Logical Methods in Computer Science
Volume 19, Issue 4, 2023, pp. 17:1–17:35
https://lmcs.episciences.org/

Submitted Oct. 18, 2022
Published Nov. 29, 2023

REWRITING WITH ACYCLIC QUERIES: MIND YOUR HEAD

GAETANO GECK, JENS KEPPELER, THOMAS SCHWENTICK, AND CHRISTOPHER SPINRATH

TU Dortmund University, Germany
e-mail address: {gaetano.geck, jens.keppeler, thomas.schwentick, christopher.spinrath}@tu-dortmund.de

Abstract. The paper studies the rewriting problem, that is, the decision problem whether, for a given conjunctive query Q and a set $\mathcal{V}$ of views, there is a conjunctive query Q' over $\mathcal{V}$ that is equivalent to Q, for cases where the query, the views, and/or the desired rewriting are acyclic or even more restricted.

It shows that, if Q itself is acyclic, an acyclic rewriting exists if there is any rewriting. An analogous statement also holds for free-connex acyclic, hierarchical, and q-hierarchical queries.

Regarding the complexity of the rewriting problem, the paper identifies a border between tractable and (presumably) intractable variants of the rewriting problem: for schemas of bounded arity, the acyclic rewriting problem is NP-hard, even if both Q and the views in $\mathcal{V}$ are acyclic or hierarchical. However, it becomes tractable if the views are free-connex acyclic (i.e., in a nutshell, their body is (i) acyclic and (ii) remains acyclic if their head is added as an additional atom).

1. Introduction

The *query rewriting problem* asks, for a query Q and a set $\mathcal{V}$ of views, whether there is a query Q' over $\mathcal{V}$ that is equivalent to Q, and to find such a *rewriting* Q'.

We emphasise that in the literature, various notions of rewriting are studied and that the one above is sometimes called *exact* [CGLV05] or *equivalent* [AC19]. Since we are in this paper exclusively interested in exact rewritings, we use the term rewriting in the sense stated above.

Over the last decades, query rewriting has been studied in a wide range of settings with varying database models (relational, XML, graph, ...), query languages, semantics (set, bag, ...) and more. A particular well-known setting is formed by conjunctive queries (CQ) under set semantics, which captures some core aspects of SQL queries. This is also the setting studied in this paper. We refer to [AC19, CY12, Hal01] for an overview of the extensive literature.

In this paper we are interested in a modified version of the query rewriting problem which does not merely ask for *any* rewriting but rather for rewritings that are structurally

---

Key words and phrases: rewriting, acyclic rewriting, acyclic conjunctive queries, free-connex queries, hierarchical queries, NP-hardness.

An abridged conference version of this paper without full proofs and with less detailed explanations has been published in the proceedings of the 25th International Conference on Database Theory [GKSS22]. Besides Lemma 3.7, the results in this version and the conference version are identical.

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.46298/LMCS-19(4:17)2023

simple and allow for efficient evaluation. This is relevant in a setting where access to the database is only possible through the views, in which case the use of an efficient rewriting is obvious. It can also be interesting in a scenario where multiple queries have to be evaluated: by evaluating some of the queries and considering their results as views, some other queries might be rewritable into acyclic queries over the database extended by these views, and therefore they could possibly be evaluated more efficiently.

The forms of structural simplicity that we consider in this paper are acyclic (ACQ) and hierarchical queries (HCQ), and their slightly stronger versions free-connex acyclic (CCQ) and q-hierarchical queries (QHCQ). For a brief discussion of the origin and applications of these classes, we refer to Section 8.

We are interested in two kinds of questions.

(1) Under which circumstances is it guaranteed that a structurally simple rewriting exists, if there exists a rewriting at all?
(2) What is the complexity to decide whether such a rewriting exists and to compute one?

We study these questions depending on the structure of the given views and the given query and we consider the same simplicity notions.

In particular, we study the decision problem $\textsc{Rewr}(\mathbb{V}, \mathbb{Q}, \mathbb{R})$ that asks whether for a given set of views from $\mathbb{V}$ and a query from $\mathbb{Q}$, there is a rewriting from $\mathbb{R}$, for various classes $\mathbb{V}$, $\mathbb{Q}$ and $\mathbb{R}$, with an emphasis on the case where $\mathbb{R}$ is ACQ or a subclass. In the following we refer to this case as the *acyclic rewriting problem.* To the best of our knowledge, there has been no previous work dedicated to the study of questions (1) and (2).

The answer to Question (1) turns out to be very simple and also quite encouraging: in the case that $\mathbb{Q} = \mathsf{ACQ}$, whenever a rewriting exists, there is also an acyclic rewriting. And the same is true for the three subclasses of ACQ. That is, for every query in CCQ, HCQ, or QHCQ, whenever a rewriting exists, there is also a rewriting in CCQ, HCQ, or QHCQ, respectively. Thus, in these cases a rewriting with efficient evaluation exists, if there is a rewriting at all. This answer to Question (1) simplifies the study of Question (2), since it implies that for $\mathbb{Q} \in \{\mathsf{ACQ}, \mathsf{CCQ}, \mathsf{HCQ}, \mathsf{QHCQ}\}$ and $\mathbb{V} \subseteq \mathsf{CQ}$ the decision problems $\textsc{Rewr}(\mathbb{V}, \mathbb{Q}, \mathsf{CQ})$ and $\textsc{Rewr}(\mathbb{V}, \mathbb{Q}, \mathbb{Q})$—and, thus, their complexities—coincide.

The study of Question (2) reveals that the complexity of the acyclic rewriting problem may depend on two parameters: the arity of the underlying schema and the arity of the views. We denote the restriction of $\textsc{Rewr}(\mathbb{V}, \mathbb{Q}, \mathbb{R})$ to database schemas of arity at most k by $\textsc{Rewr}^k(\mathbb{V}, \mathbb{Q}, \mathbb{R})$, and we indicate by $\mathbb{V}^k$ if the arity of views is at most k.

Our main findings regarding the complexity of the acyclic rewriting problem are as follows (see Table 1 for an overview).

- If the query and the views are acyclic and the arity of the views is bounded by some fixed k, then the acyclic rewriting problem is tractable, that is, for every k the decision problem $\textsc{Rewr}(\mathsf{ACQ}^k, \mathsf{ACQ}, \mathsf{ACQ})$ is in polynomial time (Corollary 5.5). Furthermore, an acyclic rewriting can be computed in polynomial time (if it exists). This follows easily with the help of the canonical rewriting approach (see [NSV10, Proposition 5.1]) and our answer to Question (1).
- If the query and the views are acyclic and the arity of the views can be unbounded, then the acyclic rewriting problem is intractable, even if the database schema has bounded arity, more precisely: $\textsc{Rewr}^3(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{ACQ})$ is NP-complete (Theorem 5.7).

TABLE 1. Complexity results for the (acyclic) rewriting problem. Note that $\mathsf{QHCQ} \subseteq \mathsf{CCQ} \subseteq \mathsf{ACQ}$ and $\mathsf{QHCQ} \subseteq \mathsf{HCQ} \subseteq \mathsf{ACQ}$ hold. The head arity of a view V is the arity of its head atom head(V). The weak head arity of a view is defined in Definition 6.3. In a nutshell, a view with weak head arity ℓ can be "split" into (multiple) views with head arity at most ℓ.

$\mathbb{V}$ Views	$\mathbb{Q}$ Query	$\mathbb{R}$ Rewriting	Restriction of views	$\text{Rewr}^k(\mathbb{V},\mathbb{Q},\mathbb{R})$ for every $k \in \mathbb{N}$ (bounded arity db-schema)	$\text{Rewr}(\mathbb{V},\mathbb{Q},\mathbb{R})$ (unbounded arity db-schema)
CQ	ACQ	CQ	Boolean views	NP-complete for $k \geq 3$ (Proposition 3.11)	
ACQ	CQ	CQ	Boolean views	NP-complete for $k \geq 3$ (Proposition 3.11)	
ACQ	ACQ	ACQ	no restriction	NP-complete for $k \geq 3$ (Theorem 5.7)	
ACQ	ACQ	ACQ	head arity $\leq \ell$, $\ell \in \mathbb{N}$	in polynomial time (Corollary 5.5)	
ACQ	ACQ	ACQ	weak head arity $\leq \ell$, $\ell \in \mathbb{N}$	in polynomial time (Theorem 6.9)	
CCQ	ACQ	ACQ	no restriction	in polynomial time (Theorem 6.2)	open
HCQ	HCQ	HCQ	no restriction	NP-complete for $k \geq 3$ (Corollary 7.5)	
QHCQ	HCQ	HCQ	no restriction	in polynomial time (Corollary 7.4)	open
QHCQ	QHCQ	QHCQ	no restriction	in polynomial time (Corollary 7.4)	open

- If the query is acyclic, the views are free-connex acyclic, and the arity of the database schema is bounded by some fixed k, then the acyclic rewriting problem is tractable, that is, for every k: $\text{Rewr}^k(\mathsf{CCQ}, \mathsf{ACQ}, \mathsf{ACQ})$ is in polynomial time (Theorem 6.2).

Many results use a characterisation of rewritability, based on the notion of *cover partitions*, that might be interesting in its own right. Similar notions have been used in earlier work, but ours is particularly suited for the study of exact (equivalent) rewritings.

Our paper is organised as follows: we introduce general basic notions in Section 2 and notions and observations related to rewritings in Section 3. The characterisation of rewritability in terms of cover partitions is given in Section 4. The answer to Question (1) and the complexity of the acyclic rewriting problem for acyclic queries and views are studied in Section 5. The feasibility for free-connex acyclic views (and an additional class) is presented in Section 6. Section 7 shows that the answer to Question (1) is the same for hierarchical and q-hierarchical queries and views. Section 8 discusses related work and, in particular, the relationship of our characterisation of rewritability with results in the literature. Section 9 concludes.

2. Preliminaries

In this section, we fix some notation and recall the basic concepts from relational databases that are relevant for this paper. Let $\mathbb{N}_0$ denote the set of non-negative integers. Let dom and var be countably infinite sets of *data values* (also called *constants*) and *variables*, respectively. We use the natural extensions of mappings of variables onto tuples and sets without notational distinction. That is, for a mapping $f\colon X \to Y$, we write $f(X')$ for $\{f(x) \mid$

$x \in X'\}$ and $f(\mathbf{x})$ for $(f(x_1), \ldots, f(x_k))$ for sets $X' \subseteq X$ and tuples $\mathbf{x} = (x_1, \ldots, x_k) \in X^k$, respectively. The *composition* $f \circ g$ of two functions $g\colon X \to Y$ and $f\colon Y \to Z$ is the function $(f \circ g)\colon X \to Z$ defined by $(f \circ g)(x) = f(g(x))$ for all $x \in X$. By id we denote the identity mapping (on any set of variables).

Databases. Databases and queries are formulated over *database schemas*. A *database schema* $\mathcal{S}$ is a finite set of relation schemas, each represented by a symbol R and associated with a fixed arity $\text{ar}(R) \in \mathbb{N}_0$.[1] A *fact* or *$R$-fact* $R(a_1, \ldots, a_r)$ comprises a relation symbol R with some arity r and data values $a_1, \ldots, a_r$. A *database* D over schema $\mathcal{S}$ is a finite set of R-facts for $R \in \mathcal{S}$.

Queries. In general, queries map databases to relations. In this paper, we consider conjunctive queries and restrictions of them. Conjunctive queries are queries that can be expressed syntactically as conjunctions of relation atoms as follows.

An *atom* is of the form $R(x_1, \ldots, x_r)$ with a relation symbol R with arity r, and variable set $\{x_1, \ldots, x_r\}$. We denote the variable set of an atom A by $\text{vars}(A)$. Analogously to facts, an atom with relation symbol R is called an *R-atom* if we want to stress the associated relation (symbol). More generally, $\mathcal{S}$-atoms are R-atoms for some symbol R in schema $\mathcal{S}$.

We represent a *conjunctive query* (*CQ* for short) over schema $\mathcal{S}$ as a rule of the form $A \leftarrow A_1, \ldots, A_m$ whose *body* $\{A_1, \ldots, A_m\}$ consists of a positive number of atoms and whose *head* is a single atom A such that the following two conditions are satisfied. First, atoms $A_1, \ldots, A_m$ refer to relation symbols from $\mathcal{S}$ and atom A, on the contrary, does not. Second, the query is *safe*, that is, every variable in the head occurs in at least one atom of the body. Let $\text{head}(Q)$ and $\text{body}(Q)$ denote the head and body of a query Q, respectively. Variables that occur in the head of a query are called *head variables*; all other variables are called *quantified variables*. A query without quantified variables is called a *full* query. Queries with two or more body atoms that refer to the same relation are said to have *self-joins*. The *arity* of a query Q is the arity of its head atom. Furthermore, a query Q is a Boolean query if the arity of $\text{head}(Q)$ is 0.

Like for sets and tuples, we also use the natural extension of mappings of variables on atoms without difference in notation: for a mapping $f : \mathsf{var} \to Y$, let $f(A)$ denote $R(f(x_1), \ldots, f(x_r))$ for an atom $A = R(x_1, \ldots, x_r)$.

A *valuation* is a mapping $\vartheta : \mathsf{var} \rightharpoonup \mathsf{dom}$. A database D *satisfies* a set $\mathcal{A}$ of atoms under a valuation ϑ, if $\vartheta(\mathcal{A}) \subseteq D$, that is for every atom $R(x_1, \ldots, x_r)$ in $\mathcal{A}$ the fact $R(\vartheta(x_1), \ldots, \vartheta(x_r))$ is contained in D. The *result* of query Q on database D is defined as

$$Q(D) = \{\vartheta(\text{head}(Q)) \mid \vartheta \text{ is a valuation and } D \text{ satisfies body}(Q) \text{ under } \vartheta\}.$$

Relationships between queries. Queries over the same schema can be compared with respect to the results they define. Let Q_1 and Q_2 be queries. We say that Q_1 is *contained* in Q_2 (notation: $Q_1 \sqsubseteq Q_2$) if $Q_1(D) \subseteq Q_2(D)$ for every database D. We say that Q_1 and Q_2 are *equivalent* (notation: $Q_1 \equiv Q_2$) if $Q_1 \sqsubseteq Q_2$ and $Q_2 \sqsubseteq Q_1$.

It is well-known that a conjunctive query Q_1 is contained in a conjunctive query Q_2 if and only if there is a homomorphism from the latter query into the first [CM77]. Such a *homomorphism* is a mapping $h\colon \text{vars}(Q_2) \to \text{vars}(Q_1)$ such that

(1) $h(\text{body}(Q_2)) \subseteq \text{body}(Q_1)$ and

[1] We refer to database schemas usually by *schema*.

(2) $h(\text{head}(Q_2)) = \text{head}(Q_1)$ hold.

We call h a *body homomorphism* if it fulfils Condition (1).

A conjunctive query Q_1 is *minimal* if there is no conjunctive query Q_2 such that $Q_2 \equiv Q_1$ and $|\text{body}(Q_2)| < |\text{body}(Q_1)|$ holds.

Structurally simple queries. Despite their simplicity and restricted expressibility, several interesting problems are intractable for conjunctive queries in general. Therefore, different fragments have been studied in the literature. In this paper, we are particularly concerned with "acyclic" queries, which allow, for instance, evaluation in polynomial time. We also consider three subclasses of acyclic queries, namely, free-connex acyclic, hierarchical, and q-hierarchical queries.

Acyclic and free-connex acyclic queries. A *join tree* for a query Q is a tree whose vertices are the atoms in the query's body that satisfies the following path property: for two atoms $A, A' \in \text{body}(Q)$ with a common variable x, all atoms on the path from A to A' contain x. A query Q is *acyclic* if it has a join tree. It is *free-connex acyclic* if Q is acyclic and the Boolean query whose body is $\text{body}(Q) \cup \{\text{head}(Q)\}$ is acyclic as well [BDG07, BB13].

Hierarchical queries. For a fixed query Q and some variable x in Q, let $\text{atoms}(x)$ denote the set of atoms in $\text{body}(Q)$ in which x appears.

Definition 2.1 [DS07], [BKS17, Definition 3.1]**.** A conjunctive query Q is *hierarchical* if, for all variables x, y in Q, one of the following conditions is satisfied.

(1) $\text{atoms}(x) \subseteq \text{atoms}(y)$
(2) $\text{atoms}(x) \supseteq \text{atoms}(y)$
(3) $\text{atoms}(x) \cap \text{atoms}(y) = \emptyset$

Thus, the "hierarchy" is established by the query's variables and the sets of atoms that contain them.

A conjunctive query Q is *q-hierarchical* if it is hierarchical and for all variables $x, y \in \text{vars}(Q)$ the following is satisfied: if $\text{atoms}(x) \subsetneq \text{atoms}(y)$ holds and x is in the head of Q, then y is also in the head of Q.

For brevity, we denote by CQ, ACQ, CCQ, HCQ, and QHCQ the classes of conjunctive queries in general and those conjunctive queries that are acyclic, free-connex acyclic, hierarchical or q-hierarchical, respectively. We note that each q-hierarchical query is free-connex acyclic [IUV17, Proposition 4.25] and each hierarchical query is acyclic.[2]

Views and rewritings. A view V over a schema $\mathcal{S}$ is just a query over schema $\mathcal{S}$.[3] A set $\mathcal{V} = \{V_1, \ldots, V_k\}$ of views induces, for each database D over schema $\mathcal{S}$, the $\mathcal{V}$-defined *database* $\mathcal{V}(D) = V_1(D) \cup \cdots \cup V_k(D)$.

In this paper, we consider only views that are conjunctive queries. We furthermore assume that every view defines its own relation in the derived database. We denote the set of relation schemas induced by the heads of the views in $\mathcal{V}$ by $\mathcal{S}_{\mathcal{V}}$.

---

[2]The latter inclusion is mentioned in, e.g. [HY19, KNOZ20]; it also follows readily from the former inclusion. For acyclicity the head of a query is of no concern, and the Boolean variant of a hierarchical query is always q-hierarchical by definition, and, thus, free-connex acyclic which implies the existence of a join tree for the body of the original query.

[3]They are called views due to their special role which distinguishes them from "normal" queries.

In a nutshell, a rewriting of a query Q over a set $\mathcal{V}$ of views is a query over $\mathcal{S}_\mathcal{V}$ that is meant to yield, for each database D, the same result over $\mathcal{V}(D)$ as Q does over D.

Definition 2.2. Let Q be a query, $\mathcal{V}$ a set of views, and Q' a query over $\mathcal{S}_\mathcal{V}$. We call Q' a *$\mathcal{V}$-rewriting of* Q if, for every database D, it holds that $Q'(\mathcal{V}(D)) = Q(D)$.

