Query Rewriting with Disjunctive Existential Rules and Mappings

Michel Leclère, Marie-Laure Mugnier, Guillaume Pérution-Kihli

Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Query Rewriting with Disjunctive Existential Rules and Mappings

Michel Leclère , Marie-Laure Mugnier , Guillaume Pérution-Kihli
LIRMM, Inria, University of Montpellier, CNRS, France
{leclere,mugnier}@lirmm.fr, guillaume.perution-kihli@inria.fr

Abstract

We consider the issue of answering unions of conjunctive queries (UCQs) with disjunctive existential rules and mappings. While this issue has already been well studied from a chase perspective, query rewriting within UCQs has hardly been addressed yet. We first propose a sound and complete query rewriting operator, which has the advantage of establishing a tight relationship between a chase step and a rewriting step. The associated breadth-first query rewriting algorithm outputs a minimal UCQ-rewriting when one exists. Second, we show that for any "truly disjunctive" nonrecursive rule, there exists a conjunctive query that has no UCQ-rewriting. It follows that the notion of finite unification sets (fus), which denotes sets of existential rules such that any UCQ admits a UCQ-rewriting, seems to have little relevance in this setting. Finally, turning our attention to mappings, we show that the problem of determining whether a UCQ admits a UCQ-rewriting through a disjunctive mapping is undecidable. We conclude with a number of open problems.

1 Introduction

Existential rules (Calì, Gottlob, and Kifer 2008; Baget et al. 2009; Calì, Gottlob, and Lukasiewicz 2009), aka tuple generating dependencies (Beeri and Vardi 1984), are an extension of datalog (i.e., first-order function-free Horn rules), which allows for existentially quantified variables in the rule heads, e.g., $\forall x(\text{human}(x) \rightarrow \exists y\ \text{isParent}(y, x))$. They have become a popular language to model ontologies and do reasoning on data. Then, a key issue is *ontology-mediated query answering*, which consists of computing the answers to a query on a knowledge base (KB), composed of a set of facts (or data) F and an ontology $\mathcal{O}$. In this context, most works focus on the prominent class of (unions of) conjunctive queries ((U)CQs). There are two main dual techniques to compute the answers to a query Q: the *chase*, which enriches the facts F by performing a fixpoint computation with the ontology $\mathcal{O}$ until a canonical model of F and $\mathcal{O}$ is obtained (then Q is evaluated on this canonical model), and *query rewriting*, where Q is rewritten using $\mathcal{O}$ into a query Q', such that for any set of facts F, the evaluation of Q' on F yields the answers to Q on the KB. Query answering with general existential rules is undecidable, however a wide range of decidable subclasses have been defined, based on syntactic restrictions that ensure the termination of chase-like or query rewriting techniques. Tuple generating dependencies (TGDs) are also the main formalism to represent *schema mappings*, which are high-level specifications of the relationships between two database schemas (Fagin et al. 2005). Schema mappings are at the core of many data interoperability tasks, such as data exchange, data integration or peer data management. More specifically, a mapping is a set of TGDs, with bodies and heads expressed on disjoint sets of predicates, namely $\mathcal{S}$ and $\mathcal{T}$, called the *source* and the *target* schemas. Given a database instance I on $\mathcal{S}$ and a mapping $\mathcal{M}$, a query expressed on $\mathcal{T}$ is posed on the set of facts produced from I by triggering $\mathcal{M}$; again, query answering can be solved by *chasing* I with $\mathcal{M}$ or *rewriting* Q with $\mathcal{M}$ into a query that is evaluated on I. Since mappings are inherently nonrecursive, both techniques always terminate. Finally, in the Ontology-Based Data Access (OBDA) framework (Poggi et al. 2008), mappings specify relationships between a database schema and an ontology. Here, existential rules can be used as a uniform language to express both the ontology and the mapping (Buron, Mugnier, and Thomazo 2021).

Existential rules generalize popular description logics (DLs) used to do reasoning on data, such as DL-Lite (Calvanese et al. 2007), $\mathcal{EL}$ (Baader, Brandt, and Lutz 2005; Lutz, Toman, and Wolter 2009) and more expressive Horn-DLs (Krötzsch, Rudolph, and Hitzler 2006). However, they do not capture *nondeterministic* features, as offered by some key DLs such as $\mathcal{ALC}$ (Schmidt-Schauß and Smolka 1991) or the Semantic Web ontology language OWL (W3C 2009).

In this paper, we consider the extension of existential rules with *disjunction*, e.g., $\forall x \forall y(\text{isGrandParent}(x, y) \rightarrow \exists z_1\ (\text{isParent}(x, z_1) \wedge \text{isMother}(z_1, y)) \vee \exists z_2\ (\text{isParent}(x, z_2) \wedge \text{isFather}(z_2, y)))$. From a KR perspective, the usefulness of such rules has long been acknowledged for ontology modeling, but also for expressing nondeterministic guessing in problem solving, see e.g., (Eiter, Gottlob, and Mannila 1997). From a database perspective, disjunction in schema mappings received considerable attention in the context of *mapping management*, where mapping composition and inversion emerged as fundamental operators (Bernstein and Ho 2007; Arenas et al. 2010). Indeed, disjunction is required to express several kinds of inverse mappings, like so-called quasi-inverses or maximum recovery mappings (Fagin et al. 2008; Arenas, Pérez, and Riveros 2008). Beside the issue of

constructing such mappings, the design of associated query answering techniques is highly relevant. For instance, in a peer data management system, a mapping $\mathcal{M}$ from peer P_1 to peer P_2 allows to rewrite a query on P_2 in terms of P_1, while an inverse of $\mathcal{M}$ allows to rewrite a query on P_1 in terms of P_2. As another example, consider a mapping $\mathcal{M}$ from schema A to schema B, and assume that A evolves into A', which is expressed by a mapping $\mathcal{M}'$; the relation between A' and B can be obtained by inverting $\mathcal{M}'$ and composing it with $\mathcal{M}$; then, a query on B can be translated into a query on A' by rewriting it first with $\mathcal{M}$, then with the inverse of $\mathcal{M}'$ (Pérez 2013). Such scenario is also relevant in OBDA, taking for B an ontology instead of a schema.

So far, reasoning with disjunctive existential rules has been mainly studied through the *chase*. It was shown that decidable classes of (conjunctive) existential rules, based on the behavior of the chase, can be generalized to disjunctive rules in a quite natural way, whether in relation to acyclicity notions (Carral, Dragoste, and Krötzsch 2017) or based on guardedness (Alviano et al. 2012; Gottlob et al. 2012; Bourhis et al. 2016), although these generalizations come with a huge increase in the complexity of query answering.

In contrast, *query rewriting* within UCQs has been barely addressed yet. A notable exception is the work in (Alfonso, Chortaras, and Stamou 2021), which provides a rewriting technique based on first-order resolution (see Section 3). A large body of work has studied the rewritability of ontology-mediated queries, i.e., pairs of the form $(Q, \mathcal{O})$ with Q a (U)CQ and $\mathcal{O}$ an ontology, into query languages of various expressivity. However, for ontologies expressed in fragments of disjunctive existential rules, most studies target expressive rewriting languages, like disjunctive datalog (Bienvenu et al. 2014; Ahmetaj, Ortiz, and Simkus 2018). As far as we are aware, the only result directly relevant to our purpose comes from the fine-grained complexity study in (Gerasimova et al. 2020), which provides syntactic rewritability conditions for ontology-mediated queries where the ontology is composed of a single specific disjunctive rule, called a covering axiom (see Section 4).

Our contributions are the following:

- We first define a sound and complete query rewriting operator for UCQs and disjunctive existential rules, which has the advantage of establishing a tight relationship between a chase step and a rewriting step (Theorem 3). The associated breadth-first query rewriting algorithm outputs a minimal UCQ-rewriting when one exists (Theorem 4).
- We then turn our attention to the notion of *finite unification sets (fus)*, which denotes sets of existential rules for which any UCQ is UCQ-rewritable, i.e., admits a finite sound and complete rewriting under the form of a UCQ. Noting that the known *fus* classes for conjunctive existential rules do not seem to be generalizable to disjunctive rules, we show that, in fact, for *any* "truly disjunctive" nonrecursive rule, there is a CQ that is not UCQ-rewritable (Theorem 5). This leads to question the relevance of *fus* for disjunctive rules and to consider the problem of whether a specific UCQ is UCQ-rewritable.
- Finally, considering (disjunctive) mappings, we show that the problem of determining whether a given UCQ on the target schema admits a UCQ-rewriting on the source schema is undecidable (Theorem 6).

Based on these results, we conclude with a number of open problems. Detailed proofs are available in a technical report (Leclère, Mugnier, and Pérution-Kihli 2023).

2 Preliminaries

Generalities. We consider logical vocabularies of the form $\mathcal{V} = (\mathcal{P}, \mathcal{C})$, where $\mathcal{P}$ is a finite set of predicates and $\mathcal{C}$ is a (possibly infinite) set of constants. A *term* on $\mathcal{V}$ is a constant from $\mathcal{C}$ or a variable. An *atom* on $\mathcal{V}$ has the form $p(\mathbf{t})$ where $p \in \mathcal{P}$ is a predicate of arity n and $\mathbf{t}$ is a tuple of terms on $\mathcal{V}$ with $|\mathbf{t}| = n$. An atom with predicate p is also called a p-atom. Given a formula or set of formulas S, we denote by $\mathtt{vars}(S)$, $\mathtt{consts}(S)$ and $\mathtt{terms}(S)$ its sets of variables, constants and terms, respectively. We will often see a tuple $\mathbf{x}$ of pairwise distinct variables as a set. We denote by $\models$ and $\equiv$ classical logical entailment and equivalence, respectively. Given two sets of atoms S_1 and S_2, a *homomorphism* h from S_1 to S_2 is a substitution of $\mathtt{vars}(S_1)$ by $\mathtt{terms}(S_2)$ such that $h(S_1) \subseteq S_2$ (we say that S_1 *maps* to S_2 by h). It is well-known that, when we see S_1 and S_2 as existentially closed conjunctions of atoms, $S_2 \models S_1$ iff S_1 maps to S_2.

A *safe copy* of an atom set S is obtained from S by a bijective renaming of its variables with *fresh* variables (i.e., that do not occur elsewhere in the context of the computation).

