Extremal Fitting Problems for Conjunctive Queries

Balder ten Cate
ILLC, Universiteit van Amsterdam
Netherlands
b.d.tencate@uva.nl

Victor Dalmau
Universitat Pompeu Fabra
Spain
victor.dalmau@upf.edu

Maurice Funk
Universität Leipzig
ScaDS.AI Center Dresden/Leipzig
Germany
mfunk@informatik.uni-leipzig.de

Carsten Lutz
Universität Leipzig
ScaDS.AI Center Dresden/Leipzig
Germany
clu@informatik.uni-leipzig.de

ABSTRACT

The *fitting problem* for conjunctive queries (CQs) is the problem to construct a CQ that fits a given set of labeled data examples. When a fitting CQ exists, it is in general not unique. This leads us to proposing natural refinements of the notion of a fitting CQ, such as *most-general fitting CQ*, *most-specific fitting CQ*, and *unique fitting CQ*. We give structural characterizations of these notions in terms of (suitable refinements of) homomorphism dualities, frontiers, and direct products, which enable the construction of the refined fitting CQs when they exist. We also pinpoint the complexity of the associated existence and verification problems, and determine the size of fitting CQs. We study the same problems for UCQs and for the more restricted class of tree CQs.

ACM Reference Format:
Balder ten Cate, Victor Dalmau, Maurice Funk, and Carsten Lutz. 2023. Extremal Fitting Problems for Conjunctive Queries. In *Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS '23), June 18–23, 2023, Seattle, WA, USA*. ACM, New York, NY, USA, 39 pages. https://doi.org/10.1145/3584372.3588655

1 INTRODUCTION

The *fitting problem* for conjunctive queries (CQs) is the problem to construct a CQ q that fits a given set of labeled data examples, meaning that q returns all positive examples as an answer while returning none of the negative examples. This fundamental problem has a long history in database research. It lies at the heart of the classic *Query-By-Example* paradigm that aims to assist users in query formation and query refinement, and has been intensively studied for CQs [4, 38, 44] and other types of queries (e.g., [3, 7, 20]). The fitting problem is also central to *Inductive Logic Programming* [21, 25], where CQs correspond to the basic case of non-recursive single-rule Datalog programs, and has close connections to fitting problems for *schema mappings* [2, 10]. More recent motivation comes from *automatic feature generation in machine learning with relational data* [5, 34]. Here, the CQ fitting problem arises because a CQ that separates positive from negative examples in (a sufficiently large subset of) a labeled dataset is a natural contender for being added as an input feature to the model [5]. In addition, there has been significant recent interest in fitting CQs and other queries in knowledge representation, typically in the presence of an ontology [9, 24, 31, 32, 37, 42].

When a fitting CQ exists, in general it need not be unique up to equivalence. In fact, there may be infinitely many pairwise non-equivalent fitting CQs. However, the fitting CQs form a *convex set*: whenever two CQs q_1, q_2 fit a set of labeled examples, then the same holds for every CQ q with $q_1 \subseteq q \subseteq q_2$, where "$\subseteq$" denotes query containment. Maximal elements of this convex set can be viewed as "most-general" fitting CQs, while minimal elements can be viewed as "most-specific" fitting CQs. The set of all most-general and all most-specific fitting CQs (when they exist), can thus be viewed as natural representatives of the entire set of fitting CQs, c.f. the version-space representation theorem used in machine learning [39, Chapter 2.5]. In the context of automatic feature generation mentioned above, it would thus be natural to compute all extremal fitting CQs and add them as features, especially when infinitely many fitting CQs exist. Likewise, in query refinement tasks where the aim is to construct a modified query that excludes unwanted answers or includes missing answers (cf. [44]), it is also natural to ask for a most-general, respectively, most-specific fitting query.

In this paper we embark on a systematic study of extremal fitting CQs. To the best of our knowledge, we are the first to do so. We show that the intuitive concepts of most-general and most-specific fitting CQs can be formalized in multiple ways. We give structural characterizations of each notion, study the associated verification, existence, and computation problems, and establish upper and lower bounds on the size of extremal fitting CQs. The characterizations link "weakly most-general" fittings to the notion of homomorphism frontiers, "complete bases" of most-general fittings to (a certain relativized version of) homomorphism dualities, and most-specific fittings to direct products. We use the structural characterizations to obtain effective algorithms and pinpoint the exact complexity of the decision and computation problems mentioned above, and to establish size bounds. Our algorithms use a combination of techniques from automata theory and from the literature on constraint satisfaction problems. We perform the same study for two other natural classes of database queries, namely *unions of conjunctive queries* (UCQs) and acyclic connected unary CQs, from now on referred to as *tree CQs*. For the latter class, which holds significance

PODS '23, June 18–23, 2023, Seattle, WA, USA

ACM ISBN 979-8-4007-0127-6/23/06.
https://doi.org/10.1145/3584372.3588655

	Verification	Existence	Construction and size
Any Fitting	DP-c (Thm 3.1)	coNExpTime-c [10, 46]	In ExpTime [46]; Exp size lower bound (Thm 3.26)
Most-Specific	NExpTime-c (Thm 3.7, 3.23)	coNExpTime-c [10, 46]	In ExpTime [46]; Exp size lower bound (Thm 3.26)
Weakly Most-General	NP-c (Thm 3.12)	ExpTime-c (Thm 3.13, 3.24)	In 2ExpTime (Thm 3.13); Exp size lower bound (Thm 3.26)
Basis of Most-General	NExpTime-c (Thm 3.17, 3.23)	NExpTime-c (Thm 3.17, 3.23)	In 3ExpTime (Thm 3.18); 2Exp size lower bound (Thm 3.27)
Unique	NExpTime-c (Thm 3.21, 3.23)	NExpTime-c (Thm 3.21, 3.23)	In ExpTime [46]; Exp size lower bound (Thm 3.26)

Table 1: Summary of results for CQs

	Verification	Existence	Construction and size
Any Fitting	DP-complete (Thm 4.5)	coNP-complete (Thm 4.5)	in PTime (Thm 4.5
Most-Specific	DP-complete (Thm 4.5)	coNP-complete (Thm 4.5)	in PTime (Thm 4.5)
Most-General	HomDual-equivalent (Thm 4.7)	NP-c (Thm 4.5)	in 2ExpTime (Thm 4.5)
Unique	HomDual-equivalent (Thm 4.7)	HomDual-equivalent (Thm 4.7)	in PTime (Thm 4.5)

Table 2: Summary of results for UCQs

	Verification	Existence	Construction and size
Any Fitting	PTime (Thm 5.2)	ExpTime-c [24]	In 2ExpTime (Thm 5.4); 2Exp size lower bound (Thm 5.21)
Most-Specific	ExpTime-c (Thm 5.7, 5.20)	ExpTime-c (Thm 5.7, 5.20)	In 2ExpTime (Thm 5.8)
Weakly Most-General	PTime (Thm 5.11)	ExpTime-c (Thm 5.12, 5.20)	In 2ExpTime (Thm 5.12)
Basis of Most-General	ExpTime-c (Thm 5.16, 5.20)	ExpTime-c (Thm 5.18, 5.20)	In 3ExpTime (2Exp upper bound on size of members) (Thm 5.18)
Unique	ExpTime-c (Thm 5.13, 5.20)	ExpTime-c (Thm 5.13, 5.20)	in 2ExpTime (Thm 5.4)

Table 3: Summary of results for tree CQs

as it corresponds to the concept language of the description logic $\mathcal{ELI}$ that is prominent in knowledge representation, we restrict our attention to relation symbols of arity one and two. The main complexity results and size bounds for CQs, UCQs, and tree CQs are summarized in Tables 1, 2, and 3. Note that, since the classical (non-extremal) fitting problem for CQs is already coNExpTime-complete [10, 46], it is not surprising that many of the problems we consider here turn out to be of similarly high complexity. We will comment on possible strategies for taming the complexity of these problems in Sect. 6.

All proofs are provided in the appendix of the long version, made available at [12].

Related Work. The fitting problem for CQs and UCQs, as well as for bounded-treewidth CQs and UCQs, was studied in [2, 4, 10, 46]. Note that, from a fitting point of view, the *GAV schema mappings* and *LAV schema mappings* studied in [2] correspond in a precise way to UCQs and CQs, respectively, cf. [10]. The fitting problem for tree CQs (equivalently, $\mathcal{ELI}$*-concept expressions*) was studied in [24]. The fitting problem for CQs is also closely related to the ILP consistency problem for Horn clauses, studied in [25], although the latter differs in assuming a bound on the size of clauses.

The notion of a *most-specific fitting query* appears in several places in this literature, largely because of the fact that some of the canonical fitting algorithms naturally produce such fittings. We are not aware of any prior work studying the verification or construction of most-general fitting queries or unique fitting queries, although [11] studies the inverse problem, namely the existence and construction of uniquely characterizing examples for a query, and we build on results from [11].

The problem of deriving queries from data examples has also been studied from the perspective of computational learning theory, cf. the related work sections in [11, 14].

2 PRELIMINARIES

Schema, Instance, CQ, Homomorphism, Core. A *schema* (or relational signature) is a finite set of relation symbols $\mathcal{S} = \{R_1, \ldots, R_n\}$, where each relation symbol R_i has an associated arity $\text{arity}(R_i) \geq 1$. A *fact* over $\mathcal{S}$ is an expression $R(a_1, \ldots, a_n)$, where $a_1, \ldots, a_n$ are *values*, $R \in \mathcal{S}$, and $\text{arity}(R) = n$. An *instance over* $\mathcal{S}$ is a finite set I of facts over $\mathcal{S}$. The *active domain* of I (denoted $\text{adom}(I)$) is the set of all values occurring in facts of I.

Let $k \geq 0$. A k-ary *conjunctive query (CQ)* q over a schema $\mathcal{S}$ is an expression of the form $q(\mathbf{x}) \text{ :- } \alpha_1 \wedge \cdots \wedge \alpha_n$ where $\mathbf{x} = x_1, \ldots, x_k$ is a sequence of variables, and each α_i is an atomic formula using a relation from $\mathcal{S}$. Note that α_i may use variables from $\mathbf{x}$ as well as other variables. The variables in $\mathbf{x}$ are called *answer variables*, and the other variables *existential variables*. Each answer variable is required to occur in at least one conjunct α_i. This requirement is known as the *safety* condition. A CQ of arity 0 is called a *Boolean CQ*.

If q is a k-ary CQ and I is an instance over the same schema as q, we denote by $q(I)$ the set of all k-tuples of values from the active domain of I that satisfy the query q in I. We write $q \subseteq q'$ if q and q' are queries over the same schema, and of the same arity, and $q(I) \subseteq q'(I)$ holds for all instances I. We say that q and q' are *logically equivalent* (denoted $q \equiv q'$) if $q \subseteq q'$ and $q' \subseteq q$ both hold.

Given two instances I, J over the same schema, a *homomorphism* $h : I \to J$ is a map from $\text{adom}(I)$ to $\text{adom}(J)$ that preserves all facts. When such a homomorphism exists, we say that I "homomorphically maps to" J and write $I \to J$. We say that I and J are *homomorphically equivalent* if $I \to J$ and $J \to I$.

It is well known that every instance I has a unique (up to isomorphism) minimal subinstance to which it is homomorphically equivalent, known as the *core* of I. Furthermore, two instances are homomorphically equivalent iff their cores are isomorphic.

Pointed Instance, Canonical Instance, Canonical CQ, UNP. A *pointed instance* for schema $\mathcal{S}$ is a pair $(I, \mathbf{a})$ where I is an instance over $\mathcal{S}$, and $\mathbf{a}$ is a tuple of values. The values in $\mathbf{a}$ are

typically elements of adom(I), but we also admit here values from outside of adom(I) as this allows us to simplify some definitions and proofs. If the tuple $\mathbf{a}$ consists of k values, then we call $(I, \mathbf{a})$ a *k-ary* pointed instance. We refer to $\mathbf{a}$ as the *distinguished elements* of the pointed instance.

The definition of a homomorphism naturally extends to pointed instances. More precisely a homomorphism $h : (I, \mathbf{a}) \to (J, \mathbf{b})$ is a map from adom$(I) \cup \{\mathbf{a}\}$ to adom$(J) \cup \{\mathbf{b}\}$ that maps every fact of I to a fact of J, and that maps every distinguished element a_i to the corresponding distinguished element b_i.

There is a natural correspondence between k-ary CQs over a schema $\mathcal{S}$ and k-ary pointed instances over $\mathcal{S}$. In one direction, the *canonical instance* of a CQ $q(\mathbf{x})$ is the pointed instance $\widehat{q} = (I_q, \mathbf{x})$, where the domain of I_q is the set of variables occurring in q and the facts of I_q are the conjuncts of q. Note that every distinguished element of $\widehat{q}$ does indeed belong to the active domain (i.e. occurs in a fact), due to the safety condition of CQs. Conversely, the *canonical CQ* of a pointed instance $(I, \mathbf{a})$ with $\mathbf{a} = a_1, \ldots, a_k$ is the CQ $q(x_{a_1}, \ldots, x_{a_k})$ that has a variable x_a for every value $a \in$ adom(I), and a conjunct for every fact of A. Here, we assume that all distinguished elements belong to the active domain.

We write $q \to q'$ when there is a homomorphism $h : \widehat{q} \to \widehat{q'}$ and $q \to (I, \mathbf{a})$ when $\widehat{q} \to (I, \mathbf{a})$. By the classic Chandra-Merlin Thm. [18], then, a tuple $\mathbf{a}$ belongs to $q(I)$ if and only if $q \to (I, \mathbf{a})$ holds; and $q \subseteq q'$ holds if and only if $q' \to q$.

A pointed instance $(I, \mathbf{a})$, with $\mathbf{a} = a_1, \ldots, a_k$, has the *Unique Names Property (UNP)* if $a_i \neq a_j$ for all $i \neq j$.

Disjoint Union, Direct Product. Let $(I, \mathbf{a})$ and $(J, \mathbf{a})$ be pointed instances over the same schema $\mathcal{S}$ with the UNP, where both pointed instances have the same tuple of distinguished elements. Furthermore, assume that adom$(I) \cap$ adom$(J) \subseteq \{\mathbf{a}\}$. Then the *disjoint union* $(I, \mathbf{a}) \uplus (J, \mathbf{a})$ is the pointed instance $(I \cup J, \mathbf{a})$, where the facts of $I \cup J$ are the union of the facts of I and J. This construction generalizes to arbitrary pairs of k-ary pointed instances with the UNP, by taking suitable isomorphic copies of the input instances (to ensure that they have the same tuple of distinguished elements, and are disjoint otherwise). This operation also naturally generalizes to finite sets of k-ary pointed instances with the UNP.

The *direct product* of two k-ary pointed instances $(I, \mathbf{a})$ and $(J, \mathbf{b})$, where $\mathbf{a} = \langle a_1, \ldots a_k \rangle$ and $\mathbf{b} = \langle b_1, \ldots, b_k \rangle$ is the k-ary pointed instance $(I \times J, \langle (a_1, b_1), \ldots, (a_k, b_k) \rangle)$, where $I \times J$ consists of all facts $R((c_1, d_1), \ldots, (c_n, d_n))$ such that $R(c_1, \ldots, c_n)$ is a fact of I and $R(d_1, \ldots, d_n)$ is a fact of J. The same operation of direct product can also be applied to CQs: for CQs $q_1(\mathbf{x}), q_2(\mathbf{y})$ (over the same schema and of the same arity), $q_1 \times q_2$ is the canonical CQ of the direct product of pointed instances $(I_{q_1}, \mathbf{x}) \times (I_{q_2}, \mathbf{y})$. However, this yields a well-defined CQ only when the designated elements of $(I_{q_1}, \mathbf{x}) \times (I_{q_2}, \mathbf{y})$ belong to the active domain, which is not necessarily the case. This construction extends naturally to finite sets of pointed instances (where the direct product of an empty set of pointed instances is, by convention, the pointed instance $(I, \langle a, \ldots, a \rangle)$ where I consists of all possible facts over the singleton domain $\{a\}$).

Data Example, Fitting Problem. A k-ary *data example* for schema $\mathcal{S}$ (for $k \geq 0$) is a pointed instance $e = (I, \mathbf{a})$ where $\mathbf{a}$ is a k-tuple of values from adom(I). A data example $(I, \mathbf{a})$ is said to be a *positive example* for a query q (over the same schema and of the same arity) if $\mathbf{a} \in q(I)$, and a *negative example* otherwise. By a *collection of labeled examples* we mean a pair $E = (E^+, E^-)$ of finite sets of data examples. The size of a data example e (as measured by the number of facts) is denoted by $|e|$, and the combined size of a set of data examples E by $||E|| = \Sigma_{e \in E} |e|$.

We say that q *fits* E if each data example in E^+ is a positive example for q and each data example in E^- is a negative example for q. The *fitting problem* (for CQs) is the problem, given as input a collection of labeled examples, to decide if a fitting CQ exists.

A special case is where the input examples involve a single database instance I, and hence can be given jointly as (I, S^+, S^-), where S^+, S^- are sets of tuples. We focus on the general version of the fitting problem here, but note that the aforementioned special case typically carries the same complexity (cf. [10, Thm. 2]).

Frontiers, Dualities, C-Acyclicity, Degree. A *frontier* for a CQ is, intuitively, a finite complete set of minimal weakenings of q. Formally, a finite set of CQs $\{q_1, \ldots, q_n\}$ is a frontier for a CQ q, with respect to a class $\mathcal{C}$ of CQs, if:

(1) for all $i \leq n$, $q_i \to q$ and $q \not\to q_i$, and
(2) for all $q' \in \mathcal{C}$ such that $q' \to q$ and $q \not\to q'$, it holds that $q' \to q_i$ for some $i \leq n$.

If $\mathcal{C}$ is the class of all CQs, we simply call $\{q_1, \ldots, q_n\}$ a frontier for q.

Another related concept is that of *homomorphism dualities*. A pair of finite sets of data examples (F, D) is a homomorphism duality if

$$\{e \mid e \text{ is a data example and } e \to e' \text{ for some } e' \in D\} =$$
$$\{e \mid e \text{ is a data example and } e' \not\to e \text{ for all } e' \in F\}.$$

Homomorphism dualities have been studied extensively in the literature on combinatorics and constraint satisfaction, and elsewhere (e.g., [6, 23, 35]).

Frontiers and homomorphism dualities were studied in [11, 23]. Their existence was characterized in terms of a structural property called *c-acyclicity*: the *incidence graph* of a CQ q is the bipartite multi-graph consisting of the variables and the atoms of q, and such that there is a distinct edge between a variable and an atom for each occurrence of the variable in the atom. A CQ q is *c-acyclic* if every cycle in the incidence graph (including every self-loop and every cycle of length 2 consisting of different edges that connect the same pair of nodes) passes through an answer variable of q.

Theorem 2.1 ([1, 11]). *For all CQs q the following are equivalent:*

(1) q has a frontier,
(2) there exists a homomorphism duality $(\{q\}, D)$,
(3) the core of the canonical instance of q is c-acyclic.

Furthermore, for any fixed $k \geq 0$, a frontier for a k-ary c-acyclic CQs can be computed in polynomial time, and a set D as in (2) can be computed in exponential time.

By the *degree* of a CQ q we mean the maximum degree of variables in the incidence graph of q (i.e., the maximum number of occurrences of a variable in q).

3 THE CASE OF CONJUNCTIVE QUERIES

In this section, we study the fitting problem for CQs. We first review results for the case where the fitting CQ needs not satisfy any further properties. After that, we introduce and study extremal

fitting CQs, including most-general, most-specific, and unique fittings. For these, we first concentrate on characterizations and upper bounds, deferring lower bounds to Sect. 3.5

To simplify presentation, when we speak of a CQ q in the context of a collection of labeled examples E, we mean that q ranges over CQs that have the same schema and arity as the data examples in E.

3.1 Arbitrary Fitting CQs

We first consider the verification problem for fitting CQs: *given a collection of labeled examples E and a CQ q, does q fit E?* This problem naturally falls in the complexity class DP (i.e., it can be expressed as the intersection of a problem in NP and a problem in coNP). Indeed:

Theorem 3.1. *The verification problem for fitting CQs is DP-complete. The lower bound holds for a schema consisting of a single binary relation, a fixed collection of labeled examples, and Boolean CQs.*

The *existence* problem for fitting CQs (*given a collection of labeled examples E, is there a CQ that fits E?*) was studied in [10, 46].

Theorem 3.2 ([10, 46]). *The existence problem for fitting CQs is coNExpTime-complete. The lower bound holds already for Boolean CQs over a fixed schema consisting of a single binary relation.*

Theorem 3.3 ([46]). *If any CQ fits a collection of labeled examples $E = (E^+, E^-)$, then the canonical CQ of the direct product $\Pi_{e \in E^+}(e)$ is well-defined and fits E.*

When we are promised that a fitting CQ exists, then we can construct one in (deterministic) single exponential time. We will see in Sect. 3.5 that this is optimal, as there is a matching lower bound.

3.2 Most-Specific Fitting CQs

There are two natural ways to define *most-specific* fitting CQs:

Definition 3.4.

- A CQ q is a **strongly most-specific fitting** CQ for a collection of labeled examples E if q fits E and for every CQ q' that fits E, we have $q \subseteq q'$.
- A CQ q is a **weakly most-specific fitting** CQ for a collection of labeled examples E if q fits E and for every CQ q' that fits E, $q' \subseteq q$ implies $q \equiv q'$.