Formally, we only consider rewritings over $\mathcal{S}_\mathcal{V}$. This is not a restriction, since database relations can be replicated as view relations. In the literature, rewritings often only aim at approximations of the query [PH01]. However, in this paper, we are only interested in *equivalent* and *complete* rewritings, where $Q' \circ \mathcal{V} \equiv Q$ holds. Here $Q' \circ \mathcal{V}$ denotes the query with query result $Q'(\mathcal{V}(D))$ for every database D.

We illustrate the above definitions by an example.

Example 2.3. Let $\mathcal{V}$ be a set of two views

$$\begin{aligned} V_1(x_1, w_1) &\leftarrow P(v_1, v_1', x_1),\ R(x_1, w_1),\ S(w_1), \\ V_2(y_2, z_2) &\leftarrow S(y_2),\ T(y_2, z_2) \end{aligned}$$

over schema $\mathcal{S} = \{P, R, S, T\}$ and let Q be the query

$$H(x, y, y') \leftarrow P(u, u', x), R(x, w), S(w), T(w, y), T(w, y').$$

For each database D, query Q yields the same result on D as the following query Q'

$$H(x, y, y') \leftarrow V_1(x, w), V_2(w, y), V_2(w, y')$$

on $\mathcal{V}(D)$. Therefore, Q' is a $\mathcal{V}$-rewriting of Q.

3. The Rewriting Problem

In this section, we define the rewriting problem, recall some known results about its complexity and discuss some relevant concepts that were used to tackle it.

Definition 3.1. Let $\mathbb{V}$, $\mathbb{Q}$, and $\mathbb{R}$ be classes of conjunctive queries. The *rewriting problem* for $\mathbb{V}$, $\mathbb{Q}$, and $\mathbb{R}$, denoted $\text{Rewr}(\mathbb{V}, \mathbb{Q}, \mathbb{R})$ asks, upon input of a query $Q \in \mathbb{Q}$ and a (finite) set $\mathcal{V} \subseteq \mathbb{V}$ of views, whether there is a $\mathcal{V}$-rewriting of Q in the class $\mathbb{R}$. We write $\text{Rewr}^k(\mathbb{V}, \mathbb{Q}, \mathbb{R})$ for the restriction, where the arity of the database schema is bounded by k.

In general, the rewriting problem for conjunctive queries is NP-complete.

Theorem 3.2 [LMSS95, Theorem 3.10]**.** $\text{Rewr}(\mathsf{CQ}, \mathsf{CQ}, \mathsf{CQ})$ *is* NP*-complete.*

There is a straightforward, albeit in general inefficient, way of finding a rewriting for a conjunctive query and views. In fact, it can be shown that, for given Q and $\mathcal{V}$, a rewriting Q' over $\mathcal{V}$ exists, if and only if a certain canonical query $\mathsf{canon}(Q, \mathcal{V})$ exists and is a rewriting.

This *canonical candidate* $\mathsf{canon}(Q, \mathcal{V})$ can be obtained by evaluating $\mathcal{V}$ on the canonical database $\mathsf{canon}(Q)$, which in turn is defined as the set of body atoms of Q viewed as facts, in which the variables are considered as fresh constants. The canonical candidate has the same head as Q and its body is just $\mathcal{V}(\mathsf{canon}(Q))$.[4] If the canonical candidate turns out to be a rewriting, then we often call it *canonical rewriting*.

An important detail should be mentioned here: the canonical candidate does not always exist, e.g., if $\mathcal{V}(\mathsf{canon}(Q))$ does not contain all head variables of Q or it is outright empty. In that case, there is no rewriting.

[4]Here the variables are considered as variables, not as constants.

Proposition 3.3 [NSV10, Proposition 5.1]**.** *Let Q be a conjunctive query and $\mathcal{V}$ a set of views. If there is a $\mathcal{V}$-rewriting of Q, then the canonical candidate* $\mathsf{canon}(Q, \mathcal{V})$ *is such a rewriting.*

For conjunctive queries without self-joins this had been shown in [CR00, Lemma 7].

Example 3.4. Let us consider the query Q defined by

$$H(x,y,z) \leftarrow C(x,y,z),\ R(x,y),\ S(y,z),\ T(z,x)$$

and the views

$$V_1(u_1,v_1,w_1) \leftarrow C(u_1,v_1,w_1) \quad \text{and} \quad V_2(x_2,y_2,z_2,u_2) \leftarrow R(x_2,y_2),\ S(y_2,z_2),\ T(z_2,u_2).$$

Evaluating the views V_1 and V_2 on the canonical database $\mathsf{canon}(Q)$ of Q yields the result $\{V_1(x,y,z), V_2(x,y,z,x)\}$. Thus, the query Q' defined by

$$H(x,y,z) \leftarrow V_1(x,y,z), V_2(x,y,z,x)$$

is the canonical candidate, which happens to be an actual $\mathcal{V}$-rewriting.

In this example it is easy to see that the canonical candidate is a $\mathcal{V}$-rewriting, without considering any (let alone all) databases. In fact, in this particular case, this can be established with the help of an expansion of the rewriting. Before we define expansions, we need one more definition.

Definition 3.5 (View Application)**.** An *application* of a view V is a substitution $\alpha\colon \mathrm{vars}(V) \to \mathsf{var}$ that does not unify any quantified variable of V with another variable of V.

This definition reflects the fact that a rewriting can access a view only through its head and therefore cannot unify any quantified variables.

We say that a collection $\alpha_1, \ldots, \alpha_m$ of applications for views $V_1, \ldots, V_m$ fulfils *quantified variable disjointness*, if for all $i, j \in \{1, \ldots, m\}$, each quantified variable x of V_i, and each variable y of V_j with $i \neq j$, it holds $\alpha_i(x) \neq \alpha_j(y)$. That is, beyond what is already required by the definition of an application, a view application never maps a quantified variable of a view to a variable in the range of any other view application. This ensures that, for all $i, j \in \{1, \ldots, m\}$ with $i \neq j$, the bodies $\alpha_i(\mathrm{body}(V_i))$ and $\alpha_j(\mathrm{body}(V_j))$ only share variables from $\mathrm{vars}(\alpha_i(\mathrm{head}(V_i))) \cap \mathrm{vars}(\alpha_j(\mathrm{head}(V_j)))$.

An expansion of a $\mathcal{V}$-rewriting Q' is, intuitively, obtained by inlining the bodies of the views from $\mathcal{V}$ in Q'.

Definition 3.6 (Expansion)**.** Let $\mathcal{V}$ be a set of views over a schema $\mathcal{S}$ and let Q' be a query with $\mathrm{body}(Q') = \{A'_1, \ldots, A'_m\}$ over the schema $\mathcal{S}_\mathcal{V}$. Furthermore, let, for each $i \in \{1, \ldots, m\}$, V_i be the view in $\mathcal{V}$ and α_i be an application for V_i such that $A'_i = \alpha_i(\mathrm{head}(V_i))$, such that $\alpha_1, \ldots, \alpha_m$ fulfil quantified variable disjointness.

The *expansion of Q' w.r.t. $\mathcal{V}$ and $\alpha_1, \ldots, \alpha_m$* is the query that has the same head as Q' and body $\bigcup_{i=1}^{m} \alpha_i(\mathrm{body}(V_i))$.

Since the applications $\alpha_1, \ldots, \alpha_m$ in Definition 3.6 are uniquely determined up to renaming of quantified variables, we usually do not mention them explicitly and just speak of an expansion of a query Q' w.r.t. $\mathcal{V}$.

Furthermore, because an expansion Q'' of a $\mathcal{V}$-rewriting Q' is, intuitively, obtained by inlining the bodies of the views, the expansion Q'' is "equivalent" to Q' in the following sense.

Lemma 3.7. *Let $\mathcal{V}$ be a set of views, Q' be a query over the schema $\mathcal{S}_\mathcal{V}$, and Q'' be an expansion of Q' w.r.t. $\mathcal{V}$. Then $Q''(D) = Q'(\mathcal{V}(D))$ holds for every database D.*

Lemma 3.7 has been proved, e.g., as part of the proof of [AC19, Theorem 3.5]. For the sake of convenience, we provide a short proof using our notation in the following.

Proof of Lemma 3.7. Following Definition 3.6, let $\{A'_1, \ldots, A'_m\}$ be the body of Q', and let $\alpha_1, \ldots, \alpha_m$ be applications fulfilling quantified variable disjointness, and $V_1, \ldots, V_m \in \mathcal{V}$ views such that $A'_i = \alpha_i(\text{head}(V_i))$ holds for all $i \in \{1, \ldots, m\}$ and $\text{body}(Q'') = \bigcup_{i=1}^m \alpha_i(\text{body}(V_i))$ holds.

We first show the inclusion $Q''(D) \subseteq Q'(\mathcal{V}(D))$. Let D be a database and ϑ be a valuation such that D satisfies $\vartheta(\text{body}(Q'')) = \vartheta(\bigcup_{i=1}^m \alpha_i(\text{body}(V_i)))$ under ϑ. By the definition of an expansion, $\vartheta(\text{head}(Q'')) = \vartheta(\text{head}(Q'))$ holds. Thus, it suffices to show that $\vartheta(\text{body}(Q')) \subseteq \mathcal{V}(D)$ holds witnessing that $\vartheta(\text{head}(Q')) \in Q'(\mathcal{V}(D))$.

Since $\vartheta(\text{body}(Q'') \subseteq D$, it holds $\vartheta(\alpha_i(\text{body}(V_i))) \subseteq D$, for all $i \in \{1, \ldots, m\}$. Interpreting $\vartheta \circ \alpha_i$ as a valuation, it follows that $\vartheta(\alpha_i(\text{head}(V_i))) \in V_i(D)$. But $\alpha_i(\text{head}(V_i)) = A'_i$ for all $i \in \{1, \ldots, m\}$ and, therefore, we can conclude that $\vartheta(\text{body}(Q')) = \vartheta(\{A'_1, \ldots, A'_m\}) \subseteq \mathcal{V}(D)$.

For the inclusion $Q''(D) \supseteq Q'(\mathcal{V}(D))$ let ϑ be a valuation such that $\mathcal{V}(D)$ satisfies $\text{body}(Q')$ under ϑ. Thus $\vartheta(A'_i) \in \mathcal{V}(D)$ and therefore $\vartheta(\alpha_i(\text{head}(V_i))) \in V_i(D)$ holds, for all $i \in \{1, \ldots, m\}$. The latter implies that there are valuations μ_i such that $\vartheta(\alpha_i(\text{head}(V_i))) = \mu_i(\text{head}(V_i))$ such that $\mu_i(\text{body}(V_i)) \subseteq D$. That is, the μ_i map $\text{body}(V_i)$ into D and agree with $\vartheta \circ \alpha_i$ on all head variables of V_i. Moreover, since the application α_i does not unify any quantified variables with other variables, the valuation ϑ can be extended to a valuation ϑ_i^+ such that $\vartheta_i^+(\alpha_i(\text{body}(V_i))) \subseteq D$.

Lastly, thanks to the quantified variable disjointness of the α_i, the extended mappings ϑ_i^+ can be combined into a valuation ϑ^+ which maps all $\alpha_i(\text{body}(V_i))$ into D. □

We further note that the expansion of a rewriting Q' can be directly compared with a query Q since it is over the same schema. In fact, we will frequently use the following result which follows readily from Lemma 3.7.

Proposition 3.8 [AC19, Theorem 3.5]**.** *Let Q be a query over a schema $\mathcal{S}$, $\mathcal{V}$ be a set of views over $\mathcal{S}$, Q' be a query over the schema $\mathcal{S}_\mathcal{V}$, and Q'' be an expansion of Q' w.r.t. $\mathcal{V}$.*

The expansion Q'' of Q' is equivalent to Q if and only if Q' is a $\mathcal{V}$-rewriting of Q.

In Example 3.4, the query Q' is a $\mathcal{V}$-rewriting of Q, because the views are full queries, and, thus, there is only one (unique) expansion of Q' w.r.t. $\mathcal{V}$ which even coincides with Q.

If there is no a priori bound on the arity of views, then the canonical candidate can be of exponential size in $|Q| + |\mathcal{V}|$. However, if there is a rewriting, then there always is one of polynomial size. In fact, at most n view applications are needed if the body of query Q contains n atoms [LMSS95, Lemma 3.5].

Example 3.9. Consider the query $H() \leftarrow S(x, y), R(x), R(y)$ and the views

$$V_1(x_1, y_1) \leftarrow S(x_1, y_1) \text{ and } V_2(x_{2,1}, \ldots, x_{2,n}) \leftarrow R(x_{2,1}), \ldots, R(x_{2,n}).$$

Evaluating the views on the canonical database $\mathsf{canon}(Q)$ yields the result

$$\{V_1(x, y)\} \cup \{V_2(a_1, \ldots, a_n) \mid a_1, \ldots, a_n \in \{x, y\}\}.$$

This result, and thus, the body of the canonical candidate, has exponential size. There is, however, a simple $\{V_1, V_2\}$-rewriting which has as many view atoms as Q has atoms, namely,

$$H() \leftarrow V_1(x, y), V_2(x, \dots, x), V_2(y, \dots, y).$$

Therefore, an NP-algorithm for $\textsc{Rewr}(\mathsf{CQ}, \mathsf{CQ}, \mathsf{CQ})$ can "guess" a rewriting of polynomial size and test whether it yields an expansion that is equivalent to the given query. Since equivalence of CQ can be tested in NP, this indeed shows that $\textsc{Rewr}(\mathsf{CQ}, \mathsf{CQ}, \mathsf{CQ})$ is in NP.

Due to their potentially exponential size, canonical candidates are, in general, not useful as a starting point for polynomial time algorithms. However, in some cases their size is only polynomial, in particular, if the views have bounded arity. In such cases, the above characterisation can sometimes be used to obtain efficient algorithms. In particular, this holds for acyclic views of bounded arity and acyclic queries. For a class $\mathbb{V}$ of views, we write $\mathbb{V}^k$ for the restriction to views of arity at most k.

Proposition 3.10. *For every $k \geq 0$, $\textsc{Rewr}(\mathsf{ACQ}^k, \mathsf{ACQ}, \mathsf{CQ})$ is in polynomial time.*

Proof. We first observe that, if it exists, the canonical candidate always obeys $Q \sqsubseteq \mathsf{canon}(Q, \mathcal{V}) \circ \mathcal{V}$. Indeed, a homomorphism from an expansion of $\mathsf{canon}(Q, \mathcal{V})$ to Q can be obtained by combining the valuations that yield $\mathcal{V}(\mathsf{canon}(Q))$ from $\mathsf{canon}(Q)$ as follows. To improve readability and avoid technical clutter we identify the variables in Q with their corresponding constants in $\mathsf{canon}(Q)$ in the following. Notably, this means that we identify $\mathsf{canon}(Q)$ with $\text{body}(Q)$, i.e. we write $\mathsf{canon}(Q) = \text{body}(Q)$.

Let $A'_1, \dots, A'_m$ be the atoms of $\mathsf{canon}(Q, \mathcal{V})$ and, for each $i \in \{1, ..., m\}$, let α_i be an application and V_i be a view such that $A'_i = \alpha_i(\text{head}(V_i))$, such that $\alpha_1, \dots, \alpha_m$ fulfil quantified variables disjointness. Furthermore, since the A'_i are also contained in $\mathcal{V}(\mathsf{canon}(Q))$ by definition of $\mathsf{canon}(Q, \mathcal{V})$, there are valuations ϑ_i such that $A'_i = \vartheta_i(\text{head}(V_i))$ and $\vartheta_i(\text{body}(V_i)) \subseteq \mathsf{canon}(Q)$ holds. In particular, we have that $\vartheta_i(\text{head}(V_i)) = \alpha_i(\text{head}(V_i))$ for all $i \in \{1, \dots, m\}$.

Let y be a variable occuring in $\alpha_i(\text{body}(V_i))$ and x such that $y = \alpha_i(x)$ for some $i \in \{1, \dots, m\}$. Then the desired homomorphism h maps y to $\vartheta_i(x)$. Note that h is the identity on variables occuring in $\alpha_i(\text{head}(V_i))$ because $\alpha_i(\text{head}(V_i)) = \vartheta_i(\text{head}(V_i))$ and, thus, $y = \alpha_i(x) = \vartheta_i(x) = h(y)$. Hence, h is well-defined, since, if a variable y occurs in $\alpha_i(\text{body}(V_i))$ and $\alpha_j(\text{body}(V_j))$ with $i \neq j$ then it also occurs in $\alpha_i(\text{head}(V_i))$ and $\alpha_j(\text{head}(V_j))$ thanks to quantified variable disjointness.

It is easy to see that h is a homomorphism from the expansion of $\mathsf{canon}(Q, \mathcal{V})$ to Q, because

$$h(\alpha_i(\text{body}(V_i))) = \vartheta_i(\text{body}(V_i)) \subseteq \mathsf{canon}(Q)$$

holds for all $i \in \{1, \dots, m\}$ and h is the identity on all variables occuring in the head of $\mathsf{canon}(Q, \mathcal{V})$.

Therefore, to test for rewritability, it suffices to check whether $\mathsf{canon}(Q, \mathcal{V}) \circ \mathcal{V} \sqsubseteq Q$ holds.

Since the views are acyclic and their arity is bounded by a constant, the polynomial-sized canonical candidate $\mathsf{canon}(Q, \mathcal{V})$ can be computed in polynomial time.[5] It then suffices to test whether a homomorphism from Q to an expansion of $\mathsf{canon}(Q, \mathcal{V})$ exists. This test can be done in polynomial time because the query Q is acyclic.

[5] Evaluating and testing containment for acyclic conjunctive queries is in polynomial time [Yan81, AHV95].

Thus, overall, it can be tested efficiently whether $\mathsf{canon}(Q, \mathcal{V}) \circ \mathcal{V} \sqsubseteq Q$ holds if the query and views are acyclic and the arity of the views is bounded by a constant.

Hence, the statement follows due to Proposition 3.3. □

Chekuri and Rajaraman have shown that the rewriting problem is in P for acyclic queries without self-joins and arbitrary views using a similar idea [CR00, Theorem 5].

However, even for Boolean views and databases over a small fixed schema, it does not suffice to restrict only the query or only the views to acyclic queries.

Proposition 3.11. *$\text{Rewr}^k(\mathsf{CQ}^0, \mathsf{ACQ}, \mathsf{CQ})$ and $\text{Rewr}^k(\mathsf{ACQ}^0, \mathsf{CQ}, \mathsf{CQ})$ are* NP*-complete, for every $k \geq 3$. This even holds for Boolean queries, view sets with only one Boolean view and a schema with two relations of maximum arity 3.*

Proof. The upper bound holds thanks to Theorem 3.2. For the lower bound, we first show, by a reduction from the three-colouring problem, that it is NP-hard to test, for an *acyclic* Boolean conjunctive query Q_1 and a Boolean conjunctive query Q_2, whether $Q_1 \sqsubseteq Q_2$ holds.[6]

To this end, let G be an undirected graph without isolated vertices and self-loops (i.e. edges of the form (v, v)). With each vertex v of G we associate a variable x_v. We define Q_2 to be the Boolean conjunctive query with head $H()$ that consists of all atoms $E(x_v, x_w)$ such that (v, w) is an edge in G. The Boolean query Q_1 is defined by the rule

$$H() \leftarrow E(b, r), E(r, y), E(y, b), E(r, b), E(y, r), E(b, y), C(b, r, y)$$

where b, r, y are variables representing the colours blue, red, and yellow. Observe that Q_1 is acyclic, because all variables are contained in the "cover-atom" $C(b, r, y)$.