Knowledge base. A *set of facts* F is a possibly infinite set of atoms, logically seen as an existentially closed conjunction. When this set is finite we call it a *fact base*. A *disjunctive existential rule* R (or simply rule hereafter) is a closed formula of the form

$$\forall \mathbf{x} \forall \mathbf{y} \; (\; B[\mathbf{x}, \mathbf{y}] \rightarrow \bigvee_{i=1}^{n} \exists \mathbf{z_i} H_i[\mathbf{x_i}, \mathbf{z_i}] \;)$$

where $n \geq 1$, B and the H_i are non-empty finite conjunctions of atoms with $\mathtt{vars}(B) = \mathbf{x} \cup \mathbf{y}$ and $\mathtt{vars}(H_i) = \mathbf{x_i} \cup \mathbf{z_i}$, $\mathbf{x} = \bigcup_{i=1}^{n} \mathbf{x_i}$ and $\mathbf{x}, \mathbf{y}$ and the $\mathbf{z_i}$ are pairwise disjoint; B is the *body* of R, also denoted by $\mathtt{body}(R)$, and $\{H_1, \ldots, H_n\}$ is the *head* of R, also denoted by $\mathtt{head}(R)$. We also denote by $\mathtt{head}_i(R)$ the i-th disjunct H_i of the head of R. The set $\mathbf{x}$ is the *frontier* of R and is denoted by $\mathtt{fr}(R)$. Its elements are called frontier variables. The set $\mathbf{z_i}$ is the set of *existential variables* of H_i, also denoted by $\mathtt{exist}(H_i)$, and the union of all the $\mathtt{exist}(H_i)$ is the set of existential variables of R, also denoted by $\mathtt{exist}(R)$. Note that constants may occur anywhere. For brevity, we often denote by $B \rightarrow H_1 \vee \ldots \vee H_n$ a rule with body B and head $\{H_1, \ldots, H_n\}$. A rule R is *conjunctive* if $n = 1$. A (disjunctive) rule R is (disjunctive) *datalog* if $\mathtt{exist}(R) = \emptyset$.

A (disjunctive) *knowledge base* (KB) is a pair $(F, \mathcal{R})$, where F is a fact base and $\mathcal{R}$ is a finite set of (disjunctive) existential rules. We assume w.l.o.g. that distinct rules in $\mathcal{R}$ have disjoint sets of variables. In examples, we may reuse variables for simplicity.

Disjunctive chase. A rule $R = B \to H_1 \vee \ldots \vee H_n$ is *applicable* on a fact base F if there is a homomorphism h from $\mathtt{body}(R)$ to F. The pair (R, h) is called a *trigger* on F. The application of (R, h) to F is denoted by $\alpha_\vee(F, R, h)$; it produces a set of n fact bases, each obtained by adding to F a set of atoms obtained from $\mathtt{head}_i(R)$ by replacing each frontier variable x by $h(x)$ and each existential variable by a fresh variable. We denote by $h^{\mathtt{safe}_i}$ the extension of h that safely renames $\mathtt{exist}(\mathtt{head}_i(R))$ by fresh variables. Then:

$$\alpha_\vee(F, R, h) = \{F \cup h^{\mathtt{safe}_i}(\mathtt{head}_i(R)) \mid 1 \leq i \leq n\}$$

The disjunctive chase procedure iteratively applies triggers towards a fixpoint. This procedure is often seen as the construction of a tree, see in particular (Bourhis et al. 2016; Carral, Dragoste, and Krötzsch 2017).

Definition 1 (Derivation tree)**.** *A* derivation tree $\mathcal{T}$ *of a KB* $(F, \mathcal{R})$ *is a (possibly infinite) rooted labeled tree* (V, E, λ)*, where* V *is the set of vertices,* E *the set of edges, and* λ *a vertex labeling function inductively defined as follows:*

- $\lambda(r) = F$ *for the root* r *of* $\mathcal{T}$*;*
- *For each vertex* v *with children* $\{v_1, ..., v_n\}$*, there is a trigger* (R, h) *on* $\lambda(v)$ *with* $R = B \to H_1 \vee \ldots \vee H_n \in \mathcal{R}$ *and the restriction of* λ *to the domain* $\{v_1, ..., v_n\}$ *is a bijection to* $\alpha_\vee(\lambda(v), R, h)$*.*

Note that we do not impose any criterion of trigger applicability, as we do not aim at studying a particular chase strategy. A *branch* γ of a rooted tree is a maximal path from the root; we denote by *nodes*(γ) its set of vertices. Given a derivation tree $\mathcal{T}$, we denote by $\Gamma(\mathcal{T})$ the set of all its branches. A trigger (R, h) on F is *satisfied* (by F) if there is an extension h' of h with $h'(\mathtt{head}_i(R)) \subseteq F$ for some i. A derivation tree (V, E, λ) is *fair* if, for each branch γ and each vertex $v \in$ *nodes*(γ), any trigger on $\lambda(v)$ is satisfied in a $\lambda(v')$ with $v' \in$ *nodes*(γ). Finally, a *chase tree* is a fair derivation tree.

Definition 2 (Disjunctive chase result)**.** *The* result of a disjunctive chase *of* F *by* $\mathcal{R}$ *is chase*$(F, \mathcal{R}) = \{\bigcup_{v \in nodes(\gamma)} \lambda(v) \mid \gamma \in \Gamma(\mathcal{T})\}$ *where* $\mathcal{T}$ *is a chase tree and* λ *its labeling function.*

From a logical viewpoint, the chase result is a *disjunction* of existentially closed conjunctions of atoms. Neither the chase tree nor the chase result are unique, however all the results entail the same queries (see next Theorem 1). Although the degree of each vertex in a chase tree is bounded by the maximal number of disjuncts in a rule head, the tree may have infinite branches, and an infinite number of them. When the chase tree is finite, the result of the chase is the (finite) set of fact bases associated with its leaves.

It is sometimes convenient to consider a linearization of a finite derivation tree, which we call a derivation. A *derivation* of $(\{F\}, \mathcal{R})$ is a finite sequence of sets of fact bases and triggers $\mathcal{D} = (\mathcal{F}_0 = \{F\}) \xrightarrow{t_1} \mathcal{F}_1 \xrightarrow{t_2} \ldots \xrightarrow{t_k} \mathcal{F}_k$ where $t_i = (R, h)$ is a trigger of $R \in \mathcal{R}$ on an $F_j \in \mathcal{F}_{i-1}$ and $\mathcal{F}_i = (\mathcal{F}_{i-1} \setminus \{F_j\}) \cup \alpha_\vee(F_j, R, h)$, for all $1 \leq i \leq k$. To each finite derivation tree can be assigned a derivation obtained from any total ordering of the trigger applications associated with the inner vertices in the tree, in a compatible way with the parent-child partial order. When $\mathcal{R}$ is a set of conjunctive rules, a derivation tree is a path and the $\mathcal{F}_i$ in a derivation are singletons; then, a derivation can be seen as a sequence of fact bases (instead of sets of fact bases).

Query Answering. A *conjunctive query* (CQ) Q takes the form $\exists \mathbf{y}\ \phi[\mathbf{x}, \mathbf{y}]$, where $\mathbf{x}$ and $\mathbf{y}$ are disjoint tuples of variables, and ϕ is a finite conjunction of atoms with $\mathtt{vars}(\phi) = \mathbf{x} \cup \mathbf{y}$. The variables in $\mathbf{x}$ are called *answer variables*. A *Boolean CQ* has no answer variables. In a *full* CQ, all variables are answer variables. An *atomic* CQ has a single atom. A (Boolean) *union of conjunctive queries* (UCQ) is a disjunction of (Boolean) CQs with the same tuple of answer variables $\mathbf{x}$. For clarity, we denote a UCQ by $\mathcal{Q}$ and a CQ by Q. A set of facts F *answers positively* to a Boolean CQ Q if $F \models Q$. More generally, a tuple of constants $\mathbf{c}$ is an *answer* to a CQ Q on F if there is a substitution s such that $s(\mathbf{x}) = \mathbf{c}$ and $F \models s(Q)$. This extends to a UCQ $\mathcal{Q}$ and a set of sets of facts $\mathcal{F}$: a tuple of constants $\mathbf{c}$ is an answer to $\mathcal{Q}$ on $\mathcal{F}$ if for every $F_i \in \mathcal{F}$, there is a CQ $Q_j \in \mathcal{Q}$ such that $\mathbf{c}$ is an answer to Q_j on F_i.

W.l.o.g. we focus in this paper on Boolean queries, to avoid technicalities related to answer variables. Hence, in the following, by UCQ and CQ we refer to *Boolean* queries, unless otherwise specified. We will often see a CQ as a set of atoms, and a UCQ as a set of atoms sets.

The following theorem states that the disjunctive chase provides a sound and complete procedure to decide whether a UCQ is entailed by a disjunctive KB.

Theorem 1 (from (Bourhis et al. 2016))**.** *Let* $\mathcal{Q}$ *be a (Boolean) UCQ and* $(F, \mathcal{R})$ *be a disjunctive KB. Then* $F, \mathcal{R} \models \mathcal{Q}$ *iff chase*$(F, \mathcal{R}) \models \mathcal{Q}$ *(i.e.,* $F_i \models \mathcal{Q}$ *for all* $F_i \in$ *chase*$(F, \mathcal{R})$*).*

Example 1 (Colorability)**.** *Let* F *be a fact base on predicates* v *(vertex) and* e *(edge) describing a graph* G*. Let* $R = v(x) \to g(x) \vee r(x)$ *("Every vertex has color green or red"). Then, chase*$(F, \{R\})$ *yields all ways of coloring each vertex. Let the UCQ* $\mathcal{Q} = \{Q_1, Q_2\}$ *with* $Q_1 = \{g(u), e(u, w), g(w)\}$ *and* $Q_2 = \{r(u), e(u, w), r(w)\}$*. The KB* $(F, \{R\})$ *answers positively to* $\mathcal{Q}$ *iff* G *is not 2-colorable.*

Given UCQs $\mathcal{Q}_1$ and $\mathcal{Q}_2$, we say that $\mathcal{Q}_1$ is *more specific* than $\mathcal{Q}_2$ if $\mathcal{Q}_1 \models \mathcal{Q}_2$. Note that $\mathcal{Q}_1 \models \mathcal{Q}_2$ iff for all $Q_1 \in \mathcal{Q}_1$, there is $Q_2 \in \mathcal{Q}_2$ such that $Q_1 \models Q_2$ (i.e., Q_2 maps to Q_1 by homomorphism). A CQ Q is *minimal* if it has no strict subset $Q' \subsetneq Q$ such that $Q' \equiv Q$ (i.e., $Q' \models Q$). A UCQ $\mathcal{Q}$ is *minimal* if it has no strict subset $\mathcal{Q}' \subsetneq \mathcal{Q}$ such that $\mathcal{Q} \equiv \mathcal{Q}'$ (whether each CQ in the UCQ is itself minimal is not relevant for our results). A *cover* of a UCQ $\mathcal{Q}$ is a minimal subset $\mathcal{Q}' \subseteq \mathcal{Q}$ such that $\mathcal{Q} \equiv \mathcal{Q}'$. It is known that, given two equivalent UCQs $\mathcal{Q}_1$ and $\mathcal{Q}_2$, there is a bijection from any cover of $\mathcal{Q}_1$ to any cover of $\mathcal{Q}_2$ that maps each CQ in $\mathcal{Q}_1$ to an equivalent CQ in $\mathcal{Q}_2$ (see, e.g., (König et al. 2015)).