It follows from Thm. 3.3 that the above two notions coincide:

Proposition 3.5. *For all CQs q and collections of labeled examples $E = (E^+, E^-)$, the following are equivalent:*

(1) q is strongly most-specific fitting for E,
(2) q is weakly most-specific fitting for E,
(3) q fits E and q is homomorphically equivalent to the canonical CQ of $\Pi_{e \in E^+}(e)$. [1]

In light of Prop. 3.5, we simply speak of *most-specific fitting CQs*, dropping "weak" and "strong".

Example 3.6. Let $\mathcal{S} = \{R, P\}$, where R is a ternary relation and P is a unary relation. Consider the collection of labeled examples $E = (E^+ = \{I_1, I_2\}, E^- = \{I_3\})$, where $I_1 = \{R(a, a, b), P(a)\}$, $I_2 = \{R(c, d, d), P(c)\}$, and $I_3 = \emptyset$. The Boolean CQs $q_1 :\!-\ \exists xyz\, R(x, y, z)$ and $q_2 :\!-\ \exists xyz(R(x, y, z) \wedge P(x))$ both fit E, but q_2 is more specific than q_1. Indeed, q_2 is most-specific fitting for E, as it is homomorphically equivalent to the canonical query of $I_1 \times I_2$.

It follows from Prop. 3.5 and Thm. 3.2 that the existence problem for most-specific fitting CQs coincides with that for arbitrary fitting CQs, and hence, is coNExpTime-complete; and that we can construct in exponential time a CQ q (namely, the canonical CQ of $\Pi_{e \in E^+}(e)$), with the property that, if there is a most-specific fitting CQ, then q is one. For the *verification* problem, finally, Thm. 3.3, with Thm. 3.1, implies:

Theorem 3.7. *The verification problem for most-specific fitting CQs is in NExpTime.*

3.3 Most-General Fitting CQs

For *most-general fitting* CQs, there are again two natural definitions.

Definition 3.8.

- A CQ q is a **strongly most-general fitting** CQ for a collection of labeled examples E if q fits E and for all CQs q' that fit E, we have $q' \subseteq q$.
- A CQ q is a **weakly most-general fitting** CQ for a collection of labeled examples E if q fits E and for every CQ q' that fits E, $q \subseteq q'$ implies $q \equiv q'$

Unlike in the case of most-specific fitting CQs, as we will see, these two notions do *not* coincide. In fact, there is a third:

Definition 3.9. A finite set of CQs $\{q_1, \ldots, q_n\}$ is a **basis of most-general fitting CQs** for E if each q_i fits E and for all CQs q' that fit E, we have $q' \subseteq q_i$ for some $i \leq n$. If, in addition, no strict subset of $\{q_1, \ldots, q_n\}$ is a basis of most-general fitting CQs for E, we say that $\{q_1, \ldots, q_n\}$ is a *minimal* basis.

Each member of a minimal basis is indeed guaranteed to be weakly most-general fitting. The same does not necessarily hold for non-minimal bases. We could have included this as an explicit requirement in the definition, but we decided not to, in order to simplify the statement of the characterizations below.

It is easy to see that minimal bases are unique up to homomorphic equivalence. Also, a strongly most-general fitting CQ is simply a basis of size 1. We will therefore consider the notions of *weakly most-general fitting CQs* and *basis of most-general fitting CQs*, only.

Example 3.10. Let $\mathcal{S} = \{R, P, Q\}$, where R is a binary relation and P, Q are unary relations. The following examples pertain to Boolean CQs. Let K_2 be the 2-element clique, i.e., $K_2 = \{R(a, b), R(b, a)\}$. Furthermore, let I_P, I_Q, and I_{PQ} be the instances consisting of the set of facts $\{P(a)\}$, $\{Q(a)\}$, and $\{P(a), Q(a)\}$, respectively.

(1) The collection of labeled examples $(E^+ = \emptyset, E^- = \{I_{PQ}\})$ has a strongly most-general fitting CQ, namely $q :\!-\ \exists xy(R(x, y))$.
(2) The collection of labeled examples $E = (E^+ = \emptyset, E^- = \{I_P, I_Q\})$ has a basis of most-general fitting CQs of size two, consisting of $q_1 :\!-\ \exists xy(R(x, y))$ and $q_2 :\!-\ \exists xy(P(x) \wedge Q(y))$. In particular, each of these two CQs is weakly most-general fitting for E.
(3) The collection of labeled examples $E = (E^+ = \emptyset, E^- = \{K_2\})$ does not have a weakly most-general fitting CQ. Indeed, a CQ q fits E iff q is not two-colorable, i.e., q, viewed as a graph, contains a cycle of odd length. Take a fitting CQ q and let k be

[1] In particular, in this case, the canonical CQ of $\Pi_{e \in E^+}(e)$ is well-defined.

the size of the smallest cycle in q of odd length. For C_{3k} the (odd) cycle of length $3k$, we have $q \subseteq C_{3k}$ and $C_{3k} \not\subseteq q$, and C_{3k} fits E.

(4) The collection of labeled examples $(E^+ = \emptyset, E^- = \{K_2, I_P, I_Q\})$ has a weakly most-general fitting CQ, namely q :- $\exists xy(P(x) \wedge Q(y))$. By the same reasoning as in (3), there is no basis of most-general fitting CQs.

Weakly most-general fitting CQs. As it turns out, weakly most-general fitting CQs can be characterized in terms of frontiers.

Proposition 3.11. *The following are equivalent for all collections of labeled examples $E = (E^+, E^-)$ and all CQs q:*

(1) q is weakly most-general fitting for E,
(2) q fits E, q has a frontier and every element of the frontier has a homomorphism to an example in E^-,
(3) q fits E and $\{q \times q_e \mid e \in E^- \text{ and } q \times q_e \text{ is a well-defined CQ}\}$ is a frontier for q,

where q_e is the canonical CQ of e.

Using Thm. 2.1, we can now show:

Theorem 3.12. *Fix $k \geq 0$. The verification problem for weakly most-general fitting k-ary CQs is NP-complete. In fact, it remains NP-complete even if the examples are fixed suitably and, in addition, the input query is assumed to fit the examples.*

Theorem 3.13. *The existence problem for weakly most-general fitting CQs is in ExpTime. Moreover, if such a CQ exists, then*

(1) there is one of doubly exponential size and
(2) we can produce one in time $2^{poly(n)} + poly(m)$ where $n = ||E||$ and m is the size of the smallest weakly most-general fitting CQ.

The proof of Thm. 3.13 uses tree automata. More precisely, we show that, given a collection of labeled examples $E = (E^+, E^-)$, (i) if there is a weakly most-general fitting CQ for E, then there is one that is c-acyclic and has a degree at most $||E^-||$; and (ii) we can construct in ExpTime a non-deterministic tree automaton $\mathfrak{A}_E$ that accepts precisely the (suitably encoded) c-acyclic weakly most-general fitting CQs for E of degree at most $||E^-||$.

Bases of most-general fitting CQs. In the same way that the weakly most-general fitting CQs are characterized in terms of frontiers, *bases of most-general fitting CQs* admit a characterization in terms of *homomorphism dualities*. To spell this out, we need a refinement of this concept, *relativized* homomorphism dualities.

Definition 3.14 (Relativized homomorphism dualities). A pair of finite sets of data examples (F, D) forms a homomorphism duality relative to a data example p, if for all data examples e with $e \to p$, the following are equivalent:

(1) e homomorphically maps to a data example in D,
(2) No data example in F homomorphically maps to e.

Proposition 3.15. *For all collections of labeled examples $E = (E^+, E^-)$, the following are equivalent for all CQs $q_1, \ldots, q_n$:*

(1) $\{q_1, \ldots, q_n\}$ is a basis of most-general fitting CQs for E,
(2) each q_i fits E and $(\{e_{q_1}, \ldots, e_{q_n}\}, E^-)$ is a homomorphism duality relative to p,

where $p = \Pi_{e \in E^+}(e)$ and e_{q_i} is the canonical instance of q_i.

While homomorphism dualities have been studied extensively, we are not aware of the relativized variant having been considered.

Theorem 3.16.

(1) The following is NP-complete: given a finite set of data examples D and a data example p, is there a finite set of data examples F such that (F, D) is a homomorphism duality relative to p?
(2) Given a finite set of data examples D and a data example p, if there is a finite set of data examples F such that (F, D) is a homomorphism duality relative to p, then we can compute in 2ExpTime such a set F, where each $e \in F$ is of size $2^{O(||D||^2 \cdot \log ||D|| \cdot |p|)}$.

Thm. 3.16(1) was proved in [35] and [8] for non-relativized dualities and where D consists of a single instance without distinguished elements. Our proof, given in the appendix, extends the one in [8]. As a consequence, we get:

Theorem 3.17. *The existence and verification problems for bases of most-general fitting CQs is in NExpTime.*

Theorem 3.18. *Let $E = (E^+, E^-)$ be a collection of labeled examples, for which a basis of most-general fitting CQs exists. Then we can compute a minimal such basis in 3ExpTime, consisting of CQs of size $2^{poly(||E^-||) \cdot 2^{O(||E^+||)}}$.*

3.4 Unique fitting CQs

By a *unique fitting CQ* for a collection of labeled examples E, we mean a fitting CQ q with the property that *every CQ that fits E is logically equivalent to q.*

Example 3.19. Let $\mathcal{S}$ consist of a single binary relation R, and let I be the instance consisting of the facts $R(a, b)$, $R(b, a)$, and $R(b, b)$. Let $E = (E^+, E^-)$, where $E^+ = \{(I, b)\}$ and $E^- = \{(I, a)\}$.

The query $q(x)$:- $R(x, x)$ is a unique fitting CQ for E. Indeed, q fits E, and it is easy to see that if $q'(x)$ is any CQ that fits E, then q' must contain the conjunct $R(x, x)$ (in order to fit E^-). From this, it is easy to see that q and q' admit homomorphisms to each other.

Proposition 3.20. *For every CQ q and collection of labeled examples $E = (E^+, E^-)$ the following are equivalent:*

(1) q is a unique fitting CQ for E,
(2) q is a most-specific and weakly most-general fitting CQ for E,
(3) q is homomorphically equivalent to $\Pi_{e \in E^+}(q_e)$ and $\{q \times e \mid e \in E^- \text{ and } q \times e \text{ is a well-defined CQ}\}$ is a frontier for q.

Our previous results on most-specific fitting CQs and weakly most-general fitting CQs now immediately imply:[2]

Theorem 3.21. *The verification and existence problems for unique fitting CQs are in NExpTime. When a unique fitting CQ exists, it can be computed in exponential time.*

3.5 Lower bounds

The lower bound proofs below involve reductions from the *Product Homomorphism Problem* (PHP) [10]. The PHP takes as input a set of instances $I_1, \ldots, I_n$ and an instance J, and asks whether the direct product $I_1 \times \cdots \times I_n$ admits a homomorphism to J. This problem is NExpTime-complete [10, 46]. We need a refinement of this:

[2] Indeed, the existence of a unique fitting CQ for $E = (E^+, E^-)$ can be tested simply by checking that $\Pi_{e \in E^+}(e)$ is weakly most-general fitting for E.

Theorem 3.22. *Let $\mathcal{S}$ consist of a single binary relation. There is a fixed k for which the following problem is NExpTime-complete: given k-ary pointed $\mathcal{S}$-instances $(I_1, \boldsymbol{a}_1) \ldots, (I_n, \boldsymbol{a}_n)$ and $(J, \boldsymbol{b})$ with the UNP, where $(J, \boldsymbol{b})$ is c-acyclic, is it the case that $\Pi_i(I_i, \boldsymbol{a}_i) \to (J, \boldsymbol{b})$?*

Using this we obtain the following results:

Theorem 3.23. *The following problems are NExpTime-hard:*

(1) The verification problem for most-specific fitting CQs.
(2) The verification problem for unique fitting CQs.
(3) The existence problem for unique fitting CQs.
(4) The verification problem for bases of most-general fitting CQs.
(5) The existence problem for bases of most-general fitting CQs.

Each problem is NExpTime-hard already for a fixed schema and arity, and, in the case of the verification problems, when restricted to inputs where the input CQ fits the examples, or, in the case of the existence problems, when restricted to inputs where a fitting CQ exists.

For the existence of weakly most-general fitting CQs, we prove an ExpTime lower bound by adapting a reduction from the word problem for certain alternating Turing machines used in [28] to prove hardness of a product simulation problem for transition systems. Unlike the previous reductions, it does not apply to the restricted case where a fitting CQ is promised to exist.

Theorem 3.24. *The existence problem for weakly most-general fitting CQs is ExpTime-hard.*

The same proof also shows that Thm. 3.24 holds for *tree CQs*, which we will introduce and study in Sect. 5.

The following results provide *size* lower bounds:

Theorem 3.25. *Fix a schema consisting of a single binary relation. For $n > 0$, we can construct a collection of Boolean data examples of combined size polynomial in n such that a fitting CQ exists, but not one of size less than 2^n.*

We can prove something stronger (but not for a fixed schema):

Theorem 3.26. *For $n \geq 0$, we can construct a schema with $O(n)$ unary and binary relations and a collection of labeled examples of combined size polynomial in n such that*

(1) There is a unique fitting CQ.
(2) Every fitting CQ contains at least 2^n variables.

Theorem 3.27. *For $n \geq 0$, we can construct a schema with $O(n)$ unary and binary relations and a collection of labeled examples of combined size polynomial in n such that*

(1) There is a basis of most-general fitting CQs.
(2) Every such basis contains at least 2^{2^n} CQs.

4 THE CASE OF UCQS

A k-ary *union of conjunctive queries* (UCQ) over a schema $\mathcal{S}$ is an expression of the form $q_1 \cup \cdots \cup q_n$, where $q_1, \ldots, q_n$ are k-ary CQs over $\mathcal{S}$. Each q_i is called a *disjunct* of q. Logically, q is interpreted as the disjunction of $q_1, \ldots, q_n$. For two UCQs q, q', we say that q *maps homomorphically to* q' (written: $q \to q'$) if, for every disjunct q'_i of q', there is a disjunct q_j of q such that $q_j \to q'_i$. Under this definition, as for CQs we have $q \to q'$ precisely if $q' \subseteq q$. All the notions and problems considered in Sect. 3 now naturally generalize to UCQs.

Example 4.1. Consider a schema consisting of the unary relations P, Q, R, and let $k = 0$. Let E consist of positive examples $\{P(a), Q(a)\}$ and $\{P(a), R(a)\}$, and negative examples $\{P(a)\}$ and $\{Q(a), R(a)\}$. Clearly, the UCQ $\exists x P(x) \wedge Q(x) \cup \exists x P(x) \wedge R(x)$ fits. Indeed, it can be shown that this UCQ is unique fitting for E. However, there is no fitting CQ for E, as the direct product of the positive examples maps to the first negative example.

We next give characterizations for most-specific fitting UCQs, most-general fitting UCQs, and unique fitting UCQs, in the style of the characterization for CQs provided in Sect. 3. For most-general fitting UCQs, the weak and the strong version turn out to coincide, unlike for CQs.

Proposition 4.2. *(Implicit in [2].) For all collections of labeled examples $E = (E^+, E^-)$ and UCQs q, the following are equivalent:*

(1) q is a strongly most-specific fitting UCQ for E,
(2) q is a weakly most-specific fitting UCQ for E,
(3) q fits E and is homomorphically equivalent to $\bigcup_{e \in E^+} q_e$.

Proposition 4.3. *For all collections of labeled examples $E = (E^+, E^-)$ and UCQs $q = q_1 \cup \cdots \cup q_n$, the following are equivalent:*

(1) q is a strongly most-general fitting UCQ for E,
(2) q is a weakly most-general fitting UCQ for E,
(3) q fits E^+ and $(\{e_{q_1}, \ldots, e_{q_n}\}, E^-)$ is a homomorphism duality.

Proposition 4.4. *For all collections of labeled examples $E = (E^+, E^-)$ and UCQs q, the following are equivalent:*

(1) q is a unique fitting UCQ for E,
(2) q fits E and the pair (E^+, E^-) is a homomorphism duality,
(3) q is homomorphically equivalent to $\bigcup_{e \in E^+} q_e$ and (E^+, E^-) is a homomorphism duality.

Based on these characterizations we obtain:

Theorem 4.5.

(1) The existence problem for fitting UCQs (equivalently, for most-specific fitting UCQs) is coNP-complete; if a fitting UCQ exists, a most-specific fitting UCQ can be computed in PTime.
(2) The existence problem for most-general fitting UCQs is NP-complete; if a most-general fitting UCQ exists, one can be computed in 2ExpTime.
(3) The verification problem for fitting UCQs is DP-complete.
(4) The verification problem for most-specific fitting UCQs is DP-complete.

In order to state the remaining complexity results, let HomDual be the problem of testing if a given pair (F, D) is a homomorphism duality. The precise complexity of this problem is not known, but we prove the following.

Proposition 4.6. *HomDual is in ExpTime and NP-hard.*

The upper bound in Prop. 4.6 is based on the observation that, in order for (F, D) to be a homomorphism duality, each $e \in F$ must be c-acyclic. For the lower bound, we use an argument that was also used in [35] to show that FO definability of a CSP is NP-hard: we reduce from 3-SAT.

Theorem 4.7. *The following problems are computationally equivalent (via polynomial conjunctive reductions) to HomDual:*

(1) The existence problem for unique fitting UCQs,
(2) The verification problem for unique fitting UCQs,
(3) The verification problem for most-general fitting UCQs.

As we mentioned, there is a PTime-computable "canonical candidate" fitting UCQ, namely $\bigcup_{e\in E^+}(q_e)$. That is, computing a fitting UCQ under the promise that one exists, is in PTime. While this can be viewed as a positive result, it is somewhat disappointing as the UCQ in question does nothing more than enumerate the positive examples. It does not "compress" or "generalize from" the input examples in a meaningful way. This, it turns out, is unavoidable:

Theorem 4.8 ([13]). *There does* not *exist an "efficient Occam algorithm" for UCQs, i.e., a PTime algorithm taking as input a collection of labeled examples E for which a fitting UCQ is promised to exists and producing a fitting UCQ of size $O(m)^\alpha \cdot poly(n)$ where m is the size of the input, n is the size of the smallest fitting UCQ, and $\alpha < 1$.*

5 THE CASE OF TREE CQS

We now consider *tree CQs*, that is, unary CQs that are acyclic and connected. Moreover, we concentrate on schemas that consist only of unary and binary relations, which we refer to as *binary* schemas. Note that this type of schema is at the core of prominent web data formalisms such as RDF and OWL. Apart from being natural per se, the class of tree CQs holds significance as it correspond to concept expressions in the description logic $\mathcal{ELI}$. Table 3 summarizes our complexity results on tree CQs.

As we shift from unrestricted CQs to tree CQs, simulations take on the role of homomorphisms. In addition, unraveling a CQ or an instance into a (finite or infinite) tree turns out to be a central operation.

Simulation. Given two instances I, J over the same binary schema, a *simulation of I in J* is a relation $S \subseteq \mathrm{adom}(I) \times \mathrm{adom}(J)$ that satisfies the following properties:

(1) if $A(a) \in I$ and $(a, a') \in S$, then $A(a') \in J$;
(2) if $R(a, b) \in I$ and $(a, a') \in S$, then there is an $R(a', b') \in J$ with $(b, b') \in S$.
(3) if $R(a, b) \in I$ and $(b, b') \in S$, then there is an $R(a', b') \in J$ with $(a, a') \in S$.

We write $(I, a) \preceq (J, b)$ if there exists a simulation S of I in J with $(a, b) \in S$. A simulation of a tree CQ $q(x)$ in an instance I is a simulation of I_q in I, and we write $q \preceq (I, a)$ as shorthand for $(I_q, x) \preceq (I, a)$, and likewise for $(I, a) \preceq q$. It is well-known that if I is a tree, then $(I, a) \preceq (J, b)$ iff there is a homomorphism h from I to J with $h(a) = b$. We thus have the following.

Lemma 5.1. *For all instances I, $a \in adom(I)$, and tree CQs $q(x)$, $I \models q(a)$ iff $q \preceq (I, a)$.*

Unraveling. Let I be an instance over a binary schema. A *role* is a binary relation symbol R or its *converse* R^-. For an instance I, we may write $R^-(a, b) \in I$ to mean $R(b, a) \in I$. A *path* in I is a sequence $p = a_1 R_1 \cdots R_{k-1} a_k$, $k \geq 1$, where $a_1, \ldots, a_k \in \mathrm{adom}(I)$ and $R_1, \ldots, R_{k-1}$ are roles such that $R_i(a_i, a_{i+1}) \in I$ for $1 \leq i < k$. We say that p is of *length* k, *starts* at a_1 and *ends* at a_k. The *unraveling of I at $a \in \mathrm{adom}(I)$* is the instance U with active domain $\mathrm{adom}(U)$ that consists of all paths starting at a. It contains the fact

- $A(p)$ for every path $p \in \mathrm{adom}(U)$ that ends with some $b \in \mathrm{adom}(I)$ such that $A(b) \in I$, and
- $R(p, pRb)$ for every path $pRb \in \mathrm{adom}(U)$.

For all $m \geq 1$, the *m-finite unraveling of I at a* is the (finite) restriction of I to all paths of length at most m.