Towards the correctness of the reduction, if $Q_1 \sqsubseteq Q_2$ holds, there is a homomorphism h from Q_2 to Q_1. This homomorphism represents a valid colouring, since two variables x_v and x_w cannot be mapped to the same variable in Q_1 if there is an edge (v, w) in G. Conversely, every valid colouring of G gives rise to a homomorphism from Q_2 to Q_1 and therefore witnesses $Q_1 \sqsubseteq Q_2$.

Since it is well-known that $Q_1 \sqsubseteq Q_2$ if and only if $Q_1 \equiv Q_2 \wedge Q_1$, we conclude in the second step that the equivalence problem for Boolean conjunctive queries Q_1, Q_2 is NP-hard, even if Q_1 is acyclic.

Finally, we reduce the equivalence problem with acyclic Q_1 to the rewriting problem. Let thus Q_1 and Q_2 be Boolean conjunctive queries. The input instance (Q_1, Q_2) for the equivalence problem is mapped to an input instance for the rewriting problem by assigning Q_1 the role of the query and Q_2 the role of (the sole) view.

The canonical rewriting, if it exists, consists only of the single atom $Q_2()$, since Q_2 is a Boolean query. Thus, there is a Q_2-rewriting of Q_1 if and only if Q_1 is equivalent to Q_2. This reduction establishes that $\text{Rewr}^3(\mathsf{CQ}^0, \mathsf{ACQ}, \mathsf{CQ})$ is NP-hard. For the NP-hardness of $\text{Rewr}^3(\mathsf{ACQ}^0, \mathsf{CQ}, \mathsf{CQ})$, the roles of Q_1 and Q_2 are simply swapped in the last reduction.

By composing the three reductions, the proposition is established. □

4. A Characterisation

In this section we give a characterisation of rewritability of a query Q with respect to a set $\mathcal{V}$ of views. It is in terms of a partition of the atoms of body(Q), where each set of the partition can be matched with a view in a specific way. We refer to such partitions as *cover*

[6] We think that this is folklore knowledge but could not find a reference for it.

partitions and the matches as *cover descriptions*. The characterisation is very similar to other such characterisations in the literature [GKC06, AC19], in particular to "MiniCon Descriptions" [PH01]. However, in its specific form and notation it is tailored for our needs in the subsequent sections. We will discuss the relationship of our characterisation with others further below in Section 8.

Next we define the notions of *cover descriptions* and *cover partitions*. Let Q be a conjunctive query. For a set $\mathcal{A} \subseteq \text{body}(Q)$ we define $\text{bvars}_Q(\mathcal{A})$ as the set of *bridge variables* of $\mathcal{A}$, that occur in $\mathcal{A}$ as well as in the head of Q or in some atom of Q not in $\mathcal{A}$. More formally, $\text{bvars}_Q(\mathcal{A}) = \text{vars}(\mathcal{A}) \cap (\text{vars}(\text{head}(Q)) \cup \text{vars}(\text{body}(Q) \setminus \mathcal{A}))$.

Example 4.1. Consider the query Q defined by $H(x, y, z) \leftarrow R(x, u), S(u, y, w), T(y, w, z)$. For the set $\mathcal{A} = \{R(x, u), S(u, y, w)\}$ of atoms from the body of Q, we have $\text{bvars}_Q(\mathcal{A}) = \{x, y, w\}$ because x and y are head variables of Q and because w, and also y, occurs in the atom $T(y, w, z)$ that does not belong to $\mathcal{A}$. The variable u in $\mathcal{A}$ is not a bridge variable since it is quantified and does not occur in any atom outside of $\mathcal{A}$.

Definition 4.2 (Cover Description)**.** A *cover description* C for a query Q is a tuple $(\mathcal{A}, V, \alpha, \psi)$ where

- $\mathcal{A}$ is a subset of body(Q),
- V is a view,
- α is an application of V, and
- ψ is a mapping from $\text{vars}(\alpha(V))$ to $\text{vars}(Q)$,

such that

(1) $\mathcal{A} \subseteq \alpha(\text{body}(V))$,
(2) $\text{bvars}_Q(\mathcal{A}) \subseteq \alpha(\text{vars}(\text{head}(V)))$,
(3) ψ is a body homomorphism from $\alpha(V)$ to Q, and
(4) ψ is the identity on $\text{vars}(\mathcal{A})$.

Intuitively, the conditions of Definition 4.2 testify that the atoms of Q in $\mathcal{A}$ can be "represented" or "covered" by a V-atom in a rewriting of Q—giving rise to the term cover description: Condition (1) means that every atom in $\mathcal{A}$ has a matching atom in V. Condition (3) ensures that every database which satisfies the query also satisfies the view (under application α). Conditions (2) and (4) establish an "interface" to combine multiple cover descriptions in a straightforward and compatible manner such that, overall, a rewriting is described.

Example 4.3. Consider the query Q given by the rule

$$H(x, y, z) \leftarrow R(x, y, z), T(x, v), F(v), E(w), S(w, z)$$

and the following views.

$$\begin{aligned} V_1(x_1, y_1, w_1) &\leftarrow R(x_1, y_1, u_1), T(x_1, v_1), F(v_1), E(w_1), S(w_1, u_1) \\ V_2(x_2, y_2, z_2) &\leftarrow R(x_2, y_2, z_2), F(v_2) \\ V_3(w_3, z_3) &\leftarrow S(w_3, z_3), E(w_3) \end{aligned}$$

The tuple $C_1 = (\mathcal{A}_1, V_1, \alpha_1, \psi_1)$ with $\mathcal{A}_1 = \{T(x, v), F(v)\}$,

$$\alpha_1 = \{x_1 \mapsto x, y_1 \mapsto y', u_1 \mapsto u', v_1 \mapsto v, w_1 \mapsto w'\}, \text{ and}$$
$$\psi_1 = \{x \mapsto x, y' \mapsto y, u' \mapsto z, v \mapsto v, w' \mapsto w\}$$

is a cover description for Q with $\text{bvars}_Q(\mathcal{A}_1) = \{x\}$. Although ψ_1 could be chosen as id (by adapting α_1 accordingly), we will see in Example 4.5, that this is not always desirable.

Now we can simply characterise rewritability of a query Q by the existence of a partition of body(Q) whose subsets have cover descriptions with views from $\mathcal{V}$.

Definition 4.4. Let Q be a query and $\mathcal{V}$ be a set of views. A collection $\mathcal{C} = C_1, \dots, C_m$ of cover descriptions $C_i = (\mathcal{A}_i, V_i, \alpha_i, \psi_i)$ for Q with $V_i \in \mathcal{V}$ is a *cover partition* for Q over $\mathcal{V}$ if the sets $\mathcal{A}_1, \dots, \mathcal{A}_m$ constitute a partition of body(Q).

We call a cover partition *consistent* if variables of any $\alpha_j(V_j)$ are in the range of any other α_i only if they also appear in $\text{bvars}_Q(\mathcal{A}_j)$. We note that, since each α_i is a view application, in a consistent cover partition, the applications obey quantified variable disjointness.[7]

Example 4.5 (Continuation of Example 4.3)**.** Let $C_1 = (\mathcal{A}_1, V_1, \alpha_1, \psi_1)$ be the cover description defined in Example 4.3. In addition, we consider the cover descriptions C_2 and C_3 with $C_i = (\mathcal{A}_i, V_i, \alpha_i, \psi_i)$ for $i \in \{2, 3\}$ where

$$\mathcal{A}_2 = \{R(x, y, z)\}, \qquad \mathcal{A}_3 = \{E(w), S(w, z)\},$$
$$\alpha_2 = \{x_2 \mapsto x, y_2 \mapsto y, z_2 \mapsto z, v_2 \mapsto v\}, \qquad \alpha_3 = \{w_3 \mapsto w, z_3 \mapsto z\},$$

and $\psi_2 = \psi_3 = \text{id}$. The cover descriptions C_1, C_2, C_3 constitute a cover partition for Q over $\{V_1, V_2, V_3\}$. It is, however, not consistent, since v is in the range of α_1 and α_2, but not in $\text{bvars}(\mathcal{A}_1)$. Replacing α_2 and ψ_2 by mappings $\hat{\alpha}_2$ and $\hat{\psi}_2$ with $\hat{\alpha}_2(v_2) = \hat{v}$ and $\hat{\psi}_2(\hat{v}) = v$ that agree with α_2 and ψ_2 on all other variables, respectively, yields a consistent cover partition. Note that there is no consistent cover partition with a cover description of the form $(\mathcal{A}_1, V_1, \alpha'_1, \text{id})$ because necessarily $\alpha'_1(w_1) = w$ would hold and thus w would be in the range of α'_1 and α_3.

A consistent cover partition $\mathcal{C}$ induces a query $Q_{\mathcal{C}}$ and an expansion $Q'_{\mathcal{C}}$ as follows. We note first that each variable in head(Q) occurs in at least one of the sets $\mathcal{A}_i$ and thus in some set $\text{bvars}_Q(\mathcal{A}_i)$. Therefore, Condition (2) of Definition 4.2 guarantees that each head variable of Q occurs in some set $\alpha_i(\text{head}(V_i))$ and thus a cover partition $\mathcal{C} = C_1, \dots, C_m$ with $C_i = (\mathcal{A}_i, V_i, \alpha_i, \psi_i)$ induces a query $Q_{\mathcal{C}}$ with

$$\text{head}(Q_{\mathcal{C}}) = \text{head}(Q) \text{ and } \text{body}(Q_{\mathcal{C}}) = \{\alpha_i(\text{head}(V_i)) \mid 1 \le i \le m\}.$$

Likewise, and thanks to quantified variable disjointness, it induces an expansion $Q'_{\mathcal{C}}$ with

$$\text{head}(Q'_{\mathcal{C}}) = \text{head}(Q) \text{ and } \text{body}(Q'_{\mathcal{C}}) = \{\alpha_i(\text{body}(V_i)) \mid 1 \le i \le m\}.$$

Next, we show that the existence of a cover partition indeed characterises rewritability.

Theorem 4.6. *Let Q be a minimal conjunctive query and $\mathcal{V}$ be a set of views. The following three statements are equivalent.*

(a) *Q is $\mathcal{V}$-rewritable.*
(b) *There is a cover partition $\mathcal{C}$ for Q over $\mathcal{V}$.*
(c) *There is a consistent cover partition $\mathcal{C}$ for Q over $\mathcal{V}$.*

---

[7]For the sake of contradiction, assume a quantified variable x and a variable y are mapped to the same variable z by two different view applications α_i and α_j, respectively. Then, due to consistency, z is in $\text{bvars}(\mathcal{A}_i)$ and, thus, in $\alpha_i(\text{head}(V_i))$ due to Definition 4.2(2). It follows that α_i unifies x with a head variable. But this is a contradiction to α_i being a view application.

If $\mathcal{C}$ is a consistent cover partition, then $Q_{\mathcal{C}}$ is a $\mathcal{V}$-rewriting of Q. If $\mathcal{C}$ is a cover partition, then there is a consistent cover partition with the same partition of body(Q) *as $\mathcal{C}$.*

Proof. We first show that (c) implies (a). To this end, let $\mathcal{C} = C_1, \ldots, C_m$ be a consistent cover partition for Q over $\mathcal{V}$, where, for each i, $C_i = (\mathcal{A}_i, V_i, \alpha_i, \psi_i)$.

We prove that the query $Q_{\mathcal{C}}$ induced by $\mathcal{C}$ is a $\mathcal{V}$-rewriting of Q: first, since $\mathcal{A}_1, \ldots, \mathcal{A}_m$ partition body(Q) and thanks to Condition (1), id is a homomorphism from Q into the expansion $Q'_{\mathcal{C}}$.

Therefore, it suffices to show that the union of the mappings $\psi_1, \ldots, \psi_m$ is a homomorphism h' from $Q'_{\mathcal{C}}$ into Q.

We first show that h' is well-defined: let us assume that a variable z occurs in $\alpha_i(V_i)$ and $\alpha_j(V_j)$ with $i \neq j$. Thanks to consistency, z is in bvars$(\mathcal{A}_i)$ and bvars$(\mathcal{A}_j)$ and therefore $\psi_i(z) = \psi_j(z)$ thanks to Condition (4).

That h' is a homomorphism follows easily, because each ψ_i is a body homomorphism by Condition (3) and h' is the identity on head(Q) by Condition (4).

Next, we show that (a) implies (b). Let us assume that Q has a $\mathcal{V}$-rewriting Q_R with an expansion Q_E and let the equivalence of Q and Q_E be witnessed by homomorphisms h from Q to Q_E and h' from Q_E to Q.

Since Q is minimal, we can assume, thanks to Lemma 4.7, that h is injective and h' is the inverse of h on the atoms of $h(\text{body}(Q))$. Since h is injective, we can further assume, without loss of generality, that h is the identity mapping on the variables in Q and that $h'(x) = x$ for every such variable.[8]

That is, we have head$(Q) =$ head$(Q_E) =$ head(Q_R) as well as body$(Q) \subseteq$ body(Q_E).

Let $\alpha_1, \ldots, \alpha_m$ be applications of views $V_1, \ldots, V_m$ that result in expansion Q_E, that is,

$$\text{body}(Q_E) = \bigcup_{i=1}^{m} \text{body}(\alpha_i(V_i)),$$

where the applications fulfil the quantified variable disjointness property. These applications induce a partition $\mathcal{A}_1, \ldots, \mathcal{A}_m$ of body(Q_E) in the following way: For every $i \in \{1, \ldots, m\}$ we define

$$\mathcal{A}_i = \{A \in \text{body}(Q) \mid i \text{ is minimal such that } A \in \alpha_i(\text{body}(V_i))\}.$$

Now, let $C_i = (\mathcal{A}_i, V_i, \alpha_i, h'_i)$, where h'_i is the restriction of h' to the variables in $\alpha_i(V_i)$, for every $i \in \{1, \ldots, m\}$. To show that $C_1, \ldots, C_m$ yield a cover partition, it only remains to show that each C_i is a cover description.

To this end, we show that all four conditions in Definition 4.2 are satisfied for C_i, for every $i \in \{1, \ldots, m\}$. Condition (1) holds by the definition of $\mathcal{A}_i$. Condition (3) is true because each h'_i is a restriction of the body homomorphism h'. Condition (4) follows since h' is the identity on all variables in body(Q) and thus also on *all* variables of $\mathcal{A}_i \subseteq$ body(Q).

Hence, it only remains to show that Condition (2) is satisfied. To this end, let x be an arbitrary bridge variable in some $\mathcal{A}_i$. If x occurs in a subset $\mathcal{A}_j$ where $j \neq i$, then it is a head variable in both $\alpha_i(V_i)$ and $\alpha_j(V_j)$ because of the quantified variable disjointness of $\alpha_1, \ldots, \alpha_m$. Otherwise, $\mathcal{A}_i$ is the only subset containing variable x, which thus has to be a head variable of Q in order to be a bridge variable in $\mathcal{A}_i$. Because of head$(Q) =$ head(Q_R),

[8]This can be easily achieved by renaming the variables of Q_R and Q_E appropriately.

it is then also a head variable of Q_R. This, in turn, implies that x is a head variable of $\alpha_i(V_i)$ because the quantified variables of $\alpha_i(V_i)$ do *not* occur in Q_R.

Finally, we show that (b) implies (c). Let thus $\mathcal{C} = C_1, \ldots, C_m$ be a cover partition for Q over $\mathcal{V}$, where, for each i, $C_i = (\mathcal{A}_i, V_i, \alpha_i, \psi_i)$. Let z be a variable that occurs in some $\alpha_i(V_i)$, but not in $\text{bvars}(\mathcal{A}_i)$, and also in some $\alpha_j(V_j)$ with $j \neq i$. Since $z \notin \text{bvars}(\mathcal{A}_i)$ we have $z \notin \text{vars}(\mathcal{A}_j)$. We define α'_j like α_j but, for some fresh variable z', we set $\alpha'_j(x) = z'$ whenever $\alpha_j(x) = z$. Accordingly, we define ψ'_j like ψ_j but with $\psi'_j(z') = \psi_j(z)$. It is easy to verify that $C'_j = (\mathcal{A}_j, V_j, \alpha'_j, \psi'_j)$ is a cover description, as well. By repeating this process, a consistent cover partition for Q over $\mathcal{V}$ can be obtained. □

The previous proof used the following lemma. It is similarly stated in the proof of Lemma 2 in [CGLP20a], see also the proof of Lemma 9 in the full version [CGLP20b] of [CGLP20a] for the proof idea. We give its short proof to keep the paper self-contained.

Lemma 4.7. *Let Q_1 be a minimal conjunctive query, Q_2 a conjunctive query equivalent to Q_1 and $h_1\colon Q_1 \to Q_2$ a homomorphism. Then, there is a homomorphism $h_2\colon Q_2 \to Q_1$ that is the inverse of h_1 on $h_1(\text{body}(Q_1))$.*

Proof. Since Q_1 and Q_2 are equivalent, there is a homomorphism $h'_2\colon Q_2 \to Q_1$.

The mapping $h'_2 \circ h_1$ is an automorphism[9] on Q_1, because Q_1 is minimal. Since the automorphisms of Q_1 constitute a group, there is some $k > 0$ such that $(h'_2 \circ h_1)^k$ is the identity on Q_1. We choose $h_2 = (h'_2 \circ h_1)^{k-1} \circ h'_2$. Clearly, h_2 is a homomorphism from Q_2 to Q_1 and h_2 is the inverse of h_1 on $h_1(Q)$. □

Remark 4.8. We note that the requirement in Theorem 4.6 for Q to be minimal is only seemingly a restriction in our setting, since we apply it only to acyclic queries. If Q is acyclic but not minimal, an equivalent minimal query Q' can be computed in polynomial time by iteratively removing atoms from its body [CM77, CR00]. Moreover, it is guaranteed that the minimal query Q' is also acyclic (we believe this to be folklore, it follows readily from more general results, cf. for instance [BPR17]).