Mappings. Given two disjoint sets of predicates $\mathcal{S}$ and $\mathcal{T}$, respectively called the *source* and the *target* predicates, a *source-to-target* (or $\mathcal{S}$-to-$\mathcal{T}$) rule R is such that $\texttt{body}(R)$ uses predicates in $\mathcal{S}$ and $\texttt{head}(R)$ uses predicates in $\mathcal{T}$. A (disjunctive) mapping $\mathcal{M}$ on $(\mathcal{S}, \mathcal{T})$ is a finite set of $\mathcal{S}$-to-$\mathcal{T}$ (disjunctive) rules. In this setting, a fact base (or database instance) is expressed on $\mathcal{S}$ and a query on $\mathcal{T}$. Note that the chase of a fact base with a mapping is always finite.

UCQ rewritability. In the following, by *rewriting* of a UCQ $\mathcal{Q}$ with a set of rules $\mathcal{R}$, we mean a possibly infinite set of CQs $\mathcal{Q}'$, such that for all fact base F, if $F \models \mathcal{Q}'$ then $F, \mathcal{R} \models \mathcal{Q}$ (in other words, a rewriting is by definition *sound*). A rewriting $\mathcal{Q}'$ of $\mathcal{Q}$ with $\mathcal{R}$ is *complete* if for all fact base F, if $F, \mathcal{R} \models \mathcal{Q}$ then $F \models \mathcal{Q}'$. A finite complete rewriting is called a *UCQ-rewriting*. A pair $(\mathcal{Q}, \mathcal{R})$ is called *UCQ-rewritable* if it admits a UCQ-rewriting. The set $\mathcal{R}$ itself is called *UCQ-rewritable* if for any UCQ $\mathcal{Q}$, the pair $(\mathcal{Q}, \mathcal{R})$ is UCQ-rewritable. In the framework of conjunctive existential rules, a UCQ-rewritable set is also called a *finite unification set (fus)* (Baget et al. 2011). We shall extend this term to disjunctive rules.

Example 2 (Transitivity)**.** *Let $R = p(x, y) \wedge p(y, z) \rightarrow p(x, z)$. The (Boolean) CQ $Q_1 = \{p(a, b)\}$, where a and b are constants, has no UCQ-rewriting with $\{R\}$, while the (Boolean) CQ $Q_2 = \{p(u, v)\}$ has one, which is $\{Q_2\}$. Indeed, any complete rewriting of Q_1 is infinite as it contains all the "paths" of p-atoms from a to b, which are pairwise incomparable by homomorphism. In contrast, the atom $p(u, v)$ maps by homomorphism to any path of p-atoms.*

Finally, we recall some fundamental notions on rewriting with *conjunctive* existential rules. We will rely on these to define rewriting with disjunctive rules.

Query rewriting with conjunctive existential rules In the setting of conjunctive existential rules, query rewriting can be performed using *piece-unifiers*; these are a generalization of classical unifiers that take care of existential variables in rule heads by unifying sets of atoms instead of single atoms (Salvat and Mugnier 1996; Baget et al. 2009). In short, a piece-unifier unifies a subset Q' of a CQ Q and a subset H' of a rule head, such that existential variables from H' are unified only with variables of Q' that do not occur in $Q \setminus Q'$. Next, we call *separating variables* of Q' (w.r.t. Q) the variables of Q' that also occur in $Q \setminus Q'$. It is convenient to represent a unifier as a partition of a set of terms rather than a substitution. Hence, we say that a partition P of a set of terms is *admissible* if no class of P contains two constants; we associate a substitution u with an admissible partition P_u by selecting one term in each class with priority given to constants: for each class C in P_u, let t_i be the selected term, then for every $t_j \in C$, we set $u(t_j) = t_i$.

Definition 3 (Piece-unifier)**.** [1] *Let Q be a CQ and $R = B \rightarrow H$ be a conjunctive existential rule such that $\texttt{vars}(Q) \cap \texttt{vars}(B \cup H) = \emptyset$. A* piece-unifier *of Q with R is a triple $\mu = (Q', H', P_u)$ with $Q' \neq \emptyset$, $Q' \subseteq Q$, $H' \subseteq H$, and P_u is an admissible partition on $\texttt{terms}(Q') \cup \texttt{terms}(H')$ such that:*

1. *$u(Q') = u(H')$, with u a substitution associated with P_u;*
2. *If a class $C \in P_u$ contains an existential variable (from H'), then the other terms in C are non-separating variables from Q'.*

Let $\mu = (Q', H', P_u)$ be a piece-unifier of Q with $R : B \rightarrow H$ and u a substitution associated with P_u. The application of μ produces the following CQ:

$$\beta(Q, R, \mu) = u(B) \cup u(Q \setminus Q')$$

Example 3 (Piece-Unifier)**.** *Let $R = p(x, y) \rightarrow \exists z\ p_1(x, z) \wedge p_2(y, z)$ and $Q_1 = \{p_1(u, v), s(v)\}$. There is no piece-unifier of Q_1 with R since v is a separating variable of $Q'_1 = \{p_1(u, v)\}$, hence cannot be unified with z. Let $Q_2 = \{p_1(u, v), s(u)\}$: now, there is a piece-unifier of Q_2 with R, namely $\mu_2 = (\{p_1(u, v)\}, \{p_1(x, z)\}, P_{u_2})$ with $P_{u_2} = \{\{x, u\}, \{y\}, \{z, v\}\}$. Taking the substitution $u_2 = \{u \mapsto x, v \mapsto z\}$, we obtain $\beta(Q_2, R, \mu_2) = \{p(x, y), s(x)\}$. Finally, let $Q_3 = \{p_1(u, v), p_2(u, w), p_1(t, v), s(t)\}$, and $Q'_3 = Q_3 \setminus \{s(t)\}$. The triple $\mu_3 = (Q'_3, \texttt{head}(R), P_{u_3})$ with $P_{u_3} = \{\{x, y, t, u\}, \{z, v, w\}\}$ is a piece-unifier of Q_3 with R. If we select x and z in P_{u_3}, $\beta(Q_3, R, \mu_3) = \{p(x, x), s(x)\}$.*

A *piece-rewriting* of a UCQ $\mathcal{Q}$ with a (conjunctive) rule set $\mathcal{R}$ is a UCQ $\mathcal{Q}_k$ obtained by a finite sequence of piece-unifier applications, i.e., $(\mathcal{Q}_0 = \mathcal{Q}), \ldots, \mathcal{Q}_k$ ($k \geq 0$) such that, for all $0 < i \leq k$, there is a piece-unifier μ of $Q \in \mathcal{Q}_{i-1}$ with $R \in \mathcal{R}$ such that $\mathcal{Q}_i = \mathcal{Q}_{i-1} \cup \{\beta(Q, R, \mu)\}$.

As stated below, piece-unifiers provide a sound and complete query rewriting procedure:

Theorem 2 (from (Baget et al. 2011))**.** *For any (conjunctive) KB $(F, \mathcal{R})$ and UCQ $\mathcal{Q}$, there is a derivation of $(F, \mathcal{R})$ leading to an F_i such that $F_i \models \mathcal{Q}$ iff there is a piece-rewriting $\mathcal{Q}_j$ of $\mathcal{Q}$ with $\mathcal{R}$ such that $F \models \mathcal{Q}_j$.*

It follows that, when a pair $(\mathcal{Q}, \mathcal{R})$ is UCQ-rewritable, a UCQ-rewriting can be obtained as a piece-rewriting. Let us point out that a conjunctive mapping is always UCQ-rewritable (or *fus*). Indeed, since it is made of $\mathcal{S}$-to-$\mathcal{T}$ rules, the application of a piece-unifier of a CQ Q produces a CQ with strictly fewer atoms on $\mathcal{T}$ than Q. Also, CQs that contain predicates on $\mathcal{T}$ are useless in a rewriting.

3 Query Rewriting with Disjunctive Rules

Our generalization of query rewriting to disjunctive rules relies on a simple idea: a query $\mathcal{Q}$ can be rewritten with a rule $R = B \rightarrow H_1 \vee \cdots \vee H_n$ if *each* H_i contributes to partially answer $\mathcal{Q}$. Therefore, a unification step consists of unifying *each* H_i (using a piece-unifier) with a safe copy Q_i of a CQ from $\mathcal{Q}$; safe copies ensure that the CQs involved in the unification have pairwise disjoint sets of variables. Note that several safe copies of the same CQ from $\mathcal{Q}$ can be involved. This yields a new CQ made of $\texttt{body}(R)$ and the remaining parts of the unified CQs, according to some aggregation of the piece-unifiers. We need a few auxiliary notions to specify this aggregation. Let $\mathcal{P}$ be a set of partitions

[1] In non-Boolean queries, answer variables have to be treated as separating variables.

(not necessarily of the same set). The *join* of $\mathcal{P}$, denoted by join($\mathcal{P}$), is the partition obtained from $\mathcal{P}$ by making the union of the partitions in $\mathcal{P}$, then merging all non-disjoint classes until fixed point. E.g., given $\mathcal{P}$ composed of partitions $\{\{x,u\},\{y,v\},\{z,w\}\}$ and $\{\{x,y,a\},\{z',t\}\}$, we obtain $join(\mathcal{P}) = \{\{x,u,y,v,a\},\{z,w\},\{z',t\}\}$. We say that a set of partitions associated with piece-unifiers is *admissible* if its join is an admissible partition (i.e., it does not contain a class with two constants).