5.1 Arbitrary Fitting Tree CQs

By Lem. 5.1, we can decide verification of arbitrary fitting tree CQs by checking the (non-)existence of simulations to positive and negative examples. Since the existence of simulations can be decided in PTime [30], we obtain the following.

Theorem 5.2. *The verification problem for fitting tree CQs is in PTime.*

The following was proved in [24] in the setting of the description logic $\mathcal{ELI}$, see Thm. 12 of that paper and its proof.

Theorem 5.3 ([24]). *The existence problem for fitting tree CQs is ExpTime-complete. The lower bound already holds for a fixed schema.*

We actually reprove the ExpTime upper bound in Thm. 5.3 using an approach based on (two-way alternating) tree automata. What we gain from this is the following result regarding the size of fitting tree CQs.

Theorem 5.4. *If any tree CQ fits a collection of labeled examples $E = (E^+, E^-)$, then we can produce a DAG representation of a fitting tree CQ with a minimal number of variables in single exponential time and the size of such a tree CQ is at most double exponential.*

A matching lower bound is given in Sect. 5.5.

5.2 Most-Specific Fitting Tree CQs

We may define a strong and a weak version of most-specific fitting tree CQs, in analogy with Def. 3.4. We then observe the following counterpart of Prop. 3.5.

Proposition 5.5. *For all tree CQs q and collections of labeled examples $E = (E^+, E^-)$, the following are equivalent:*

(1) q is a weakly most-specific fitting for E,
(2) q is a strongly most-specific fitting for E,
(3) q fits E and $\Pi_{e\in E^+}(e) \preceq q$.

We thus simply speak of *most-specific* fittings tree CQs. Unlike in the non-tree case, the existence of most-specific fitting tree CQs does not coincide with the existence of arbitrary fitting tree CQs.

Example 5.6. Consider the single positive example $R(a, a)$ and the empty set of negative examples. Then $q(x) \text{ :- } R(x, y)$ is a fitting tree CQ and $p(x) \text{ :- } R(x, x)$ is a most-specific fitting CQ, but there is no most-specific fitting tree CQ. In fact, any m-finite unraveling of $p(x)$ is a fitting tree CQ and there is no (finite) fitting tree CQ that is more specific than all these.

In [33] it is shown that verification and existence of a most-specific fitting tree CQ are in ExpTime and PSpace-hard when there are only positive examples, but no negative examples. We extend the upper bounds to the case with negative examples.

Theorem 5.7. *Verification and existence of most-specific fitting tree CQs is in ExpTime.*

The upper bound for verification follows from Prop. 5.5, and the upper bound for existence follows from Prop. 5.5 and the results in [33]. However, we reprove the latter using tree automata to show the following.

Theorem 5.8. *If a collection of labeled examples $E = (E^+, E^-)$ admits a most-specific tree CQ fitting, then we can construct a DAG representation of such a fitting with a minimal number of variables in single exponential time and the size of such a tree CQ is at most double exponential.*

The proof of Thm. 5.8 comes with a characterization of most-specific fitting tree CQs in terms of certain initial pieces of the unraveling of $\prod_{e \in E^+}(e)$.

5.3 Weakly Most-General and Unique Fitting Tree CQs

We define weakly and strongly most-general tree CQs in the obvious way and likewise for bases of most-general fitting tree CQs and unique fitting tree CQs, see Def. 3.8 and 3.9. The following example illustrates that the existence of weakly most-general tree CQs does not coincide with the existence of weakly most-general CQs.

Example 5.9. Let $(E^+ = \emptyset, E^- = \{\{P(a_0)\}, \{R(a_0, a_0)\}\}$. Then there are no weakly most-general fitting tree CQs. To see this, let $q(x)$ be a tree CQ that fits the examples. Clearly, $q(x)$ must contain both an R-atom and a P atom. Let n be the shortest distance, in the graph of q, from x to some y that satisfies P, and let π be the path from x to y, written as a sequence of roles R and R^-. If π is empty, then the query $R(x, y) \wedge R(y, x) \wedge P(x)$ is homomorphically strictly weaker than q, but still fits. If π is non-empty, then the query $x(\pi; \pi^-; \pi)y \wedge P(y)$ is homomorphically weaker than q, but fits. Thus q is not weakly most-general.

However, weakly most-general fitting CQs exist that are not tree CQs. In fact, we obtain a complete basis of most-general fitting CQs of size 3 by taking the CQs $q(x)$:- $\exists yzu(\alpha(x) \wedge R(y, z) \wedge P(u))$ where $\alpha(x)$ is $P(x)$ or $\exists v R(x, v)$ or $\exists v R(v, x)$, serving purely to make the CQ safe.

As in the case of unrestricted CQs, we may characterize weakly most-general fitting tree CQs using frontiers. The following is an immediate consequence of the definition of frontiers.

Proposition 5.10. *The following are equivalent for all collections of labeled examples $E = (E^+, E^-)$ and tree CQs q:*

(1) q is a weakly most-general fitting for E,
(2) q fits E and every element of the frontier for q w.r.t. tree CQs simulates to an example in E^-.

As every tree CQ is c-acyclic, it has a frontier that can be computed in polynomial time. We have the choice of using the same frontier construction as in the proofs for Sect. 3.3 or one that is tailored towards trees and 'only' yields a frontier w.r.t. tree CQs [11]. Both constructions need only polynomial time and, together with Prop. 5.10, yield a PTime upper bound for the verification problem.

Theorem 5.11. *Verification of weakly most-general fitting tree CQs is in PTime.*

For the existence problem, we choose the frontier construction from [11] and then again use an approach based on tree automata. We also obtain the same results regarding the size and computation of weakly most-general fitting tree CQs as in Sect. 5.1 and 5.2.

Theorem 5.12. *Existence of weakly most-general fitting tree CQs is in ExpTime. Moreover, if a collection of labeled examples $E = (E^+, E^-)$ admits a weakly most-general tree CQ fitting, then we can construct a DAG representation of such a fitting with a minimal number of variables in single exponential time and the size of such a tree CQ is at most double exponential.*

For uniquely fitting tree CQs, we observe that a fitting tree CQ is a unique fitting iff it is both a most-specific and a weakly most-general fitting. This immediately gives an ExpTime upper bound for verification, from the ExpTime upper bounds for verifying most-specific and weakly most-general tree CQs. We obtain an ExpTime upper bound for the existence of uniquely fitting tree CQs by combining the automata constructions for these two cases.

Theorem 5.13. *Verification and existence of unique fitting tree CQs is in ExpTime.*

We remark that Thm. 5.4 clearly also applies to unique fitting tree CQs: if a unique fitting tree CQ exists, then the algorithm from the proof of Thm. 5.4 must compute it.

5.4 Bases of Most General Fitting Tree CQs

In Sect. 3, we have characterized bases of most-general fitting CQs in terms of relativized homomorphism dualities. Here, we do the same for tree CQs, using simulation dualities instead.

Definition 5.14 (Relativized simulation dualities). A pair of finite sets of data examples (F, D) forms a *simulation duality* if, for all data examples e, the following are equivalent:

(1) $e \preceq e'$ for some $e \in D$,
(2) $e' \not\preceq e$ for all $e' \in F$.

We say that (F, D) forms a simulation duality *relative to a data example p* if the above conditions hold for all e with $e \preceq p$.

Proposition 5.15. *For all collections of labeled examples $E = (E^+, E^-)$, the following are equivalent, for $p = \Pi_{e \in E^+}(e)$:*

(1) $\{q_1, \ldots, q_n\}$ is a basis of most-general fitting tree CQs for E,
(2) each q_i fits E and $(\{q_1, \ldots, q_n\}, E^-)$ is a simulation duality relative to p.

We use Prop. 5.15 and the fact any homomorphism duality (F, D) where F consists only of trees is also a simulation duality to show that the verification problem for bases of most-general fitting tree CQs is in ExpTime.

Theorem 5.16. *The verification problem for bases of most-general fitting tree CQs is in ExpTime.*

For the existence problem, we use the following characterization: let D be a finite collection of data examples. A tree CQ q is a *critical tree obstruction* for D if $q \not\preceq e$ for all $e \in D$ and every tree CQ q' that can be obtained from q by removing subtrees satisfies $q' \preceq e$ for some $e \in D$.

Proposition 5.17. *Let D be a finite set of data examples and $\widehat{e}$ a data example. Then the following are equivalent:*

(1) there is a finite set of tree data examples F such that (F, D) is a simulation duality relative to $\widehat{e}$,

(2) there is a finite number of critical tree obstructions q for D that satisfy $q \to \widehat{e}$ (up to isomorphism).

We use Prop. 5.2 to provide a reduction to the infinity problem for tree automata. This also yields bounds for the construction and size of bases of most-general fitting tree CQs

Theorem 5.18. *The existence problem for bases of most-general fitting tree CQs is in ExpTime. Moreover, if a collection of labeled examples E has a basis of most-general fitting tree CQs, then it has such a basis in which every tree CQ has size at most double exponential in $||E||$.*

5.5 Lower Bounds

All the complexity upper bounds stated for tree CQs above are tight. We establish matching ExpTime lower bounds by a polynomial time reduction from the product simulation problem into trees (with one exception).

Product Simulation Problem Into Trees. The *product simulation problem* asks, for finite pointed instances $(I_1, a_1), \ldots, (I_n, a_n)$ and (J, b), whether $\Pi_{1 \leq i \leq n}(I_i, a_i) \preceq (J, b)$. A variant of this problem was shown to be ExpTime-hard in [28] where simulations are replaced with $\downarrow$-simulations, meaning that the third condition of simulations is dropped, and certain transition systems are used in place of instances. This result was adapted to database instances in [24]. Here, we consider instead the *product simulation problem into trees* where the target instance J is required to be a tree (and full simulations are used in place of $\downarrow$-simulations). We prove ExpTime-hardness by a non-trivial reduction from the $\downarrow$-simulation problem studied in [24].

Theorem 5.19. *The product simulation problem into trees is ExpTime-hard, even for a fixed schema.*

This improves a PSpace lower bound from [33] where, however, *all* involved instances were required to be trees. It is easy to prove an ExpTime upper bound by computing the product and then deciding the existence of a simulation in polynomial time [30].

Theorem 5.20. *The verification problem and the existence problem are ExpTime-hard for:*

(1) weakly most-general fitting tree CQs,
(2) most-specific fitting tree CQs,
(3) unique fitting tree CQs, and
(4) complete bases of most-general fitting tree CQs.

Each problem is ExpTime-hard already for a fixed schema and arity, and, in the case of the verification problems, when restricted to inputs where the input CQ fits the examples, or, in the case of the existence problems, when restricted to inputs where a fitting CQ exists.

Points (2) to (4) of Thm. 5.20 are proved by reductions from the product simulation problem into trees. Point (1) is proved simultaneously with Thm. 3.24 by adapting a reduction from the word problem for alternating Turing machines used in [28].

We also establish a double exponential lower bound on the size of (arbitrary) fitting tree CQs.

Theorem 5.21. *For all $n \geq 0$, there is a collection of labeled examples of combined size polynomial in n such that a fitting tree CQ exists and the size of every fitting tree CQ is at least 2^{2^n}. This even holds for a fixed schema.*

We do not currently have a similar lower bound for any of the other types of fitting tree CQs listed in Table 3.

6 CONCLUSION

The characterizations and complexity results we presented, we believe, give a fairly complete picture of extremal fitting problems for CQs, UCQs, and tree CQs. Similar studies could be performed, of course, for other query and specification languages (e.g., graph database queries, schema mappings). In particular, the problem of computing fitting queries has received considerable interest in knowledge representation, where, additionally, background knowledge in the form of an ontology is considered. The existence of a fitting $\mathcal{ELI}$ concept (corresponding to a tree CQ) is undecidable in the presence of an $\mathcal{ELI}$ ontology [24], but there are more restricted settings, involving e.g. $\mathcal{EL}$ concept queries, that are decidable and have received considerable interest [9, 24, 36].

Since the non-extremal fitting problem for CQs is already coNExpTime-complete [10, 46], it is not surprising that many of our complexity bounds are similarly high. In [4], it was shown that the (non-extremal) fitting problem for CQs can be made tractable by a combination of two modifications to the problem: (i) "desynchronization", which effectively means to consider UCQs instead of CQs, and (ii) replacing homomorphism tests by k-consistency tests, which effectively means to restrict attention to queries of bounded treewidth. Similarly, in our results we also see improved complexity bounds when considering UCQs and tree CQs. While we have not studied unions of tree CQs in this paper, based on results in [4] one may expect that they will exhibit a further reduction in the complexity of fitting. We leave this as future work. Another way to reduce the complexity is to consider size-bounded versions of the fitting problem, an approach that also has learning-related benefits [15].

A question that we have not addressed so far is what to do if an extremal fitting query of interest does not exist. For practical purposes, in such cases (and possibly in general) it may be natural to consider relaxations where the fitting query is required to be, for instance, most-general, only as compared to other queries *on some given (unlabeled) dataset*. It is easy to see that, under this relaxation, a basis of most-general fitting queries always exists.

It would also be interesting to extend our extremal fitting analysis to allow for approximate fitting, for instance using a threshold based approach as in [5] or an optimization-based approach as in [16, 26].

ACKNOWLEDGMENTS

Balder ten Cate is supported by the European Union's Horizon 2020 research and innovation programme (MSCA-101031081), Victor Dalmau is supported by the MiCin (PID2019-109137GB-C/AEI/10.13039/501100011033), and Carsten Lutz by the DFG Collaborative Research Center 1320 EASE.

REFERENCES

[1] Bogdan Alexe, Balder ten Cate, Phokion G. Kolaitis, and Wang-Chiew Tan. 2011. Characterizing Schema Mappings via Data Examples. *ACM Trans. Database Syst.* 36, 4 (2011), 23:1–23:48. https://doi.org/10.1145/2043652.2043656

[2] Bogdan Alexe, Balder ten Cate, Phokion G. Kolaitis, and Wang-Chiew Tan. 2011. Designing and Refining Schema Mappings via Data Examples. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data* (Athens, Greece) *(SIGMOD '11)*. Association for Computing Machinery, New York, NY, USA, 133–144. https://doi.org/10.1145/1989323.1989338

[3] Marcelo Arenas, Gonzalo I. Diaz, and Egor V. Kostylev. 2016. Reverse Engineering SPARQL Queries. In *Proceedings of the 25th International Conference on World Wide Web* (Montréal, Québec, Canada) *(WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 239–249. https://doi.org/10.1145/2872427.2882989

[4] Pablo Barceló and Miguel Romero. 2017. The Complexity of Reverse Engineering Problems for Conjunctive Queries. In *20th International Conference on Database Theory, ICDT 2017, March 21-24, 2017, Venice, Italy (LIPIcs, Vol. 68)*, Michael Benedikt and Giorgio Orsi (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 7:1–7:17. https://doi.org/10.4230/LIPIcs.ICDT.2017.7

[5] Pablo Barceló, Alexander Baumgartner, Victor Dalmau, and Benny Kimelfeld. 2021. Regularizing conjunctive features for classification. *J. Comput. System Sci.* 119 (2021), 97–124. https://doi.org/10.1016/j.jcss.2021.01.003

[6] Meghyn Bienvenu, Balder ten Cate, Carsten Lutz, and Frank Wolter. 2014. Ontology-Based Data Access: A Study through Disjunctive Datalog, CSP, and MMSNP. *ACM Trans. Database Syst.* 39, 4 (2014), 33:1–33:44. https://doi.org/10.1145/2661643

[7] Angela Bonifati, Radu Ciucanu, and Aurélien Lemay. 2015. Learning Path Queries on Graph Databases. In *EDBT*. 109–120. https://doi.org/10.5441/002/edbt.2015.11

[8] Raimundo Briceño, Andrei Bulatov, Víctor Dalmau, and Benoît Larose. 2021. Dismantlability, connectedness, and mixing in relational structures. *J. of Comb. Theory, Ser. B* 147 (2021), 37–70. https://doi.org/10.1016/j.jctb.2020.10.001

[9] Lorenz Bühmann, Jens Lehmann, and Patrick Westphal. 2016. DL-Learner - A framework for inductive learning on the Semantic Web. *J. Web Semant.* 39 (2016), 15–24. https://doi.org/10.1016/j.websem.2016.06.001

[10] Balder ten Cate and Víctor Dalmau. 2015. The Product Homomorphism Problem and Applications. In *18th International Conference on Database Theory, ICDT 2015, March 23-27, 2015, Brussels, Belgium (LIPIcs, Vol. 31)*, Marcelo Arenas and Martín Ugarte (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 161–176. https://doi.org/10.4230/LIPIcs.ICDT.2015.161

[11] Balder ten Cate and Victor Dalmau. 2022. Conjunctive Queries: Unique Characterizations and Exact Learnability. *ACM Trans. Database Syst.* 47, 4 (2022), 14:1–14:41. https://doi.org/10.1145/3559756

[12] Balder ten Cate, Victor Dalmau, Maurice Funk, and Carsten Lutz. 2022. Extremal Fitting Problems for Conjunctive Queries. https://doi.org/10.48550/arXiv.2206.05080 arXiv:2206.05080 [cs.DB]

[13] Balder ten Cate, Víctor Dalmau, and Phokion G. Kolaitis. 2013. Learning schema mappings. *ACM Trans. Database Syst.* 38, 4 (2013), 28:1–28:31. https://doi.org/10.1145/2539032.2539035

[14] Balder ten Cate, Maurice Funk, Jean Christoph Jung, and Carsten Lutz. 2022. On the non-efficient PAC learnability of acyclic conjunctive queries. https://doi.org/10.48550/arXiv.2208.10255 arXiv:2208.10255 [cs.DB]

[15] Balder ten Cate, Maurice Funk, Jean Christoph Jung, and Carsten Lutz. 2023. SAT-Based PAC Learning of Description Logic Concepts. Forthcoming.

[16] Balder ten Cate, Phokion G. Kolaitis, Kun Qian, and Wang-Chiew Tan. 2017. Approximation Algorithms for Schema-Mapping Discovery from Data Examples. *ACM Trans. Database Syst.* 42, 2 (2017), 12:1–12:41. https://doi.org/10.1145/3044712

[17] Ashok K. Chandra, Dexter Kozen, and Larry J. Stockmeyer. 1981. Alternation. *J. ACM* 28, 1 (1981), 114–133. https://doi.org/10.1145/322234.322243

[18] Ashok K. Chandra and Philip M. Merlin. 1977. Optimal Implementation of Conjunctive Queries in Relational Data Bases. In *ACM Symposium on Theory of Computing (STOC)*. 77–90.

[19] Hubie Chen, Victor Dalmau, and Berit Grußien. 2011. Arc Consistency and Friends. https://doi.org/10.48550/arXiv.1104.4993 arXiv:1104.4993 [cs.AI]

[20] Sara Cohen and Yaacov Y. Weiss. 2016. The Complexity of Learning Tree Patterns from Example Graphs. *ACM Trans. Database Syst.* 41, 2 (2016), 14:1–14:44. https://doi.org/10.1145/2890492

[21] Andrew Cropper, Sebastijan Dumančič, Richard Evans, and Stephen H. Muggleton. 2022. Inductive logic programming at 30. *Mach. Learn.* 111, 1 (2022), 147–172. https://doi.org/10.1007/s10994-021-06089-1

[22] Tomás Feder and Moshe Y. Vardi. 1998. The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory. *SIAM J. on Computing* 28, 1 (1998), 57–104.

[23] Jan Foniok, Jaroslav Nesetril, and Claude Tardif. 2008. Generalised dualities and maximal finite antichains in the homomorphism order of relational structures. *Eur. J. Comb.* 29, 4 (2008), 881–899. https://doi.org/10.1016/j.ejc.2007.11.017

[24] Maurice Funk, Jean Jung, Carsten Lutz, Hadrien Pulcini, and Frank Wolter. 2019. Learning Description Logic Concepts: When can Positive and Negative Examples be Separated?. In *Proceedings of IJCAI 2019*. 1682–1688. https://doi.org/10.24963/ijcai.2019/233

[25] Georg Gottlob, Nicola Leone, and Francesco Scarcello. 1999. On the Complexity of Some Inductive Logic Programming Problems. *New Generation Comput.* 17, 1 (1999), 53–75. https://doi.org/10.1007/BF03037582

[26] Georg Gottlob and Pierre Senellart. 2010. Schema mapping discovery from data instances. *J. ACM* 57, 2 (2010), 6:1–6:37. https://doi.org/10.1145/1667053.1667055

[27] Wolfgang Gutjahr, Emo Welzl, and Gerhard J. Woeginger. 1992. Polynomial graph-colorings. *Discret. Appl. Math.* 35, 1 (1992), 29–45. https://doi.org/10.1016/0166-218X(92)90294-K

[28] David Harel, Orna Kupferman, and Moshe Y. Vardi. 2002. On the Complexity of Verifying Concurrent Transition Systems. *Inf. Comput.* 173, 2 (2002), 143–161. https://doi.org/10.1006/inco.2001.2920

[29] Pavol Hell and Jaroslav Nešetřil. 2004. *Graphs and homomorphisms*. Oxford lecture series in mathematics and its applications, Vol. 28. Oxford University Press.