The same is true for free-connex acyclic queries: every homomorphism from Q to Q' is also a homomorphism from $\text{body}(Q) \cup \{\text{head}(Q)\}$ to $\text{body}(Q') \cup \{\text{head}(Q')\}$ and vice versa, since the relation symbol of $\text{head}(Q) = \text{head}(Q')$ does not occur in $\text{body}(Q')$. Thus, Q' is minimal if and only if the Boolean query whose body is $\text{body}(Q') \cup \{\text{head}(Q')\}$ is minimal. Therefore, if $\text{body}(Q) \cup \{\text{head}(Q)\}$ is acyclic, so is $\text{body}(Q') \cup \{\text{head}(Q')\}$. In other words, if Q is free-connex acyclic, then Q' is free-connex acyclic as well.

For hierarchical and q-hierarchical queries the same holds: it is easy to see that removing atoms does not change the conditions in their respective definitions.

5. Towards Acyclic Rewritings

In this section, we turn our focus to the main topic of this paper: acyclic rewritings and the decision problem that asks whether such a rewriting exists. We study the complexity of $\text{Rewr}(\mathbb{V}, \mathbb{Q}, \mathsf{ACQ})$ for the case that $\mathbb{V}$ and $\mathbb{Q}$ are the class of conjunctive queries and for various subclasses. It will be helpful to analyse the case that $\mathbb{Q} = \mathsf{ACQ}$ and $\mathbb{V} = \mathsf{CQ}$ first.

[9] We note that compositions of homomorphisms are applied from right to left.

5.1. **A Characterisation of Acyclic Rewritability for Acyclic Queries.** The following example illustrates that, even if an acyclic rewriting exists, the canonical rewriting need not be acyclic. Furthermore, it may be that each "sub-rewriting" of the canonical candidate is cyclic or not a rewriting, and thus none of them is an acyclic rewriting.

Example 5.1. Consider the query Q given by the rule

$$H() \leftarrow R_1(x), R_2(y), S(x,z), T_1(z), T_2(y)$$

and the following views $\mathcal{V} = \{V_1, V_2, V_3\}$.

$$V_1(u_1,v_1) \leftarrow R_1(u_1), R_2(v_1) \qquad V_2(u_2,v_2) \leftarrow S(u_2,v_2) \qquad V_3(u_3,v_3) \leftarrow T_1(u_3), T_2(v_3)$$

The canonical candidate $Q_R = H() \leftarrow V_1(x,y), V_2(x,z), V_3(z,y)$ is a $\mathcal{V}$-rewriting, but it is *cyclic*. Each query whose body is a proper subset of the body of Q_R is not a $\mathcal{V}$-rewriting for Q. However, an *acyclic* $\mathcal{V}$-rewriting for Q exists. One such rewriting is the query Q'_R.

$$H() \leftarrow V_1(x,y), V_2(x,z), V_3(z,y'), V_3(z',y)$$

Even though Example 5.1 suggests that general and acyclic rewritings can be seemingly unrelated, it turns out that there is a close connection between them. In fact, we will show next that an acyclic $\mathcal{V}$-rewriting exists if and only if an arbitrary $\mathcal{V}$-rewriting exists. Furthermore, from an arbitrary rewriting an acyclic rewriting can always be constructed.

Towards a proof, we study the relationship between a query Q and the view applications that can occur in any rewriting of Q more closely, to determine the circumstances under which a decomposition of view atoms is possible.

Example 5.2 (Continuation of Example 5.1)**.** We first have a closer look at the V_3-atoms in the rewritings of the previous example. The two V_3-atoms in Q'_R can be understood as a (sub-)query with body $\{V_3(z,y'), V_3(z',y)\}$ and head variables z, y, whereas the V_3-atom in Q_R can be understood as a (sub-)query with body $\{V(z,y)\}$ and head variables z, y. The expansions of these two (sub-)queries are equivalent. That is, the acyclic rewriting can be obtained from the canonical rewriting by replacing an atom by a set of atoms that is equivalent with respect to view expansions.[10]

In Example 5.1 the acyclic rewriting was obtained from the canonical rewriting by replacing a view atom by a set that contained one view atom for each connected component of (the hypergraph induced by) V_3. The following example shows that the required modifications can be more involved.

Example 5.3. Consider the query Q given by the rule

$$H(x,y,z) \leftarrow R(x,y,z), E_1(x), E_2(y), E_3(w), S(w,z)$$

and the following views.

$$\begin{aligned} V_1(x_1,y_1,w_1) &\leftarrow R(x_1,y_1,v_1), E_1(x_1), E_3(w_1), S(w_1,v_1) \\ V_2(x_2,y_2,z_2) &\leftarrow R(x_2,y_2,z_2), E_2(y_2) \\ V_3(w_3,z_3) &\leftarrow S(w_3,z_3), E_3(w_3) \end{aligned}$$

The canonical candidate $H(x,y,z) \leftarrow V_1(x,y,w), V_2(x,y,z), V_3(w,z)$ is a cyclic $\{V_1,V_2,V_3\}$-rewriting of Q. In contrast to Example 5.1, all view bodies are connected. Therefore, replacing a view atom by a set of representatives of connected components of some views

[10]We note that one could also replace the V_1-atom in Q_R.

does not yield an acyclic rewriting. However, there is an acyclic $\{V_1, V_2, V_3\}$-rewriting of Q, namely $H(x, y, z) \leftarrow V_1(x, y', w'), V_2(x, y, z), V_3(w, z)$.

Now we turn to the main result of this section.

Theorem 5.4. *Let Q be a conjunctive query and $\mathcal{V}$ be a set of views.*

(a) *If Q is acyclic and $\mathcal{V}$-rewritable, then it has is an acyclic $\mathcal{V}$-rewriting.*
(b) *If Q is free-connex acyclic and $\mathcal{V}$-rewritable, then it has a free-connex acyclic $\mathcal{V}$-rewriting.*

Proof. Since Q is acyclic, we can assume that it is minimal (cf. Remark 4.8). Moreover, it has a join tree J_Q, and, thanks to Theorem 4.6, since Q is $\mathcal{V}$-rewritable, there is a consistent cover partition $\mathcal{C} = C_1, \ldots, C_m$ for Q over $\mathcal{V}$ and the query $Q_{\mathcal{C}}$ is a $\mathcal{V}$-rewriting of Q. For each i, let $C_i = (\mathcal{A}_i, V_i, \alpha_i, \psi_i)$.

We show first that $Q_{\mathcal{C}}$ is acyclic if each set $\mathcal{A}_i$ is connected in J_Q. Afterwards we show that a consistent cover partition with that property can always be constructed from $\mathcal{C}$.

To this end, we show how to construct a join tree J for $Q_{\mathcal{C}}$: we first cluster, for each j, the nodes for $\mathcal{A}_j$ in J_Q together into one node that is labelled by $\alpha_j(\text{head}(V_j))$. Since the $\mathcal{A}_j$ are connected in J_Q, the resulting graph J is a tree.

To verify that J is a join tree, let us consider two nodes u, v of J labelled by $\alpha_j(\text{head}(V_j))$ and $\alpha_k(\text{head}(V_k))$, respectively, and x be a variable that appears in $\alpha_j(\text{head}(V_j))$ *and* $\alpha_k(\text{head}(V_k))$. Thanks to consistency of $\mathcal{C}$, variable x appears in $\text{bvars}_Q(\mathcal{A}_j)$ and in $\text{bvars}_Q(\mathcal{A}_k)$. It follows that x appears in two atoms $A \in \mathcal{A}_j$ and $A' \in \mathcal{A}_k$. Moreover, since J_Q is a join tree, x appears in every node on the (shortest) path from A to A' in J_Q. Again thanks to consistency, x is in $\text{bvars}_Q(\mathcal{A}_\ell)$ for every $\alpha_\ell(\text{head}(V_\ell))$ along the corresponding contracted path in J from u to v. In particular, x appears in all sets $\alpha_\ell(\text{head}(V_\ell))$ on the path from u to v. Thus, u and v are x-connected. We conclude that J is a join tree for $Q_{\mathcal{C}}$. Hence, $Q_{\mathcal{C}}$ is acyclic.

If Q is also free-connex acyclic, there is a join tree J_Q^+ for $\text{body}(Q) \cup \{\text{head}(Q)\}$. As we show later, we can assume that the sets $\mathcal{A}_i$ of the cover partition $\mathcal{C}$ are also connected in J_Q^+. Clustering the nodes[11] of J_Q^+ analogously as for J_Q yields a join tree for $\text{body}(Q_{\mathcal{C}}) \cup \{\text{head}(Q_{\mathcal{C}})\}$ since $\text{head}(Q_{\mathcal{C}}) = \text{head}(Q)$ and all head variables in a set $\mathcal{A}_i$ also occur in the new label $\alpha_i(\text{head}(V_i))$ thanks to Condition (2) of Definition 4.2.

We now show how, from a consistent cover partition $\mathcal{C}$, we can construct a consistent cover partition $\mathcal{C}' = C'_1, \ldots, C'_m$ such that each set $\mathcal{A}'_i$ is connected. To this end, let us assume that some set $\mathcal{A}_j$ is not connected in J_Q. Let $\mathcal{B}_j \subseteq \mathcal{A}_j$ be a maximal connected subset and let $\mathcal{A}'_j = \mathcal{A}_j \setminus \mathcal{B}_j$. We observe that each variable x that appears in $\mathcal{B}_j$ and $\mathcal{A}'_j$, is also in $\text{bvars}_Q(\mathcal{A}_j)$, since x has to occur in at least one other atom in J_Q (otherwise, $\mathcal{B}_j$ and $\mathcal{A}'_j$ would be connected in J_Q). Thus, $\text{bvars}_Q(\mathcal{B}_j) \subseteq \text{bvars}_Q(\mathcal{A}_j)$. We conclude that $C_{\mathcal{B}_j} = (\mathcal{B}_j, V_j, \alpha_j, \psi_j)$ is a cover description. Likewise, $C'_j = (\mathcal{A}'_j, V_j, \alpha_j, \psi_j)$ is a cover description. By repeated application of this modification step, we eventually obtain a cover partition $\mathcal{C}' = C'_1, \ldots, C'_m$ in which each set $\mathcal{A}'_i$ is connected. If Q is free-connex, then $\mathcal{C}'$ can be further refined as described above but w.r.t. J_Q^+ instead of J_Q. Refining the cover partition iteratively w.r.t. J_Q and J_Q^+ yields a cover partition $\mathcal{C}'' = C''_1, \ldots, C''_p$ in which each set $\mathcal{A}''_i$ is connected in J_Q and J_Q^+. The number of iterations is bounded by the number of atoms of Q.

Thanks to Theorem 4.6, $\mathcal{C}''$ can be turned into a consistent cover partition (and the required renamings of variables do not affect the connectedness). □

[11] The node labelled $\text{head}(Q)$ is not clustered with any other node, since it does not occur in any $\mathcal{A}_i$.

Theorem 5.4 delivers good news as well as bad news. The good news is that, since the proofs of Theorem 5.4 and Theorem 4.6 are constructive, we altogether have a procedure to construct an acyclic rewriting from an arbitrary rewriting Q_R if the query Q is acyclic. Furthermore, if a homomorphism from an expansion of Q_R to Q can be computed in polynomial time, the overall procedure can be performed in polynomial time. In particular, we get the following collary.

Corollary 5.5. *For every k,* $\textsc{Rewr}(\mathsf{ACQ}^k, \mathsf{ACQ}, \mathsf{ACQ})$ *is in polynomial time, and an acyclic rewriting can be computed in polynomial time (if it exists).*

That the decision problem is in polynomial time follows immediately from Proposition 3.10 and Theorem 5.4. Moreover, the canonical candidate for an acyclic query Q and a set $\mathcal{V}$ of acyclic views with bounded arity can be computed in polynomial time, since the size of $\mathcal{V}(\mathsf{canon}(Q))$ is bounded by a polynomial and query evaluation for acyclic queries is in polynomial time [Yan81, AHV95]. In case the canonical candidate is a rewriting, valuations witnessing $\mathcal{V}(\mathsf{canon}(Q))$ can be computed in polynomial time [KS06] and combined into a homomorphism from an expansion of the rewriting to Q. Thus, an acyclic rewriting can then be efficiently constructed by the above procedure.

However, this leaves open the complexity of $\textsc{Rewr}(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{ACQ})$ (as well as that of $\textsc{Rewr}(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{CQ})$), and of their restrictions to schemas of bounded arity. The bad news is that, since $\textsc{Rewr}(\mathbb{V}, \mathsf{ACQ}, \mathsf{CQ})$ and $\textsc{Rewr}(\mathbb{V}, \mathsf{ACQ}, \mathsf{ACQ})$ are basically the same problem, lower bounds on $\textsc{Rewr}(\mathbb{V}, \mathsf{ACQ}, \mathsf{CQ})$ transfer to $\textsc{Rewr}(\mathbb{V}, \mathsf{ACQ}, \mathsf{ACQ})$. In fact, we get the following, due to NP-hardness of $\textsc{Rewr}^k(\mathsf{CQ}, \mathsf{ACQ}, \mathsf{CQ})$ in the general case (Proposition 3.11).[12]

Corollary 5.6. *For every $k \geq 3$, the problem* $\textsc{Rewr}^k(\mathsf{CQ}, \mathsf{ACQ}, \mathsf{ACQ})$ *and, therefore, also* $\textsc{Rewr}^k(\mathsf{CQ}, \mathsf{CQ}, \mathsf{ACQ})$ *is* NP*-hard.*

In the next part of this section, we resolve the complexity of $\textsc{Rewr}(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{CQ})$ and $\textsc{Rewr}(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{ACQ})$ and their restrictions to schemas of bounded arity.

5.2. **The Complexity of Acyclic Rewritability for Acyclic Queries.** It may be tempting to assume that, since acyclic queries are so well-behaved in general, it should be tractable to decide whether for an acyclic query and a set of acyclic views there exists an acyclic rewriting. However, as we show next, this is (probably) not the case, and this surprising finding even holds for the even better behaved hierarchical queries as well.

Theorem 5.7. *The problems* $\textsc{Rewr}^k(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{CQ})$ *and* $\textsc{Rewr}^k(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{ACQ})$*, as well as* $\textsc{Rewr}^k(\mathsf{HCQ}, \mathsf{HCQ}, \mathsf{CQ})$ *are* NP*-complete, for $k \geq 3$.*[13] *The lower bounds even hold for instances with a single view.*

Of course, Theorem 5.7 immediately implies NP-hardness of $\textsc{Rewr}^k(\mathbb{V}, \mathbb{Q}, \mathsf{ACQ})$, for all pairs $\mathbb{V}, \mathbb{Q}$ of classes with $\mathsf{ACQ} \subseteq \mathbb{V} \subseteq \mathsf{CQ}$ and $\mathsf{ACQ} \subseteq \mathbb{Q} \subseteq \mathsf{CQ}$.

We will see in Section 7 that Theorem 5.4 has an analogue for hierarchical queries from which it can be concluded that deciding the existence of a hierarchical rewriting, given a hierarchical query and hierarchical views, is still NP-complete (cf. Corollary 7.5).

[12] We note that Corollary 5.6 also follows from the proof of Proposition 3.11, since the canonical candidate constructed in that proof is trivially acyclic. Indeed, NP-hardness of $\textsc{Rewr}(\mathsf{ACQ}, \mathsf{CQ}, \mathsf{ACQ})$ is also implied.

[13] These results hold for schemas of unbounded arity, as well.

Theorem 5.7 will easily follow from NP-hardness of a seemingly simpler problem. From the characterisation in Theorem 4.6, we already know that deciding the existence of a rewriting is the same as deciding the existence of a cover partition. We show next that it is even NP-hard to decide the existence of a *cover description*, given a query, a set of atoms, and a single view.

Definition 5.8. Let $\mathbb{V} \subseteq \mathsf{CQ}$ and $\mathbb{Q} \subseteq \mathsf{CQ}$ be classes of conjunctive queries. The *cover description problem* for $\mathbb{V}$ and $\mathbb{Q}$, denoted $\textsc{CovDesc}(\mathbb{V}, \mathbb{Q})$ asks, upon input of a query $Q \in \mathbb{Q}$, a subset $\mathcal{A} \subseteq \text{body}(Q)$ and a view $V \in \mathbb{V}$, whether mappings α and ψ exist such that $(\mathcal{A}, V, \alpha, \psi)$ is a cover description.

Theorem 5.9. $\textsc{CovDesc}(\mathsf{ACQ}, \mathsf{ACQ})$ *is* NP*-hard. Indeed, even* $\textsc{CovDesc}(\mathsf{HCQ}, \mathsf{HCQ})$ *is* NP*-hard and even if the input is restricted to* $\mathcal{A} = \text{body}(Q)$*.*

Proof. We reduce problem 3SAT to $\textsc{CovDesc}(\mathsf{HCQ}, \mathsf{HCQ})$. This lower bound directly translates to $\textsc{CovDesc}(\mathsf{ACQ}, \mathsf{ACQ})$ since every hierarchical query is acyclic. For a formula f in 3CNF, we describe how a query Q and a view V can be derived in polynomial time such that both Q and V are hierarchical and such that f is satisfiable if and only if there are mappings α and ψ such that $(\text{body}(Q), V, \alpha, \psi)$ is a cover description for Q. In the following we will use the term *proposition* for propositional variables of a formula in order to easily distinguish them from variables occurring in queries and views.

Construction. Let $f = f_1 \wedge \cdots \wedge f_k$ be a propositional formula in 3CNF over propositions $x_1, \ldots, x_n$ in clauses $f_1, \ldots, f_k$, where $f_j = (\ell_{j,1} \vee \ell_{j,2} \vee \ell_{j,3})$ for each $j \in \{1, \ldots, k\}$.

We start with the construction of query Q. This query is Boolean, with head $H()$, and uses only three variables w_0, w_1 and u, where the first two are intended to represent the truth values false and true. The structure of Q is depicted in Figure 1(a). The body of Q is defined as the union

$$\{\texttt{Form}(w_0, w_1, u)\} \uplus \mathcal{C}_0 \uplus \mathcal{C}_1 \uplus \mathcal{N}_1 \uplus \cdots \uplus \mathcal{N}_n$$

of the following sets of atoms.

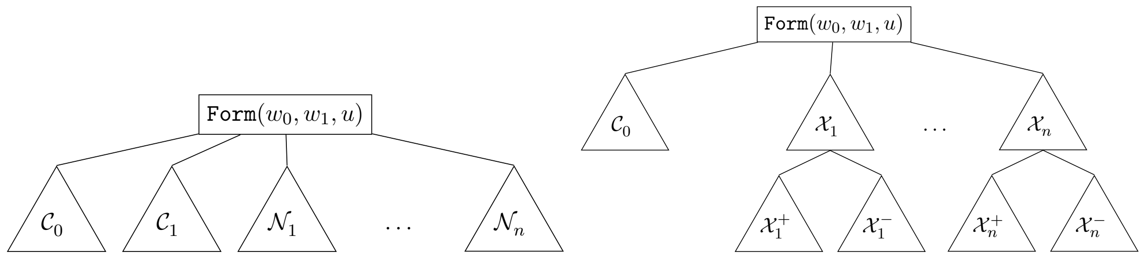
- Set $\mathcal{C}_0$ contains an atom $\texttt{C}_j(w_0, w_1, u)$ for each clause f_j in f.

