} c \xleftarrow{R'} b'$ in $\mathbb{B}$, for some relations $R, R'$ and elements $b, b', c$ so that $b \neq b'$ or $R \neq R'$.

In the case (a), suppose $\mathbb{B}$ has a $\Sigma$-cycle $\mathbb{B}'$ consisting of $a_1 \xrightarrow{R_1} \cdots \xrightarrow{R_n} a_{n+1} = a_1$. Let $\hat{\mathbb{B}}$ be the result of removing the edge $a_n \xrightarrow{R_n} a_{n+1}$ from $\mathbb{B}$, and let $\hat{\mathbb{B}}'$ be the result of renaming every element $a_i$ of $\mathbb{B}'$ with a fresh element $b_i$, for all $2 \leq i \leq n$ (*i.e.*, so that $\hat{\mathbb{B}}' \cong \mathbb{B}'$ and the domain of $\hat{\mathbb{B}}'$ is $\{a_1, b_2, \ldots, b_n\}$). Note that

(i) $\hat{\mathbb{B}}$ and $\hat{\mathbb{B}}'$ have only $a_1$ in common,
(ii) $\hat{\mathbb{B}} \cup \hat{\mathbb{B}}' \Rightarrow^*_\Sigma \mathbb{B}$,
(iii) $(\hat{\mathbb{B}} \cup \hat{\mathbb{B}}'){\restriction}_\Sigma a = \hat{\mathbb{B}} \cup \hat{\mathbb{B}}'$.

On the one hand, (i) and (iii) are immediate. Item (ii) can be shown by proving, by induction, that in at most $i$ steps of $\Rightarrow_\Sigma$ we can identify every $b_j$ with $a_j$ for $2 \leq j \leq i$. Thus, in $\leq n$ steps the substructures $\mathbb{B}'$ and $\hat{\mathbb{B}}'$ are fused together, and we obtain precisely $\mathbb{B}$. In the second case (b), this implies that there is a $\Sigma$-confluence rooted at $a$ with some paths as in ($\star$) so that $a_n = b$, $a'_m = b'$, $a_{n+1} = a'_{m+1} = c$, $R_n = R$ and $R'_m = R'$. We can assume, without any loss of generality, that $(a'_i, R'_i, a'_{i+1}) \neq (b', R', c)$ for all $i$. Indeed, there are always paths with these properties whenever $b'$ is at smaller or equal distance to $b$ from $a$ —distance measured in minimum number of $\Sigma$-edges to reach them from $a$. Otherwise we can simply invert the roles of $b$ and $b'$. Let $\mathbb{B}'$ be such $\Sigma$-confluence. Let $\hat{\mathbb{B}}$ be the result of removing the edge $b \xrightarrow{R} c$ from $\mathbb{B}$; and let $\hat{\mathbb{B}}'$ be the result of renaming every element except $a$ with a fresh element. Note that the properties (i)–(iii) above continue to hold also in this case. As before, (i) is immediate by construction; while (ii) follows from the fact that in $\leq i$ steps of $\Rightarrow_\Sigma$ we identify every $a_j$ with $2 \leq j \leq i$ with its copy and in $\leq n + i$ steps we identify every $a'_j$ with $2 \leq j \leq i$ and every $a_t$ with $2 \leq t \leq n$ with its copy. Item (iii) follows from the property above, namely that $(a'_i, R'_i, a'_{i+1}) \neq (b', R', c)$ for all $i$, and thus after removing $b \xrightarrow{R} c$ from $\mathbb{B}$ we will still have that every element is reachable from $a$.

It is easy to see that by applying iteratively these two operations eventually we obtain a structure whose underlying undirected graph is a cactus. Note that item (iii) enables us to repeat the operation, since it tells us that these constructions preserve the "$a$-rootedness". $\square$

*Detailed proof of Lemma 7.1.* The (i) $\Rightarrow$ (ii) part is straightforward since the homomorphic image of a complex path is a complex path of equal moving length. For the (ii) $\Rightarrow$ (i) part, it is not hard to prove the statement for $\mathbb{A} \Rightarrow_\Sigma \mathbb{A}'$. Indeed, suppose $a \xrightarrow{R} b$ and $a \xrightarrow{R} b'$ are in $\mathbb{A}$ for some $R$ in $\Sigma$, and $\mathbb{A}'$ is obtained by identifying $b$ with $b'$ with the provenance homomorphism $h(b') = b$ or the identity otherwise. For any complex path of $\mathbb{A}'$, either it doesn't go through $b$, in which case the same path exists in $\mathbb{A}$, or it is of the form $p(c \overset{R_1}{\leftrightsquigarrow_1} b)(b \overset{R_2}{\leftrightsquigarrow_2} c')p'$ for some paths $p, p'$ and $\leftrightsquigarrow_1, \leftrightsquigarrow_2 \in \{\leftarrow, \rightarrow\}$. Assume that neither $p$ nor $p'$ goes through $b$. Since complex paths are closed under inserting static paths, it follows that $p(c \overset{R_1}{\leftrightsquigarrow_1} b)(b \xleftarrow{R} a)(a \xrightarrow{R} b)(b \overset{R_2}{\leftrightsquigarrow_2} c')p'$ is also a complex path of $\mathbb{A}'$, and that $p(c \overset{R_1}{\leftrightsquigarrow_1} b_1)(b_1 \xleftarrow{R} a)(a \xrightarrow{R} b_2)(b_2 \overset{R_2}{\leftrightsquigarrow_2} c')p'$ is a complex path of $\mathbb{A}$ for some choice $b_1, b_2 \in \{b, b'\}$. For the case where $p, p'$ have more appearances of $b$ the reasoning is similar. By iterating the argument we obtain it for $\mathbb{A} \Rightarrow^*_\Sigma \mathbb{A}'$. $\square$