[30] Monika Rauch Henzinger, Thomas A. Henzinger, and Peter W. Kopke. 1995. Computing Simulations on Finite and Infinite Graphs. In *36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, USA, 23-25 October 1995*. IEEE Computer Society, 453–462. https://doi.org/10.1109/SFCS.1995.492576

[31] Jean Christoph Jung, Carsten Lutz, Hadrien Pulcini, and Frank Wolter. 2020. Logical Separability of Incomplete Data under Ontologies. In *Proceedings of KR 2020*, D. Calvanese, E. Erdem, and M. Thielscher (Eds.). 517–528. https://doi.org/10.24963/kr.2020/52

[32] Jean Christoph Jung, Carsten Lutz, Hadrien Pulcini, and Frank Wolter. 2021. Separating Data Examples by Description Logic Concepts with Restricted Signatures. In *Proceedings of KR 2021*, M. Bienvenu, G. Lakemeyer, and E. Erdem (Eds.). 390–399. https://doi.org/10.24963/kr.2021/37

[33] Jean Christoph Jung, Carsten Lutz, and Frank Wolter. 2020. Least General Generalizations in Description Logic: Verification and Existence. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*. 2854–2861. https://doi.org/10.1609/aaai.v34i03.5675

[34] Benny Kimelfeld and Christopher Ré. 2018. A Relational Framework for Classifier Engineering. *ACM SIGMOD Record* 47 (2018), 6–13. https://doi.org/10.1145/3277006.3277009

[35] Benoit Larose, Cynthia Loten, and Claude Tardif. 2007. A Characterisation of First-Order Constraint Satisfaction Problems. *Log. Methods Comput. Sci.* 3, 4 (2007). https://doi.org/10.2168/LMCS-3(4:6)2007

[36] Jens Lehmann and Christoph Haase. 2009. Ideal Downward Refinement in the $\mathcal{EL}$ Description Logic. In *Inductive Logic Programming, 19th International Conference, ILP 2009, Leuven, Belgium, July 02-04, 2009. Revised Papers (Lecture Notes in Computer Science, Vol. 5989)*, Luc De Raedt (Ed.). Springer, 73–87. https://doi.org/10.1007/978-3-642-13840-9_8

[37] Jens Lehmann and Pascal Hitzler. 2010. Concept learning in description logics using refinement operators. *Mach. Learn.* 78, 1-2 (2010), 203–250. https://doi.org/10.1007/s10994-009-5146-2

[38] Hao Li, Chee-Yong Chan, and David Maier. 2015. Query from Examples: An Iterative, Data-Driven Approach to Query Construction. *Proc. VLDB Endow.* 8, 13 (2015), 2158–2169. https://doi.org/10.14778/2831360.2831369

[39] Tom M. Mitchell. 1997. *Machine Learning*. McGraw-Hill, New York.

[40] David E. Muller and Paul E. Schupp. 1987. Alternating Automata on Infinite Trees. *Theor. Comput. Sci.* 54 (1987), 267–276. https://doi.org/10.1016/0304-3975(87)90133-2

[41] Jaroslav Nesetril and Vojtech Rödl. 1989. Chromatically optimal rigid graphs. *J. Comb. Theory, Ser. B* 46, 2 (1989), 133–141. https://doi.org/10.1016/0095-8956(89)90039-7

[42] Giuseppe Rizzo, Nicola Fanizzi, and Claudia d'Amato. 2020. Class expression induction as concept space exploration: From DL-FOIL to DL-FOCL. *Future Gener. Comput. Syst.* 108 (2020), 256–272. https://doi.org/10.1016/j.future.2020.02.071

[43] Jörg Rothe. 2003. Exact complexity of Exact-Four-Colorability. *Inform. Process. Lett.* 87, 1 (2003), 7–12. https://doi.org/10.1016/S0020-0190(03)00229-1

[44] Quoc Trung Tran, Chee-Yong Chan, and Srinivasan Parthasarathy. 2014. Query reverse engineering. *The VLDB Journal* 23, 5 (2014), 721–746. https://doi.org/10.1007/s00778-013-0349-3

[45] Moshe Y. Vardi. 1998. Reasoning about The Past with Two-Way Automata. In *Automata, Languages and Programming, 25th International Colloquium, ICALP'98, Aalborg, Denmark, July 13-17, 1998, Proceedings (Lecture Notes in Computer Science, Vol. 1443)*, Kim Guldstrand Larsen, Sven Skyum, and Glynn Winskel (Eds.). Springer, 628–641. https://doi.org/10.1007/BFb0055090

[46] Ross Willard. 2010. Testing Expressibility Is Hard. In *Principles and Practice of Constraint Programming - CP 2010 - 16th International Conference, CP 2010, St. Andrews, Scotland, UK, September 6-10, 2010. Proceedings (Lecture Notes in Computer Science, Vol. 6308)*, David Cohen (Ed.). Springer, 9–23. https://doi.org/10.1007/978-3-642-15396-9_4

A ADDITIONAL PRELIMINARIES

The following fundamental fact about direct products (cf. [29]) will be used in many proofs.

Proposition A.1. *For all pointed instances $(I, \boldsymbol{a})$, $(J_1, \boldsymbol{b}_1)$, $(J_2, \boldsymbol{b}_2)$, the following are equivalent:*

(1) $(I, \boldsymbol{a}) \to (J_1, \boldsymbol{b}_1)$ and $(I, \boldsymbol{a}) \to (J_2, \boldsymbol{b}_2)$
(2) $(I, \boldsymbol{a}) \to (J_1, \boldsymbol{b}_1) \times (J_2, \boldsymbol{b}_2)$

Recall that, for CQs q, q', we denote by $q \times q'$ the canonical CQ of the direct product of the canonical instances of q, q', *if well-defined*. It follows, by the Chandra-Merlin theorem, that the following analogue of Prop. A.1 holds for CQs:

Proposition A.2. *For all CQs q, q_1, q_2, the following are equivalent:*

(1) $q \to q_1$ and $q \to q_2$,
(2) $q_1 \times q_2$ is a well-defined CQ and $q \to q_1 \times q_2$.

Proof. In light of the previous proposition and the definition of direct products for CQs, we only need to show that, if $q \to q_1$ and $q \to q_2$, then $q_1 \times q_2$ is well-defined. Indeed, let $(I_1, \langle a_1, \ldots, a_k \rangle)$ and $(I_2, \langle b_1, \ldots, b_k \rangle)$ be the canonical instances of q_1 and q_2, and consider their direct product $(I_1 \times I_2, \langle (a_1, b_1), \ldots, (a_n, b_n) \rangle)$. We must show that every distinguished element (pair) (a_i, b_i) belongs to the active domain, that is, occurs in some fact of $I_1 \times I_2$. Let $q = q(x_1, \ldots, x_n)$ and consider any fact f of q containing x_i. Such a fact must exist, by the safety condition of CQs. Let $h_1 : q \to (I_1, \mathbf{a})$ and $h_2 : q \to (I_2, \mathbf{b})$, and let h be the map given by $h(x) = (h_1(x), h_2(x))$. Then then h-image of f must belong to $I_1 \times I_2$ and must contain (a_i, b_i). □

B DETAILED PROOFS FOR SECT. 3

Theorem 3.1. *The verification problem for fitting CQs is DP-complete. The lower bound holds for a schema consisting of a single binary relation, a fixed collection of labeled examples, and Boolean CQs.*

Proof. Clearly it is equivalent to a conjunction of problems that are in NP or coNP. For the lower bound we can reduce from exact-4-colorability, i.e., testing that a graph is 4-colorable and not 3-colorable [43]. Fix a schema consisting of a single binary relation R. Let K_3 be the 3-clique (viewed as an instance with a symmetric, irreflexive relation), and let K_4 be the 4-clique. Let $E^- = \{K_3\}$ and $E^+ = \{K_4\}$. Then a graph G is exact-4-colorable if and only if the canonical CQ of G fits (E^+, E^-). □

Remark B.1. A special case of the existence problem for arbitrary fitting CQs, is *CQ definability*, where the input is a input is a pair (I, S) with I an instance and $S \subseteq adom(I)^k$ a k-ary relation, and the task is to decide whether there exists a CQ q such that $q(I) = S$. Note that the CQ definability problem is meaningful only for $k \geq 1$. For fixed $k \geq 1$, this polynomially reduces to a fitting problem, namely for $E = (E^+, E^-)$ with $E^+ = \{(I, \mathbf{a}) \mid \mathbf{a} \in S\}$, and $E^- = \{(I, \mathbf{a}) \mid \mathbf{a} \in adom(I)^k \setminus S\}$. In other words, CQ definability can be viewed as a special case of the CQ fitting, where all input examples share the same instance I and where the k-tuples appearing in the positive and negative examples cover the complete set of all k-tuples over $adom(I)$.

The lower bound of Thm. 3.1, in fact, already holds for this more restricted CQ definability problem. For the second variant of the lower bound, we let I be the disjoint union $K_3 \uplus K_4$, let E^+ be the set of all triples (I, a) where a lies on the 4-clique, and let E^- be the set of all data examples (I, a) where a lies on the 3-clique. Then, for any connected graph G, if a is an arbitrarily chosen vertex of G, then we have that G is exact-4-colorable if and only if the canonical unary CQ of (G, a) fits E. Disconnected graphs G can be handled similarly: in this case, we can linearly order its connected components and add a connecting edge from each component to the next one, without affecting the 3-colorability or 4-colorability of the graph.

Proposition 3.5. *For all CQs q and collections of labeled examples $E = (E^+, E^-)$, the following are equivalent:*

(1) q is strongly most-specific fitting for E,
(2) q is weakly most-specific fitting for E,
(3) q fits E and q is homomorphically equivalent to the canonical CQ of $\Pi_{e \in E^+}(e)$. [3]

Proof. (1 ⇒ 2) is trivial. (2 ⇒ 3): If q fits E, then, by Thm. 3.3, the canonical CQ q^* of $\Pi_{(I,\mathbf{a}) \in E^+}(I, \mathbf{a})$ is well defined and fits. It then follows from the basic properties of direct products that q^* is a most-specific fitting CQ for E: if q' is any fitting CQ, then q' has a homomorphism to each positive example, and hence, q' has a homomorphism to their product. It follows that $q \to q^*$. Since q is weakly most-specific, $q \subseteq q^*$, which means that $q^* \to q$ and hence q and q^* are homomorphically equivalent. (3 ⇒ 1): Let q^* be the canonical CQ of $\Pi_{e \in E^+}(e)$. As we already pointed out, if q^* fits then it is a strongly most-general fitting CQ for E. □

Theorem 3.7. *The verification problem for most-specific fitting CQs is in NExpTime.*

Proof. we first verify that q fits E (in DP by Thm. 3.1). If this test succeeds, we apply by Thm. 3.3 and Prop. 3.5, we test that q is homomorphically equivalent to the canonical CQ of $\Pi_{e \in E^+}(e)$. This puts us in NExpTime because $\Pi_{e \in E^+}(e)$ can be computed in exponential time. □

Proposition 3.11. *The following are equivalent for all collections of labeled examples $E = (E^+, E^-)$ and all CQs q:*

(1) q is weakly most-general fitting for E,
(2) q fits E, q has a frontier and every element of the frontier has a homomorphism to an example in E^-,
(3) q fits E and $\{q \times q_e \mid e \in E^-$ and $q \times q_e$ is a well-defined CQ$\}$ is a frontier for q,

where q_e is the canonical CQ of e.

Proof.

(1 ⇒ 3): If $\{q \times q_e \mid e \in E^-$ and $q \times q_e$ is a well-defined CQ$\}$ is not a frontier for q, then there exists an query q' that is homomorphically strictly weaker than $\widehat{q}$ but that does not map to $\{q \times q_e \mid e \in E^-$ and $q \times q_e$ is a well-defined CQ$\}$. It follows that q' has no homomorphism to any example in $e \in E^-$ (for, if it did, then, by Prop. A.2, we would have that $q' \to q \times q_e$ and $q \times q_e$ would be well-defined). Hence, q' is a fitting CQ that is strictly weaker than q, showing that q is not a weakly most-general fitting CQ. (3

[3] In particular, in this case, the canonical CQ of $\Pi_{e \in E^+}(e)$ is well-defined.

$\Rightarrow 2$) is trivial. ($2 \Rightarrow 1$): let q' be homomorphically strictly weaker than q. Then $\widehat{q'}$ maps to the frontier of q and hence to a negative example. Therefore q' does not fit E. □

Theorem 3.12. *Fix $k \geq 0$. The verification problem for weakly most-general fitting k-ary CQs is NP-complete. In fact, it remains NP-complete even if the examples are fixed suitably and, in addition, the input query is assumed to fit the examples.*

Proof. The algorithm showing NP-membership is as follows. We check by means of a non-deterministic guess that q is homomorphically equivalent to a c-acyclic CQ q' (which can be assumed to be of polynomial size by Thm. 2.1(3)). Then we check whether q' fits. This can be done in polynomial time since q' is c-acyclic by means of an easy dynamic programming argument. Finally, we compute the frontier of q' (which we can do in polynomial time by Thm. 2.1 since k is fixed), and we check each member of the frontier has a homomorphism to a negative example.

Let us now turn our attention to the NP-hardness. For every graph T, let $CSP(T)$ be the problem consisting to determine whether an input graph G is homomorphic to T. It is known ([27]) that there exists some directed trees T such that $CSP(T)$ is NP-complete. Fix any such directed tree T, and let us turn it into a pointed instance $(T, \mathbf{a})$ by selecting $\mathbf{a}$ to be any tuple of k (non necessarily different) values from T.

Since $(T, \mathbf{a})$ is c-acyclic, it has a frontier F. Now, given any graph G, we have that G is homomorphic to T if and only if F is a frontier for $(T, \mathbf{a}) \uplus G$. This, in turn, holds if and only if the canonical CQ of $(T, \mathbf{a}) \uplus G$ is a weakly most-general fitting CQ for $(E^+ = \emptyset, E^- = F)$. To see that this is the case, note that if G is homomorphic to T, then $(T, \mathbf{a}) \uplus G$ is homomorphically equivalent to $(T, \mathbf{a})$ itself, whereas if G is not homomorphic to T, then $(T, \mathbf{a}) \uplus G$ is strictly greater than $(T, \mathbf{a})$ in the homomorphism order.

□

Theorem 3.13. *The existence problem for weakly most-general fitting CQs is in ExpTime. Moreover, if such a CQ exists, then*

1. *there is one of doubly exponential size and*
2. *we can produce one in time $2^{2^{poly(n)}} + poly(m)$ where $n = ||E||$ and m is the size of the smallest weakly most-general fitting CQ.*

The proof is lengthy and is given in Appendix B.1.

Proposition 3.15. *For all collections of labeled examples $E = (E^+, E^-)$, the following are equivalent for all CQs $q_1, \ldots, q_n$:*

1. *$\{q_1, \ldots, q_n\}$ is a basis of most-general fitting CQs for E,*
2. *each q_i fits E and $(\{e_{q_1}, \ldots, e_{q_n}\}, E^-)$ is a homomorphism duality relative to p,*

where $p = \Pi_{e \in E^+}(e)$ and e_{q_i} is the canonical instance of q_i.

Proof. ($1 \Rightarrow 2$): By assumption, each q_i fits. Let e be any data example such that $e \to p$. We need to show that $e_{q_i} \to e$ for some $i \leq n$ iff e does not map to any data example in E^-. First, assume $e_{q_i} \to e$, and assume for the sake of a contradiction that e has a homomorphism to a data example in E^-. Then, by transitivity, e_{q_i} also has a homomorphism to the same negative example, contradicting the fact that q_i fits E. For the converse direction, assume e does not have a homomorphism to a data example in E^-. Since $e \to p$, by Prop. A.1, e has a homomorphism to every data example in E^+. Therefore, the canonical CQ q_e of e fits E. Hence, we have $q_e \subseteq q_i$ for some q_i, and therefore, $e_{q_i} \to e$. ($2 \Rightarrow 1$): let q' be any CQ that fits (E^+, E^-). Then, by Prop. A.1, $e_{q'} \to p$, and $e_{q'}$ does not map to any negative example in E^-. It follows that some e_{q_i} maps to $e_{q'}$, and hence, $q_i \to q'$, which means that $q' \subseteq q_i$. □

Theorem 3.16.

1. *The following is NP-complete: given a finite set of data examples D and a data example p, is there a finite set of data examples F such that (F, D) is a homomorphism duality relative to p?*
2. *Given a finite set of data examples D and a data example p, if there is a finite set of data examples F such that (F, D) is a homomorphism duality relative to p, then we can compute in 2ExpTime such a set F, where each $e \in F$ is of size $2^{O(||D||^2 \cdot \log ||D|| \cdot |p|)}$.*

The proof is lengthy and is given in Appendix B.2.

Theorem 3.17. *The existence and verification problems for bases of most-general fitting CQs is in NExpTime.*

Proof. For the existence problem, let a collection of labeled examples $E = (E^+, E^-)$ be given. Let $p = \Pi_{e \in E^+}(e)$. We claim that the following are equivalent:

(1) a basis of most-general fitting CQs exists for E,
(2) there exists a finite set of data examples F such that (F, E^-) is a homomorphism duality relative to p.

The direction from (i) to (ii) is immediate from Prop. 3.15. For the converse direction, let Q be the set of all canonical CQs of data examples in F that fit E. Then Q is a basis of most-general fitting CQs: let q' be any query that fits E. Then, by Prop. A.1, $e_{q'} \to p$. Furthermore $e_{q'}$ does not have a homomorphism to any data example in E^-. Therefore, we have $e \to e_{q'}$ for some $e \in F$. Since $e \to e_{q'}$ and $e_{q'} \to p$, e has a homomorphism to every data example in E^+. Therefore, q_e fits E, and hence, belongs to Q.

This puts the problem in NExpTime, since p can be computed in exponential time, and, by Thm. 3.16, (ii) can be tested in NP given p.

For the verification problem, let $E = (E^+, E^-)$ and $\{q_1, \ldots, q_n\}$ be given. We may assume that each q_i fits E (as this can be checked in DP by Thm. 3.1). We may also assume that $q_1, \ldots, q_n$ are pairwise homomorphically incomparable (if not, we can select a minimal subset, with the property that the queries in this subset homomorphically map into all others), and core. It is then straightforward to see that, in order for $\{q_1, \ldots, q_n\}$ to be a basis of most-general fitting CQs, each q_i must be a weakly most-general fitting CQ. By 3.11 and Thm. 2.1, for this to be the case, each q_i must be c-acyclic.

Since, at this point, we have that $q_1, \ldots, q_n$ are c-acyclic, by Thm. 2.1, we can compute, in single exponential time, for each q_i, a set of data examples D_{q_i}, such that $(\{e_{q_i}\}, D_{q_i})$ is a homomorphism duality. Let $D = \{e_1 \times \cdots \times e_n \mid e_i \in D_{q_i}\}$. It is easy to see (using Prop. A.1) that $(\{e_{q_1}, \ldots, e_{q_n}\}, D)$ is a homomorphism duality.

Finally, we claim that the following are equivalent:

(1) $\{q_1, \ldots, q_n\}$ is a basis of most-general fitting CQs for E,
(2) $(\{e_{q_1}, \ldots, e_{q_n}\}, E^-)$ is a homomorphism duality relative to p, where $p = \Pi_{e \in E^+}(e)$,
(3) For each $e \in D$, there is $e' \in E^-$ such that $e \times p \to e'$.

The equivalence of 1 and 2 is given by Prop. 3.15. ($2 \Rightarrow 3$): let $e \in D$. Since $(\{e_{q_1}, \ldots, e_{q_n}\}, D)$ is a homomorphism duality and $e \in D$, we have $e_{q_i} \not\to e$ for all $i \leq n$. Hence, by Prop. A.1, also $e_{q_i} \not\to e \times p$.

Therefore, since $e \times p \to p$, we have that $e \times p \to e'$ for some $e' \in E^-$. ($3 \Rightarrow 2$): let e be any data example such that $e \to p$. If some $e_{q_i} \to e$, then, since q_i fits E, we know that $e \not\to e'$ for all $e' \in E^-$. If, on the other hand, no e_{q_i} has a homomorphism to e, then $e \to e'$ for some $e' \in D$. Hence, since $e \to p$, by Prop. A.1, we have that $e \to e' \times p$, and therefore $e \to e''$ for some $e'' \in E^-$.

This concludes the proof since (3) can be tested in NExpTime. □

Theorem 3.18. *Let $E = (E^+, E^-)$ be a collection of labeled examples, for which a basis of most-general fitting CQs exists. Then we can compute a minimal such basis in 3ExpTime, consisting of CQs of size $2^{poly(||E^-||)\cdot 2^{O(||E^+||)}}$.*

Proof. It follows from Prop. 3.15 together with Thm. 3.16(2) that there exists a basis consisting of CQs of size $2^{poly(||E^-||)\cdot 2^{O(||E^+||)}}$. Trivially, this means that the set of all fitting such CQs is a basis. Since the fitting problem is in DP by Thm. 3.1, this basis can be enumerated, and, subsequently, minimized, in 3ExpTime. □

Theorem 3.22. *Let $\mathcal{S}$ consist of a single binary relation. There is a fixed k for which the following problem is NExpTime-complete: given k-ary pointed $\mathcal{S}$-instances $(I_1, \boldsymbol{a}_1) \ldots, (I_n, \boldsymbol{a}_n)$ and $(J, \boldsymbol{b})$ with the UNP, where $(J, \boldsymbol{b})$ is c-acyclic, is it the case that $\Pi_i(I_i, \boldsymbol{a}_i) \to (J, \boldsymbol{b})$?*

Proof. The NExpTime-hardness results in [10, 46] are based on encodings of a (single, fixed) domino system whose tiling problem is NExpTime-hard. Let N be the number of tile types of this domino system. Specifically, Thm. 1(3) in [10] shows that PHP is NExpTime-hard *for instances over a fixed schema consisting of a single binary relation, and without distinguished elements.* The target instance J used in this construction contains a value for each of the N tile type as well as a variable number of other values. This instance J is, in general, not acyclic. However, as it turns out, careful inspection shows that every cycle in the incidence graph of J passes through one of the N values that correspond to tile types. We can exploit this as follows: for each instance I_i let I_i^* be the instance that extends I_i with N distinct, isolated distinguished elements. Furthermore, let J^* be a copy of the instance J where each of the N values that denotes a tile type, becomes a distinguished element. It is easy to see that $\Pi_i I_i^* \to J^*$ if and only if $\Pi_i I_i \to J$. By construction, J^* is c-acyclic.