  Intuitively, these atoms represent *unsatisfied* clauses as opposed to the next atoms that represent *satisfied* clauses. Note that these atoms differ in the order of variables w_0 and w_1 only.
- Set $\mathcal{C}_1$ contains an atom $\texttt{C}_j(w_1, w_0, u)$ for each clause f_j in f.
- There is a set $\mathcal{N}_i$ for each proposition x_i in formula f. The sets differ only in the name of the relation they address and contain two atoms $\texttt{Neg}_i(w_0, w_1, u)$ and $\texttt{Neg}_i(w_1, w_0, u)$ each.

Query Q is hierarchical since every atom in each of the sets above refers to all three variables w_0, w_1 and u. Thus, query Q is acyclic in particular.

We now proceed with the construction of view V, which refers to the same relations as query Q. Like query Q, the view uses variables w_0, w_1 and u but also additional variables for the propositions $x_1, \ldots, x_n$ in formula f. For each proposition x_i, there are two variables x_i and $\bar{x}_i$, intended to represent the positive and negated literal over the proposition, respectively. With the exception of variable u, all other variables are in the head $H(w_0, w_1, x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n)$ of view V. The body of V, whose structure is depicted in Figure 1(b), is defined as the union

$$\{\texttt{Form}(w_0, w_1, u)\} \uplus \mathcal{C}_0 \uplus \left(\mathcal{X}_1 \uplus \mathcal{X}_1^+ \uplus \mathcal{X}_1^-\right) \uplus \cdots \uplus \left(\mathcal{X}_n \uplus \mathcal{X}_n^+ \uplus \mathcal{X}_n^-\right)$$



(a) Structure of query $\boldsymbol{Q}$, induced by formula $\boldsymbol{f}$. (b) Structure of view $\boldsymbol{V}$, induced by formula $\boldsymbol{f}$.

FIGURE 1. Structure of the query $\boldsymbol{Q}$ and the view $\boldsymbol{V}$, induced by formula $\boldsymbol{f}$.

of sets of atoms, where the first two sets $\{\texttt{Form}(w_0, w_1, u)\}$ and $\mathcal{C}_0$ are the same as above. The bodies of query Q and view V thus differ in the sets $\mathcal{C}_1$ and $\mathcal{N}_1, \ldots, \mathcal{N}_n$ of atoms on the one hand as well as $\mathcal{X}_1, \ldots, \mathcal{X}_n$ and $\mathcal{X}_1^+, \ldots, \mathcal{X}_n^+$ and $\mathcal{X}_1^-, \ldots, \mathcal{X}_n^-$ on the other hand. Sets $\mathcal{N}_1, \ldots, \mathcal{N}_n$ and $\mathcal{X}_1, \ldots, \mathcal{X}_n$ are intended to guide the assignment of propositions, while sets $\mathcal{X}_1^+, \ldots, \mathcal{X}_n^+$ and $\mathcal{X}_1^-, \ldots, \mathcal{X}_n^-$ represent the occurrences of literals in the clauses of formula f, and set $\mathcal{C}_1$ represents the aim to satisfy all of them. The new sets are defined as follows.

- For each proposition x_i, set $\mathcal{X}_i$ contains atoms $\texttt{Neg}_i(x_i, \bar{x}_i, u)$ and $\texttt{Neg}_i(\bar{x}_i, x_i, u)$, intended to enforce the mapping of $(x_i, \bar{x}_i)$ to either (w_0, w_1) or (w_1, w_0), that is, to "complementary truth values".
- For each positive literal x_i and each clause f_j that contains literal x_i, set $\mathcal{X}_i^+$ contains an atom $\texttt{C}_j(x_i, \bar{x}_i, u)$.
- For each negated literal $\bar{x}_i$ and each clause f_j that contains literal $\neg x_i$, set $\mathcal{X}_i^-$ contains an atom $\texttt{C}_j(\bar{x}_i, x_i, u)$.

Thus defined, the view is also hierarchical as shown in the following. Every atom in every set contains variable u. Therefore, it follows that $\text{atoms}(x) \subseteq \text{atoms}(u)$ for all $x \in \text{vars}(V)$. Furthermore, all atoms in $\{\texttt{Form}(w_0, w_1, u)\}$ and $\mathcal{C}_0$ additionally contain both variables w_0 and w_1 and they are the only atoms to contain these variables. Thus, we have that $\text{atoms}(w_0) = \text{atoms}(w_1)$ and $\text{atoms}(y) \cap \text{atoms}(w_0) = \emptyset$ and $\text{atoms}(y) \cap \text{atoms}(w_1) = \emptyset$ for all $y \in \text{vars}(V) \setminus \{u, w_0, w_1\}$. Similarly, for each $i \in \{1, \ldots, n\}$, all atoms in $\mathcal{X}_i \cup \mathcal{X}_i^+ \cup \mathcal{X}_i^-$ contain both variables x_i and $\bar{x}_i$, which do not occur in any other atom. It follows that $\text{atoms}(x_i) = \text{atoms}(\bar{x}_i)$ and $\text{atoms}(\bar{x}_i) \cap \text{atoms}(y) = \emptyset$ and $\text{atoms}(x_i) \cap \text{atoms}(y) = \emptyset$ for all $y \in \text{vars}(V) \setminus \{x_i, \bar{x}_i, u\}$ and $i \in \{1, \ldots, n\}$. Therefore, we obtain that each pair of variables in V satisfies at least one of the conditions in Definition 2.1 and thus V is hierarchical—and acyclic in particular.

For the correctness argument, however, the structure of the join tree of view V, depicted in Figure 1(b) is probably more helpful. We close the description of the construction by remarking that the query and view can be computed in polynomial time for any given propositional formula in 3CNF.

Correctness. We prove now that formula f is satisfiable if and only if there are mappings α and ψ such that $(\text{body}(Q), V, \alpha, \psi)$ is a cover description for Q.

First, we show that if formula f is satisfiable, then there are mappings α and ψ such that $(\text{body}(Q), V, \alpha, \psi)$ *is a cover description for Q.*

To this end, let us assume that f is satisfiable and that this is witnessed by a satisfying truth assignment β. From this assignment for propositions $x_1, \dots, x_n$, we derive the application α of V. Let α be the mapping that is the identity on variables w_0, w_1 and u and behaves as follows on the literal variables, for every $i \in \{1, \dots, n\}$.

$$\alpha(x_i, \bar{x}_i) = \begin{cases} (w_0, w_1) & \text{if } \beta(x_i) = 0 \\ (w_1, w_0) & \text{if } \beta(x_i) = 1 \end{cases}$$

We choose mapping ψ to be the identity on the variables of $\alpha(V)$. Then $(\text{body}(Q), V, \alpha, \psi)$ is a cover description for Q because application α establishes the following relationships between the atoms of view V (on the left-hand side) and the atoms of query Q (on the right-hand side).

(i) $\alpha(\mathcal{C}_0) = \mathcal{C}_0$
(ii) $\alpha(\mathcal{X}_1) = \mathcal{N}_1, \dots, \alpha(\mathcal{X}_n) = \mathcal{N}_n$
(iii) $\alpha(\mathcal{X}_1^+ \cup \mathcal{X}_1^- \cup \cdots \cup \mathcal{X}_n^+ \cup \mathcal{X}_n^-) \subseteq \mathcal{C}_0 \cup \mathcal{C}_1$
(iv) $\alpha(\mathcal{X}_1^+ \cup \mathcal{X}_1^- \cup \cdots \cup \mathcal{X}_n^+ \cup \mathcal{X}_n^-) \supseteq \mathcal{C}_1$

Property (i) is rather obvious since the application α behaves like the identity on the variables w_0, w_1 and u in $\mathcal{C}_0$. Next, let us consider the set $\mathcal{X}_i = \{\texttt{Neg}_i(w_0, w_1, u), \texttt{Neg}_i(w_1, w_0, u)\}$ for an arbitrary $i \in \{1, \dots, n\}$. By definition, application α maps $(x_i, \bar{x}_i)$ either to (w_0, w_1) or (w_1, w_0). In each case, we get

$$\alpha(\mathcal{X}_i) = \{\texttt{Neg}_i(w_0, w_1, u), \texttt{Neg}_i(w_1, w_0, u)\} = \mathcal{N}_i.$$

Hence, Property (ii) holds too. Again by definition of α, the atoms in $\mathcal{X}_1^+ \cup \cdots \cup \mathcal{X}_n^+$, which are of the form $\texttt{C}_j(x_i, \bar{x}_i, u)$, are mapped to $\texttt{C}_j(w_0, w_1, u)$ or $\texttt{C}_j(w_1, w_0, u)$, contained in $\mathcal{C}_0 \cup \mathcal{C}_1$. An analogous argument holds for the atoms in $\mathcal{X}_1^- \cup \cdots \cup \mathcal{X}_n^-$. Thus, Property (iii) is also satisfied. Finally, we show that Property (iv) holds. To this end, let $\texttt{C}_j(w_1, w_0, u)$ be an arbitrary atom in $\mathcal{C}_1$. By our assumption, clause f_j in formula f is satisfied by truth assignment β. Hence, there is a literal $\ell_{j,h}$ in clause f_j such that $\beta \models \ell_{j,h}$. Let us assume $\ell_{j,h} = x_i$ for some $i \in \{1, \dots, n\}$; the argument for a negated literal is analogous. This assumption implies $\alpha(x_i) = w_1$ by the definition of α, which depends on β. Since literal x_i is a literal in clause f_j, we have $\texttt{C}_j(x_i, \bar{x}_i, u) \in \mathcal{X}_i^+$ and thus $\alpha(\mathcal{X}_i^+) \ni \texttt{C}_j(w_1, w_0, u)$.

Since Properties (i) – (iv) consider all body atoms of query Q and view V, they imply that the bodies are identical: $\text{body}(Q) = \alpha(\text{body}(V))$. For $\psi = \text{id}$, this trivially implies $\text{body}(Q) \subseteq \alpha(\text{body}(V))$ and $\psi(\alpha(\text{body}(V))) \subseteq \text{body}(Q)$, namely Conditions (1) and (3) in Definition 4.2. Furthermore, since query Q is Boolean, set $\mathcal{A} = \text{body}(Q)$ has *no* bridge variables. Thus, Conditions (4) and (2) are trivially satisfied.

Therefore, $(\mathcal{A}, V, \alpha, \psi)$ is a cover description for Q.

Now, we show that formula f is satisfiable if there are mappings α and ψ such that $(\text{body}(Q), V, \alpha, \psi)$ is a cover description for Q.

Consider the set $\mathcal{B} = \alpha(\text{body}(V))$ of atoms. From Definition 4.2, we know that $\text{body}(Q) \subseteq \mathcal{B}$ and $\psi(\mathcal{B}) \subseteq \text{body}(Q)$ hold. Since both sets body(Q) and $\mathcal{B}$ contain only one Form-atom $B = \texttt{Form}(w_0, w_1, u)$, we can conlude $\alpha(B) = B$ by Condition (1) and $\psi(\alpha(B)) = \psi(B) = B$ by Condition (3). Therefore, each mapping α and ψ is the identity on variables w_0, w_1 and u. Hence, application α induces unambiguously a truth assignment β that maps proposition $x_i \mapsto 1$ if $\alpha(x_i) = w_1$; and maps $x_i \mapsto 0$ otherwise.

It suffices to show that β satisfies formula f. Let f_j be an arbitrary clause of the formula. From $\text{body}(Q) \subseteq \alpha(\text{body}(V))$, we know that atom $\texttt{C}_j(w_1, w_0, u)$ is in $\alpha(\text{body}(V))$ because

$\mathtt{C}_j(w_1, w_0, u)$ is in $\mathcal{C}_1$ and, therefore, in body(Q). Since body(V) does not contain atoms from $\mathcal{C}_1$, it has to contain a different $\mathtt{C}_j$-atom which is mapped to $\mathtt{C}_j(w_1, w_0, u)$ by α. This atom cannot be $\mathtt{C}_j(w_0, w_1, u)$ from $\mathcal{C}_0$ because α is the identity on w_0, w_1, and u. Hence, there has to be an atom $\mathtt{C}_j(x_i, \bar{x}_i, u) \in \mathcal{X}_i^+$ or $\mathtt{C}_j(\bar{x}_i, x_i, u) \in \mathcal{X}_i^-$ for some $i \in \{1, \dots, n\}$ in body(V) that is mapped to $\mathtt{C}_j(w_1, w_0, u)$ by α. In the case that an atom $\mathtt{C}_j(x_i, \bar{x}_i, u) \in \mathcal{X}_i^+$ is mapped to $\mathtt{C}_j(w_1, w_0, u)$, it follows by construction that the literal x_i occurs in clause f_j. Furthermore, from $\alpha(\mathtt{C}_j(x_i, \bar{x}_i, u)) = \mathtt{C}_j(w_1, w_0, u)$, we can infer $\alpha(x_i) = w_1$. Hence, truth assignment β satisfies literal x_i in clause f_j because $\beta(x_i) = 1$ if $\alpha(x_i) = w_1$. The other case is analogous.

Thus, truth assignment β satisfies every clause of formula f and formula f is satisfiable if a rewriting of query Q in terms of view V exists.

This concludes our proof. □

The proof of Theorem 5.7 can now be stated easily.

Proof of Theorem 5.7. The upper bound follows from Theorem 3.2.

For the lower bound, we show that the reduction of Theorem 5.9 can be adapted to a reduction from 3SAT to $\textsc{Rewr}^k(\mathsf{HCQ}, \mathsf{HCQ}, \mathsf{CQ})$. To this end, we show that the query Q constructed in the proof of Theorem 5.9 has a $\{V\}$-rewriting *if* the formula f is satisfiable.

If the formula f is satisfiable then there are mappings α and ψ such that $C = (\text{body}(Q), V, \alpha, \psi)$ is a cover description. Due to Theorem 4.6 it follows that Q is $\{V\}$-rewritable, since C trivially constitutes a cover partition.

If Q is $\{V\}$-rewritable then there is a cover partition $\mathcal{C}$ for Q over $\{V\}$ due to Theorem 4.6. We will show that $\mathcal{C}$ consists of a single cover description C. Since C then necessarily has the shape $(\text{body}(Q), V, \alpha', \psi')$, this implies that the formula f is satisfiable.

Suppose for the sake of contradiction that $\mathcal{C}$ consists of at least two cover descriptions. Let C_1 be a cover description from $\mathcal{C}$. Since V is the only view and C_1 is not the only cover description in $\mathcal{C}$, C_1 has the shape $C_1 = (\mathcal{A}_1, V, \alpha_1, \psi_1)$ with $\mathcal{A}_1 \subsetneq \text{body}(Q)$. We observe that the variable u is a bridge variable of $\mathcal{A}_1$ since it occurs in every atom of Q, and, thus, in $\mathcal{A}_1$ and outside $\mathcal{A}_1$. Since u is always the last variable in every atom in Q and V, it follows from Definition 4.2(1) that $\alpha_1(u) = u$. But then $\text{bvars}_Q(\mathcal{A}_1) \not\subseteq \alpha_1(\text{vars}(\text{head}(V)))$, since u does not occur in the head of V and α_1 does not unify quantified variables. This contradicts Definition 4.2(2), and, thus, C_1 being a cover description. Therefore, $\mathcal{C}$ consists of a single cover description.

To conclude, we have established that Q is $\{V\}$-rewritable if and only of formula f is satisfiable. Hence, the NP-hardness shown for $\textsc{CovDesc}(\mathsf{HCQ}, \mathsf{HCQ})$ also holds for $\textsc{Rewr}^k(\mathsf{HCQ}, \mathsf{HCQ}, \mathsf{CQ})$. It trivially transfers to $\textsc{Rewr}^k(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{CQ})$ and by Theorem 5.4 also to $\textsc{Rewr}^k(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{ACQ})$. □

The fact that, for Theorem 5.7, a single view application is sufficient if there is any rewriting, allows to draw yet another conclusion for a related problem, defined next.

Definition 5.10. Given a query class $\mathbb{Q} \subseteq \mathsf{CQ}$, the *select-full-project equivalence* problem for $\mathbb{Q}$, denoted $\textsc{SelProjEquiv}(\mathbb{Q})$, asks, upon input of a Boolean query $Q \in \mathbb{Q}$ and a query $Q' \in \mathbb{Q}$ whether there is a Boolean query Q'' which is equivalent to Q and whose body can be obtained from the body of Q' by unifying head variables of Q' in its body.[14]

[14] The problem is called the select-full-project equivalence problem because the query Q'' can, if it exists, be expressed in the relational algebra by applying select-operators and a (full) project-operator to an expression for Q'.

Example 5.11. Consider the Boolean query Q defined by

$$H() \leftarrow R(x), S(y,x,y), T(y,z), T(z,x)$$

and the query Q' defined by

$$H'(y,z,u,v) \leftarrow R(x), S(y,x,v), S(y,x,w), T(y,z), T(u,x).$$

The body of the Boolean query Q'' given by

$$H() \leftarrow R(x), S(y,x,y), S(y,x,w), T(y,z), T(z,x)$$

can be obtained by unifying the head variables y and v as well as z and u in body(Q'). The query Q'' is equivalent to Q. Thus, $(Q, Q') \in$ SelProjEquiv(CQ).

Interpreting the query Q' in Definition 5.10 as a view, the unification of head variables can be realised by an application α which does not unnecessarily rename variables. Whether the desired Boolean query Q'' exists then boils down to whether there is a $\{Q'\}$-rewriting for Q whose body consists of a single view atom, i.e. $\alpha(\text{head}(Q'))$.

Example 5.12 (Continuation of Example 5.11)**.** Recall that the body of the Boolean query Q'' in Example 5.11 is obtained from body(Q') by unifying the head variables y and v as well as z and u. Let α be the application which maps v to y, u to z, and is the identity on every other variable. The Boolean query with body $\{\alpha(\text{head}(Q'))\} = \{H'(y,z,z,y)\}$ is a $\{Q'\}$-rewriting of Q. In fact, its expansion is Q'' which is equivalent to Q.

Due to the relationship between the select-full-project equivalence problem and rewritability explained above and Theorem 5.9 the following holds.

Corollary 5.13. SelProjEquiv(ACQ) *and* SelProjEquiv(HCQ) *are* NP*-hard.*

Proof. The proof is essentially the same as for Theorem 5.7.

Let Q and V be the Boolean query and the view constructed in the proof for Theorem 5.7, respectively. If $(\text{body}(Q), V, \alpha, \psi)$, for some mappings α and ψ, is a cover description for Q, then the Boolean query with body $\alpha(\text{head}(V))$ is a rewriting of Q since the cover description constitutes a consistent cover partition for Q (a cover partition consisting of a single cover description is consistent by definition).