*Detailed proof of Lemma 7.3.* The NFA accepting $\mathcal{L}_\mathbb{C}$ works over the alphabet $\{a \xrightarrow{R} b, b \xleftarrow{R} a \mid a \xrightarrow{R} b \text{ in } \mathbb{C}\}$. It is a polynomial union of languages, each of these being basically described by the tree unravelling $\mathbb{C}'$ of $\mathbb{C}$ and the canonical homomorphism $h_{tree} : \mathbb{C}' \to \mathbb{C}$. We build one automaton $\mathcal{A}_a$ for each element $a$ of $\mathbb{C}'$. The language $L(\mathcal{A}_a)$ of $\mathcal{A}_a$ consists in all regular static paths of $\mathbb{C}$ beginning and ending in $h_{tree}(a)$. The automaton $\mathcal{A}_a$ for element $a$ is built as having the elements $X = \{a' \mid a' \preceq_{\mathbb{C}'} a\}$ of $\mathbb{C}'$ as state space; $a$ as initial and final state; and a transition $(a, h_{tree}(a) \xrightarrow{R} h_{tree}(b), b)$ and $(b, h_{tree}(b) \xleftarrow{R} h_{tree}(a), a)$ for every edge $a \xrightarrow{R} b$ in $\mathbb{C}'|_X$. We prove that $\mathcal{A}_a$ recognizes all regular static paths starting and ending in $a$ by induction on the $\Sigma$-rank of $a$. For rank 0, $\mathbb{C}'|_X = \mathbb{C}|_\Sigma$ and $h_{tree}$ is the identity, and it follows that $L(\mathcal{A}_a)$ is the set of all regular static paths starting and ending in $a$. For higher rank, take a regular static path of $\mathbb{C}$ and consider the first time it applies a rule that decreases the rank:

$$p(b \xleftarrow{R} c)p'(c \xrightarrow{R'} b)p''$$

where $b \prec_\mathbb{C} a$. By inductive hypothesis $p' \in L(\mathcal{A}_{c'})$ for any $c' \in h^{-1}_{tree}(c)$. It is easy to see that $\mathcal{A}_a$ can accept this path by first reading $p$ in the SCC of $a$ reaching some $b' \in h^{-1}_{tree}(b)$, and then reading $(b \xleftarrow{R} c)$ by going to the SCC corresponding to $c$, that is, to some $c' \in h^{-1}_{tree}(c)$. At this point, the substructure generated by $\{c'' \mid c'' \preceq_{\mathbb{C}'} c'\}$ is isomorphic to that of $\mathcal{A}_{c'}$, and hence by inductive hypothesis $\mathcal{A}_a$ can read $p'$ coming back to $c'$. From this point it can read $(c \xrightarrow{R'} b)$ arriving back to $b'$, and we repeat the same argument for the remaining path $p''$.

On the other hand, it is not hard to see that every word accepted by $\mathcal{A}_a$ is a regular complex path of $\mathbb{C}$. This is because of the functionality of the edges that increase/decrease the rank, forcing the path to have a correct nesting of the regular static rules of the form $(a \xleftarrow{R} b)p(b \xrightarrow{R} a)$. Using these automata one can build a NFA for $\mathcal{L}_\mathbb{C}$ with an automaton $\mathcal{A}_\mathbb{C}$ having $\mathbb{C}$ as transition graph, where $a \xrightarrow{R} b$ is replaced with $(a, (a \xrightarrow{R} b), b)$ and $(b, (b \xleftarrow{R} a), a)$, and with every node being initial and final. Further, from every state/vertex $a$ there is an $\varepsilon$-transition from and to the initial/final state of $\mathbb{A}_{a'}$ for some $a' \in h^{-1}_{tree}(a)$. It follows that the resulting automaton is polynomial in $\{\mathcal{A}_a \mid a \text{ in } \mathbb{C}'\}$. $\square$