□

Lemma B.1. *For pointed instances $(I_1, \boldsymbol{a}_1), \ldots, (I_n, \boldsymbol{a}_n)$ and $(J, \boldsymbol{b})$ of the same arity and with the UNP, the following are equivalent:*

(1) $\Pi_i(I_i, \boldsymbol{a}_i) \to (J, \boldsymbol{b})$
(2) $\Pi_i((I_i, \boldsymbol{a}_i) \uplus (J, \boldsymbol{b})) \to (J, \boldsymbol{b})$

Proof. (sketch) A homomorphism from $\Pi_i(I_i, \mathbf{a}_i)$ to $(J, \mathbf{b})$, can be extended to a homomorphism from $\Pi_i((I_i, \mathbf{a}_i) \uplus (J, \mathbf{b}))$ to $(J, \mathbf{b})$ by sending every k-tuple that contains at least one value from $(J, \mathbf{b})$, to the first element of the k-tuple in question that is a value from $(J, \mathbf{b})$. The converse direction is trivial. □

Theorem 3.23. *The following problems are NExpTime-hard:*

(1) The verification problem for most-specific fitting CQs.
(2) The verification problem for unique fitting CQs.
(3) The existence problem for unique fitting CQs.
(4) The verification problem for bases of most-general fitting CQs.
(5) The existence problem for bases of most-general fitting CQs.

Each problem is NExpTime-hard already for a fixed schema and arity, and, in the case of the verification problems, when restricted to inputs where the input CQ fits the examples, or, in the case of the existence problems, when restricted to inputs where a fitting CQ exists.

Proof. (1) By a reduction from the PHP, which is NExpTime-hard already for $k = 0$ and over a schema consisting of a single binary relation [10]. Given $I_1, \ldots, I_n$ and J, let E be the set of labeled examples that contains $(I_1 \uplus J), \ldots, (I_n \uplus J)$ as positive examples, and that does not contain any negative examples. Let q be the canonical query of J. It is clear from the construction that q fits E. Furthermore, q is a most-specific fitting CQ for E if and only if (by Prop. 3.5) $\Pi_i(I_i \uplus J) \to J$ if and only if (by Lemma B.1) $\Pi_i I_i \to J$.

(2–3) By reduction from Thm. 3.22. It follows from Prop. 3.20 and Lemma B.1 that, if J is c-acyclic, then the following are equivalent:

(1) $(I_1, \mathbf{a}_1) \times \cdots \times (I_n, \mathbf{a}_n) \to (J, \mathbf{b})$
(2) q is a unique fitting CQ for $(E^+ = \{(I_i, \mathbf{a}_i) \uplus (J, \mathbf{b}) \mid 1 \leq i \leq n\}, E^- = F)$
(3) There is a unique fitting CQ for $(E^+ = \{(I_i, \mathbf{a}_i) \uplus (J, \mathbf{b}) \mid 1 \leq i \leq n\}, E^- = F)$

where F is the frontier of $(J, \mathbf{b})$ (which can be computed in polynomial time, since $(J, \mathbf{b})$ is c-acyclic) and where q is the canonical CQ of $(J, \mathbf{b})$. Therefore, the verification and the existence problem for unique fitting CQs are both NExpTime-hard.

(4–5) Next, we show how to modify this reduction to also show hardness for the verification and existence problems for *bases* of most-general CQs. For any k-ary pointed instance $(C, \mathbf{a})$ over schema $\mathcal{S}$, we will denote by $(C, \mathbf{a})^*$ the $k+1$-ary pointed instance over schema $\mathcal{S}^* = \mathcal{S} \cup \{R, P\}$, that extends $(C, \mathbf{a})$ with a fresh designated element a_{k+1} and a non-designated element d, and with facts $R(a_{k+1}, d)$ and $P(d)$. In addition, we denote by $(C_{sink}, \mathbf{c})$ the pointed instance with distinguished elements $\mathbf{c} = c_1, \ldots, c_{k+1}$ and non-distinguished element d, consisting of all possible $\mathcal{S}$-facts over $\{c_1, \ldots, c_{k+1}\}$ and all possible $\mathcal{S}^*$ facts over $\{d\}$. We claim that the following are equivalent:

(1) $(I_1, \mathbf{a}_1) \times \cdots \times (I_n, \mathbf{a}_n) \to (J, \mathbf{b})$
(2) $(I_1, \mathbf{a}_1)^* \times \cdots \times (I_n, \mathbf{a}_n)^* \to (J, \mathbf{b})^*$
(3) q is a unique fitting CQ for $(E^+ = \{(I_i, \mathbf{a}_i)^* \uplus (J, \mathbf{b})^* \mid 1 \leq i \leq n\}, E^- = F \cup \{(C_{sink}, \mathbf{c})\})$
(4) There is a basis of most-general fitting CQ for $(E^+ = \{(I_i, \mathbf{a}_i)^* \uplus (J, \mathbf{b})^* \mid 1 \leq i \leq n\}, E^- = F \cup \{(C_{sink}, \mathbf{c})\})$

where F is the frontier of $(J, \mathbf{b})^*$ (which can be computed in polynomial time, since $(J, \mathbf{b})^*$ is c-acyclic) and where q is the canonical CQ of $(J, \mathbf{b})^*$. Therefore, the verification and the existence problem for unique fitting CQs are both NExpTime-hard.

The equivalence between (1) and (2) is easy to see. Note that $\Pi_i((I_i, \mathbf{a}_i)^*)$ is homomorphically equivalent to $(\Pi_i(I_i, \mathbf{a}_i))^*$. The argument for the implication from (2) to (3) is similar to the one we gave above with the simpler reduction (note that q clearly does not map homomorphically to $(C_{sink}, \mathbf{c})$). The implication from (3) to (4) is trivial, because a unique fitting CQ, by definition, constitutes a singleton basis of most-general fitting CQs. It therefore remains only to show that (4) implies (2).

Let $\{q_1, \ldots, q_m\}$ be a basis of most-general fitting CQs for (E^+, E^-) and assume towards a contradiction that (2) fails. Let

$p = \Pi_{e \in E^+}(e)$. It is not hard to see that p is homomorphically equivalent to $\Pi_i((I_i, \mathbf{a}_i) \uplus (J, \mathbf{b}))^*$. With a slight abuse of notation, in what follows we will identify p with $\Pi_i((I_i, \mathbf{a}) \uplus (J, \mathbf{b}))^*$, so that we can speak about the unique R-edge in p. For $i \geq 1$, let p'_i be obtained from p by replacing this R-edge by a zig-zag path of length i, i.e., an oriented path of the form $\rightarrow (\leftarrow \rightarrow)^i$. Clearly, $p'_i \rightarrow p$. Therefore, in particular p'_i fits the positive examples E^+. Also, clearly, p'_i fits the negative example $(C_{sink}, \mathbf{c})$. It also fits the other negative examples: Suppose p'_i had a homomorphism h to $e \in F$. Since F is a frontier for $(J, \mathbf{b})^*$, this implies that $p'_i \rightarrow (J, \mathbf{b})^*$. Since $(J, \mathbf{b})^*$ has a unique R-edge it follows that every R-edge in the zigzag of p'_i must necessarily be mapped to it by h. In turn, this implies that p is homomorphic to $(J, \mathbf{b})^*$, contradiction the fact that (2) fails. Thus, each p'_i fits (E^+, E^-). Hence, some member q_j of the basis must homomorphically map to infinitely many p'_i. It follows that q_j does not contain any finite undirected R-path from a designated element to a value satisfying the unary P (for if such a path existed, of length ℓ, then q_j would not map to p'_i for any $i > \ell$). It follows that q_j maps to $(C_{sink}, \mathbf{c})$, a contradiction. □

Theorem 3.24. *The existence problem for weakly most-general fitting CQs is ExpTime-hard.*

The proof of Theorem 3.24 is given in Appendix D.6, where the result is established simultaneously for CQs and for tree CQs.

Theorem 3.25. *Fix a schema consisting of a single binary relation. For $n > 0$, we can construct a collection of Boolean data examples of combined size polynomial in n such that a fitting CQ exists, but not one of size less than 2^n.*

Proof. For $i \geq 1$, let C_{p_i} denote the directed cycle of length p_i, with p_i the i-th prime number (where $p_1 = 2$). Note that, by the prime number theorem, C_{p_i} is of size $O(i \log i)$. Let $E_n^+ = \{C_{p_i} \mid i = 2, \ldots, n\}$ and let $E_n^- = \{C_{p_1}\}$. Then it is easy to see that a fitting CQ for (E_n^+, E_n^-) exists (namely the any cycle whose length is a common multiple of the lengths of the cycles in E_n^+). Furthermore, every fitting CQ must necessarily contain a cycle of odd length (in order not to fit the negative example), and the length of this cycle must be a common multiple of the prime numbers $p_2, \ldots, p_n$ (in order to fit the positive examples). This shows that the query must have size at least 2^n. □

Remark B.2. Continuing from Remark B.1, the above proof can be adapted to also apply also to the *CQ definability* problem: let I_n be the disjoint union $\biguplus_{i=1\ldots n} C_{p_i}$, let $E_n^+ = \{(I_n, a) \mid a \text{ lies on the cycle of length 2}\}$ and $E_n^- = \{(I_n, a) \mid a \text{ lies on a cycle of length greater than 2}\}$. By the same reasoning as before, there is a unary CQ that fits these examples, but every unary CQ that fits must have size at least 2^n.

Theorem 3.26. *For $n \geq 0$, we can construct a schema with $O(n)$ unary and binary relations and a collection of labeled examples of combined size polynomial in n such that*

(1) There is a unique fitting CQ.
(2) Every fitting CQ contains at least 2^n variables.

Proof. The follow construction is inspired by the lower bound arguments in [10]. The schema contains relations $T_1, \ldots, T_n$, $F_1, \ldots, F_n$ (used to encode bit-strings of length n) and $R_1, \ldots, R_n$ where the intended interpretation of R_i is "the successor relation on bit-strings, restricted to pairs of bit-strings where the i-th bit is the one that flips to from 0 to 1". Note that the union of these R_i's is precisely the ordinary successor relation on bit-strings.

Next, we describe the positive examples. For each $i \leq n$, let P_i be the two-element instance that has domain $\{0, 1\}$ and contains the following facts:

- $F_i(0)$ and $T_i(1)$
- all facts involving unary relations T_j and F_j for $j \neq i$
- all facts $R_j(0, 0)$ and $R_j(1, 1)$ for $j < i$
- all facts $R_i(0, 1)$
- all facts $R_j(1, 0)$ for $j > i$

Let P be the direct product $P_1 \times \cdots \times P_n$. It can easily be verified that P is a directed path of length 2^n, starting with $\langle 0, \ldots, 0 \rangle$ and ending with $\langle 1, \ldots, 1 \rangle$, where the unary and binary relations have the intended interpretation as described above. For example, if $n = 2$, then the instance P can be depicted as follows:

$$\langle 0,0 \rangle \xrightarrow{R_2} \langle 0,1 \rangle \xrightarrow{R_1} \langle 1,0 \rangle \xrightarrow{R_2} \langle 1,1 \rangle$$

Our negative example is the instance N, whose domain consists of $3n$ values, $a_1, \ldots, a_n$ and $b_1, \ldots, b_n$, and $c_1, \ldots, c_n$, and such that N contains the following facts:

- All facts over domain $A = \{a_1, \ldots, a_n\}$ except $T_i(a_i)$ for $i \leq n$;
- All facts over domain $B = \{b_1, \ldots, b_n\}$ except $F_i(b_i)$ for $i \leq n$;
- All facts over domain $C = \{c_1, \ldots, c_n\}$ except $T_i(c_i)$ and $F_i(c_i)$ for $i \leq n$;
- All binary facts $R_j(x, y)$ where $x \in B$ and $y \in A$
- All binary facts $R_j(x, y)$ where $x \in C$ or $y \in C$

In particular, note that there are no directed edges going from the A cluster to the B cluster.

We claim that P does not map to N. Indeed, the only values in N that satisfy $F_1(x) \wedge \cdots \wedge F_n(x)$ (and hence, to which the value $\langle 0, \ldots, 0 \rangle$ of P could be mapped) are $a_1, \ldots, a_n$, while the only values in N that satisfy $T_1(x) \wedge \cdots \wedge T_n(x)$ (and hence, to which the value $\langle 1, \ldots, 1 \rangle$ of P could be mapped), are $b_1, \ldots, b_n$. By construction, the latter cannot be reached from the former by a path consisting of forward edges only, except if the path goes through the C cluster. Note that the values in the C cluster are not viable candidates for the homomorphism because each fails to satisfy $T_i \vee F_i$ for some i. It follows that the canonical CQ of P is a fitting CQ, and, indeed, a most-specific fitting CQ. In the remainder of the proof, we show that it is in fact a unique fitting CQ.

Let q' be any fitting CQ and let I be its canonical instance. Then I maps to P and not to N. Consider any connected component of I that does not map to N.

The component in question must contain a node a satisfying all $F_1, \ldots, F_n$ (otherwise the entire component could be mapped to the $\{b_1, \ldots, b_n\}$-subinstance of N. Similarly, it must contain a node b satisfying all $T_1, \ldots, T_n$ (otherwise the entire component could be mapped to the $\{a_1, \ldots, a_n\}$-subinstance of N.

We can now distinguish three cases:

- There is no directed path in I from a node a satisfying $F_1, \ldots, F_n$ to a node b satisfying $T_1, \ldots, T_n$. In this case, let X be the set of all values of I that are reachable by a directed path from an value satisfying $F_1, \ldots, F_n$. Take any map h that sends every

$x \in X$ to some a_j (where j is chosen so that x omits T_j) and that sends every value y outside X to a b_j of N (where j is chosen so that y omits F_j). Then h is a homomorphism from I to N, contradicting the fact that q' is a fitting CQ. Therefore, this cannot happen.

- Every directed path in I from a node a satisfying $F_1, \ldots, F_n$ to a node b satisfying $T_1, \ldots, T_n$, contains a "bad" node, by which we mean a value that, for some $j \leq n$, fails to satisfy either T_j or F_j. In this case, let X be the set of all values of I that can be reached from a node satisfying $F_1, \ldots, F_n$ by a path that does not contain bad nodes. We construct a homomorphism from I to N by sending all the values in X to a suitable a_j; all bad nodes to c_j (where j is such that the bad node in question fails to satisfy T_j or F_j); all other nodes to suitable b_j. Therefore, again, we have a contradiction, showing that this cannot happen.
- There are nodes a and b satisfying $F_1, \ldots, F_n$ and $T_1, \ldots, T_n$, respectively, such that there is a directed path from a to b that does not contain any bad node. In this case, we can easily see that the homomorphism from I to P maps this path bijectively to P and hence its inverse contains a homomorphism from P to I. It follows that I and P are homomorphically equivalent, and hence, q' is logically equivalent to q.

□

Theorem 3.27. *For $n \geq 0$, we can construct a schema with $O(n)$ unary and binary relations and a collection of labeled examples of combined size polynomial in n such that*

(1) There is a basis of most-general fitting CQs.
(2) Every such basis contains at least 2^{2^n} CQs.

Proof. It suffices to make minor changes to the construction used in the proof of Thm. 3.26. Specifically, (i) we expand the schema with unary relation symbols Z_0 and Z_1, (ii) we extend the positive examples and the negative example with all possible Z_0- and Z_1-facts over their domain, and (iii) we extend the negative example N with one further value z where z satisfies all possible unary facts except $Z_0(z)$ and $Z_1(z)$, as well as all binary facts $R_j(x, y)$ for which it holds that $z \in \{x, y\}$.

Let P be the direct product $P_1 \times \cdots \times P_n$. It can again be verified that P is a directed path of length 2^n, starting with $\langle 0, \ldots, 0\rangle$ and ending with $\langle 1, \ldots, 1\rangle$, where the unary relations $T_1, \ldots, T_n, F_1, \ldots, F_n$ and the binary relations $R_1, \ldots R_n$ have the intended interpretation, and such that the unary relation symbols Z_0 and Z_1 are true everywhere.

Let X be the set containing all subinstances of P obtained by removing, for each node x in its domain, exactly one of the facts $Z_0(x)$ or $Z_1(x)$. We shall show that $(X, \{N\})$ is a homomorphism duality relative to P.

First, note that every instance in X is not homomorphic to P. Now, let Q be any instance satisfying $Q \to P$ and $Q \not\to N$. We need to show that Q admits an homomorphism from some instance in X. To do so, let Q' be any connected component of Q such that $Q' \not\to N$.

By the same arguments as in the proof of Thm. 3.26, Q' contains nodes a and b satisfying $F_1, \ldots, F_n$ and $T_1, \ldots, T_n$ respectively and there a directed path from a to b containing no bad nodes (that is a value that for some $j \leq n$, fails to satisfy either T_j or F_j). Mimicking the same arguments it is immediate to show that, additionally, every node in this directed path satisfies Z_0 or Z_1. Since $Q \to P$ there is an homomorphism h from this directed path to P. It is easy to see that h must be bijective. Since every node x in the path satisfies Z_0 or Z_1 it follows that the inverse of h defines an homomorphism from some instance in X to Q.

Finally, note that X has 2^{2^n} values and that every pair of instances in X is not homomorphically equivalent. By Proposition 3.15, the set containing the canonical queries of instances in X, which clearly fits, is a minimal basis of most general fitting CQs. □

B.1 Proof of Thm. 3.13 (via Automata that accept Weakly Most-General Fitting CQs)

This section is devoted to the proof of:

Theorem 3.13. *The existence problem for weakly most-general fitting CQs is in ExpTime. Moreover, if such a CQ exists, then*

(1) there is one of doubly exponential size and
(2) we can produce one in time $2^{poly(n)} + poly(m)$ where $n = ||E||$ and m is the size of the smallest weakly most-general fitting CQ.

The main result of this section is that, given collection of labeled examples E, we can construct in exponential time a tree automaton that accepts (suitable encodings of) weakly most-general fitting c-acyclic CQs for E. The existence problem for weakly most-general fitting CQs then reduces to the emptiness problem for the corresponding automaton, and hence can be solved in ExpTime. Similarly, the other claims in Theorem 3.13 follow by basic facts from automata theory, cf. Theorem B.5 below. Note that, by Prop. 3.11 and Thm. 2.1, the core of a weakly most-general fitting CQ is always c-acyclic, and hence we can restrict attention to c-acyclic CQs here. In fact, we will restrict attention to c-acyclic CQs with the Unique Names Property (UNP). We will explain afterwards how to lift the UNP restriction.

First, we must introduce tree automata.

Definition B.2 (d-ary Σ-trees). Fix a finite alphabet Σ and a $d > 0$. A *d-ary Σ-tree* is a pair (T, Lab) where T is a non-empty prefix-closed finite subset of $\{1, \ldots, d\}^*$, and $Lab : T \to \Sigma$. By abuse of notation, we will sometimes use the symbol T to refer to the pair (T, Lab) and we will write Lab^T for Lab. For $(i_1, \ldots, i_n) \in T$, we will denote $(i_1, \ldots, i_n)$ also by $Suc_{i_n}((i_1, \ldots, i_{n-1}))$. Note that the empty sequence ε belongs to every tree.

This permits a node to have a Suc_i-successor without having a Suc_j-successor for $j < i$.

Definition B.3 (Non-Deterministic Tree Automaton). A d-ary non-deterministic tree automaton (NTA) is a tuple $\mathfrak{A} = (Q, \Sigma, \Delta, F)$ where

- Q is a finite set of *states*
- Σ is a finite alphabet
- $\Delta \subseteq (Q \cup \{\bot\})^{\{1,\ldots,d\}} \times \Sigma \times Q$ is the *transition relation*
- $F \subseteq Q$ is the set of *accepting states*.

When specifying an automaton, for the sake of readability, we will use the notation $\langle q_1, \ldots, q_d\rangle \overset{\sigma}{\Longrightarrow} q$ for transitions, instead of writing $\langle q_1, \ldots, q_d, \sigma, q\rangle \in \Delta$.

Definition B.4 (Accepting Run; Acceptance). An *accepting run* of a d-ary NTA $\mathcal{A} = (Q, \Sigma, \Delta, F)$ on a d-ary Σ-tree T is a mapping $\rho : T \to Q$ such that:

- $\rho(\varepsilon) \in F$, and
- for each $t \in T$, the transition

$$\langle \rho_1(t), \ldots, \rho_d(t) \rangle \xRightarrow{Lab^T(t)} \rho(t)$$

belongs to Δ, where $\rho_i(t) = \rho(t \cdot i))$ if $(t \cdot i) \in T$, and $\rho_i(t) = \bot$ otherwise.