Definition 4 (Disjunctive Piece-Unifier and One-step Piece-Rewriting)**.** *Let a rule $R = B \to H_1 \vee \cdots \vee H_n$ and a UCQ $\mathcal{Q}$. A* disjunctive piece-unifier $\mu_\vee$ *of $\mathcal{Q}$ with R is a set $\{\mu_1, \ldots, \mu_n\}$ such that:*

- *for $1 \leq i \leq n$, $\mu_i = (Q'_i, H'_i, P_{u_i})$ is a (conjunctive) piece-unifier of Q_i, a safe copy of a CQ from $\mathcal{Q}$, with the (conjunctive) rule $B \to H_i$;*
- *and $\mathcal{P}_{u_\vee} = \{P_{u_1}, \ldots, P_{u_n}\}$ is admissible.*

Given a substitution $u_\vee$ associated with join($\mathcal{P}_{u_\vee}$), the application of $\mu_\vee$ produces the CQ

$$\beta_\vee(\mathcal{Q}, R, \mu_\vee) = u_\vee(B) \cup \bigcup_{1 \leq i \leq n} u_\vee(Q_i \setminus Q'_i)$$

The one-step piece-rewriting *of $\mathcal{Q}$ w.r.t. $\mu_\vee$ is*

$$\mathcal{Q} \cup \{\beta_\vee(\mathcal{Q}, R, \mu_\vee)\}$$

Example 4. *Let $R = p(x,y) \to \exists z_1\, r(x,z_1) \vee \exists z_2\, r(y,z_2)$ and the UCQ $\mathcal{Q} = \{Q\}$ with $Q = \{s(u), r(u,v)\}$. Let $Q_1 = \{s(u_1), r(u_1,v_1)\}$ and $Q_2 = \{s(u_2), r(u_2,v_2)\}$ be two safe copies of Q, and let $\mu_\vee = \{\mu_1, \mu_2\}$ with $\mu_1 = (\{r(u_1,v_1)\}, \{r(x,z_1)\}, \{\{u_1,x\}, \{v_1,z_1\}\})$ and $\mu_2 = (\{r(u_2,v_2)\}, \{r(y,z_2)\}, \{\{u_2,y\}, \{v_2,z_2\}\})$. Assume we give priority to variables from R, i.e., we take the substitution $u_\vee = \{u_1 \mapsto x, v_1 \mapsto z_1, u_2 \mapsto y, v_2 \mapsto z_2\}$. Then $\beta_\vee(\mathcal{Q}, R, \mu_\vee) = \{p(x,y), s(x), s(y)\}$.*

Definition 5 (Piece-Rewriting)**.** *Given a disjunctive rule set $\mathcal{R}$, a UCQ $\mathcal{Q}'$ is a* piece-rewriting *(or simply* rewriting *when clear from the context) of a UCQ $\mathcal{Q}$ with $\mathcal{R}$ if there is a finite sequence (called* rewriting sequence*) $\mathcal{Q} = \mathcal{Q}_0, \mathcal{Q}_1, \ldots, \mathcal{Q}_k = \mathcal{Q}'$ $(k \geq 0)$, such that for all $0 < i \leq k$, there is a disjunctive piece-unifier $\mu_\vee$ of $\mathcal{Q}_{i-1}$ with $R \in \mathcal{R}$ such that $\mathcal{Q}_i$ is the one-step rewriting of $\mathcal{Q}_{i-1}$ w.r.t. $\mu_\vee$.*

The following lemmas highlight fundamental properties of $\alpha_\vee$ and $\beta_\vee$.

Lemma 1 (Preservation of entailment by $\alpha_\vee$ and $\beta_\vee$)**.** *Let R be a disjunctive rule.*

1. *For any fact bases F_1 and F_2 such that $F_2 \models F_1$: if there is a trigger (R, h_1) on F_1 then there is a trigger (R, h_2) on F_2 such that $\alpha_\vee(F_2, R, h_2) \models \alpha_\vee(F_1, R, h_1)$.*
2. *For any UCQs $\mathcal{Q}_1$ and $\mathcal{Q}_2$ such that $\mathcal{Q}_2 \models \mathcal{Q}_1$: if there is a (disjunctive) piece-unifier μ_2 of $\mathcal{Q}_2$ with R then either $\beta_\vee(\mathcal{Q}_2, R, \mu_2) \models \mathcal{Q}_1$, or there is a (disjunctive) piece-unifier μ_1 of $\mathcal{Q}_1$ with R such that $\beta_\vee(\mathcal{Q}_2, R, \mu_2) \models \beta_\vee(\mathcal{Q}_1, R, \mu_1)$.*

The second lemma clarifies the tight relationship between $\alpha_\vee$ and $\beta_\vee$ (we recall that fact bases and CQs have the same logical form; this is also true of finite sets of fact bases and UCQs).

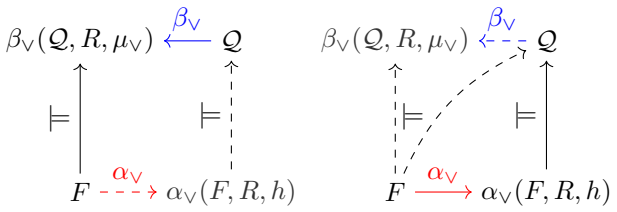

Figure 1: Correspondences between $\beta_\vee$ (in blue) and $\alpha_\vee$ (in red)

Lemma 2 (Composition of $\alpha_\vee$ and $\beta_\vee$)**.** *Let R be a disjunctive rule.*

1. *For any fact base F: if there is a trigger (R, h) on F then there is a (disjunctive) piece-unifier μ of $\alpha_\vee(F, R, h)$ with R such that $F \models \beta_\vee(\alpha_\vee(F, R, h), R, \mu)$.*
2. *For any UCQ $\mathcal{Q}$: if there is a piece-unifier μ of $\mathcal{Q}$ with R then there is a trigger (R, h) on $\beta_\vee(\mathcal{Q}, R, \mu)$ such that $\alpha_\vee(\beta_\vee(\mathcal{Q}, R, \mu), R, h) \models \mathcal{Q}$.*

These two lemmas are keys to establish the soundness and completeness of piece-rewriting, as stated next.

Theorem 3 (Soundness and completeness of piece-rewriting)**.** *Let $\mathcal{R}$ be a set of disjunctive rules and $\mathcal{Q}$ be a UCQ. Then, for any fact base F, holds $F, \mathcal{R} \models \mathcal{Q}$ iff there is a piece-rewriting $\mathcal{Q}'$ of $\mathcal{Q}$ such that $F \models \mathcal{Q}'$.*

Proof. (Sketch) We show that there is a derivation of $(\{F\}, \mathcal{R})$ leading to an $\mathcal{F}_i$ such that $\mathcal{F}_i \models \mathcal{Q}$ iff there is a rewriting $\mathcal{Q}_j$ of $\mathcal{Q}$ with $\mathcal{R}$ such that $F \models \mathcal{Q}_j$ (with moreover $j \leq i$). This equivalence relies on the following two lemmas, which are corollaries of previous Lemmas 1 and 2. Given any Boolean UCQ $\mathcal{Q}$, disjunctive rule R and fact base F, the following holds (see Figure 1):

- (Backward-forward Lemma) For any disjunctive piece-unifier $\mu_\vee$ of $\mathcal{Q}$ with R, if $F \models \beta_\vee(\mathcal{Q}, R, \mu_\vee)$ then there is a trigger (R, h) on F such that $\alpha_\vee(F, R, h) \models \mathcal{Q}$;
- (Forward-backward Lemma) For any trigger (R, h) on F, if $\alpha_\vee(F, R, h) \models \mathcal{Q}$ then either $F \models \mathcal{Q}$ or there is a disjunctive piece-unifier $\mu_\vee$ of $\mathcal{Q}$ with R, such that $F \models \beta_\vee(\mathcal{Q}, R, \mu_\vee)$.

The $(\Rightarrow)$ direction of the theorem is proved by induction on the length k of a derivation from $\{F\}$ to $\mathcal{F}_k$ such that $\mathcal{F}_k \models \mathcal{Q}$, using forward-backward Lemma (which itself follows from Lemma 2 (Point 1) and Lemma 1 (Point 2)). The $(\Leftarrow)$ direction is proved by induction on the length k of a rewriting sequence from $\mathcal{Q}$ to $\mathcal{Q}_k$ such that $F \models \mathcal{Q}_k$, using backward-forward Lemma (which itself follows from Lemma 2 (Point 2) and Lemma 1 (Point 1)). □

To actually compute a UCQ-rewriting of $\mathcal{Q}$ when one exists, it is convenient to proceed in a breadth-first manner, i.e., extend $\mathcal{Q}$ at each step with all the CQs that can be generated with (new) disjunctive piece-unifiers. More specifically, we inductively define the following operator W, which takes as input a UCQ $\mathcal{Q}$ and a disjunctive rule set $\mathcal{R}$, and returns a possibly infinite set of CQs:

- $W_0(\mathcal{Q}, \mathcal{R}) = \mathcal{Q}$

- For $i > 0$, $W_i(\mathcal{Q},\mathcal{R}) = W_{i-1}(\mathcal{Q},\mathcal{R}) \cup \{\beta_\vee(W_{i-1}(\mathcal{Q},\mathcal{R}), R, \mu_\vee) | \mu_\vee \text{ piece-unifier with } R \in \mathcal{R}\}$
- Finally, $W(\mathcal{Q},\mathcal{R}) = \bigcup_{i\in\mathbb{N}} W_i(\mathcal{Q},\mathcal{R})$.