*Detailed proof of Lemma 7.4.* It is not hard to see that every time we apply one step of $\Rightarrow_\Sigma$ we maintain the invariant of points 1–3. That is, if $\hat{\mathbb{A}} \Rightarrow_\Sigma \mathbb{A}'$ by a provenance homomorphism $f : \hat{\mathbb{A}} \to \mathbb{A}'$, there must be $a \xrightarrow{R} b$ and $a \xrightarrow{R} b'$ in $\hat{\mathbb{A}}$ so that $b, b'$ are identified in $\mathbb{A}'$ (that is, $f(b) = b'$ and the identity otherwise). Then it must be that $h(b) = h(b')$, as otherwise we would have $h(a) \xrightarrow{R} h(b)$ and $h(a) \xrightarrow{R} h(b')$ in $\mathbb{C}$ where $h(b) \neq h(b')$ which would mean that $\mathbb{C}$ is not a chase structure. Thus, $h$ is still a homomorphism from $\mathbb{A}'$ to $\mathbb{C}$, where $h(\mathbb{A}')|_{\sigma\setminus\Sigma} = \mathbb{C}|_{\sigma\setminus\Sigma}$. Finally, every $h$-regular complex path $p$ present in $\hat{\mathbb{A}}$ appears also in $\mathbb{A}'$ as $f(p)$, that is, by applying $f$ to each element of the path.

Using the properties of the cactus decomposition of the previous section (Lemma 6.1), one can show by induction that for any $h$-regular complex path departing from $a_i$ leading to some $a$ in $\hat{\mathbb{A}}$, and the provenance homomorphism $f : \hat{\mathbb{A}} \to \mathit{chase}_\Sigma(\hat{\mathbb{A}})$ one obtains: $\mathit{chase}_\Sigma(\hat{\mathbb{A}}){\restriction}_\Sigma f(a_i) \cong \mathbb{C}{\restriction}_\Sigma c_i$ and $f(a) = h(a)$ is in $\mathit{chase}_\Sigma(\hat{\mathbb{A}}){\restriction}_\Sigma f(a_i)$. In plain words, after some applications of $\Rightarrow_\Sigma$

we obtain precisely the structure $\mathbb{C}\restriction_\Sigma c_i$, plus perhaps something else that can be homomorphically mapped to $\mathbb{C}$. Indeed, first note that since $\hat{\mathbb{A}}$ is expanded, after some iterations of $\Rightarrow_\Sigma$ we can obtain the structure $\mathbb{A}$ whose every element $a'$ intersects a copy of $\mathbb{C}$ at $h(a')$. That is, for every $a'$ we add a fresh copy of $\mathbb{C}$ and we associate $h(a')$ of that copy with $a'$. Let us call $\mathbb{A}'$ to the structure just described. The fact that $\hat{\mathbb{A}} \Rightarrow_\Sigma^* \mathbb{A}'$ follows from Lemma 6.1. Note that, in particular, $\mathbb{A}'\restriction g(a_i)$ contains $\mathbb{C}\restriction c_i$, for $g$ the provenance homomorphism $\hat{\mathbb{A}} \to \mathbb{A}'$. Consider $g(a_i)$ and its copy of $\mathbb{C}\restriction c_i$. Note that any $h$-regular complex path $p$ of $\hat{\mathbb{A}}$ from $a_i$ to $a$ induces an $h'$-regular complex path $g(p)$ of $\mathbb{A}'$ departing from $g(a_i)$ arriving to $g(a)$ for some suitable $h' : \mathbb{A}' \to \mathbb{C}$. It is not hard to see that if $p$ never decreases the rank then, after some iterations of $\Rightarrow_\Sigma$, $g(a)$ (and all the elements of the path) gets 'glued' to the corresponding element from the copy of $\mathbb{C}\restriction c_i$ hanging from $a_i$. For rank-decreasing $h$-regular complex paths such as $p_l(a \xleftarrow{R} b)p(b' \xrightarrow{R} a')p_r$, where we have that $h(b) = h(b')$, one can show by induction that $b, b'$ are identified after some iterations of $\Rightarrow_\Sigma$ and thus so are $a, a'$ and thus, by induction, after some applications of chase we arrive to a structure where $b, b'$ are mapped to the same element, and we can then apply one more $\Rightarrow_\Sigma$ and have $a, a'$ be mapped to the same element. Hence, assuming $p_l, p_r$ are non rank-decreasing, we have that $p_l p_r$ is transformed into $h(p_l p_r)$ which is a regular complex path of the copy of $\mathbb{C}\restriction c_i$ inside $\mathit{chase}_\Sigma(\hat{\mathbb{A}})$. Applying the reasoning before $a$ is glued to the corresponding element of the copy of $\mathbb{C}\restriction c_i$ hanging from $a_i$.

Repeating this argument for each complex path of 1–3, we obtain that $\bigcup_i \mathit{chase}_\Sigma(\hat{\mathbb{A}})\restriction_\Sigma f(a_i) = \mathbb{C}|_\Sigma$. This, together with point 2, implies that $\bigcup_i \mathit{chase}_\Sigma(\hat{\mathbb{A}})\restriction f(a_i) = \mathbb{C}$, and that therefore there is a homomorphism $\mathbb{C} \to \mathit{chase}_\Sigma(\hat{\mathbb{A}})$. Since there is also a homomorphism $\mathit{chase}_\Sigma(\hat{\mathbb{A}}) \to \mathbb{C}$ by the $\Rightarrow_\Sigma$-invariance of 1–3, and since $\mathbb{C}$ is a chase and core structure, we have that $\mathit{core}(\mathit{chase}_\Sigma(\hat{\mathbb{A}})) = \mathbb{C}$. $\square$