When such an accepting run exist, we say that $\mathfrak{A}$ *accepts* T. The *tree language recognized by* $\mathfrak{A}$ (denoted by $L(\mathcal{A})$) is the set of all d-ary Σ-trees $\mathfrak{A}$ accepts.

The following theorem lists a number of well-known facts about non-deterministic tree automata, which can be found in any standard textbook on tree automata.[4]

Theorem B.5.

(1) The problem to decide, given an NTA $\mathfrak{A}$, whether $L(\mathfrak{A})$ is non-empty, is in PTime.
(2) Given an NTA $\mathfrak{A}$ for which $L(\mathfrak{A})$ is non-empty, we can compute in polynomial time a succinct representation (in the form of a directed acyclic graph) of a tree T of minimal size accepted by $L(\mathfrak{A})$.
(3) For a Boolean combinations of NTAs, we can construct in ExpTime an NTA that defines the same tree language.
(4) For a constant number of NTAs, we can construct in polynomial time an NTA that defines the intersection.

Here, and in what follows, d and Σ are not treated as a fixed constant in the complexity analysis, but as part of the input (and it's assumed d is given in unary).

Step 1: Encoding c-acyclic CQs (with UNP) as trees

To simplify the presentation in the remainder of this section, let us fix a schema $\mathbf{S}$ and arity $k \geq 0$. Furthermore choose some $d >$ max-arity($\mathbf{S}$) (Proposition B.10 below will tell us more precisely how to choose d).

We will encode k-ary c-acyclic CQs over $\mathbf{S}$ by trees over the alphabet

$$\Sigma = \{\langle R, \pi \rangle \mid R \in \mathbf{S} \text{ and } \pi \in (\{\mathsf{up}, \mathsf{down}, \mathsf{ans}_1, \ldots, \mathsf{ans}_k\})^{\text{arity}(R)})\} \cup \{\nu\}$$

where ν is a new symbol. The intuition behind this choice of alphabet is as follows: each node of the tree, other than the root node, represents an existentially quantified variable or a fact (i.e., an atomic conjunct) of the query. The nodes at even distance from the root represent existentially quantified variables while the nodes at odd distance from the root represent facts. Each node of the tree that represents a fact has a label of the form $\langle R, \pi \rangle$, where R indicates the relation of the fact, and π describes the arguments of the fact.

Definition B.6 (Proper d-ary Σ-tree). We say that a d-ary Σ-tree (T, Lab) is *proper* if the following hold for every $t \in T$ (where $|t|$ denotes the length of the sequence t):

[4]Item (2) can be shown by a straightforward dynamic programming algorithm that computes, for each state s of the automaton (starting with the final states) a pair (T, n), where T is a succinct representation of a tree accepted by the automaton from starting state s, and n is the size of the corresponding non-succinct representation.

C-acyclic query: $q(x_1, x_2) \text{ :- } \exists z, z'(R(x_1, z) \wedge R(z, z') \wedge R(x_1, z') \wedge P(x_2))$

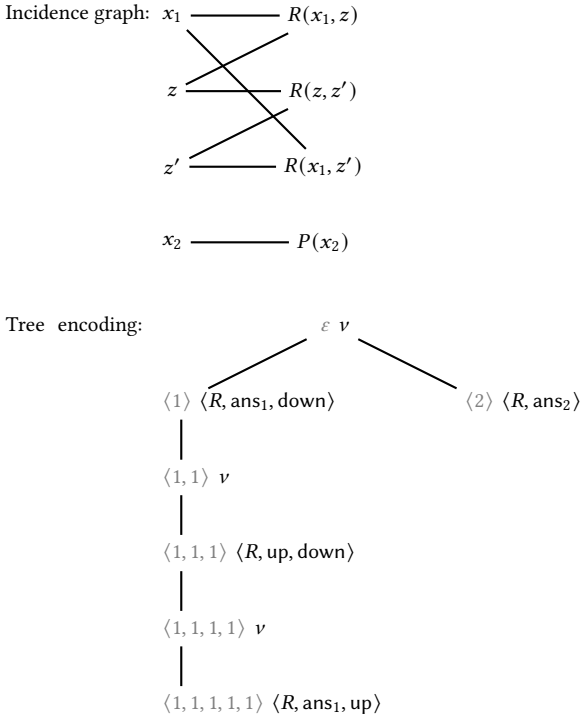

Figure 1: Example of a c-acyclic CQ and its encoding as a d-ary Σ-tree

- If $|t|$ is even, then $Lab(t) = \nu$. In particular, $Lab(\varepsilon) = \nu$
- If $|t| = 1$, then $Lab(t)$ is of the form $\langle R, \pi \rangle$ for some π not containing up.
- If $|t|$ is odd and $|t| > 1$, then $Lab(t)$ is of the form $\langle R, \pi \rangle$ for some π that includes exactly one occurrence of up.
- If $Lab(t) = \langle R, \pi \rangle$ with $\pi = dir_1 \ldots dir_n$, then we have that
  (1) for all $i \leq n$, $dir_i = \mathsf{down}$ iff $(t \cdot i) \in T$, and
  (2) for all $i > n$, $(t \cdot i) \notin T$.
- For each $i \leq k$, there exists at least one node whose label is of the form $\langle R, \pi \rangle$ where π contains ans_i.

Definition B.7 (The CQ encoded by a proper tree). Let $T = (T, Lab)$ be a proper d-ary Σ-tree. We construct a corresponding CQ $q_T(x_1, \ldots, x_k)$ as follows:

- For each non-root $t \in T$ labeled ν, q_T contains an existentially quantified variable y_t.
- For each $t \in T$ labeled $\langle R, \pi \rangle$ with $\pi = dir_1, \ldots, dir_n$, q_T contains a conjunct of the form $R(u_1, \ldots, u_n)$ where $u_i = x_j$ if $dir_i = \mathsf{ans}_j$; $u_i = y_{(t \cdot i)}$ if $dir_i = \mathsf{down}$; and $u_i = y_{t'}$ if $dir_i = \mathsf{up}$ and t' is the parent of t in T.

We leave it to the reader to verify that q_T is indeed a well-defined CQ. In particular the 5th bullet in the above definition of proper trees guarantees that q_T satisfies the safety condition.

An example of a tree encoding of a c-acyclic CQ is given in Figure 1.

Remark B.3. *The size of q_T, as counted by the number of existential quantifiers plus the number of conjuncts, is at most the number of nodes of T minus 1.*

Definition B.8. A CQ is said to be *encodable by a d-ary Σ-tree* if there is a proper d-ary Σ-tree T such that q_T is equal to q (up to a one-to-one renaming of variables).

For the next proposition, we need to introduce some terminology. The *fact graph* of q is the (undirected) graph whose nodes are the facts of q and such that there is an edge connecting two facts if they share an existential variable. By an *fg-connected component* of q we will mean a connected component of the fact graph of q.

Proposition B.9. *A k-ary CQ q over $\mathbf{S}$ is encodable by a d-ary Σ-tree (for $d \geq$ max-arity($\mathbf{S}$)) if and only if the following hold:*

(1) q is c-acyclic,
(2) q has the UNP,
(3) q has at most d fg-connected components, and
(4) every existential variable of q occurs in at most $d+1$ facts.

Indeed, if a c-acyclic CQ q meets these conditions, then q is encodable by a d-ary Σ-tree T where the number of nodes of T is equal to the number of existential variables of q plus the number of conjuncts of q plus 1.

Proof. (sketch) If T is a proper d-ary Σ-tree, then, it is clear from the definitions that q_T satisfies (1) – (4). Conversely, let $q(x_1, \ldots, x_k)$ be a k-ary query that satisfies conditions (1) – (4). Let $q_1, \ldots, q_n$ be the fg-connected components of q (where $n \leq d$). Furthermore, choose arbitrarily one fact from each fg-connected component. We will denote the chosen fact from q_i by f_i. We construct a mapping g from existential variables and facts of q to sequences in $\{1, \ldots, d\}^*$ as follows, iteratively:

- $g(f_i) = \langle i \rangle$ for $i \leq n$.
- Let $f = R(z_1, \ldots, z_m)$ be a fact of q_i (for some $i \leq n$) and suppose that z_j is an existential variable (for some $i \leq m$). We say that z_i is a *parent variable* of f, if z_i lies on the shortest path from f_i to f in the incidence graph of q_i. Note that, by c-acyclicity, there can be at most one such parent variable. If z_j is *not* a parent variable of f and $g(f) = t$, then $g(z_j) = t \cdot j$.
- Let z be an existential variable of q_i and let $f'_1, \ldots, f'_m$ be the facts in which z occurs (with $m \leq d+1$). Exactly one of of these facts must lie on the shortest path from f_i to z in the incidence graph of q_i, and we will will refer to it as the *parent fact* of z. For each fact f'_j that is *not* the parent fact of z, and $g(z) = t$, then we set $g(f'_j) = \pi \cdot j$.

The image of the map g thus obtained, is a non-empty prefix closed subset of $\{1, \ldots, d\}^*$ if we add to it also the empty sequence ε. We can expand this into a d-ary Σ-tree T by defining a suitable labeling function λ. Specifically, we set $\lambda(x) = \nu$ if $x = \varepsilon$ or if $x \in \{1, \ldots, d\}^*$ is the g-image of an existential variable; if $x \in \{1, \ldots, d\}^*$ is the g-image of a fact $R(z_1, \ldots, z_m)$, we set $\lambda(x) = (R, \langle dir_1, \ldots, dir_m \rangle)$, where $dir_j = \mathsf{ans}_i$ if $z_j = x_i$; $dir_j =$ up if z_j is the answer variable of f, and $dir_j =$ down otherwise. See Figure 1 for an example. It is then easy to see that q_T is isomorphic to q. □

Proposition B.10. *Let $E = (E^+, E^-)$ be a collection of k-ary labeled examples over a schema $\mathcal{S}$, and let $d = \max\{\text{max-arity}(\mathcal{S}), ||E^-||\}$ If there exists a weakly most-general fitting CQ with UNP for E, then there is one that is core and encodable by a d-ary Σ-tree.*

Proof sketch. Let $q(\mathbf{x})$ be a weakly most-general fitting CQ with UNP for E. Without loss of generality, we may assume that q is minimal, in the sense that no strict sub-query of q fits E. In particular, then, q is a core, and, it follows from Prop. 3.11 and Thm. 2.1 that q is c-acyclic.

We know that q fits the negative examples, which means that for each negative example, there is an fg-connected component of q that does not homomorphically map to that negative example. It follows from the minimality assumption that the number of fg-connected components of q is at most $|E^-| \leq d$.

Next, we tackle the degree. It follows from the minimality assumption that q is a core. Hence, by Prop. 3.11 and Thm. 2.1, q is c-acyclic. That is, the incidence graph of q can be represented as a tree, whose leaves may be marked by answer variables. Since $d >$ max-arity($\mathcal{S}$), every fact-node in the incidence graph has at most d variable-children. Now, let y be any variable-node in the incidence graph, and let $f_1, \ldots, f_n$ be its fact-children. We will show that $n \leq ||E^-|| \leq d$.

For each $i \neq n$, let q_i be the subquery of q that is rooted at that fact. Let

$$S_i = \{(e, d) \mid e = (I, \mathbf{c}) \in E^-, d \in adom(I), (q_i, \mathbf{x}, y) \to (I, \mathbf{c}, d)\} .$$

In other words, S_i is set of all values from the negative examples, to which y can be mapped by a homomorphism from q_i. Let $T_i = \bigcap_{j=1\ldots i} S_i$. Clearly, the sequence $T_1, T_2, \ldots$ is decreasing in the sense that $T_{i+1} \subseteq T_i$. Furthermore, we know that $T_n = \emptyset$, because q fits E and hence q does not have a homomorphism to any of the negative examples. We claim that the sequence $T_1, T_2, \ldots$ must be *strictly* decreasing, and hence, its length is bounded by $||E^-||$. Suppose, for the sake of a contradiction, that $S_i = S_{i+1}$. In particular, then $\bigcap_{j=1\ldots,i,i+2\ldots n} S_j = \emptyset$. It follows that the subquery q' be obtained from q by removing the subquery consisting of all facts and variables belonging to the subtree q_{i+1}, fits E. This contradicts our initial minimality assumption on q. □

Step 2: A Tree Automaton that Accepts Fitting C-Acyclic CQs

Lemma B.11. *Given a schema, $k \geq 0$, and $d > 0$, we can construct in polynomial time a d-ary NTA $\mathfrak{A}$ that accepts precisely the proper d-ary Σ-trees.*

Proof. (sketch) To simplify the presentation, here, we describe the automaton that accepts a d-ary Σ-tree T if and only if T satisfies the first four conditions of Definition B.6. The automaton can easily be extended to test the last condition as well. Our automaton has four states: $Q = \{q_{\text{root}}, q_{\text{root-fact}}, q_{\text{fact}}, q_{\text{exvar}}\}$, where $F = \{q_{\text{root}}\}$. The transition relation Δ consists of all transitions of the form

- $\langle q_1, \ldots, q_d \rangle \xrightarrow{\nu} q_{\text{root}}$ where $\langle q_1, \ldots, q_d \rangle \in \{q_{\text{root-fact}}\}^+\{\bot\}^*$
- $\langle q_1, \ldots, q_d \rangle \xrightarrow{\langle R, dir_1, \ldots, dir_\ell \rangle} q_{\text{root-fact}}$, where, for each $i \leq \ell$, either $dir_i =$ down and $q_i = q_{\text{exvar}}$, or $dir_i \in \{\mathsf{ans}_1, \ldots, \mathsf{ans}_k\}$ and $q_i = \bot$; furthermore $q_i = \bot$ for $i > \ell$
- $\langle q_1, \ldots, q_d \rangle \xrightarrow{\nu} q_{\text{exvar}}$ where $\langle q_1, \ldots, q_d \rangle \in \{q_{fact}\}^*\{\bot\}^*$

- $\langle q_1, \ldots, q_d \rangle \xrightarrow{\langle R, dir_1, \ldots, dir_\ell \rangle} q_{\text{fact}}$, where, for each $i \leq \ell$, either $dir_i = \text{down}$ and $q_i = q_{\text{exvar}}$, or $dir_i \in \{\text{up}, \text{ans}_1, \ldots, \text{ans}_k\}$ and $q_i = \bot$; furthermore, $q_i = \bot$ for $i > \ell$, and there is exactly one $i \leq \ell$ for which $dir_i = up$.

It is easily verified that this automaton accepts precisely the d-trees that satisfy the first four conditions of Definition B.6. □

Lemma B.12. *Given a schema, $k \geq 0$, $d > 0$ and a k-ary data example $e = (I, \boldsymbol{a})$, we can construct in polynomial time, a NTA $\mathfrak{A}_e$, such that, for all proper d-ary Σ-trees T, we have $T \in L(\mathfrak{A}_e)$ if and only if q_T fits e as a positive example (i.e., $\boldsymbol{a} \in q_T(I)$).*

Proof. The set of states Q of the automaton consists of:

- an accepting state q^{root},
- a state $q^{\text{root-fact}}_{R(b_1,\ldots,b_n)}$ for each fact $R(b_1, \ldots, b_n)$ of I,
- a state $q^{\text{fact}}_{R(b_1,\ldots,b_n),j}$ for each fact $R(b_1, \ldots, b_n)$ of I, and $j \leq n$,
- a state q^{exvar}_b for each $b \in adom(I)$.

The transition relation Δ contains all transitions of the form:

- $\langle q_1, \ldots, q_d \rangle \xrightarrow{\nu} q^{\text{root}}$ where each $q_i \in \{\bot, q^{\text{root-fact}}_{R(b_1,\ldots,b_n)} \mid R(b_1, \ldots, b_n) \text{ is a fact of } I\}$,
- $\langle q_1, \ldots, q_d \rangle \xrightarrow{\nu} q_a$ where each $q_i \in \{\bot, q^{\text{fact}}_{R(b_1,\ldots,b_n),j} \mid R(b_1, \ldots, b_n) \text{ is a fact of } I \text{ and } b_j = a\}$
- $\langle q_1, \ldots, q_n, \bot, \ldots, \bot \rangle \xrightarrow{\langle S, dir_1, \ldots, dir_n \rangle} q^{\text{root-fact}}_{S(b_1,\ldots,b_n)}$ where for each $i \leq n$, either $dir_i = \text{ans}_\ell$ for some $\ell \leq k$ and $b_i = a_\ell$, or else $dir_i = \text{down}$ and $q_i = q^{\text{exvar}}_{b_i}$
- $\langle q_1, \ldots, q_n, \bot, \ldots, \bot \rangle \xrightarrow{\langle S, dir_1, \ldots, dir_n \rangle} q^{\text{fact}}_{(S(b_1,\ldots,b_n),i)}$ where $dir_i = \text{up}$, and, for each $j \leq n$ with $j \neq i$, either $dir_j = \text{ans}_\ell$ for some $\ell \leq k$ and $b_j = a_\ell$, or else $dir_j = \text{down}$ and $q_j = q_{b_j}$

It is easy to verify that every accepting run of $\mathfrak{A}_e$ on a tree encoding T corresponds to a homomorphism from $q_T(\mathbf{a})$ to e, and vice versa. In particular, $\mathfrak{A}_e$ accepts T if and only if q_T fits e as a positive example. □

From the above two lemmas, as well as Thm. B.5, we immediately get:

Theorem B.13. *Given a schema, $k \geq 0$, $d > 0$, and a collection of labeled examples $E = (E^+, E^-)$, we can construct in exponential time an NTAs $\mathfrak{A}_E$ that defines the tree language consisting of all proper d-ary Σ-trees T for which it holds that q_T fits E.*

Step 3: An Automaton that Accepts Weakly-Most-General Fitting C-Acyclic CQs

Recall that every c-acyclic CQ has a frontier. Our next aim is to show that we can create, for a given collection of labeled examples E, an automaton that accepts (tree encodings of) those c-acyclic CQs with the UNP whose frontier consists of queries that *do not fit* E. Combining this with Thm. B.13, by Propositions 3.11, we then obtain an automaton that accepts precisely weakly most-general fitting c-acyclic CQs with the UNP.

We make use of a frontier construction from [11]. The presentation given here is slightly different to the one in [11] (because it is phrased in terms of conjunctive queries instead of finite structures), but it is equivalent.

We will denote the set of answer variables of a CQ q by ANSVAR_q and we will denote the set of existential variables by EXVAR_q. We denote the set of facts of q by FACTS_q.

Definition B.14 ($F(q)$). Let q be any fg-connected c-acyclic CQ with the UNP. Then $F(q)$ is the possible-unsafe CQ defined as follows:

- $\text{ANSVAR}_{F(q)} = \text{ANSVAR}_q$
- $\text{EXVAR}_{F(q)} = \{u_{(y,f)} \mid y \in \text{EXVAR}_q \text{ occurs in } f \in \text{FACTS}_q\} \cup \{u_x \mid x \in \text{ANSVAR}_q\}$. We will call each variable of the form $u_{y,f}$ a *replica* of the existential variable y. By the replicas of an answer variable $x \in \text{ANSVAR}_k$ we will mean u_x and x itself.
- $\text{FACTS}_{F(q)}$ consists of all acceptable instances of facts in FACTS_q, where an *acceptable instance* of a fact $f = R(z_1, \ldots, z_n) \in \text{FACTS}_q$ is a fact of the form $R(z'_1, \ldots, z'_n)$ where each z'_i is a replica of z_i, and for some $i \leq n$, either z'_i is of the form $u_{(z_i,f')}$ with $f' \neq f$, or z'_i is of the form u_{z_i}.

Definition B.15 (Frontier construction for c-acyclic CQs with the UNP [11]). Let $q(\mathbf{x})$ be a c-acyclic CQ with the UNP with m fg-connected components. Then $\mathcal{F}_q = \{F^i(q) \mid i \leq m)\}$, where $F^i(q)$ denotes the possibly-unsafe CQ obtained from q by performing the $F(\cdot)$ operation on the i-th fg-connected component (and leaving all other fg-connected components unchanged).

Proposition B.16 ([11]). *Let $q(\boldsymbol{x})$ be a c-acyclic CQ with the UNP.*

(1) Each query in $\mathcal{F}_q$ maps homomorphically to q, and
(2) If q is a core, then $\mathcal{F}_q$ is a frontier for q.

It is worth pointing out that $F(q)$ as constructed above is not necessarily c-acyclic. Furthermore, it may in fact not satisfy the safety condition that is part of the definition of CQs. Indeed, consider the (fg-connected) c-acyclic CQ $q(x) :- P(x)$. Then, $F(q)$ is the query $q'(x) :- Py$, which is an unsafe CQ. Consequently, $\mathcal{F}_q$ in general includes unsafe CQs. This will however not be a problem for what follows, because the characterization of weakly most-general fitting CQs in terms of frontiers (Prop. 3.11) applies also if the frontier is taken to include unsafe CQs.