Proposition 1 (Properties of W)**.** *For any UCQ $\mathcal{Q}$ and disjunctive rule set $\mathcal{R}$, the following holds:*

1. *$W(\mathcal{Q},\mathcal{R})$ is a complete rewriting of $(\mathcal{Q},\mathcal{R})$.*
2. *If $(\mathcal{Q},\mathcal{R})$ admits a UCQ-rewriting $\mathcal{Q}'$, then there is $i \geq 0$ such that $\mathcal{Q}' \equiv W_i(\mathcal{Q},\mathcal{R})$.*

Proof. (1) Each $W_i(\mathcal{Q},\mathcal{R})$ is a piece-rewriting of $\mathcal{Q}$ with $\mathcal{R}$ and, for any piece-rewriting $\mathcal{Q}'$ of $\mathcal{Q}$ with $\mathcal{R}$, there is i such that $\mathcal{Q}' \subseteq W_i(\mathcal{Q},\mathcal{R})$. Hence, the union of all the $W_i(\mathcal{Q},\mathcal{R})$ is a complete rewriting of $\mathcal{Q}$. (2) If $(\mathcal{Q},\mathcal{R})$ admits a UCQ-rewriting $\mathcal{Q}'$, then by Theorem 3 it admits a complete piece-rewriting $\mathcal{Q}''$, and both are necessarily equivalent. Then, $\mathcal{Q}'' \subseteq W_i(\mathcal{Q},\mathcal{R})$ for some i and, since $\mathcal{Q}''$ is complete, $\mathcal{Q}'' \equiv W_i(\mathcal{Q},\mathcal{R})$. □

We propose a query rewriting algorithm (see Algorithm 1) that mimics the computation of $W(\mathcal{Q},\mathcal{R})$, while including two optimizations at each step $i > 0$. First, it only considers *new* disjunctive piece-unifiers, i.e., those that involve at least one CQ generated at step $i-1$. Second, it removes redundant CQs in the rewriting under construction, by the computation of a cover. More specifically, $\mathcal{Q}^\star$ denotes the rewriting under construction and $\mathcal{Q}_{new}$ the set of CQs generated at a given step. The function `cover` (Lines 1 and 6) returns a cover of the given set. The function `generate` (Line 5) takes as input the current rewriting $\mathcal{Q}^\star$, its subset $\mathcal{Q}_{prev}$ of CQs generated at the previous step, as well as $\mathcal{R}$, and returns the set of generated CQs, i.e., all the $\beta_\vee(\mathcal{Q}^\star, R, \mu_\vee)$ where $\mu_\vee$ is a new disjunctive piece-unifier. This yields the set $\mathcal{Q}_{new}$. To compute a cover of $\mathcal{Q}^\star \cup \mathcal{Q}_{new}$, priority is given to $\mathcal{Q}^\star$ in case of query equivalence, for termination reasons. The function `removeMoreSpecific` takes as input two sets of CQs and returns the first set minus its queries more specific than a query of the second set. The computation of a cover of $\mathcal{Q}^\star \cup \mathcal{Q}_{new}$ is decomposed into three steps (Lines 6-8): compute a cover of $\mathcal{Q}_{new}$; remove from $\mathcal{Q}_{new}$ the queries more specific than a query from $\mathcal{Q}^\star$; and remove from $\mathcal{Q}^\star$ the queries more specific than a query from $\mathcal{Q}_{new}$. Then, $\mathcal{Q}_{new}$ is added to $\mathcal{Q}^\star$ (Line 9). We remind that a query may have rewritings of unbounded size but still a UCQ-rewriting (see Example 2), hence the role of the cover computation is not only to remove redundancies but also to ensure that the algorithm halts when a UCQ-rewriting has been found.

The correctness of the algorithm is based on the soundness and completeness of the W operator, however attention should be paid to the potential impact of query removal on completeness (Lines 6 to 8). Indeed, when a CQ Q_2 is removed because it is more specific than another CQ Q_1, we have to ensure that any CQ that could be generated using Q_2 is more specific than another CQ already present in the curent rewriting, or than a CQ that can be generated using Q_1. Fortunately, this property is ensured by Lemma 1 (Point 2), considering $\mathcal{Q}^\star$ and $\mathcal{Q}_{new}$ at the end of Line 5, then taking $\mathcal{Q}_2 = \mathcal{Q}^\star \cup \mathcal{Q}_{new}$ and $\mathcal{Q}_1 = \mathcal{Q}_2 \setminus \{Q_2\}$.

Algorithm 1: BREADTH-FIRST REWRITING

Data: UBCQ $\mathcal{Q}$ and set of disjunctive rules $\mathcal{R}$
Result: A sound and complete rewriting of $\mathcal{Q}$

1 $\mathcal{Q}_{new} \leftarrow$ `cover`$(\mathcal{Q})$; *// new CQs*
2 $\mathcal{Q}^\star \leftarrow \mathcal{Q}_{new}$; *// result*
3 **while** $\mathcal{Q}_{new} \neq \emptyset$ **do**
4 $\quad \mathcal{Q}_{prev} \leftarrow \mathcal{Q}_{new}$ *// CQs from the preceding step*
5 $\quad \mathcal{Q}_{new} \leftarrow$ `generate`$(\mathcal{Q}^\star, \mathcal{Q}_{prev}, \mathcal{R})$; *// new CQs*
6 $\quad \mathcal{Q}_{new} \leftarrow$ `cover`$(\mathcal{Q}_{new})$
7 $\quad \mathcal{Q}_{new} \leftarrow$ `removeMoreSpecific`$(\mathcal{Q}_{new}, \mathcal{Q}^\star)$
8 $\quad \mathcal{Q}^\star \leftarrow$ `removeMoreSpecific`$(\mathcal{Q}^\star, \mathcal{Q}_{new})$
9 $\quad \mathcal{Q}^\star \leftarrow \mathcal{Q}^\star \cup \mathcal{Q}_{new}$
10 **return** $\mathcal{Q}^\star$

Theorem 4. *Algorithm 1 computes a sound and complete rewriting. Moreover, it halts and outputs a minimal rewriting when $(\mathcal{Q},\mathcal{R})$ is UCQ-rewritable.*

Proof. By induction on the number of iterations of the while loop, we prove the following invariant of the algorithm, using Lemma 1 (Point 2): after step i, $\mathcal{Q}^\star$ is equivalent to $W_i(\mathcal{Q},\mathcal{R})$. Then, soundness and completeness follow from Proposition 1. Line 7 ensures that $\mathcal{Q}_{new}$ becomes empty when $\mathcal{Q}^\star$ is a complete rewriting. Since a cover of $\mathcal{Q}^\star$ is computed at each step, the output set is of minimal size. □

Further remarks on completeness. When it comes to practical implementations, one may find simpler to rely on (conjunctive) piece-unifiers that unify the smallest possible subsets of a CQ. Such piece-unifiers are called *single-piece* (König et al. 2015). In the specific case of datalog, a single-piece unifier unifies a single atom of a CQ with a rule head. Piece-rewriting restricted to single-piece unifiers is complete for conjunctive rules (König et al. 2015), but it is no longer so with disjunctive rules. This occurs already in the case of disjunctive datalog, as illustrated next.

Example 5. *Consider again the colorability example (Ex. 1) with $R = v(x) \rightarrow g(x) \vee r(x)$ and $\mathcal{Q} = \{Q_1, Q_2\}$ with $Q_1 = \{g(u), e(u,w), g(w)\}$ and $Q_2 = \{r(u), e(u,w), r(w)\}$. With single-piece unifiers we obtain CQs that have the shape of "chains" with a g-atom or an r-atom at each extremity. However, there are also rewritings without any occurrence of g nor r, and the only way of obtaining them is to unify two query atoms together. For instance, the CQ $\{v(u), e(u,u)\}$ is obtained by unifying, on the one hand both g-atoms of a safe copy of Q_1 with $g(x)$, and on the other hand both r-atoms of a safe copy of Q_2 with $r(x)$. More generally, using such piece-unifiers, one can produce all the CQs that describe the odd-length cycles in the graph. Note that these CQs are incomparable with the CQs generated with single-piece unifiers. This example also shows that a UCQ may have no UCQ-rewriting although each of its CQs has one (which is here the CQ itself).*

Related work. To the best of our knowledge, (Alfonso, Chortaras, and Stamou 2021) is the only previous work proposing a UCQ rewriting technique for general disjunctive existential rules. This technique is based on a restricted form of first-order resolution, where at each step a CQ is unified with a disjunct of a rule head (using a conjunctive piece-unifier), which produces a new disjunctive rule with fewer disjunctions; when the unified rule is conjunctive, (the negation of) a CQ is produced. In comparison, the main advantages of our proposal are the following: *(1)* a rewriting step directly produces a CQ and not a rule, *(2)* intermediate rules, which may not lead to a CQ, are avoided, and *(3)* there is a direct correspondence between a chase step and a rewriting step, which makes it easier to study the properties of query rewriting, especially as the rule set is not updated.

4 What are *fus* Disjunctive Rules?

We now address the question of identifying classes of disjunctive rules that are UCQ-rewritable. By extension of the term coined for conjunctive existential rules, we also call them *fus*. To the best of our knowledge, the only *fus* class of disjunctive rules mentioned in the literature (Alfonso, Chortaras, and Stamou 2021) is actually a slight extension of *fus* conjunctive rules: this class consists of disjunctive rules with an empty frontier and it is shown that such rules can be safely added to a set of *fus* conjunctive rules. As a matter of fact, known *fus* classes of conjunctive rules do not seem to be extensible to the disjunctive case. And worse, the straightforward extension of syntactic criteria that underlie *fus* in the conjunctive case seems to easily lead to undecidability of query answering, as shown for example in (Morak 2021) for the syntactic restriction called stickiness (Calì, Gottlob, and Pieris 2010).

At first glance, one may expect *nonrecursive* disjunctive rule sets to be *fus*, as it happens for conjunctive rules. However, it is not the case, as shown by the next example: a CQ (on unary predicates) may have no UCQ-rewriting even with a *single non-recursive body-atomic* (disjunctive) *datalog* rule.

Example 6. *Let the rule $R = p(x, y) \to t_1(x) \vee t_2(y)$ and the BCQ $Q = \{t_1(u), t_2(u)\}$. Then the pair $(\{Q\}, \{R\})$ has no UCQ-rewriting. Indeed, a complete rewriting contains all the CQs of the following shape for any $n \in \mathbb{N}$:*

$$t_2(u_0) \wedge \left(\bigwedge_{i=1}^{n} p(u_{i-1}, u_i) \right) \wedge t_1(u_n)$$

All these queries are pairwise incomparable w.r.t. homomorphism. Let us detail the first rewriting step. To unify $\{Q\}$ with R, we have to make two safe copies of Q, let Q_1 and Q_2, which are respectively unified with $t_1(x)$ and $t_2(y)$. This produces the CQ $\{t_2(x), p(x, y), t_1(y)\}$, isomorphic to $\{t_2(u_0), p(u_0, u_1), t_1(u_1)\}$. If we switch the unified atoms of $\mathtt{head}(R)$, we obtain an isomorphic CQ. All subsequent rewriting steps lead to longer paths of p-atoms.