Thus, Q is equivalent to the Boolean query Q'' whose body is obtained from body(V) by unifying exactly the (head) variables in body(V) the application α unifies. In other words, Q'' equals $\alpha(V)$ up to renaming variables.

Conversely, if Q is equivalent to a Boolean query Q'' whose body can be obtained from body(V) by unifying head variables of V in body(V), then there is an application α with $\alpha(\text{body}(V)) = \text{body}(Q'')$ and, thus, Q'' is the expansion of a V-rewriting, namely the Boolean query with body $\alpha(\text{head}(V))$, for Q.

Hence, by choosing the second input query to be $Q' = V$, Corollary 5.13 follows because Q and Q' are hierarchical. ☐

We note that the restriction of the select-full-project equivalence problem to Boolean queries Q' is just the equivalence problem for Boolean queries which is in P for acyclic queries [CR00].

But, surprisingly, the select-full-project equivalence problem is NP-hard not only for hierarchical but even for q-hierarchical queries—while the rewriting problem for q-hierarchical views is in P (Corollary 7.4).

Corollary 5.14. SelProjEquiv(QHCQ) *is* NP*-hard, even over database schemas with fixed arity.*

Proof. To lift Corollary 5.13 to q-hierarchical queries, we have to adapt the query Q' in the proof of Corollary 5.13 since it is not q-hierarchical. Adding the variable u to the head of Q' yields the desired q-hierarchical query, since every full hierarchical query is q-hierarchical. We note that the query Q is a hierarchical Boolean query and, therefore, trivially q-hierarchical.

Note that, for the proof of Theorem 5.9, which builds upon the proof for Theorem 5.7, it is required that u is *not* in the head of V. This ensures that the whole query has to be covered by a single application of the view V. In other words, if there is a rewriting of Q, there is a rewriting that consists only of a single atom. Since this restriction is not necessary in the setting of the select-full-project equivalence problem, we can indeed simply add[15] u to the head of Q'. □

6. A Tractable Case

In this section, we first show that the acyclic rewriting problem becomes tractable for free-connex acyclic views and acyclic queries over database schemas of bounded arity. We then define a slightly larger class of views, for which this statement holds as well. For the first result, we mainly show that rewritability with respect to a set $\mathcal{V}$ of free-connex acyclic views can be reduced to rewritability with respect to a set $\mathcal{W}$ of views of bounded arity.

Proposition 6.1. *There is a polynomial-time algorithm that computes from each set $\mathcal{V}$ of free-connex acyclic views a set $\mathcal{W}$ of acyclic*[16] *views such that*

(a) *the arity of the views of $\mathcal{W}$ is bounded by the arity of the underlying schema, and*
(b) *every conjunctive query Q is $\mathcal{V}$-rewritable if and only if it is $\mathcal{W}$-rewritable.*

Furthermore, given a $\mathcal{W}$-rewriting of Q, a $\mathcal{V}$-rewriting of Q can be computed in polynomial time.

The proof splits each view V into a set of views obtained by the subtrees of the root $\text{head}(V)$ of a join tree of $\text{body}(V) \cup \{\text{head}(V)\}$. The arities of the children of $\text{head}(V)$ yield the desired arity bound.

Proof. Let $\mathcal{V}$ be a set of free-connex acyclic views over a schema $\mathcal{S}$ with arity at most k. Let $V \in \mathcal{V}$ and J be a join tree for V including its head atom $\text{head}(V)$. Since a join tree is undirected, we can assume that the root of J is the node labelled with $\text{head}(V)$. Let $A_1, \dots, A_n$ be the labels of the children of the root node. Furthermore, let, for each $1 \le i \le n$, $\mathcal{B}_i$ be the set of atoms in the subtree of J with root A_i. Since J is a join tree, each variable that occurs in some atom of a set $\mathcal{B}_i$ and in $\text{head}(V)$ also occurs in A_i. Furthermore, variables that occur in two sets $\mathcal{B}_i$, $\mathcal{B}_j$, $i \neq j$, necessarily occur also in $\text{head}(V)$. Thus, the following two conditions hold, for every $i \in \{1, \dots, n\}$ and every $j < i$,

(1) $|\text{vars}(\mathcal{B}_i) \cap \text{vars}(\text{head}(V))| \le k$, and
(2) $\text{vars}(\mathcal{B}_i) \cap \text{vars}(\mathcal{B}_j) \subseteq \text{head}(V)$.

[15] In fact, we could also remove u from Q' altogether. In this case, we also have to remove it from Q.

[16] In fact, the views in $\mathcal{W}$ are even free-connex acyclic again. However, we do not claim it here, since it is not needed in the following, and it does not need to hold for the subsequent generalisation.

For each $1 \le i \le n$ we define the view V_i as the projection of V to the variables that occur in head(V) and in $\mathcal{B}_i$. That is, the body of V_i is just the body of V and the head of V_i contains precisely those head variables of V that occur in $\mathcal{B}_i$. By construction, the views V_i have arity $\text{ar}(A_i) \le k$. Furthermore, the views V_i are acyclic since they have the same body as V which is acyclic.[17]

The desired set $\mathcal{W}$ of views can thus be obtained by replacing each view $V \in \mathcal{V}$ by views $V_1, \dots, V_n$ as constructed above. It is easy to see that $\mathcal{W}$ can be computed in polynomial time.

It remains to show that an arbitrary CQ Q is $\mathcal{V}$-rewritable if and only if it is $\mathcal{W}$-rewritable. For the direction from right to left, let $\mathcal{C}_\mathcal{W}$ be a cover partition witnessing that Q is $\mathcal{W}$-rewritable. Consider a cover description $(\mathcal{A}, V_i, \alpha, \psi)$ where V_i is a view constructed as above, originating from a view $V \in \mathcal{V}$. Since the only difference between V and V_i is that V has more head variables, replacing V_i with V yields a cover description $(\mathcal{A}, V, \alpha, \psi)$. Analogous replacements in all cover descriptions in $\mathcal{C}_\mathcal{W}$ yield a cover partition witnessing $\mathcal{V}$-rewritability of Q. Moreover, given a $\mathcal{W}$-rewriting of Q, a $\mathcal{V}$-rewriting of Q can be obtained by replacing every V_i-atom in the $\mathcal{W}$-rewriting by a V-atom where V is, as above, the view V_i originated from. Variables not occurring in $\mathcal{B}_i$ but in the head of V are replaced by fresh variables (i.e. variables not occurring anywhere else) in the V-atom.

For the direction from left to right, let $\mathcal{C}_\mathcal{V}$ be a consistent cover partition witnessing $\mathcal{V}$-rewritability of Q. We replace the cover descriptions in $\mathcal{C}_\mathcal{V}$ to obtain a cover partition witnessing $\mathcal{W}$-rewritability of Q. To this end, let $C = (\mathcal{A}, V, \alpha, \psi)$ be a cover description in $\mathcal{C}_\mathcal{V}$. Furthermore, let $\mathcal{B}_1, \dots, \mathcal{B}_n$ and $V_1, \dots, V_n$ be as in the construction of $\mathcal{W}$ above. For each $1 \le i \le n$, let $\mathcal{A}_i$ be the set of all atoms of $\mathcal{A}$ which are in $\alpha(\mathcal{B}_i)$ but in no $\alpha(\mathcal{B}_j)$, for $j < i$. Since $\text{vars}(\mathcal{A}) \subseteq \alpha(\text{body}(V))$, this yields a partition of $\mathcal{A}$. We claim that, for each $1 \le i \le n$, $C_i = (\mathcal{A}_i, V_i, \alpha, \psi)$ is a cover description. Since V_i and V have the same body, Conditions (1) and (3) of Definition 4.2 hold. Condition (4) of Definition 4.2 holds since $\text{vars}(\mathcal{A}_i) \subseteq \text{vars}(\mathcal{A})$.

In the remainder, we prove that Condition (2) of Definition 4.2 holds. Let $x \in \text{bvars}(\mathcal{A}_i)$. Then x is either in $\text{bvars}(A)$ or it is a "new" bridge variable that also occurs in some $\mathcal{A}_j \subseteq \mathcal{A}$, $j \ne i$. In the former case, $x \in \alpha(\text{head}(V))$ since Condition (2) holds for C. Thus, there are variables $y \in \text{head}(V)$ and y' in $\mathcal{B}_i$ such that $\alpha(y) = x = \alpha(y')$. Since, thanks to quantified variable disjointness, α maps quantified and head variables disjointly, it follows that y' must be from head(V) as well. But then y' occurs in head(V_i) since it is in head(V) and in $\mathcal{B}_i$. Therefore, $x = \alpha(y') \in \alpha(\text{head}(V_i))$.

In the other case, let $j \ne i$ be such that x occurs in $\mathcal{A}_j$ and, hence, in $\alpha(\mathcal{B}_j)$. Let y, z be variables from $\mathcal{B}_i$ and $\mathcal{B}_j$, respectively, such that $\alpha(y) = x = \alpha(z)$. If $y = z$, then y is a head variable of V, since $\mathcal{B}_i$ and $\mathcal{B}_j$ have only head variables of V in common. If $y \ne z$, then both y and z occur in the head of V thanks to quantified variable disjointness. In both cases, we can conclude that $x = \alpha(y)$ occurs in $\alpha(\text{head}(V_i))$. Thus, Condition (2) of Definition 4.2 holds for C_i.

Since the $\mathcal{A}_i$ form a partition of $\mathcal{A}$, replacing C in $\mathcal{C}_\mathcal{V}$ with $C_1, \dots, C_n$ yields a cover partition (with respect to $\mathcal{V} \cup \mathcal{W}$) and iterating this process for all cover descriptions in the original partition $\mathcal{C}_\mathcal{V}$ yields a cover partition witnessing $\mathcal{W}$-rewritability of Q. □

The main result of this section is now a simple corollary to Proposition 6.1.

[17] We observe that, since all head variables of V_i occur in A_i, each V_i is even free-connex acyclic.

Theorem 6.2. *For every fixed k,* $\text{Rewr}^k(\mathsf{CCQ}, \mathsf{ACQ}, \mathsf{ACQ})$ *is in polynomial time and an acyclic rewriting can be computed in polynomial time, if it exists.*

Proof. Let $Q \in \mathsf{ACQ}$ be an acyclic query and $\mathcal{V} \subseteq \mathsf{CCQ}$ be a set of free-connex acyclic views over a schema $\mathcal{S}$ with arity at most k. Thanks to Proposition 6.1, from $\mathcal{V}$ an equivalent set $\mathcal{W}$ of acyclic views of arity at most k can be computed in polynomial time. The statements of the theorem thus follow immediately from Corollary 5.5. □

We leave the complexity of $\text{Rewr}(\mathsf{CCQ}, \mathsf{ACQ}, \mathsf{ACQ})$ as an open problem.

A closer inspection of the proof of Proposition 6.1 reveals that it does not exactly require that the views are free-connex acyclic. In fact, it suffices that each view has a partition $\mathcal{B}_1 \uplus \cdots \uplus \mathcal{B}_m$ of its body that obeys Conditions (1) and (2) defined in the proof of Proposition 6.1. Therefore, we turn these two requirements into a new notion. We formulate and study this notion for conjunctive queries Q, but we emphasise that we will use it for queries that define views only.

Definition 6.3. The *weak head arity* of a query Q is the smallest k for which there is a partition $\mathcal{B}_1 \uplus \cdots \uplus \mathcal{B}_n$ of body(Q) such that, for every $i \in \{1, \dots, n\}$ and every $j < i$,

(1) $|\text{vars}(\mathcal{B}_i) \cap \text{vars}(\text{head}(Q))| \leq k$, and
(2) $\text{vars}(\mathcal{B}_i) \cap \text{vars}(\mathcal{B}_j) \subseteq \text{head}(Q)$.

From the proof of Proposition 6.1 it follows that free-connex acyclic queries over a fixed schema have bounded weak head arity. The following example illustrates that there are indeed views over a fixed schema that have bounded weak head arity but are not free-connex acyclic.

Example 6.4. Let us consider the family $\left(V_n\right)_{n\in\mathbb{N}}$ of views with

- head $V_n(x, y_1, \dots, y_n, z_1, \dots, z_n)$, and
- body $\{R(x, u_i, y_i), S(x, u_i, z_i), T(y_i) \mid 1 \leq i \leq n\}$.

For $n \geq 1$ the view V_n is acyclic but *not* free-connex acyclic. It has, however, weak head arity 3. This is witnessed by the sets $\mathcal{B}_i = \{R(x, u_i, y_i), S(x, u_i, z_i), T(y_i)\}$ for $1 \leq i \leq n$ which form a partition satisfying the conditions of Definition 6.3.

To generalise Theorem 6.2 for views of bounded weak head arity, we thus only need to show that partitions obeying Conditions (1) and (2) of Definition 6.3 can be efficiently computed. In the remainder of this section we design an algorithm that determines the weak head arity of a given conjunctive query Q and computes a corresponding partition.

The algorithm relies on the concept of a cover graph for Q.

Definition 6.5. The cover graph $G(Q)$ of a conjunctive query Q is the undirected graph with node set body(Q) and edges (A, A') for atoms A and A' that share a variable that does not belong to head(Q).

Example 6.6. The cover graph of the view V_3 from Example 6.4 is depicted in Figure 2.

The following lemma states the relationship between weak head arity and cover graph.

Lemma 6.7. *Let Q be a conjunctive query and $\mathcal{B}_1, \dots, \mathcal{B}_n$ the connected components of its cover graph. The weak head arity of Q is the maximal number ℓ of head variables of Q in a set $\mathcal{B}_i$. Moreover, the connected components $\mathcal{B}_1, \dots, \mathcal{B}_n$ witness that Q has weak head arity ℓ.*

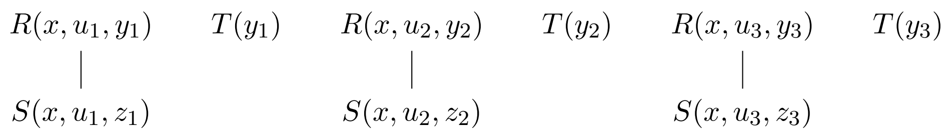


Figure 2. The cover graph $G(V_3)$ of the query V_n for $n = 3$ defined in Example 6.4.

Proof. Let Q be a conjunctive query and let $\mathcal{B}'_1 \uplus \cdots \uplus \mathcal{B}'_m$ be a partition of body(Q) witnessing that Q has weak head arity k.

Since by definition of the cover graph, two different sets $\mathcal{B}_i$ and $\mathcal{B}_j$ only share head variables, $\mathcal{B}_1 \uplus \cdots \uplus \mathcal{B}_n$ is a partition of body(Q) witnessing that Q has weak head arity at most ℓ, hence it holds $k \leq \ell$.

To show that $\ell \leq k$, it suffices to show that every connected component $\mathcal{B}$ in $G(Q)$ is contained in some atom set $\mathcal{B}'_i$. Towards a contradiction we assume that there is a connected component $\mathcal{B}$ with atoms from two different subsets of the partition. Since $\mathcal{B}$ is connected there must be some $A_1, A_2 \in \mathcal{B}$, connected by an edge, such that $A_1 \in \mathcal{B}'_i$ and $A_2 \in \mathcal{B}'_j$, for some $i \neq j$. By definition of $G(Q)$, A_1 and A_2 share a variable that is not part of the head of Q. But that contradicts Condition (2) in Definition 6.3. Thus, we can conclude $\ell \leq k$. □

Lemma 6.7 offers an algorithm to compute the weak head arity of a conjunctive query Q and a witness partition for it. It simply computes the cover graph $G(Q)$ of Q and its connected components. Then the weak head arity is the maximum number of head variables that occur in any connected component. Furthermore, the connected components form the desired partition satisfying the conditions of Definition 6.3. We thus have the following corollary.

Corollary 6.8. *There is an algorithm that, upon input of a conjunctive query Q, computes in polynomial time the weak head arity of Q and a partition of* body(Q) *that witnesses it.*

By combining Corollary 6.8 with the proofs of Proposition 6.1 and Theorem 6.2, we obtain the following generalisation of Theorem 6.2.

Theorem 6.9. *For each fixed $k \in \mathbb{N}$, the acyclic rewriting problem for acyclic queries and acyclic views with weak head arity k is in polynomial time.*

7. The Existence of Hierarchical and q-hierarchical Rewritings

In Section 5, we have shown that every *acyclic* query has an *acyclic* rewriting if it has a rewriting at all. This gives us a guarantee that there is a rewriting that has the same complexity benefits for query evaluation as the original query.

It is natural to ask whether other, stronger properties transfer in the same fashion. In this section, we consider this question for hierarchical and q-hierarchical queries.

The following example illustrates that, as for acyclic rewritings, even if a hierarchical rewriting for a hierarchical query exists, the canonical rewriting is not necessarily hierarchical.

Example 7.1. Consider the hierarchical query $H(x, y) \leftarrow R(x), S(y), T(x), T(y)$ and the views $V_1(x_1, y_1) \leftarrow R(x_1), S(y_1)$ and $V_2(z_2) \leftarrow T(z_2)$. The canonical rewriting

$$H(x, y) \leftarrow V_1(x, y), V_2(x), V_2(y)$$

is not hierarchical, since atoms(x) and atoms(y) are neither disjoint nor subsets of one another. However, a hierarchical rewriting exists, for instance

$$H(x, y) \leftarrow V_1(x, y'), V_1(x', y), V_2(x), V_2(y)$$

is a hierarchical rewriting.

It turns out that Theorem 5.4 also holds for hierarchical and q-hierarchical queries.

Theorem 7.2. *Let Q be a conjunctive query and $\mathcal{V}$ be a set of views.*
(a) *If Q is hierarchical and $\mathcal{V}$-rewritable, then it has a hierarchical $\mathcal{V}$-rewriting.*
(b) *If Q is q-hierarchical and $\mathcal{V}$-rewritable, then it has a q-hierarchical $\mathcal{V}$-rewriting.*

Similarly as for Theorem 5.4, the proof of Theorem 7.2 partitions cover descriptions. However, the strategy for doing so is different here: instead of defining the partition as the connected components of an atom set with respect to a join tree of the query at hand, here the partition guaranteed by the following lemma is used.