Lemma B.17. *Given a schema, $k \geq 0$, $d > 0$, and a set E of k-ary data examples, we can construct, in exponential time, for every $i \leq d$, a NTA $\mathfrak{A}^{frontier}_{E,i}$, such that, for all proper d-ary Σ trees T, we have that $T \in L(\mathfrak{A}^{frontier}_{E,i})$ if and only if q_T has at least i fg-connected components and $F^i(q_T)$ admits a homomorphism to an example in E.*

Proof. We will show how to construct the automaton in the case of a single data example. The general result then follows because we can construct $\mathfrak{A}^{\text{frontier}}_{E,i}$ as the union of the (polynomially many) automata $\mathfrak{A}^{\text{frontier}}_{e,i}$ for all $e \in E$ (using non-determinism in the initial state transitions effectively to select the example).

Let $e = (I, \mathbf{a})$ with $\mathbf{a} = a_1, \ldots, a_k$.

The automaton $\mathfrak{A}^{\text{frontier}}_{e,i}$, intuitively, has to check two things: (1) it has to check that subtree from the i-th child of the root (which encodes the i-th fg-connected component of the query, encodes a subquery q' such that $F(q')$ fits e as a positive example, and (2) it has to check that, for every $j \neq i$, the subtree rooted at the j-th child of the root (if it exists), encodes a query that fits e as a

positive example. For (2) we already showed in Lemma B.12 how to do this, and in fact, we can include in our automaton a copy of the automaton from Lemma B.12 to handle this part. Note that no alternation is needed for this, as this is effectively a conjunction where each conjunct pertains to a different subtree of the input tree (i.e., a different child of the root).

Therefore, it suffices to focus on (1). Before we spell out the details of the automaton, we make two observations that provide the idea behind the automaton. First of all, recall that $F(q')$ has an existential variable $u_{(y,f)}$ for every pair (y, f), where y is an existential variable of q' and f is a fact in which y occurs. When we look at the tree encoding of the query, then we can see that each existential variable y of q' is a node in the tree encoding, and the facts in which y occurs are precisely the children and the parent of y in the tree encoding.

The second observation is that every existential variable of q' may have up to $d+1$ many replicas in $F(q')$, and a homomorphism from $F(q')$ to e is a map that, among other things, has to send each replica to a value in $adom(I)$. We want to set things up so that, from an accepting run of the automaton, we can obtain such a homomorphism. To do this, we can create a different state of the automaton for every $d+1$-length vector of values from $adom(I)$. Such a state then encodes, for a given variable, the value in $adom(I)$ that each of its replicas gets mapped to. In addition, for each answer-variable x of q', $F(q')$ also includes an existential variable u_x that has to be mapped to some value in $adom(I)$. We can include this information in the states as well by further increasing the length of the vector by k.

Based on these ideas, the construction of the automaton is now a relatively straightforward extension of the automaton given in the proof of Lemma B.12.

By a "vector" v we will mean a $d+k+1$-length vector of values from $adom(I)$. We say that two vectors v, v' are *compatible* if the last k items of the vectors are identical, i.e., $v[d+2,\ldots,d+k+1] = v'[d+2,\ldots,d+k+1]$. (Recall that the last k components of the vector are used to encode what u_{x_i} gets mapped to, for each answer variable x_i. By requiring all vectors to be compatible, we ensure that this choice is effectively made only once for the entire accepting run of the automaton.)

The states of the automaton include all states of the automaton $\mathfrak{A}_e$ given in the proof of Lemma B.12, plus:

- A state q^{exvar}_v for every vector v,
- A state $q^{\text{root-fact}}_{R,\langle v_1,\ldots,v_n\rangle}$ where $n = \text{arity}(R)$ and $v_1,\ldots,v_n$ are pairwise compatible vectors.
- A state $q^{\text{fact}}_{R,\langle v_1,\ldots,v_n\rangle,j,\ell}$ where $n = \text{arity}(R)$ and $j \le n$, $\ell \le d$, and $v_1,\ldots,v_n$ are pairwise compatible vectors.

For a state of the form $q^{\text{fact}}_{R,\langle v_1,\ldots,v_n\rangle,j,\ell}$, intuitively, j represents the index at which the parent existential variable occurs in the fact, while ℓ will merely be used to keep track that the current node is going to be the ℓ-th child of its parent in the d-tree.

The transitions include all the transitions of the automaton $\mathfrak{A}$, except for those going to the root state. In addition, we have:

- $\langle q_1,\ldots,q_d\rangle \xRightarrow{v} q^{\text{root}}$ where q_i is of the form $q^{\text{root-fact}}_{R,\langle v_1,\ldots,v_n}\rangle$, and where each q_j for $j \neq i$ is either $\bot$ or is a state from $\mathfrak{A}_e$ of the form $q^{\text{root-fact}}_{R(b_1,\ldots,b_n)}$,
- $\langle q_1,\ldots,q_d >\xRightarrow{v} q^{\text{exvar}}_v$ where each q_j is either $\bot$ or of the form $q^{\text{fact}}_{R,\langle v_1,\ldots,v_n\rangle,j',j}$ with $v_{j'} = v$,
- $\langle q_1,\ldots,q_n,\bot,\ldots,\bot\rangle \xRightarrow{\langle R,dir_1,\ldots,dir_n\rangle} q^{\text{root-fact}}_{R,\langle v_1,\ldots,v_n\rangle}$ where (i) for each $j \le n$, if $dir_j = \text{down}$ then $q_j = q^{\text{exvar}}_{v_j}$, and (ii) $\langle R, dir_1,\ldots,dir_n\rangle$ is "fulfilled" by $\langle v_1,\ldots,v_n\rangle$ at $\ell = -1$,
- $\langle q_1,\ldots,q_n,\bot,\ldots,\bot\rangle \xRightarrow{\langle R,dir_1,\ldots,dir_n\rangle} q^{\text{fact}}_{R,\langle v_1,\ldots,v_n\rangle,j,\ell}$ where (i) $dir_j = \text{up}$, (ii) for each $j' \le n$, if $dir_{j'} = \text{down}$ then $q_{j'} = q^{\text{exvar}}_{v_{j'}}$, and (iii) $\langle R, dir_1,\ldots,dir_n\rangle$ is "fulfilled" by $\langle v_1,\ldots,v_n\rangle$ at ℓ.

The above definition of the transitions refer to the notion of "being fulfilled", which we define now. This definition naturally reflects the frontier construction in Def. B.14. A node label $\sigma = \langle R, dir_1,\ldots,dir_n\rangle \in \Sigma$ is "fulfilled" by n-tuple of vectors $(v_1,\ldots,v_n)$ at ℓ, if every "acceptable instance" of π relative to $(v_1,\ldots,v_n)$ and ℓ is a tuple in R. Here, an *acceptable instance* of σ relative to $(v_1,\ldots,v_n)$ and ℓ is a tuple $(y_1,\ldots,y_n)$ where

- for each $i \le n$, either $dir_i \in \{\text{down}, \text{up}\}$ and $y_i \in v_i[1\ldots d+1]$, or $dir_i = \text{ans}_j$ and $y_i \in \{a_j, v_i[d+1+j]\}$, and
- for some $i \le n$, either (i) $dir_i = \text{down}$ and $y_i \in v_i[2,\ldots,d+1]$, or (ii) $dir_i = \text{up}$ and $y_i = v_i[m+1]$ for some $m \in \{1,\ldots,d+1\}$ with $m \neq \ell+1$ or (iii) $dir_i = \text{ans}_j$ and $y_i = v_i[d+1+j]$.

Note that, the first entry in the vector corresponds to the replica $u_{(x,f)}$ where f is the parent fact of x, while the $i+1$-th entry in the vector (for $i \le d$) corresponds to the replica $u_{(x,f)}$ where f is the i-th child fact of x.

If the automaton has an accepting run on input T, then, it already follows from the above root transition and the proof of Lemma B.12, that every fg-connected component of q_T other than the i-th one, admits a homomorphism to e. It should also be clear from the above discussion that if we denote the i-th fg-connected component by q', then $F(q')$ has a homomorphism to e. The converse holds as well, and therefore the automaton accepts T if and only if $F^i(q_T)$ admits a homomorphism to e. □

By taking the intersection of the NTA $\mathfrak{A}_E$ from Thm. B.13 with the NTAs $\mathfrak{A}^{frontier}_{E,i}$ (for $i = 1,\ldots,d$) from Lemma B.17, we get:

Theorem B.18. *Given a schema, $k \ge 0$, $d > 0$, and a collection of labeled examples E, we can construct in exponential time an NTA $\mathfrak{A}$ such that $L(\mathfrak{A})$ consists of proper d-ary Σ-trees T for which it holds that q_T is weakly most-general fitting for E. In particular, $L(\mathfrak{A})$ is non-empty iff there is a CQs with UNP that is weakly most-general fitting for E and that is encodable by a d-ary Σ-tree.*

Since Prop. B.10 allows us to polynomially bound d, and the core of a weakly most-general fitting CQ is c-acyclic, we get

Corollary B.19. *Given a collection of labeled examples E, we can decide in ExpTime the existence of a CQ with UNP that is weakly most-general fitting for E, and we can produce in ExpTime a succinct DAG-representation of a minimal-size such CQ if it exists.*

Step 4: Lifting the restriction to UNP

In the above, for simplicity we restricted attention to CQs with the UNP. However, the same techniques applies in the general case. We will restrict ourselves here to giving a high-level explanation of the

changes required. As a concrete example, let us consider the 3-ary CQ

$$q(x_1, x_2, x_3) \text{ :- } R(x_1, x_2, x_3), P_1(x_1), P_2(x_2), P_3(x_3), (x_1 = x_2)$$

Note that we use here equalities in the body of the CQ, but the same query could be expressed equivalently using repeated occurrences of variables in the head.

By an *equality-type* we will mean an equivalence relation over the set of answer variables $\{x_1, x_2, x_3\}$. The equality type $\equiv_q$ of the above query q is the equivalence relation that identifies x_1, x_2 with each other but not with x_3.

A frontier for such a query can be obtained by taking the set F of all instances that can be obtained in one of the following two ways:

(1) Compute a CQ q' of lower arity by replacing every equivalence class of answer variable by a single representative of that class. By construction, q' has the UNP. We take all queries belonging to the frontier of q', and, finally, add equalities to obtain queries of the original arity. Specifically, in the case of our example query q above, q' is the 2-ary query $q'(x_1, x_3)$:- $R(x_1, x_1, x_3), P_1(x_1), P_2(x_1), P_3(x_3)$. We then take each CQ belonging to the frontier of q', and extend it with a conjunct $x_2 = x_1$ to make that CQ ternary again, and add the result to F.
(2) Let $\equiv$ be the equality type of the query at hand. A *minimal weakening* of $\equiv$ is an equivalence relation in which some equivalence class is divided in two. In our example, the only minimal weakening of $\equiv_q$ is the equality type $\equiv'$ that does not identify any answer variables with each other. For each such weakening (i.e., in the case of our example, $\equiv'$), we then construct another CQ q' where we drop from q all equalities and replace them with the equalities $(x_i = x_j)$ for $(x_i, x_j) \in \equiv'$. We add q' to our set F. In our specific example, q' ends up being identical to q except without the $x_1 = x_2$ conjunct.

It follows from results in [11] that F, thus constructed, is a frontier for q.

All of the above can be implemented by a tree automaton. First of all, this requires a minor modification to the tree representation of c-acyclic CQs: we will store the equality type of the query in the root label of the tree. Next, with a non-deterministic root transitions, we guess whether item 1 or 2 as described above, applies. In first case, we reuse (with minimal modification) the automata we constructed earlier. In the second case, we guess a minimally weaker equality type. It is not hard to write an automaton that accepts a tree-encoding of a c-acyclic CQ if and only its corresponding minimal weakening fits the labeled examples. We omit the details here.

B.2 Proof of Thm. 3.16 (Relativized homomorphism dualities)

This section is dedicated to the proof of:

Theorem 3.16.

(1) The following is NP-complete: given a finite set of data examples D and a data example p, is there a finite set of data examples F such that (F, D) is a homomorphism duality relative to p?

(2) Given a finite set of data examples D and a data example p, if there is a finite set of data examples F such that (F, D) is a homomorphism duality relative to p, then we can compute in 2ExpTime such a set F, where each $e \in F$ is of size $2^{O(||D||^2 \cdot \log ||D|| \cdot |p|)}$.

We proceed in two steps: we first reduce to the case where D is a single example, and then we provide a characterization for this case, which leads to an criterion that can be evaluated in NP.

Reduction to the case where D is a single example

It will be convenient to regard equivalence relations α on some set A, as subsets of A^2. In this way we can write $\alpha \subseteq \beta$ to indicate that α refines β. We shall use $\wedge$ and $\vee$ to indicate the meet and join of equivalence relations.

Let be $(I, \mathbf{a})$ be a pointed instance and let α be an equivalence relation on $\text{adom}(I)$. For every array $\mathbf{b}$ containing values from $\text{adom}(I)$ we define $\mathbf{b}_\alpha$ to be the tuple obtained by replacing every value in $\mathbf{b}$ by its α-class. Also, we define $(I, \mathbf{a})_\alpha$ to be $(I_\alpha, \mathbf{a}_\alpha)$ where the facts of I_α are $\{R(\mathbf{b}_\alpha) \mid R(\mathbf{b}) \in I\}$.

For this part of the proof it will be convenient to generalize the disjoint union of a pair of pointed instances $q_1 = (I, \mathbf{a}_1)$ and $q_2 = (I, \mathbf{a}_2)$ which, so far, has only been defined under the assumption that q_1 and q_2 have the UNP. In its full generality we define $q_1 \uplus q_2$ as $(I_1 \cup I_2, \mathbf{a}_1)_\alpha$ where α is the more refined equivalence such that $\mathbf{a}_1[i]$ and $\mathbf{a}_2[i]$ are α-related for every $i \in [k]$. Note that $(I_1, \mathbf{a}_1) \uplus (I_2, \mathbf{a}_2)$ is defined so that for every pointed instance $(I, \mathbf{a})$ the following is equivalent:

(1) $(I_i, \mathbf{a}_i) \to (I, \mathbf{a})$ for $i = 1, 2$
(2) $(I_1, \mathbf{a}_1) \uplus (I_2, \mathbf{a}_2) \to (I, \mathbf{a})$.

We say that $(I, \mathbf{a})$ is *fg-connected* if it cannot be expressed as the disjoint union $(I_1, \mathbf{a}_1) \uplus (I_2, \mathbf{a}_1)$ where both I_1 and I_2 are non-empty.[5]

We say that an example $e \in D$ is *strictly subsumed* relative to D and p if, for some $e' \in D$, $p \times e \to e'$ and $p \times e' \not\to e$. We say that an example $e \in D$ is *non-subsumed* if it is not strictly subsumed by any $e' \in D$. We note that a pair (F, D) is a homomorphism duality relative to p if and only if (F, D') is a homomorphism duality relative to p, where D' is the set of non-subsumed examples in D.

The next lemma reduces our problem to the case where D contains a single example.

Lemma B.20. *For every finite set of data examples D and data example p, the following are equivalent:*

(1) There exists a finite set F such that (F, D) is a generalized duality relative to p.
(2) For every non-subsumed $e \in D$ there exists a finite set F_e such that $(F_e, \{e\})$ is a generalized duality relative to p.

Proof. Let $D' = \{e_1, \ldots, e_n\}$ be the set of non-subsumed examples in D.

(2) $\Rightarrow$ (1) It can easily verified that (F, D') (and hence (F, D)) is a generalized duality relative to p when $F = \{q_1 \uplus \cdots \uplus q_n \mid q_i \in F_{e_i}, i \in [n]\}$.

(1) $\Rightarrow$ (2) Let $e_i \in D'$. For every pointed instance $x = (X, \mathbf{x})$ with k distinguished elements we shall use $\gamma(x)$ to denote the

[5] This terminology stems from the fact that it corresponds to connectedness of the "fact graph", i.e., the graph consisting of the facts, with an edge between two facts if they share a non-designated element.

equivalence relation on $[k]$ where $i, j \in [k]$ are related if $\mathbf{x}[i] = \mathbf{x}[j]$.

Let $\delta = \gamma(e) \wedge \gamma(p)$. For every pointed instance $x = (X, \mathbf{x})$ satisfying $\delta \subseteq \gamma(x)$ we shall use x' to denote the pointed instance $x' = (X, \mathbf{x}')$ where $\mathbf{x}'$ is obtained from $\mathbf{x}$ by removing elements that belong to the same δ-class. Formally, fix a representative $i_1, \ldots, i_r$ for each one of the δ-classes and define $\mathbf{x}'[j]$ to be $\mathbf{x}[i_j]$.

Let m be the maximum domain size of any pointed instance in F and let F'_i be the set of all pointed instances with at most m values that are not homomorphic to e'_i. We shall show that $(F'_i, \{e'_i\})$ is a generalized duality relative to p'. This completes the proof of (1) $\Rightarrow$ (2) as it is easily verified that e_i and p satisfy Lemma B.21(1) iff e'_i and p satisfy it as well.

Let us then show that $(F'_i, \{e'_i\})$ is a generalized duality relative to p'. Let y be satisfying $y \nrightarrow e'_i$ and $y \rightarrow p'$. We note that $y = x'$ for some $x = (X, \mathbf{x})$ satisfying $x \nrightarrow e_i$, $x \rightarrow p$, and $\delta \subseteq \gamma(x)$. We can assume that $\gamma(x) \subseteq \gamma(e_i)$ since otherwise, $(\emptyset, \mathbf{x}')$ (which belongs to F'_{e_i}) is homomorphic to x' and we are done.

We note that $\delta = \gamma(p \times e_i)$. Also, since $\gamma(x) \subseteq \gamma(e_i)$ and $\gamma(x) \subseteq \gamma(p)$ it follows that $\gamma(x) \subseteq \delta$. Hence, we have $\gamma(x) = \delta$.

Consider $x \uplus (p \times e_i)$. We note that in general we cannot assume that the disjoint union $y_1 \uplus y_2$ of two pointed instances contains y_1 (or y_2) as a subinstance since some values can be identified while computing the disjoint union. However, since $\gamma(x) = \delta = \gamma(p \times e_i)$ it follows that $x \uplus (p \times e_i)$ contains x as a subinstance. This fact will be necessary later in the proof.

We claim that $x \uplus (p \times e_i) \nrightarrow e_j$ for every $j \in [n]$. The case $j = i$ follows from $x \nrightarrow e_i$. The case $j \neq i$ follows from the fact that e_i is non-subsumed. Also, since $x \rightarrow p$ and $(p \times e_i) \rightarrow p$ we have that $x \uplus (p \times e_i) \rightarrow p$. By (1) it follows that there exists q satisfying $q \rightarrow x \uplus (p \times e_i)$ and $q \nrightarrow e_j$ for every $j \in [n]$.

Let $h : q \rightarrow x \uplus (p \times e_i)$. We can assume that h is injective since otherwise we could replace q by q_α where α is the equivalence relation that partitions $\mathrm{adom}(q)$ according to the image of h. This implies, in particular, that $\gamma(q) = \delta$. To simplify notation we will assume further that q is a subinstance of $x \uplus (p \times e_i)$, i.e, q has been just obtained by (possibly) removing facts from $x \uplus (p \times e_i)$.

Let $q_j, j \in J$ be the fg-connected components of q and let $t = \biguplus_{j \in J'} q_j$ where $j \in J'$ if $q_j \nrightarrow e_i$. We note that by definition $t \nrightarrow e_i$.

For every $j \in J'$, $q_j \nrightarrow (p \times e_i)$ and, since q_j is fg-connected, we can conclude that q_j is a subinstance of x (here we are using implicitly $x \uplus (p \times e_i)$ contains x as a subinstance). Hence, $t \rightarrow x$.

Finally, we have $\delta \subseteq \gamma(t)$. Consequently $t' \rightarrow x'$ and $t' \in F'_e$, completing the proof. □

The case where D contains a single example

We shall now deal with the case where D contains a single example. To this end, we adapt the techniques introduced in [8] to a broader setting, since the setup in [8] did not consider distinguished elements and, more importantly, did not include the relativized version considered here. In addition, the proof given here is more streamlined as, unlike in [8], it does not go via mixing properties.

We shall need to introduce a few extra definitions. Let A be an instance. We define a *walk* in A as a walk in its incidence graph that starts and finishes at values from $\mathrm{adom}(A)$. That is, a walk ρ in A is a sequence

$$a_0, R_1(\mathbf{a}_1), a_1, \ldots, a_{n-1}, R_n(\mathbf{a}_n), a_n$$

for some $n \geq 0$, such that, for all $1 \leq \ell \leq n$,

- $R_\ell(\mathbf{a}_\ell)$ is a fact of A, and
- $a_{\ell-1}, a_\ell \in \{\mathbf{a}_\ell\}$.

In this case, we will say that a_0 and a_n are the starting and ending point of ρ, and that the *length* of the walk ρ is n. The *distance* between two values is defined to be the smallest length among all the walks that join them. The *diameter* of a connected instance is the maximum distance of any pair of its values, while the diameter of an instance with multiple connected components is the maximum of the diameters of its components.

Let a, b be values from $\mathrm{adom}(A)$. We say that b *dominates* a (in A) if for every fact $R(a_1, \ldots, a_r)$ in A and for every $i \in [r]$ with $a_i = a$, we also have that the fact $R(a_1, \ldots, a_{i-1}, b, a_{i+1}, \ldots, a_r)$ belongs to A. Additionally, if $a \neq b$, then the instance A' obtained from A by removing a and all the facts in which a participates is said to be obtained from A *by folding* a.