A similar observation follows from (Gerasimova et al. 2020), which focuses on a specific disjunctive rule of the form $A(x) \to T(x) \vee F(x)$, called a covering axiom and denoted by cov_A; their complexity results imply that the singleton set $\{cov_A\}$ is not *fus*,[2] which can be checked for instance by considering the query $Q = \{T(u), p(u, v), F(v)\}$.

Next, we show that such observations can be generalized to almost *any* source-to-target disjunctive rule. Evidently, we have to exclude disjunctive rules that are equivalent to a conjunctive rule, as classes of *fus* conjunctive rules are known. We also exclude *disconnected* rules, i.e., rules R such that $\mathtt{body}(R) \cup \mathtt{head}(R)$ is not a connected set of atoms (where connectivity is defined in the obvious way based on shared variables). Note that a rule with a head H_i that has an empty frontier is disconnected, as well as a rule whose body has a connected component with an empty frontier. However, a rule with a disconnected body may not be disconnected, since head atoms may connect several connected components of the body (e.g., a "product" rule like $b_1(x) \wedge b_2(y) \to t_1(x) \vee t_2(y) \vee p(x, y)$ is not disconnected).

Example 7 (*Fus* disconnected rule)**.** *Let the disconnected rule $R = b(x) \to t_1(x) \vee \exists z\ t_2(z)$. R is not equivalent to a conjunctive rule. Let us check that it is* fus*. Given any UCQ $\mathcal{Q}$, let $\mathcal{Q}_2$ be the subset of $\mathcal{Q}$ that contains all the CQs that can be unified with $\exists z\ t_2(z)$. Any $Q \in \mathcal{Q}_2$ necessarily contains a disconnected component of the form $\exists u\ t_2(u)$. Moreover, it is useless to unify Q with $t_1(x)$: in such case, let Q_2 be the CQ unified with $\exists z\ t_2(z)$, then the obtained rewriting is more specific than Q_2. Hence, we can ignore all the produced CQs that contain a connected component of the form $\exists u\ t_2(u)$. Rewriting $\mathcal{Q}$ with $\{R\}$ amounts to rewriting $\mathcal{Q} \setminus \mathcal{Q}_2$ with the conjunctive rule set $\mathcal{R} = \{b(x) \wedge (Q_2 \setminus \{\exists u\ t_2(u)\}) \to t_1(x) \mid Q_2 \in \mathcal{Q}_2\}$, which belongs to the* fus *class called* domain restricted *(Baget et al. 2011).*

In the next theorem, we restrict the head of the rule to a disjunction of two atom sets, to keep the proof simple.

Theorem 5. *Let $R = B \to H_1 \vee H_2$ be a source-to-target rule that is not disconnected nor equivalent to a conjunctive rule. Then, there is a CQ Q such that $(\{Q\}, \{R\})$ is not UCQ-rewritable.*

Proof. (Sketch) Let $R = B[\mathbf{x_1}, \mathbf{x_2}, \mathbf{y}] \to \exists \mathbf{z_1}\ H_1[\mathbf{x_1}, \mathbf{z_1}] \vee \exists \mathbf{z_2}\ H_2[\mathbf{x_2}, \mathbf{z_2}]$, where:

- $\mathtt{fr}(R) = \mathbf{x_1} \cup \mathbf{x_2}$; $\mathbf{x_1}$ and $\mathbf{x_2}$ may share variables;
- $\mathbf{x_i} \neq \emptyset$ $(i = 1, 2)$ since R is not disconnected.

We build the following (Boolean) CQ:

$$Q = \{H_1^s[\mathbf{v_1}, \mathbf{w_1}], p(\mathbf{v_1}, \mathbf{v_2}), H_2^s[\mathbf{v_2}, \mathbf{w_2}]\}$$

where each $H_i^s[\mathbf{v_i}, \mathbf{w_i}]$ is a safe copy of $H_i[\mathbf{x_i}, \mathbf{z_i}]$ and p is a fresh predicate. Note that, since R is connected, both H_1

[2] That paper studies syntactic conditions on ontology-mediated CQs of the form (Q, cov_A) that determine the data complexity of query answering and the rewritability in some target query language. In particular, it is shown that if a (connected) CQ Q has no term x with both atoms $T(x)$ and $F(x)$ and contains at least one F-atom and one T-atom then answering (Q, cov_A) is L-hard for data complexity. Since answering a UCQ-rewritable ontology-mediated query is in AC^0 for data complexity, and $AC^0 \subset L$, it follows that no cov_A is fus.

and H_2 have a frontier variable, and frontier variables being renamed in each H_i^s, the arity of p is at least 2. In $p(\mathbf{v_1}, \mathbf{v_2})$ the order on the variables is important: a fixed order is chosen on $\mathbf{x_i}$ (hence, $\mathbf{v_i}$) and the tuple $\mathbf{v_1}$ comes before the tuple $\mathbf{v_2}$. Hence, $p(\mathbf{v_1}, \mathbf{v_2})$ can be seen as "directed" from $\mathbf{v_1}$ to $\mathbf{v_2}$. We then proceed in two steps.

1. We show that we can produce an infinite set $\mathcal{Q}$ whose element CQs are pairwise incomparable by homomorphism. Let $Q_0 = Q$. At each step $i \geq 1$, Q_i is produced from a safe copy of Q unified with H_1 and a safe copy of Q_{i-1} unified with H_2. The piece-unifiers unify H_1^s (resp. H_2^s) in Q (resp. Q_{i-1}) according to the isomorphism from H_1^s (resp. H_2^s) to H_1 (resp. H_2). Any CQ Q_k in $\mathcal{Q}$ is connected and follows the "pattern" $H_2^s.p.(B.p)^k.H_1^s$, where occurrences of p-atoms all have the same direction; hence, two "adjacent" p-atoms, i.e., that share variables with the same copy B_i of a B, cannot be mapped one onto the other (by a homomorphism that maps B_i to itself).
2. We show that no CQ Q' that can be produced by piece-rewriting maps by homomorphism to a CQ from $\mathcal{Q}$, except by isomorphism. When there is no (conjunctive) piece-unifier that unifies $H_1[\mathbf{v_1}, \mathbf{w_1}]$ in Q with $H_2[\mathbf{x_2}, \mathbf{z_2}]$ (the same holds if we exchange H_1 and H_2), all the produced Q' are more specific than (including isomorphic to) CQs from $\mathcal{Q}$. Otherwise, assume that a CQ Q' is produced by unifying $H_1[\mathbf{v_1}, \mathbf{w_1}]$ with $H_2[\mathbf{x_2}, \mathbf{z_2}]$. If Q' can be mapped by homomorphism to a $Q_n \in \mathcal{Q}$, the arguments of any p-atom in Q' must be pairwise distinct variables. We show that it leads to have R equivalent to the conjunctive rule $B \to H_i$ (with $i = 1$ or $i = 2$), which contradicts the hypothesis on R.

It follows that $\mathcal{Q}$ is a subset of any sound and complete rewriting of $\{Q\}$ with $\{R\}$, hence the pair $(\{Q\}, \{R\})$ does not admit a UCQ-rewriting. □

One interest of the above proof is to provide a general construction that applies to any rule (fulfilling the conditions of the theorem). Also, the proof can be generalized to a rule head with k disjuncts, taking Q containing a safe copy of each H_i plus a p-atom that connects these copies through their frontier variables.

Given this result, the notion of *fus* disjunctive rules does not seem to be particularly relevant. Studying the problem of deciding whether a pair $(\mathcal{Q}, \mathcal{R})$ is UCQ-rewritable seems more interesting, although it is known to be undecidable already for (conjunctive) datalog rules.[3] Again, little is known about classes of disjunctive rules and UCQs for which this problem would be decidable. Let us point out a few immediate cases of UCQ-rewritable pairs $(\mathcal{Q}, \mathcal{R})$:

- $\mathcal{Q}$ is composed of atomic CQs and $\mathcal{R}$ is a set of disjunctive linear existential rules (i.e., rules with an atomic body). Indeed, only atomic CQs can be produced, and there is a finite number of them on a given set of predicates. This case was already noticed in (Bourhis et al. 2016).
- $\mathcal{Q}$ is composed of atomic queries and $\mathcal{R}$ is a set of $\mathcal{S}$-to-$\mathcal{T}$ rules. The produced CQs are obtained from the rule bodies by specializing their frontier (i.e., merging variables and replacing them by constants occurring in $\mathcal{Q}$ and rule heads). Hence, there is a finite number of them.
- $\mathcal{Q}$ is composed of variable-free CQs[4] and $\mathcal{R}$ is a set of lossless existential rules (i.e., such that all the variables in a rule body are frontier). Then, no variable is introduced by rewriting, hence the number of terms in a CQ is bounded by $|\texttt{consts}(\mathcal{Q}) \cup \texttt{consts}(\mathcal{R})|$.

5 Disjunctive Mappings

We now consider UCQ-rewritability with (disjunctive) mappings. Let $\mathcal{S}$ and $\mathcal{T}$ be the sets of source and target predicates, respectively, and let $\mathcal{M}$ be a mapping on $(\mathcal{S}, \mathcal{T})$. Given a query on $\mathcal{T}$, the aim is to obtain a complete rewriting w.r.t. fact bases on $\mathcal{S}$. Because $\mathcal{S}$ and $\mathcal{T}$ are disjoint, CQs that contain atoms on $\mathcal{T}$ are useless in a rewriting. Hence, we define a *mapping rewriting* as a rewriting on $\mathcal{S}$ and use the notation *$\mathcal{S}$-rewriting* to distinguish it from a rewriting on $\mathcal{S} \cup \mathcal{T}$. An $\mathcal{S}$-rewriting $\mathcal{Q}'$ of a UCQ $\mathcal{Q}$ with $\mathcal{M}$ is *complete* if, for all fact base F on $\mathcal{S}$, if $F, \mathcal{M} \models \mathcal{Q}$ then $F \models \mathcal{Q}'$. A finite complete $\mathcal{S}$-rewriting is called a *UCQ-$\mathcal{S}$-rewriting*.