Lemma 7.3. *Let Q be a hierarchical query and $(\mathcal{A}, V, \alpha, \psi)$ a cover description for Q. There is a partition $\mathcal{A}_1 \uplus \cdots \uplus \mathcal{A}_n = \mathcal{A}$ such that the following conditions hold.*
(A) *Each variable $y \notin \text{vars}(\alpha(\text{head}(V)))$ appears in at most one set $\mathcal{A}_i$.*
(B) *Each $\mathcal{A}_i$ is*
    (Ba) *a singleton set or*
    (Bb) *there is a variable $x \notin \text{vars}(\alpha(\text{head}(V)))$ that appears in every atom in $\mathcal{A}_i$.*

Proof. The algorithm uses an undirected graph G similar to the cover graph of Section 6. The graph G has vertex set $\mathcal{A}$ and an edge labelled by variable x between two atoms A_1, A_2, if $x \notin \text{vars}(\text{head}(\alpha(V)))$ and x occurs in A_1 and A_2. We note that there can be more than one edge between two atoms.

Clearly, for each variable $x \notin \text{vars}(\text{head}(\alpha(V)))$, the atoms that contain x constitute a clique in G with edges labelled by x. In particular, each variable $x \notin \text{vars}(\text{head}(\alpha(V)))$ can occur in at most one connected component. Since Q is hierarchical, for two cliques C_x, C_y induced by different variables x and y it holds that either they are disjoint or one contains the other. It readily follows that for each connected component H of G there is a variable x, such that C_x contains all atoms of H.

Therefore the partition $\mathcal{A}_1 \uplus \cdots \uplus \mathcal{A}_n = \mathcal{A}$ given by the connected components of G fulfils the conditions of the Lemma. □

Now we are prepared to prove Theorem 7.2.

Proof of Theorem 7.2. Towards (a), let Q be a $\mathcal{V}$-rewritable hierarchical query. W.l.o.g. we can assume that Q is minimal (cf. Remark 4.8). Thanks to Theorem 4.6, there is a cover partition $\mathcal{C}' = C'_1, \ldots, C'_k$ for Q over $\mathcal{V}$. We replace every cover description $(\mathcal{A}, V, \alpha, \psi)$ in $\mathcal{C}'$ by $(\mathcal{A}_1, V, \alpha, \psi), \ldots, (\mathcal{A}_n, V, \alpha, \psi)$ were $\mathcal{A}_1 \uplus \cdots \uplus \mathcal{A}_n = \mathcal{A}$ is the partition of $\mathcal{A}$ from Lemma 7.3 and let $\mathcal{C}$ be the collection we obtain after the replacement. For each $(\mathcal{A}_i, V, \alpha, \psi)$ Condition (2) of Definition 4.2 is satisfied thanks to Condition (A) of Lemma 7.3. It is easy to verify that the other conditions in Definition 4.2 are satisfied for each $(\mathcal{A}_i, V, \alpha, \psi)$. All in all, the collection $\mathcal{C}$ is a cover partition for Q over $\mathcal{V}$. Thanks to Theorem 4.6 we can assume that $\mathcal{C}$ is a consistent one with the same partition of body(Q).

Let R be the rewriting obtained from $\mathcal{C}$, i.e., the query $Q_{\mathcal{C}}$. In the remainder of the proof we show that R is hierarchical, i.e., every pair of variables in x and y in R satisfies the conditions in Definition 2.1. Let x and y be two variables in R.

If x occurs in only one atom of body(R) then there are two cases: (i) if in the only atom in which x occurs, y occurs as well, then $\text{atoms}_R(x) \subseteq \text{atoms}_R(y)$, and (ii) otherwise, $\text{atoms}_R(x) \cap \text{atoms}_R(y) = \emptyset$ holds. Analogously, if y occurs in only one atom of body(R) then either $\text{atoms}_R(y) \subseteq \text{atoms}_R(x)$ or $\text{atoms}_R(x) \cap \text{atoms}_R(y) = \emptyset$ holds.

Let us finally assume that both x and y occur in at least two atoms of body(R) each. Thanks to $\mathcal{C}$ being a consistent cover partition and Condition (2) of Definition 4.2, x and y are bridge variables, and occur in atom sets of at least two different cover descriptions of $\mathcal{C}$ each.

Since Q is hierarchical we have the following three cases.

Case 1: $\text{atoms}_Q(x) \subseteq \text{atoms}_Q(y)$. Let $(\mathcal{A}, V, \alpha, \psi)$ in $\mathcal{C}$ be a cover description which fulfils $x \in \alpha(\text{head}(V))$. In particular, x occurs in an atom $A \in \mathcal{A}$. Therefore, y also occurs in A, since $\text{atoms}_Q(x) \subseteq \text{atoms}_Q(y)$. Since y is a bridge variable by assumption, it follows that $y \in \alpha(\text{head}(V))$ thanks to Condition (2) of Definition 4.2. We conclude that $\text{atoms}_R(x) \subseteq \text{atoms}_R(y)$ holds.

Case 2: $\text{atoms}_Q(y) \subseteq \text{atoms}_Q(x)$. This case is analogous to the first case.

Case 3: $\text{atoms}_Q(x) \cap \text{atoms}_Q(y) = \emptyset$. If there is no cover description in $\mathcal{C}$, in which x and y occur together, then $\text{atoms}_R(x) \cap \text{atoms}_R(y) = \emptyset$ holds. Let us thus assume that there is a cover description $(\mathcal{A}, V, \alpha, \psi)$ in $\mathcal{C}$ in which both x and y occur. Thanks to $\text{atoms}_Q(x) \cap \text{atoms}_Q(y) = \emptyset$, set $\mathcal{A}$ contains at least two atoms. Thanks to Lemma 7.3, there is a variable $u \notin \text{vars}(\text{head}(\alpha(V)))$ that appears in all atoms of $\mathcal{A}$. Since Q is hierarchical, we must have $\text{atoms}_Q(x) \subsetneq \text{atoms}_Q(u)$. However, this yields a contradiction, because x occurs in at least two cover descriptions, and therefore outside $\mathcal{A}$, whereas u does not.

This concludes the proof of Statement (a).

Towards (b), let us assume that Q is q-hierarchical. Our goal is to show that for all variables $x, y \in \text{vars}(R)$, if $\text{atoms}_R(x) \subsetneq \text{atoms}_R(y)$ holds and x is in the head of R, then y is also in the head of R. Note that x and y are both bridge variables because x occurs in the head of Q which is the same as the head of R, and y occurs in at least two atoms of body(R) due to $\text{atoms}_R(x) \subsetneq \text{atoms}_R(y)$. In particular, x and y both occur in Q. Since the heads of Q and R are the same, whenever $\text{atoms}_Q(x) \subsetneq \text{atoms}_Q(y)$ holds, we can conclude that if x is in the head of R, then y is also in the head of R.

In the remainder we assume, for the sake of a contradiction, that $\text{atoms}_Q(x) \subsetneq \text{atoms}_Q(y)$ does not hold. Since Q is hierarchical this means that either $\text{atoms}_Q(x) \supseteq \text{atoms}_Q(y)$ or $\text{atoms}_Q(x) \cap \text{atoms}_Q(y) = \emptyset$ holds.[18]

If x also occurs in at least two atoms of body(R), then the preconditions for Case 2 and Case 3 in the proof for Statement (a) above are met. Thus, $\text{atoms}_Q(x) \supseteq \text{atoms}_Q(y)$ and $\text{atoms}_Q(x) \cap \text{atoms}_Q(y) = \emptyset$ imply $\text{atoms}_R(x) \supseteq \text{atoms}_R(y)$ and $\text{atoms}_R(x) \cap \text{atoms}_R(y) = \emptyset$, respectively. But this is a contradiction to $\text{atoms}_R(x) \subsetneq \text{atoms}_R(y)$.

In the remainder, consider the case that x occurs in exactly one atom B of body(R). Variable y occurs in B and outside B. Since x and y are bridge variables, it follows that x occurs in exactly one atom set $\mathcal{A}$ of $\mathcal{C}$ and y occurs in and outside $\mathcal{A}$. Hence, $\text{atoms}_Q(x) \cap \text{atoms}_Q(y) = \emptyset$ holds because $\text{atoms}_Q(x) \supseteq \text{atoms}_Q(y)$ cannot. This implies that $\mathcal{A}$ consists of at least two atoms, since x and y co-occur in $\mathcal{A}$. Therefore, there is a non-bridge variable u that occurs in all atoms of $\mathcal{A}$ thanks to Lemma 7.3(B). But then

[18]Note, that the case $\text{atoms}_Q(x) = \text{atoms}_Q(y)$ is a special case of $\text{atoms}_Q(x) \supseteq \text{atoms}_Q(y)$.

$\text{atoms}_Q(x) \subsetneq \text{atoms}_Q(u)$ holds, and, since u is a non-head variable, x is a non-head variable as well, because Q is q-hierarchical. This is a contradiction to x being a head variable.

All in all, we can conclude that $\text{atoms}_Q(x) \subsetneq \text{atoms}_Q(y)$ holds, and therefore, that y is a head variable. Thus, R is q-hierarchical. □

Similarly to Theorem 5.4, Theorem 7.2 delivers good news as well as bad news. The good news is that, since $\mathsf{QHCQ} \subseteq \mathsf{CCQ}$ and $\mathsf{QHCQ} \subseteq \mathsf{HCQ} \subseteq \mathsf{ACQ}$ hold, the rewriting problem for q-hierarchical views and hierarchical queries over a fixed schema is tractable thanks to Theorem 6.2 and Theorem 7.2. The bad news is that Theorem 5.7 and Theorem 7.2 imply NP-completeness for hierarchical queries and views.

Corollary 7.4. $\text{Rewr}^k(\mathsf{QHCQ}, \mathsf{HCQ}, \mathsf{HCQ})$ *and* $\text{Rewr}^k(\mathsf{QHCQ}, \mathsf{QHCQ}, \mathsf{QHCQ})$ *are in polynomial time for every* $k \in \mathbb{N}$*.*

Corollary 7.5. $\text{Rewr}^k(\mathsf{HCQ}, \mathsf{HCQ}, \mathsf{HCQ})$ *is* NP*-complete for every* $k \geq 3$*.*

Of course, Corollary 7.5 implies NP-hardness of $\text{Rewr}^k(\mathbb{V}, \mathbb{Q}, \mathsf{ACQ})$, for all pairs $\mathbb{V}, \mathbb{Q}$ of classes with $\mathsf{HCQ} \subseteq \mathbb{V} \subseteq \mathsf{CQ}$ and $\mathsf{HCQ} \subseteq \mathbb{Q} \subseteq \mathsf{CQ}$.

8. Related Work

We already mentioned that our notion of cover partitions is similar to various notions from the literature. We mention three of them here.

In [AC19] algorithms for finding exact rewritings with a minimal number of atoms and (maximally) contained rewritings are presented. For this purpose triples (S, S', h) are considered which are comparable to cover descriptions [AC19, Definition 3.12]. Namely, S' corresponds to the set $\mathcal{A}$ in a cover description, $S = \text{body}(Q)$, and h is a homomorphism from S' into a view V. In our characterisation we can assume $h = \text{id}$ (and, thus, omit it in the specification of a cover description) thanks to the view application α. We note that h can be assumed to be one-to-one for (equivalent) $\mathcal{V}$-rewritings [AC19, Theorem 3.15]. Furthermore, for (equivalent) rewritings, only candidates whose body is a proper subset of the canonical rewriting's body are considered (cf. [AC19, Section 3.2.3]). Note that the existence of body homomorphisms ψ_i mapping the $\alpha_i(V_i)$ into Q is guaranteed for such candidates and ψ_i determines the application α_i (in particular, the unification of variables in the head of a view). This is, however, too restrictive for our purposes (cf., for instance, Example 5.1) which is why α and ψ are part of a cover description. We emphasise that Condition (2) is equivalent to the *shared-variable property* [AC19, Definition 3.12]. An analogue of the implication (a) ⇒ (b) of Theorem 4.6 is proven [AC19, Theorem 3.15] and, based on that, an algorithm CoreCover for finding (equivalent) rewritings is derived. This algorithm, however, considers only triples with maximal sets S' (called *tuple cores*) and they are allowed to overlap, i.e. they do not have to form a partition. In contrast, we consider non-maximal sets, for instance in the proof of Theorem 5.4 which depends upon the possibility to split these sets. For (maximally) contained rewritings, an analogue of (b) ⇒ (a) is implied by [AC19, Theorem 4.19]. Interestingly, the proof of this result (and the associated algorithm) exploits triples with minimal sets S' (that still satisfy the shared-variable property) and a partition property. We discuss the relation of this minimality constraint with our results below in more detail. An analogue of (b) ⇒ (a) for (equivalent) $\mathcal{V}$-rewritings, and, therefore, a characterisation for $\mathcal{V}$-rewritability, is not stated (nor implied).

In [GKC06] a characterisation for $\mathcal{V}$-rewritability is employed to find rewritings efficiently. It is in terms of *tuple coverages* and a *partition condition* corresponding to cover descriptions and partitions, respectively [GKC06, Theorem 5, Theorem 6]. A tuple coverage, denoted $s(t_V, Q)$, is a (non-empty) set G, where t_V corresponds to $\alpha(V)$ and G to the set $\mathcal{A}$ of a cover description. Similar to [AC19] the most crucial differences in comparison with our characterisation are that only rewritings whose body is contained in the canonical rewriting, i.e. $\text{vars}(\alpha(V)) \subseteq \text{vars}(Q)$ holds for all tuple coverages, are considered. Also the mappings ψ and α and their associated conditions are not denoted explicitly, instead it is required that G is isomorphic to a subset of $\alpha(V)$ and Condition (2) holds. This is equivalent to our conditions if restricted to the rewritings considered in [GKC06].

Lastly, in [PH01] *MiniCon descriptions (MCDs)* are used to compute (maximally) contained rewritings (which are unions of conjunctive queries, in general). A MCD is a tuple of the form (h, V, φ, G) which relates to a cover description $(\mathcal{A}, V, \alpha, \psi)$ as follows: h is called a head homomorphism and is basically the restriction of an application α to the head variables of the view V, G corresponds to the set $\mathcal{A}$, and φ is mapping embedding G into $h(V)$. The component φ has no counterpart in a cover description because, thanks to α being able to rename any variable in V, we can assume $\varphi = \text{id}$. We note that the applications α_i of a consistent cover partition $\mathcal{C}$ also allow us to conveniently denote the expansion of the associated rewriting $Q_{\mathcal{C}}$. On the other hand, there is no counterpart for the body-homomorphism ψ in a MCD (since the ψ_i in a consistent cover partition ensure that the query is contained in the associated expansion of the rewriting, there is also no need for them if contained rewritings are considered). Condition (2) is stated as [PH01, Property 1] and the idea of a cover partition in [PH01, Property 2].

Minimal cover descriptions. In [AC19], "minimal" triples are used to construct maximally contained rewritings. In terms of cover descriptions, an analogous definition of minimality is as follows.

Definition 8.1 (Minimal Cover Descriptions)**.** A cover description $(\mathcal{A}, V, \alpha, \psi)$ for a query Q is called *minimal* if there is no partition $\mathcal{A}_1 \uplus \cdots \uplus \mathcal{A}_k = \mathcal{A}$ for $k \geq 2$ with nonempty subsets $\mathcal{A}_1, \ldots, \mathcal{A}_k$ such that cover descriptions $C_1, \ldots, C_k$ with $C_i = (\mathcal{A}_i, V, \alpha_i, \psi_i)$ for Q exist.

It seems obvious to exploit such minimal cover description for the constructions in Theorem 5.4 and Theorem 7.2, instead of partitioning the set $\mathcal{A}$ of a cover description "manually". However, in contrast to our constructions, this would (possibly) not result in efficient algorithms to compute an acyclic or hierarchical rewriting from an arbitrary one, since it turns out that deciding whether a cover description is minimal is coNP-hard.

Proposition 8.2. *Let Q be a hierarchical conjunctive query. It is* coNP*-hard to decide whether a cover description $(\mathcal{A}, Q, \alpha, \psi)$ is minimal.*

Proof. We reduce the problem Rewr(HCQ, HCQ, CQ), which is NP-hard due to Theorem 5.7, to the *complement* of the minimisation problem. For a hierarchical query Q and a set of hierarchical views $\mathcal{V}$, we describe how we can derive a cover description $C^Q_{\mathcal{V}}$ and a query Q' such that $C^Q_{\mathcal{V}}$ is *not* minimal for Q' if and only if there is a $\mathcal{V}$-rewriting for Q.

For convenience, we introduce some notation. For an atom $A = R(x_1, \ldots, x_r)$ and a variable u we denote by A^u the atom $R(u, x_1, \ldots, x_r)$ resulting from extending A by the variable u. We lift this notation to sets $\mathcal{A}$ of atoms in the natural way, i.e.

$$\mathcal{A}^u \;=\; \{R(u, x_1, \ldots, x_r) \mid R(x_1, \ldots, x_r) \in \mathcal{A}\}\,.$$

If a set of atoms or a query consists of atoms extended by (possibly different) variables as described above, we often signify this with a "$_+$"-symbol in the superscript, e.g. we write $\mathcal{A}^+$.

Let $V_1, \ldots, V_n$ be the views in $\mathcal{V}$. We assume that the views in $\mathcal{V}$ and the query Q refer to distinct variables, which is no restriction since the variables can be renamed accordingly in polynomial time.

Construction. We start with the construction of the cover description $C^Q_{\mathcal{V}}$. To define a view and a query for the cover description, we consider views $V_1^+, \ldots, V_n^+$ that are copies of the input views $V_1, \ldots, V_n$ where each atom will be expanded by a special variable, that only appears in the associated view, in the first component of each body atom and of the head, i.e. for every $i \in \{1, \ldots, n\}$ we define a view V_i^+ with $\text{head}(V_i^+) = \text{head}(V_i)^{v_i}$ and $\text{body}(V_i^+) = \text{body}(V_i)^{v_i}$ where $v_1, \ldots, v_n$ are distinct variables that do not occur in the query or in any of the views in $\mathcal{V}$. In the same fashion, we define a set of atoms for the atoms in the query, but we add a new atom whose relation symbol does not occur in Q or in any of the views in $\mathcal{V}$. For a new variable u and a new relation symbol S, we define $\mathcal{A}^+_Q = \text{body}(Q)^u \cup \{S(u)\}$.

Let us now turn to the definition of the view W that is used in the cover description. The view W contains the atoms in $\mathcal{A}^+_Q$, the atoms in $\text{body}(V_i^+)$ for all $i \in \{1, \ldots, n\}$, and a special atom $S(v_{n+1})$ with a new variable v_{n+1} in the body. The head of W contains the variables in the head of Q and variables in the heads of the views V_i for all $i \in \{1, \ldots, n\}$ as well as the variables $v_1, \ldots, v_{n+1}$, but not u. Let us emphasize that, for each view V_i^+, the set $\text{body}(V_i^+)$ is contained in $\text{body}(W)$.

The query Q' is defined by the rule $\text{head}(Q) \leftarrow \text{body}(W)$, that is Q' has the head variables of Q and the body of Q' has the same atoms as the body of W.