Example 8 (Colorability)**.** *We adapt Example 5 to transform the rule into a mapping. Let $\mathcal{S} = \{v, e\}$, $\mathcal{T} = \{\hat{e}, g, r\}$ and $\mathcal{M} = \{m_1, m_2\}$, with:*
$m_1 = e(x, y) \to \hat{e}(x, y)$
$m_2 = v(x) \to g(x) \vee r(x)$.
Let $\mathcal{Q} = \{Q_1, Q_2\}$ with $Q_1 = \{g(u), \hat{e}(u, w), g(w)\}$ and $Q_2 = \{r(u), \hat{e}(u, w), r(w)\}$. Any complete $\mathcal{S}$-rewriting of $\mathcal{Q}$ contains CQs that describe all the cycles of odd length (in other words, it defines non-2-colorability). All the other CQs that can be produced by piece-rewriting contain predicates g and r, hence are discarded.

Note that a query may have a UCQ-$\mathcal{S}$-rewriting, while it does not have any UCQ-rewriting (on $\mathcal{S} \cup \mathcal{T}$), as illustrated by the next example.

Example 9. *Let $\mathcal{S} = \{p\}$ and $\mathcal{T} = \{t_1, t_2\}$. Consider the (Boolean) CQ $Q = \{t_1(u), t_2(u)\}$ and the rule $R = p(x, y) \to t_1(x) \vee t_2(y)$ from Example 6. While the pair $(\{Q\}, \{R\})$ has no UCQ-rewriting, it has a UCQ-$\mathcal{S}$-rewriting, which is empty. Indeed, all the CQs that can be obtained by piece-rewriting contain an atom on $\mathcal{T}$.*

Let *disjunctive mapping rewritability* be the following problem: Given a disjunctive mapping $\mathcal{M}$ on $(\mathcal{S}, \mathcal{T})$ and a UCQ $\mathcal{Q}$ on $\mathcal{T}$, does $(\mathcal{Q}, \mathcal{M})$ have a UCQ-$\mathcal{S}$-rewriting ?

Theorem 6. *Disjunctive mapping rewritability is undecidable.*

Proof. (Sketch) We build a reduction from the following undecidable problem: Given a (Boolean) CQ Q and a set of

[3] This follows from the undecidability of determining whether a datalog program is uniformly bounded (Gaifman et al. 1993). Indeed, a datalog program $\mathcal{R}$ is uniformly bounded iff the pair $(Q, \mathcal{R})$ is UCQ-rewritable for any *full* atomic query Q. In turn, UCQ-rewritability of $(Q, \mathcal{R})$ can be reduced to UCQ-rewritability of $(Q', \mathcal{R})$ with Q' a Boolean CQ.

[4] If non-Boolean CQs are considered, $\mathcal{Q}$ can be extended to a set of full CQs.

(conjunctive) datalog rules $\mathcal{R}$, is the pair $(\{Q\}, \mathcal{R})$ UCQ-rewritable? W.l.o.g. we assume that rules in $\mathcal{R}$ have no constants (and an atomic head). The reduction translates each instance $(Q, \mathcal{R})$ defined on a set of predicates $\mathcal{P}$, into an instance $(\mathcal{Q}^{Q,\mathcal{R}}, \mathcal{M}^{Q,\mathcal{R}})$ of the disjunctive mapping rewritability problem, defined on a pair of predicats sets $(\mathcal{S}, \mathcal{T})$ such that:

- $\mathcal{S} = \mathcal{P} \cup \{T\}$, where T is a fresh unary predicate,
- $\mathcal{T}$ is the union of: (1) a set of predicates in bijection with $\mathcal{S}$, where $\hat{p}$ denotes the predicate obtained from $p \in \mathcal{S}$, and (2) a set of fresh predicates in bijection with $\mathcal{R}$, where p_{R_i} denotes the predicate associated with the rule R_i; the arity of each p_{R_i} is $|\mathtt{fr}(R_i)|$.

Given a conjunction Q (on $\mathcal{P}$), we denote by Q^T the conjunction (on $\mathcal{S}$) obtained from Q by adding a T-atom on each term; given a conjunction Q (on $\mathcal{S}$), we denote by $\hat{Q}$ the conjunction (on $\mathcal{T}$) obtained from Q by renaming all the predicates p into $\hat{p}$. Hence, $\widehat{Q^T}$ is obtained by performing the first operation, then the second. Given $\mathbf{x} = x_1, \ldots, x_n$, $T[\mathbf{x}]$ denotes the conjunction $T(x_1) \wedge \cdots \wedge T(x_n)$. Similarly, $\hat{T}[\mathbf{x}] = \hat{T}(x_1) \wedge \cdots \wedge \hat{T}(x_n)$.

Let Q and $\mathcal{R} = \{R_1, \ldots, R_n\}$, where $R_i = B_i[\mathbf{x_i}, \mathbf{y_i}] \rightarrow H_i[\mathbf{x_i}]$. The instance $(\mathcal{Q}^{Q,\mathcal{R}}, \mathcal{M}^{Q,\mathcal{R}})$ is defined as follows:

- $\mathcal{Q}^{Q,\mathcal{R}} = \{Q_Q\} \cup \mathcal{Q}_\mathcal{R}$ with:
  $Q_Q = \widehat{Q^T}$,
  $\mathcal{Q}_\mathcal{R} = \{Q_{R_i} = \exists \mathbf{x_i}, \mathbf{y_i}\ \widehat{(B_i)^T}[\mathbf{x_i}, \mathbf{y_i}] \wedge p_{R_i}(\mathbf{x_i}) | R_i \in \mathcal{R}\}$
- $\mathcal{M}^{Q,\mathcal{R}} = \mathcal{M}_\mathcal{R} \cup \mathcal{M}_{trans}$ with:
  $\mathcal{M}_\mathcal{R} = \{m_{R_i} = T[\mathbf{x_i}] \rightarrow p_{R_i}(\mathbf{x_i}) \vee \hat{H}_i(\mathbf{x_i}) \mid R_i \in \mathcal{R}\}$
  $\mathcal{M}_{trans} = \{p(\mathbf{x}) \rightarrow \hat{p}(\mathbf{x}) \mid p \in \mathcal{S}\}$

Based on the natural bijection between the CQs $Q_\mathcal{P}$ defined on $\mathcal{P}$ and the CQs $(Q_\mathcal{P})^T$ defined on $\mathcal{S}$, we prove that $Q_\mathcal{P}$ belongs to a rewriting of $\{Q\}$ with $\mathcal{R}$ iff $(Q_\mathcal{P})^T$ belongs to a rewriting of $\mathcal{Q}^{Q,\mathcal{R}}$ with $\mathcal{M}^{Q,\mathcal{R}}$. Note that set membership is up to isomorphism throughout the proof. More specifically, we first prove the following lemmas:

1. For any CQ Q_w in a piece-rewriting of $\{Q\}$ with $\mathcal{R}$, $(Q_w)^T$ belongs to a piece-rewriting of $\mathcal{Q}^{Q,\mathcal{R}}$ with $\mathcal{M}^{Q,\mathcal{R}}$. Indeed, to each R_i are associated a CQ Q_{R_i} and a rule m_{R_i} that allow to simulate any rewriting step performed with R_i, using fresh predicate p_{R_i}.
2. Any CQ Q_S in an $\mathcal{S}$-rewriting of $\mathcal{Q}^{Q,\mathcal{R}}$ with $\mathcal{M}^{Q,\mathcal{R}}$ is of the form $Q_S = (Q_\mathcal{P})^T$, with $Q_\mathcal{P}$ the subset of Q_S on $\mathcal{P}$.
3. For any CQ of the form $(Q_\mathcal{P})^T$, with $Q_\mathcal{P}$ on $\mathcal{P}$, that belongs a piece-rewriting of $\mathcal{Q}^{Q,\mathcal{R}}$ with $\mathcal{M}^{Q,\mathcal{R}}$, $Q_\mathcal{P}$ belongs to a piece-rewriting of $\{Q\}$ with $\mathcal{R}^\star$, where $\mathcal{R}^\star$ is the reflexive and transitive closure of $\mathcal{R}$ by unfolding (i.e., rule composition). Note that $\mathcal{R}^\star$ is logically equivalent to $\mathcal{R}$.

We rely on these lemmas to prove the following: if there is a UCQ-rewriting of $(\{Q\}, \mathcal{R})$ then there is a UCQ-$\mathcal{S}$-rewriting of $(\mathcal{Q}^{Q,\mathcal{R}}, \mathcal{M}^{Q,\mathcal{R}})$. The proof of the opposite direction is similar. Let $\mathcal{Q}$ be a UCQ-rewriting of $(\{Q\}, \mathcal{R})$. Then there is a piece-rewriting $\mathcal{Q}_i$ of $\{Q\}$ with $\mathcal{R}$ such that $\mathcal{Q}_i \equiv \mathcal{Q}$. By Lemma 1, there is a piece-rewriting $\mathcal{Q}_j$ of $\mathcal{Q}^{Q,\mathcal{R}}$ with $\mathcal{M}^{Q,\mathcal{R}}$ that contains all the CQs of the form $(Q_w)^T$ in bijection with the Q_w in $\mathcal{Q}_i$. By definition, $\mathcal{Q}_j$ is a finite rewriting of $(\mathcal{Q}^{Q,\mathcal{R}}, \mathcal{M}^{Q,\mathcal{R}})$ and the subset $\mathcal{Q}_j^\mathcal{S}$ of $\mathcal{Q}_j$ that contains only the CQs on $\mathcal{S}$ is a finite $\mathcal{S}$-rewriting of $(\mathcal{Q}^{Q,\mathcal{R}}, \mathcal{M}^{Q,\mathcal{R}})$. Now, assume $\mathcal{Q}_j^\mathcal{S}$ is not complete, i.e., there is a CQ that belongs to an $\mathcal{S}$-rewriting of $(\mathcal{Q}^{Q,\mathcal{R}}, \mathcal{M}^{Q,\mathcal{R}})$ but that is not more specific than a CQ in $\mathcal{Q}_j^\mathcal{S}$; by Lemma 2, such CQ is of the form $(Q_\mathcal{P})^T$. Then there is a piece-rewriting $\mathcal{Q}'_j$ of $\mathcal{Q}^{Q,\mathcal{R}}$ with $\mathcal{M}^{Q,\mathcal{R}}$ that contains a CQ entailed by $(Q_\mathcal{P})^T$; hence such CQ is also on $\mathcal{S}$, and by Lemma 2 it is of the form $(Q'_\mathcal{P})^T$. By Lemma 3, $Q'_\mathcal{P}$ belongs to a piece-rewriting of $\{Q\}$ with $\mathcal{R}^\star$. Since $\mathcal{R}^\star \equiv \mathcal{R}$, there is a CQ equivalent to $Q'_\mathcal{P}$ in some rewriting of $(\{Q\}, \mathcal{R})$. Since $\mathcal{Q}_i$ is complete, there is $Q_c \in \mathcal{Q}_i$ such that $Q'_\mathcal{P} \models Q_c$. Hence, $(Q'_\mathcal{P})^T \models (Q_c)^T$, so $(Q_\mathcal{P})^T \models (Q_c)^T$; by Lemma 1, $(Q_c)^T \in \mathcal{Q}_j$, hence $(Q_c)^T \in \mathcal{Q}_j^\mathcal{S}$, which contradicts the fact that $(Q_\mathcal{P})^T$ is not more specific than a CQ in $\mathcal{Q}_j^\mathcal{S}$. □