The cover description $C^Q_{\mathcal{V}}$ is defined as $(\mathcal{A}^+_Q, W, \text{id}, \text{id})$. Note that the bridge variables of $\mathcal{A}^+_Q$ with respect to Q' are contained in the head variables in W. Hence, $C^Q_{\mathcal{V}}$ is a cover description for Q'.

Note that the views V_i^+ are hierarchical since for the new variables v_i we have that $\text{atoms}_{V_i^+}(v_i) \supseteq \text{atoms}_{V_i^+}(y)$ for all $y \in \text{vars}(V_i)$ and the input views are hierarchical. The same is true for the set $\mathcal{A}^+_Q$ (viewed as a Boolean query here). Moreover, W and Q' are also hierarchical since the views and the query, and, hence, the V_i^+ and $\mathcal{A}^+_Q$ do not share any variable.

Correctness. In the following, we show that Q has a $\mathcal{V}$-rewriting if and only if $C^Q_{\mathcal{V}}$ for Q' is not minimal.

For the only if-direction assume that Q has a $\mathcal{V}$-rewriting. Then there is a consistent cover partition $\mathcal{C}$ for Q. Note that by construction each cover description $(\mathcal{A}, V_j, \alpha, \psi)$ in $\mathcal{C}$ can be turned into a cover description $(\mathcal{A}^u, V_j^+, \alpha \cup \{v_j \mapsto u\}, \psi \cup \{u \mapsto u\})$ for Q' and, furthermore into a cover description $(\mathcal{A}^u, W, \alpha', \psi')$ for Q' where α' and ψ' coincide with $\alpha \cup \{v_j \mapsto u\}$ and $\psi \cup \{u \mapsto u\}$ on their domain, respectively, and are the identity on all other variables. Let $\mathcal{C}'$ be the collection of cover descriptions we obtain by transforming every cover description in $\mathcal{C}$ to a cover description as described above and the additional cover description $(\{S(u)\}, W, \{v_{n+1} \mapsto u\}, \text{id})$ for Q'.

Since $\mathcal{C}$ is a cover partition for Q and Q does not contain an S-atom, the atom sets of the cover descriptions in $\mathcal{C}'$ form a partition of $\mathcal{A}_Q^+ = \text{body}(Q)^u \cup \{S(u)\}$. But then $(\mathcal{A}_Q^+, W, \text{id}, \text{id})$ is not minimal for Q' because $\mathcal{C}'$ consists of at least two cover descriptions.

For the if-direction assume that $C_{\mathcal{V}}^Q$ is not minimal for Q'. Let

$$\mathcal{C}' = (\mathcal{A}_1^+, W, \alpha_1, \psi_1), \ldots, (\mathcal{A}_k^+, W, \alpha_k, \psi_k)$$

be a collection of cover descriptions witnessing that $C_{\mathcal{V}}^Q = (\mathcal{A}_Q^+, W, \text{id}, \text{id})$ is not minimal for Q'. From $\mathcal{C}'$ we will derive a cover partition $\mathcal{C}$ for Q witnessing that Q is indeed $\mathcal{V}$-rewritable. For this purpose, we first analyse to which atoms in $\alpha_i(W)$ the atoms of a set $\mathcal{A}_i^+$ are mapped to and then associate views V_j^+ with (subsets of) the sets $\mathcal{A}_i^+$.

We can assume that the α_i fulfil the quantified variable disjointness. Note, that $k \geq 2$ since $C_{\mathcal{V}}^Q$ is not minimal, and u is a bridge variable of each set $\mathcal{A}_i^+$ since it occurs in every atom in $\mathcal{A}_Q^+$. Therefore, no atom in $\mathcal{A}_i^+$ can be mapped into the copy of $\alpha_i(\mathcal{A}_Q^+)$ in $\alpha_i(W)$ because the variable u is not a head variable of W. Hence, $\mathcal{A}_i^+$ is a subset of $\alpha_i(\text{body}(V_1)^{v_1}) \cup \ldots \cup \alpha_i(\text{body}(V_n)^{v_n}) \cup \{\alpha_i(S(v_{n+1}))\}$.

Since the sets $\alpha_i(\text{body}(V_j)^{v_j})$ do not share any variable that is not in the head$(\alpha_i(W))$, each cover description in $\mathcal{C}'$ can be partitioned into cover descriptions

$$(\mathcal{B}_{i,1}^+, W, \alpha_i, \psi_i), \ldots, (\mathcal{B}_{i,n}^+, W, \alpha_i, \psi_i)$$

for Q' where $\mathcal{B}_{i,j}^+ = \mathcal{A}_i^+ \cap \alpha_i(\text{body}(V_j)^{v_j})$ and, in case $S(u) \in \mathcal{A}_i^+$, a cover description $(\{S(u)\}, W, \alpha_i, \psi_i)$. For the sake of readability we assume that $\mathcal{B}_{i,j}^+ \neq \emptyset$ holds for all i, j. If not, the respective cover descriptions can just be removed from the sequence.

Note that the view W in a cover description $(\mathcal{B}_{i,j}^+, W, \alpha_i, \psi_i)$ can be replaced by V_j^+, because $\mathcal{B}_{i,j}^+ \subseteq \alpha_i(\text{body}(V_j^+))$ by definition. Hence, each cover description $(\mathcal{B}_{i,j}^+, W, \alpha_i, \psi_i)$ can be transformed into a cover description $(\mathcal{B}_{i,j}^+, V_j^+, \alpha_{i,j}, \psi_{i,j})$ where $\alpha_{i,j}$ and $\psi_{i,j}$ are the restriction of α_i and ψ_i resp. on vars(V_j^+). A cover description $(\mathcal{B}_{i,j}^+, V_j^+, \alpha_{i,j}, \psi_{i,j})$ can, in turn, be transformed into a cover description $(\mathcal{B}_{i,j}, V_j, \widehat{\alpha}_{i,j}, \widehat{\psi}_{i,j})$ for Q where $\mathcal{B}_{i,j}$ is the atom set with $\mathcal{B}_{i,j}^{v_j} = \mathcal{B}_{i,j}^+$, and $\widehat{\alpha}_{i,j}$ and $\widehat{\psi}_{i,j}$ are the restriction of $\alpha_{i,j}$ and $\psi_{i,j}$ on vars(V_j).

Let $\mathcal{C}$ be the collection of cover descriptions we obtain by applying the transformations described above to all cover descriptions in $\mathcal{C}'$ and removing cover descriptions with atom set $\{S(u)\}$. Note, that by construction, the atom sets in $\mathcal{C}'$ form a partition of body(Q). Thus $\mathcal{C}'$ is a cover partition and hence, Q is $\mathcal{V}$-rewritable. ☐

Applications of structurally simple queries. It is well known that many problems are tractable for acyclic conjunctive queries but (presumably) not for conjunctive queries in general. Notably, the evaluation, minimisation, and the containment problem are tractable for acyclic queries [Yan81, CR00, GLS01] but NP-complete for the class of conjunctive queries [CM77].

The class of free-connex conjunctive queries plays a central role in the enumeration complexity of conjunctive queries. In [BDG07], Bagan, Durand and Grandjean showed that the result of a free-connex acyclic conjunctive query can be enumerated with constant delay after a linear time preprocessing phase. Moreover, they also showed that the result of an acyclic conjunctive query without self-joins that is not free-connex cannot be enumerated with constant delay after a linear time preprocessing, unless $n \times n$ matrices can be multiplied in time $O(n^2)$.

Hierarchical queries play a central role in different contexts. On the one hand, Dalvi and Suciu [DS07] showed that the class of hierarchical Boolean conjunctive queries without self-joins characterises precisely the Boolean CQs without self-joins that can be answered in polynomial time on probabilistic databases. This has been extended by Fink and Olteanu [FO14] to the notion of non-Boolean queries and queries with negation. On the other hand, Koutris and Suciu [KS11] studied hierarchical join queries in the context of query evaluation on massively parallel architectures. We refer to [FO16] for further applications of hierarchical queries.

The notion of q-hierarchical queries has played a central role in the evaluation of conjunctive queries under single tuple updates [BKS17]. In [BKS17] it is shown that the result of a q-hierarchical conjunctive query can be evaluated (by answering a Boolean CQ in constant time, enumerating the result of a non-Boolean CQ with constant delay, or outputting the number of results in constant time) with constant update time after a linear time preprocessing. Moreover, they showed that the result of a conjunctive query without self-joins that is not q-hierarchical cannot be evaluated with constant update time after a linear time preprocessing, unless some algorithmic conjectures on Online Matrix-Vector multiplication (see [HKNS15] for more information about the conjecture) do not hold. The notion of q-hierarchical queries is also related to factorised databases [Kep20]. The notion of factorised databases has already been considered in various contexts [OZ15, BKOZ13, SOC16]. A further recent source of information on structurally simple queries is [KNOZ20].

9. Conclusion

We studied rewritability by acyclic queries or queries from CCQ, HCQ, or QHCQ. Based on a new characterisation of (exact) rewritability, we showed that acyclic queries have acyclic rewritings, if they have any CQ rewriting. The same holds for the other three query classes.

We showed that for acyclic queries and views the decision problem, whether an acyclic rewriting exists, is intractable, even for schemas with bounded arity, but becomes tractable if views have a bounded arity (even with unbounded schema arity) or are free-connex acyclic.

We leave the case of free-connex acyclic views and unbounded schemas open. Another interesting open question is the complexity of rewriting problems $\textsc{Rewr}(\mathbb{V}, \mathbb{Q}, \mathbb{R})$ with $\mathbb{R} \subsetneq \mathbb{Q}$, e.g. $\textsc{Rewr}(\mathsf{ACQ}, \mathsf{ACQ}, \mathsf{QHCQ})$. So far we have only NP-hardness results for problems $\textsc{Rewr}(\mathbb{V}, \mathbb{Q}, \mathsf{ACQ})$ and $\textsc{Rewr}(\mathbb{V}, \mathbb{Q}, \mathsf{HCQ})$ where $\mathbb{V}$ and $\mathbb{Q}$ encompass all acyclic or all hierarchical queries, respectively.

Finally, it would be interesting to study whether our results can be extended to other classes of queries that can be evaluated efficiently like conjunctive queries of bounded treewidth.

References

[AC19] Foto N. Afrati and Rada Chirkova. *Answering Queries Using Views, Second Edition*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2019. doi:10.2200/S00884ED2V01Y201811DTM054.

[AHV95] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995. URL: http://webdam.inria.fr/Alice/.

[BB13] Johann Brault-Baron. *De la pertinence de l'énumération : complexité en logiques propositionnelle et du premier ordre*. Theses, Université de Caen, 2013. URL: https://hal.archives-ouvertes.fr/tel-01081392.

[BDG07] Guillaume Bagan, Arnaud Durand, and Etienne Grandjean. On acyclic conjunctive queries and constant delay enumeration. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *Lecture Notes in Computer Science*, pages 208–222. Springer Berlin Heidelberg, 2007. doi:10.1007/978-3-540-74915-8_18.

[BKOZ13] Nurzhan Bakibayev, Tomás Kociský, Dan Olteanu, and Jakub Zavodny. Aggregation and ordering in factorised databases. *Proceedings of the VLDB Endowment*, 6(14):1990–2001, 2013. doi:10.14778/2556549.2556579.

[BKS17] Christoph Berkholz, Jens Keppeler, and Nicole Schweikardt. Answering conjunctive queries under updates. In Emanuel Sallinger, Jan Van den Bussche, and Floris Geerts, editors, *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017, Chicago, IL, USA, May 14-19, 2017*, pages 303–318. ACM, 2017. doi:10.1145/3034786.3034789.

[BPR17] Pablo Barceló, Andreas Pieris, and Miguel Romero. Semantic optimization in tractable classes of conjunctive queries. *ACM SIGMOD Record*, 46(2):5–17, 2017. doi:10.1145/3137586.3137588.

[CGLP20a] Hubie Chen, Georg Gottlob, Matthias Lanzinger, and Reinhard Pichler. Semantic width and the fixed-parameter tractability of constraint satisfaction problems. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 1726–1733. International Joint Conferences on Artificial Intelligence Organization, 2020. doi:10.24963/ijcai.2020/239.

[CGLP20b] Hubie Chen, Georg Gottlob, Matthias Lanzinger, and Reinhard Pichler. Semantic width and the fixed-parameter tractability of constraint satisfaction problems. *CoRR*, abs/2007.14169, 2020. URL: https://arxiv.org/abs/2007.14169, arXiv:2007.14169.

[CGLV05] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. View-based query processing: On the relationship between rewriting, answering and losslessness. In Thomas Eiter and Leonid Libkin, editors, *Database Theory - ICDT 2005, 10th International Conference, Edinburgh, UK, January 5-7, 2005, Proceedings*, volume 3363 of *Lecture Notes in Computer Science*, pages 321–336. Springer Berlin Heidelberg, 2005. doi:10.1007/978-3-540-30570-5_22.

[CM77] Ashok K. Chandra and Philip M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In John E. Hopcroft, Emily P. Friedman, and Michael A. Harrison, editors, *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*, pages 77–90. ACM Press, 1977. doi:10.1145/800105.803397.

[CR00] Chandra Chekuri and Anand Rajaraman. Conjunctive query containment revisited. *Theor. Comput. Sci.*, 239(2):211–229, 2000. doi:10.1016/S0304-3975(99)00220-0.

[CY12] Rada Chirkova and Jun Yang. Materialized views. *Foundations and Trends® in Databases*, 4(4):295–405, 2012. doi:10.1561/1900000020.

[DS07] Nilesh N. Dalvi and Dan Suciu. The dichotomy of conjunctive queries on probabilistic structures. In Leonid Libkin, editor, *Proceedings of the Twenty-Sixth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 11-13, 2007, Beijing, China*, pages 293–302. ACM, 2007. doi:10.1145/1265530.1265571.

[FO14] Robert Fink and Dan Olteanu. A dichotomy for non-repeating queries with negation in probabilistic databases. In Richard Hull and Martin Grohe, editors, *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS'14, Snowbird, UT, USA, June 22-27, 2014*, pages 144–155. ACM, 2014. doi:10.1145/2594538.2594549.

[FO16] Robert Fink and Dan Olteanu. Dichotomies for queries with negation in probabilistic databases. *ACM Transactions on Database Systems*, 41(1):4:1–4:47, 2016. doi:10.1145/2877203.

[GKC06] Gang Gou, Maxim Kormilitsin, and Rada Chirkova. Query evaluation using overlapping views: completeness and efficiency. In Surajit Chaudhuri, Vagelis Hristidis, and Neoklis Polyzotis, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, Chicago, Illinois, USA, June 27-29, 2006*, pages 37–48. ACM, 2006. doi:10.1145/1142473.1142479.

[GKSS22] Gaetano Geck, Jens Keppeler, Thomas Schwentick, and Christopher Spinrath. Rewriting with Acyclic Queries: Mind Your Head. In Dan Olteanu and Nils Vortmeier, editors, *25th International Conference on Database Theory (ICDT 2022)*, volume 220 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 8:1–8:20, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICDT.2022.8.

[GLS01] Georg Gottlob, Nicola Leone, and Francesco Scarcello. The complexity of acyclic conjunctive queries. *Journal of the ACM*, 48(3):431–498, 2001. doi:10.1145/382780.382783.

[Hal01] Alon Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, 2001. doi:10.1007/s007780100054.

[HKNS15] Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 21–30. ACM, 2015. doi:10.1145/2746539.2746609.

[HY19] Xiao Hu and Ke Yi. Instance and output optimal parallel algorithms for acyclic joins. In Dan Suciu, Sebastian Skritek, and Christoph Koch, editors, *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 450–463. ACM, 2019. doi:10.1145/3294052.3319698.

[IUV17] Muhammad Idris, Martín Ugarte, and Stijn Vansummeren. The dynamic yannakakis algorithm: Compact and efficient query processing under updates. In Semih Salihoglu, Wenchao Zhou, Rada Chirkova, Jun Yang, and Dan Suciu, editors, *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*, pages 1259–1274. ACM, 2017. doi:10.1145/3035918.3064027.

[Kep20] Jens Keppeler. *Answering Conjunctive Queries and FO+MOD Queries under Updates*. PhD thesis, Humboldt University of Berlin, Germany, 2020. doi:10.18452/21483.

[KNOZ20] Ahmet Kara, Milos Nikolic, Dan Olteanu, and Haozhe Zhang. Trade-offs in static and dynamic evaluation of hierarchical queries. In Dan Suciu, Yufei Tao, and Zhewei Wei, editors, *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, pages 375–392. ACM, 2020. doi:10.1145/3375395.3387646.

[KS06] Benny Kimelfeld and Yehoshua Sagiv. Incrementally computing ordered answers of acyclic conjunctive queries. In Opher Etzion, Tsvi Kuflik, and Amihai Motro, editors, *Next Generation Information Technologies and Systems, 6th International Workshop, NGITS 2006, Kibbutz Shefayim, Israel, July 4-6, 2006, Proceedings*, volume 4032 of *Lecture Notes in Computer Science*, pages 141–152. Springer Berlin Heidelberg, 2006. doi:10.1007/11780991_13.

[KS11] Paraschos Koutris and Dan Suciu. Parallel evaluation of conjunctive queries. In Maurizio Lenzerini and Thomas Schwentick, editors, *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2011, June 12-16, 2011, Athens, Greece*, pages 223–234. ACM, 2011. doi:10.1145/1989284.1989310.

[LMSS95] Alon Y. Levy, Alberto O. Mendelzon, Yehoshua Sagiv, and Divesh Srivastava. Answering queries using views. In Mihalis Yannakakis and Serge Abiteboul, editors, *Proceedings of the Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 22-25, 1995, San Jose, California, USA*, pages 95–104. ACM Press, 1995. doi:10.1145/212433.220198.

[NSV10] Alan Nash, Luc Segoufin, and Victor Vianu. Views and queries: Determinacy and rewriting. *ACM Transactions on Database Systems*, 35(3):21:1–21:41, 2010. doi:10.1145/1806907.1806913.

[OZ15] Dan Olteanu and Jakub Závodný. Size bounds for factorised representations of query results. *ACM Transactions on Database Systems*, 40(1):2:1–2:44, 2015. doi:10.1145/2656335.

[PH01] Rachel Pottinger and Alon Y. Halevy. Minicon: A scalable algorithm for answering queries using views. *The VLDB Journal*, 10(2-3):182–198, 2001. doi:10.1007/s007780100048.

[SOC16] Maximilian Schleich, Dan Olteanu, and Radu Ciucanu. Learning linear regression models over factorized joins. In Fatma Özcan, Georgia Koutrika, and Sam Madden, editors, *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 3–18. ACM, 2016. doi:10.1145/2882903.2882939.

[Yan81] Mihalis Yannakakis. Algorithms for acyclic database schemes. In *Very Large Data Bases, 7th International Conference, September 9-11, 1981, Cannes, France, Proceedings*, pages 82–94. IEEE Computer Society, 1981.