6 Perspectives

In conclusion, UCQ rewriting with disjunctive existential rules appears to be extremely challenging. The main classes that ensure termination for conjunctive rules fail to be generalized. As suggested by previous work in (Gerasimova et al. 2020) and our Theorem 5, the *fus* notion applied to disjunctive rules does not seem to add much w.r.t. *fus* conjunctive rules. However, it might be more relevant in the context of mappings (when it becomes UCQ-$\mathcal{S}$-rewritability), which still has to be studied. Beside, a number of interesting issues remain open, in relationship with the finite rewritability of a pair $(\mathcal{Q}, \mathcal{R})$. We list here some of them:

1. Clarify the boundary between decidability and undecidability for the problem of determining whether a pair $(\mathcal{Q}, \mathcal{R})$ is UCQ-rewritable, according to specific classes of rules (and queries). In particular, UCQ-rewritability is decidable for guarded conjunctive rules and some of their generalizations (Barceló et al. 2018), does this extend to the disjunctive case?
2. We have shown that the UCQ-$\mathcal{S}$-rewritability of a pair $(\mathcal{Q}, \mathcal{M})$ is undecidable (Theorem 6). Is it still the case for a pair $(\{Q\}, \mathcal{M})$ where Q is a CQ?
3. Our undecidability proof for UCQ-$\mathcal{S}$-rewritability (Theorem 6) exploits the fact that rewritings are restricted to predicates in $\mathcal{S}$. If we consider instead UCQ-rewritings with source-to-target rules, we know that the problem can only be simpler, as there is an easy reduction from UCQ-rewritability with $\mathcal{S}$-to-$\mathcal{T}$-rules to UCQ-$\mathcal{S}$-rewritability with mappings (one simply has to add a mapping rule per target predicate to give it an existence at the source level). Is the UCQ-rewritability of a pair $(\mathcal{Q}, \mathcal{R})$ decidable when $\mathcal{R}$ is a set of $\mathcal{S}$-to-$\mathcal{T}$ rules?
4. Design an algorithm that, given a pair $(\mathcal{Q}, \mathcal{M})$, outputs a UCQ-$\mathcal{S}$-rewriting for this pair when one exists.

Acknowledgements

This work is partly supported by the ANR project CQFD (ANR-18-CE23-0003).

References

Ahmetaj, S.; Ortiz, M.; and Simkus, M. 2018. Rewriting guarded existential rules into small datalog programs. In *21st International Conference on Database Theory, ICDT 2018*, volume 98 of *LIPIcs*, 4:1–4:24.

Alfonso, E. M.; Chortaras, A.; and Stamou, G. 2021. Ucq-rewritings for disjunctive knowledge and queries with negated atoms. *Semantic Web* 12(4):685–709.

Alviano, M.; Faber, W.; Leone, N.; and Manna, M. 2012. Disjunctive datalog with existential quantifiers: Semantics, decidability, and complexity issues. *Theory Pract. Log. Program.* 12(4-5):701–718.

Arenas, M.; Pérez, J.; Reutter, J. L.; and Riveros, C. 2010. Foundations of schema mapping management. In *29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2010*, 227–238. ACM.

Arenas, M.; Pérez, J.; and Riveros, C. 2008. The recovery of a schema mapping: bringing exchanged data back. In *27th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2008*, 13–22. ACM.

Baader, F.; Brandt, S.; and Lutz, C. 2005. Pushing the EL envelope. In *19th International Joint Conference on Artificial Intelligence, IJCAI 2005*, 364–369.

Baget, J.-F.; Leclère, M.; Mugnier, M.-L.; and Salvat, E. 2009. Extending Decidable Cases for Rules with Existential Variables. In *21st International Joint Conference on Artificial Intelligence, IJCAI 2009*, 677–682.

Baget, J.; Leclère, M.; Mugnier, M.; and Salvat, E. 2011. On rules with existential variables: Walking the decidability line. *Artif. Intell.* 175(9-10):1620–1654.

Barceló, P.; Berger, G.; Lutz, C.; and Pieris, A. 2018. First-order rewritability of frontier-guarded ontology-mediated queries. In *27th International Joint Conference on Artificial Intelligence, IJCAI 2018*, 1707–1713. ijcai.org.

Beeri, C., and Vardi, M. Y. 1984. A proof procedure for data dependencies. *J. ACM* 31(4):718–741.

Bernstein, P. A., and Ho, H. 2007. Model management and schema mappings: Theory and practice. In *33rd International Conference on Very Large Data Bases, VLDB, 2007*, 1439–1440. ACM.

Bienvenu, M.; ten Cate, B.; Lutz, C.; and Wolter, F. 2014. Ontology-based data access: A study through disjunctive datalog, csp, and MMSNP. *ACM Trans. Database Syst.* 39(4):33:1–33:44.

Bourhis, P.; Manna, M.; Morak, M.; and Pieris, A. 2016. Guarded-Based Disjunctive Tuple-Generating Dependencies. *ACM Trans. Database Syst.* 41(4):27:1–27:45.

Buron, M.; Mugnier, M.; and Thomazo, M. 2021. Parallelisable existential rules: a story of pieces. In *18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021*, 162–173.

Calì, A.; Gottlob, G.; and Kifer, M. 2008. Taming the infinite chase: Query answering under expressive relational constraints. In *11th International Conference on Principles of Knowledge Representation and Reasoning, KR 2008*, 70–80. AAAI Press.

Calì, A.; Gottlob, G.; and Lukasiewicz, T. 2009. A General Datalog-Based Framework for Tractable Query Answering over Ontologies. In *28th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2009*, 77–86. ACM.

Calì, A.; Gottlob, G.; and Pieris, A. 2010. Advanced processing for ontological queries. *VLDB Endow.* 3(1):554–565.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family. *Journal of Automated Reasoning* 39(3):385–429.

Carral, D.; Dragoste, I.; and Krötzsch, M. 2017. Restricted Chase (Non)Termination for Existential Rules with Disjunctions. In *26th International Joint Conference on Artificial Intelligence, IJCAI 2017*, 922–928. ijcai.org.

Eiter, T.; Gottlob, G.; and Mannila, H. 1997. Disjunctive datalog. *ACM Trans. Database Syst.* 22(3):364–418.

Fagin, R.; Kolaitis, P. G.; Miller, R. J.; and Popa, L. 2005. Data exchange: semantics and query answering. *Theor. Comput. Sci.* 336(1):89–124.

Fagin, R.; Kolaitis, P. G.; Popa, L.; and Tan, W. C. 2008. Quasi-inverses of schema mappings. *ACM Trans. Database Syst.* 33(2):11:1–11:52.

Gaifman, H.; Mairson, H. G.; Sagiv, Y.; and Vardi, M. Y. 1993. Undecidable optimization problems for database logic programs. *J. ACM* 40(3):683–713.

Gerasimova, O.; Kikot, S.; Kurucz, A.; Podolskii, V. V.; and Zakharyaschev, M. 2020. A data complexity and rewritability tetrachotomy of ontology-mediated queries with a covering axiom. In *17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020*, 403–413.

Gottlob, G.; Manna, M.; Morak, M.; and Pieris, A. 2012. On the complexity of ontological reasoning under disjunctive existential rules. In *Mathematical Foundations of Computer Science 2012 - 37th International Symposium, MFCS 2012*, volume 7464 of *LNCS*, 1–18. Springer.

König, M.; Leclère, M.; Mugnier, M.; and Thomazo, M. 2015. Sound, complete and minimal ucq-rewriting for existential rules. *Semantic Web* 6(5):451–475.

Krötzsch, M.; Rudolph, S.; and Hitzler, P. 2006. On the complexity of Horn description logics. In *2nd Workshop on OWL: Experiences and Directions*, volume 216 of *CEUR WS Proceedings*. CEUR-WS.org.

Leclère, M.; Mugnier, M.; and Pérution-Kihli, G. 2023. Query rewriting with disjunctive existential rules and mappings. *CoRR* abs/2306.05973.

Lutz, C.; Toman, D.; and Wolter, F. 2009. Conjunctive Query Answering in the Description Logic $\mathcal{EL}$ Using a Re-

lational Database System. In *21st International Joint Conference on Artificial Intelligence, IJCAI 2009*, 2070–2075.

Morak, M. 2021. Sticky Existential Rules and Disjunction are Incompatible. In *18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021*, 691–695.

Pérez, J. 2013. The inverse of a schema mapping. In *Data Exchange, Integration, and Streams*, volume 5 of *Dagstuhl Follow-Ups*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. 69–95.

Poggi, A.; Lembo, D.; Calvanese, D.; Giacomo, G. D.; Lenzerini, M.; and Rosati, R. 2008. Linking Data to Ontologies. *J. Data Semant.* 10:133–173.

Salvat, E., and Mugnier, M. 1996. Sound and Complete Forward and Backward Chainings of Graph Rules. In *4th International Conference on Conceptual Structures, ICCS 1996*, volume 1115 of *Lecture Notes in Computer Science*, 248–262. Springer.

Schmidt-Schauß, M., and Smolka, G. 1991. Attributive concept descriptions with complements. *Artif. Intell.* 48(1):1–26.

W3C. 2009. *OWL 2 Web Ontology Language: Document Overview*. http://www.w3.org/TR/owl2-overview/.