A sequence of instances $A_0, \ldots, A_\ell$ is a *dismantling sequence* if for every $0 \leq j < \ell$, A_{j+1} has been obtained from A_j by folding some value a_j dominated in A_j. In this case, we say that A_0 *dismantles to* A_ℓ.

In what follows, let $(P, \mathbf{p})$, $(E, \mathbf{e})$ be pointed instances over a common schema σ. Consider the new schema $\sigma \subseteq \overline{\sigma}$ containing, in addition, a new unary relation symbol R_p for each $p \in \mathrm{adom}(P)$ and consider the instances $\overline{P}$ and $\overline{E}$ over schema $\overline{\sigma}$ defined as $\overline{P} = P \cup \{R_p(p) \mid p \in \mathrm{adom}(P)\}$ and $\overline{E} = E \cup \{R_p(e) \mid p \in \mathrm{adom}(P), e \in \mathrm{adom}(E)\}$.

Let $u = \langle (p_1, d_1), (p_2, d_2) \rangle \in \mathrm{adom}(P \times E)^2$. We shall use $\pi_i u$ to denote the ith projection (p_i, d_i), $i = 1, 2$ of u. We say that u is *P-diagonal* if $p_1 = p_2$. If additionally, $a_1 = a_2$ then we say that u is *diagonal*. The *symmetric pair* of u is the value $\langle (p_2, d_2), (p_1, d_1) \rangle$.

For every subinstance $\bar{I}$ of $(\overline{P} \times \overline{E})^2$ we shall use $\mathrm{diag}_P(\bar{I})$ (resp. $\mathrm{diag}(\bar{I})$) to denote the subinstance of $\bar{I}$ induced by its P-diagonal (resp. diagonal) values.

An *endomorphism* is a homomorphism from a instance into itself. A *retraction* is an endomorphism h with the property that $h(x) = x$ for every x in the range of h.

We will say that a pointed instance $(A, \mathbf{a})$ is a *critical obstruction of* $(E, \mathbf{e})$ *relative to* $(P, \mathbf{p})$, if $(A, \mathbf{a}) \rightarrow (P, \mathbf{p})$, $(A, \mathbf{a}) \nrightarrow (E, \mathbf{e})$, and $(A' \mathbf{a}) \rightarrow (E, \mathbf{e})$ for any $A' \subsetneq A$. It is easy to see that critical obstructions are always fg-connected.

Lemma B.21. *Let $(P, \mathbf{p})$ and $(E, \mathbf{e})$ be pointed instances. The following are equivalent:*

1. *There exists a retraction from $(\overline{P}, \mathbf{p}) \times (\overline{E}, \mathbf{e})$ to some subinstance $(\bar{I}, \mathbf{i})$ such that $\mathrm{diag}_P(\bar{I}^2)$ dismantles to $\mathrm{diag}(\bar{I}^2)$.*
2. *Every critical obstruction $(A, \mathbf{a})$ for $(E, \mathbf{e})$ relative to $(P, \mathbf{p})$ has diameter at most $m = |\mathrm{adom}(P)| \cdot |\mathrm{adom}(E)|^2 + 2$.*
3. *There are finitely many (modulo isomorphism) critical obstructions of $(E, \mathbf{e})$ relative to $(P, \mathbf{p})$.*
4. *There exists F such that $(F, \{(E, \mathbf{e})\})$ is a generalized duality relative to $(P, \mathbf{p})$.*

Proof. (3) $\Leftrightarrow$ (4): The equivalence (3) $\Leftrightarrow$ (4) is immediate. Indeed, in the direction from (3) to (4), we can set F to be the set

of critical obstructions, while in the direction from (4) to (3), it is easy to see that the size of each critical obstruction is bounded by the size of the instances in F.

(2) $\Rightarrow$ (3). This proof is an adaptation of [35]. Let $(A, \mathbf{a})$ be a critical obstruction of $(E, \mathbf{e})$ relative to $(P, \mathbf{p})$. We shall use the sparse incomparability lemma (SIL) [41] (see also Theorem 5 in [22]). However, since SIL was originally only proved for instances without constants we need to do some adjustments. By the *girth* of an instance A, we will mean the length of the shortest cycle in the incidence graph of A, where the length is measured by the number of facts the lie on the cycle. If the incidence graph is acyclic, the instance is said to have girth ∞. Let $\mathbf{a} = (a_1, \ldots, a_k)$ and associate to $(A, \mathbf{a})$ the instance A' obtained from A by adding facts $R_i(a_i)$ where $R_i, i \in [k]$ are new relation symbols. We define similarly E' and P'. Note that $A' \to P'$ and $A' \not\to E'$. Then, according to SIL there is an instance B' with girth greater than m satisfying $B' \to A'$ and $B' \not\to E'$.

Consider the pointed instance $(B, \mathbf{b})$ obtained from B' in the following way. For every $i \in [k]$ we remove all facts with relation symbol R_i and glue all the values occurring in them into a single value, which then we place in the ith coordinate of $\mathbf{b}$. Clearly, we have $(B, \mathbf{b}) \to (A, \mathbf{a})$ (and, hence, $(B, \mathbf{b}) \to (P, \mathbf{p})$) and $(B, \mathbf{b}) \not\to (E, \mathbf{e})$. Further, remove facts and values from $(B, \mathbf{b})$ until becomes a critical obstruction of $(E, \mathbf{e})$ relative to $(P, \mathbf{p})$. Note that by assumption B has diameter at most m.

Now, consider the subinstance, C, of B induced by its non-distinguished elements. By the minimality of B, C must be connected. Note that C is a subinstance of B' as well and, hence, it has girth larger than m. Since B has diameter at most m it follows that C is a tree (i.e., an acyclic connected instance). We claim that every value c of C appears in at most n_E facts where n_E is the number of values in E. Indeed, let $f_1, \ldots, f_r$ be the facts in which c participates. For every $I \subseteq [r]$, let C_I be the maximal subinstance of C containing all the facts $f_i, i \in I$ and none of the facts $f_i, i \in [r] \setminus I$, let B_I be the subinstance of B induced by $C_I \cup \{\mathbf{b}\}$ and let

$$S_I = \{h(c) \mid h \text{ is an homomorphism from } (B_I, \mathbf{b}) \text{ to } (E, \mathbf{e})\}$$

We have that $S_{[1]}, S_{[2]}, \ldots, S_{[k]}$ are all different since, otherwise, say $S_{[i-1]} = S_{[i]}$, then $S_{[k]\setminus i} = \emptyset$ contradicting the minimality of $(B, \mathbf{b})$.

Since both the diameter and the branching of C are bounded it follows that there is a bound (depending only on m and e) on the number of values of C (and, hence, of B). Indeed, $|\mathrm{adom}(B)| \leq 2^{O(|\mathrm{adom}(P)|\cdot|\mathrm{adom}(E)|^2\cdot\log(|\mathrm{adom}(E)|))}$. Since $(A, \mathbf{a})$ is critical and admits an homomorphism from $(B, \mathbf{b})$ it follows that the size of the domain of A is not larger than the size of the domain of B.

(1) $\Rightarrow$ (2) Assume that (1) holds. Let $\overline{J} = \mathrm{diag}_P(\overline{I}^2)$ and let $\overline{J}_0, \overline{J}_1, \ldots, \overline{J}_n$ be a dismantling sequence where $\overline{J}_0 = \overline{J}$ and $\overline{J}_n = \mathrm{diag}(\overline{J})$. For every $i \in [n]$ there is a natural retraction $s_i : \overline{J}_{i-1} \to \overline{J}_i$ that sends the folded value $a \in \mathrm{adom}(\overline{J}_{i-1}) \setminus \mathrm{adom}(\overline{J}_i)$ to a value in $\mathrm{adom}(\overline{J}_i)$ that dominates it.

We shall use $I, J, J_0, \ldots, J_n$ to denote be the instance on schema σ obtained by removing all facts with relation symbol R_p, $p \in P$ from $\overline{I}, \overline{J}, \overline{J}_0, \ldots, \overline{J}_n$ respectively.

We claim that for every critical obstruction $(A, \mathbf{a})$ for $(E, \mathbf{e})$ relative to $(P, \mathbf{p})$ the diameter of A is bounded above by $m = n + 2 \leq |\mathrm{adom}(P)| \cdot |\mathrm{adom}(E)|^2 + 2$.

Towards a contradiction, let $(A, \mathbf{a})$ contradicting the claim. Since the diameter of A is larger than $2n+2$ it follows that there exists two facts $f_1 = R_1(\mathbf{a}_1), f_2 = R(\mathbf{a}_2)$ in A such that $N_n(\{\mathbf{a}_1\}) \cap N_n(\{\mathbf{a}_2\}) = \emptyset$ where for every $X \subseteq \mathrm{adom}(A)$ and $i \geq 0$, $N_i(X)$ denotes the set of all values from A that are at distance at most i from some value in X.

Let A_i, $i = 1, 2$ be the instance obtained removing fact f_i from A. It follows that there are homomorphisms $g_i : (A_i, \mathbf{a}) \to (E, \mathbf{e})$, $i = 1, 2$. Let $v : (A, \mathbf{a}) \to (P, \mathbf{p})$. Hence, for every $i = 1, 2$, mapping $a \mapsto (v(a), g_i(a))$ defines an homomorphism from A_i to $P \times E$. If we let $h_i = u \cdot (v, g_i)$ where $u : (\overline{P}, \mathbf{p}) \times (\overline{E}, \mathbf{e}) \to (\overline{I}, \mathbf{i})$ is the retraction guaranteed to exist from (1), we have $h_i : A_i \to I$.

Let B be the subinstance of A obtained by removing both f_1 and f_2 and let $h = (h_1, h_2) : B \to I^2$. Also, note that due to the facts with relation symbol $R_p, p \in \mathrm{adom}(P)$ added in $\overline{P} \times \overline{E}$ it follows that the image of h is necessarily in $\mathrm{adom}(J)$, and, hence $h : B \to J$. Since $g_1(\mathbf{a}) = g_2(\mathbf{a}) = \mathbf{e}$ it follows that $h(\mathbf{a})$ contains only diagonal values.

Let $r_0, \ldots, r_n : \mathrm{adom}(B) \to \mathrm{adom}(J)$ be the sequence of mappings where $r_0 = h$ and $r_i(b)$, $b \in \mathrm{adom}(B), i \in [n]$, is defined as follows:

$$r_i(b) = \begin{cases} r_{i-1}(b) & \text{if } b \in N_i(\{\mathbf{a}_1\}) \cup N_i(\{\mathbf{a}_2\}) \\ s_i \cdot r_{i-1}(b) & \text{otherwise} \end{cases}$$

It follows that $r_i : B \to J$ for every $i \leq n$. Further, since $N_n(\{\mathbf{a}_1\}) \cap N_n(\{\mathbf{a}_2\}) = \emptyset$ it follows that there is no fact in B that contains at the same time at least one value from $N_{n-1}(\{\mathbf{a}_1\})$ and at least one value from $N_{n-1}(\{\mathbf{a}_2\})$. In addition $r_n(b)$ is a diagonal value whenever $b \notin N_n(\{\mathbf{a}_1\}) \cup N_n(\{\mathbf{a}_2\})$. Consequently the mapping z, defined as

$$z(b) = \begin{cases} \pi_1 \cdot r_n(b) & \text{if } b \in N_{n-1}(\{\mathbf{a}_1\}) \\ \pi_2 \cdot r_n(b) & \text{otherwise} \end{cases}$$

is an homomorphism from B to $P \times E$. Note that r_i, $i \leq n$ agrees with h on $\{\mathbf{a}_1\} \cup \{\mathbf{a}_2\}$. This implies that z agrees with h_i on $\{\mathbf{a}_i\}$ for $i = 1, 2$. In consequence, mapping z preserves facts f_1 and f_2 as well. Further, since $h(\mathbf{a})$ contains only diagonal values it follows that r_n agrees with h on $\mathbf{a}$, and, hence $z(\mathbf{a}) = \mathbf{e}$. Putting all together we have $z : (A, \mathbf{a}) \to (E, \mathbf{b})$, contradicting our assumptions on $(A, \mathbf{a})$.

(3) $\Rightarrow$ (1) Assume that (1) does not hold. Let $(\overline{I}, \mathbf{i})$ be the core of $(\overline{P}, \mathbf{p}) \times (\overline{E}, \mathbf{e})$ and let $\overline{J} = \mathrm{diag}_P(\overline{I}^2)$. We define $\overline{K}$ be any instance obtained from a distmantling sequence starting at $\overline{J}$ until no further distmantling if possible. Since $\overline{J}$ does not distmantle to its diagonal it adom($\overline{K}$) contains some non-diagonal value. It is easy to see (see [8], Remark 3.8) that such distmantling sequence can be done by folding symmetric pairs so that $\overline{K}$ contains the symmetric pair of any of its values.

Define I, J and K to be obtained by removing all the facts with relation symbol $R_p, p \in \mathrm{adom}(P)$ from $\overline{I}, \overline{J}$, and $\overline{K}$ respectively. We shall show that for every $m \geq 0$ there is a critical obstruction of $(E, \mathbf{e})$ relative to $(P, \mathbf{p})$ with at least m values.

Let k_0 be a non-diagonal value in K. We construct an instance T (without distinguished elements) in the following way. The domain of T consists of all the walks in K of length $m + 1$ starting

at k_0. For every fact $f = R(k_1, \ldots, k_r)$ in K, for every $i \in [r]$, and for every walk ρ ending at k_i, we include in T the fact $R(\rho_1, \ldots, \rho_{i-1}, \rho, \rho_{i+1}, \ldots, \rho_r)$, where ρ_j, $j \neq i$, is the walk obtained from ρ by extending it with f, k_j. The *root* of T is the walk ρ_0 of length 0 consisting only of k_0 and the *leafs* of T are the walks of length $m+1$.

It is immediate that the map u sending every walk ρ to its last node is an homomorphism from T to $(P \times E)$. Also let $v : J \to P$ be the mapping sending value $\langle (p, a_1), (p, a_2) \rangle$ to p.

Claim 1. *Let $g : T \to K$ be any homomorphism that agrees with u on the leaves of T. Further, assume that $v \circ g = v \circ u$. Then we have that $g = u$.*

Proof of claim. This follows by induction by showing that for every $\ell \in [m+1]$, if g agrees with u for every walk of length ℓ it also agrees for every walk of length $\ell - 1$. In particular, let $\rho \in T$ of length $\ell - 1$. It follows from the definition of T and the inductive hypothesis that $g(\rho)$ dominates $u(\rho)$ in K. Additionally, since $v \circ g(\rho) = v \circ u(\rho)$ it follows that $g(\rho)$ dominates $u(\rho)$ also in $\overline{K}$. Since $\overline{K}$ cannot be further dismantled it follows that $g(\rho) = u(\rho)$ as desired. □

Let $(A_i, \mathbf{i}_\alpha)$, $i = 1, 2$ be the pointed instances $(T \cup I, \mathbf{i})_\alpha$ where α can be informally described as gluing every leaf $\rho \in \text{adom}(T)$ with value $\pi_i \circ u(\rho)$ in $\text{adom}(I)$. We also glue the root ρ_0 to some leaf ρ' such that $u(\rho_0)$ and $u(\rho')$ are symmetric (here we use the fact that $\text{adom}(\overline{K})$ is closed under symmetric pairs).

We note that all the values in $\text{adom}(I)$ have been glued to some value in T. This allows to simplify a bit our notation as we can naturally extend u to $\text{adom}(A_1)$ (and $\text{adom}(A_2)$) by defining $u(a)$ as $u(\rho)$ for any walk ρ contained in α-class a.

Claim 2. *$(A_1, \mathbf{i}_\alpha) \nrightarrow (E, \mathbf{e})$ or $(A_2, \mathbf{i}_\alpha) \nrightarrow (E, \mathbf{e})$.*

Proof of claim. Assume towards a contradiction that there are homomorphisms $f_i : (A_i, \mathbf{i}_\alpha) \to (E, \mathbf{e})$ for $i = 1, 2$. Then, mapping $g_1(a) = (v \cdot u(a), f_1(a))$ defines an homomorphism from $(A_1, \mathbf{i}_\alpha)$ to $(P, \mathbf{p}) \times (E, \mathbf{e})$. Let $z : (\overline{P}, \mathbf{p}) \times (\overline{E}, \mathbf{e}) \to (\overline{I}, \mathbf{i})$ be any retraction and let $h_1 = z^n \circ g_1$ for some $n \geq 1$ to be chosen later. Note that $h_1 : (A_1, \mathbf{i}_\alpha) \to (I, \mathbf{i})$. Note that map $i \mapsto i_\alpha$ defines an isomorphism from $(I, \mathbf{i})$ to $(A_1, \mathbf{i}_\alpha)$. We might abuse slightly notation and refer to the copy or $(I, \mathbf{i})$ in $(A_1, \mathbf{i}_\alpha)$ simply as $(I, \mathbf{i})$. Then, the restriction of h_1 to $\text{adom}(I)$ must be a bijection since otherwise this would contradict the fact that $(\overline{I}, \mathbf{i})$ is a core. Hence we can choose n so that h_1 acts as the identity on $(I, \mathbf{i})$. Note that $v \cdot h_1 = v \cdot u$

We can similarly show that there exists $h_2 : (A_2, \mathbf{i}_\alpha) \to (\overline{I}, \mathbf{i})$ that acts as the identity on $(\overline{I}, \mathbf{i})$ and $v \cdot h_2 = v \cdot u$. It follows that the mapping $h(\rho) = (h_1(\rho_\alpha), h_2(\rho_\alpha))$ defines an homomorphism from T to J satisfying $v \cdot h = v \cdot u$. Note also that h acts as the identity on the leaves of T, which implies that, indeed, $h : T \to K$. Hence Claim 1 implies that $h = u$.

Since α_0 and α' have been glued in both A_1 and A_2 it follows that h agrees on α_0 and α'. However this is impossible since we have chosen $u(\alpha_0)$ to be non-diagonal and $u(\alpha')$ and $u(\alpha_0)$ are symmetric pairs. □

Assume that $(A_1, \mathbf{i}_\alpha) \nrightarrow (E, \mathbf{e})$ (the case $(A_2, \mathbf{i}_\alpha) \nrightarrow (E, \mathbf{e})$ is analogous). Then there exists some $B \subseteq A_1$ such that $(B, \mathbf{i}_\alpha)$ is a critical obstruction for $(E, \mathbf{e})$ relative to $(P, \mathbf{p})$. To conclude our proof it is only necessary to note that B has at least m values as a consequence of the following claim.

Claim 3. *For every $\ell = 1, \ldots, m$, B contains at least one value of T of level ℓ.*

Proof of claim. Assume that some ℓ falsifies the claim. Then the mapping g defined as follows is an homomorphism from $(B, \mathbf{i}_\alpha)$ to $(E, \mathbf{e})$, a contradiction. Let $b \in B$ (recall that b is a α-class). If b contains some value (a, e) in I then define $g(b)$ to be e. If b contains some ρ in T we define $g(\rho)$ in the following way: Let i be the length of ρ and let $u(\rho) = \langle (a, e_1), (a, e_2) \rangle$ be its last value. Then $g(\rho)$ is defined to be e_2 if $i < \ell$ and e_1 if $i > \ell$. It is immediate to see that g is well defined and that defines an homomorphism. □

This concludes the main proof. □

Putting everything together

Proof of Thm. 3.16. (1) By combining Lemmas B.20 and B.21 we obtain an NP algorithm to decide whether there exists F such that (F, D) is a generalized duality relative to p: for every $e \in D$, we non-deterministically verify that there is some homomorphism $(p \times e) \to e'$ for some $e' \in D$ different than e or that condition (1) in Lemma B.21 is satisfied for e and p. This condition can be tested in NP by guessing the retraction.

The NP lower bound holds already in the non-relativized case without designated elements (i.e., where $k = 0$ and $(P, \mathbf{p})$ is the pointed instance containing all possible facts over a single-element domain) [35].

(2) By Lemma B.21, in the single-example case, when the set of critical obstructions is finite, then, in fact, each critical obstruction has domain size at most $2^{O(|\text{adom}(P)| \cdot |\text{adom}(E)|^2 \cdot \log(|\text{adom}(E)|))}$. Hence, F can be constructed to consist of instances of this size. In the general case with a set of examples D, inspection of the proof of Lemma B.20 shows that we take $F = \{e'_1 \uplus \cdots \uplus e'_n \mid e'_i \in F_{e_i}, i \in [n]\}$, where $\{e_1, \ldots, e_n\}$ be the set of all non-subsumed examples in D. It follows that each member of F has size at most $2^{O(|\text{adom}(P)| \cdot ||\text{adom}(D)||^2 \cdot \log(||\text{adom}(D)||))}$, as claimed. □

C DETAILED PROOFS FOR SECT. 4

Proposition 4.2. *(Implicit in [2].) For all collections of labeled examples $E = (E^+, E^-)$ and UCQs q, the following are equivalent:*

(1) q is a strongly most-specific fitting UCQ for E,
(2) q is a weakly most-specific fitting UCQ for E,
(3) q fits E and is homomorphically equivalent to $\bigcup_{e \in E^+} q_e$.

Proof. The implication from 1 to 2 is trivial.

For the implication from 2 to 3, suppose that q is a weakly most-specific fitting UCQ for E. Let $q' = \bigcup_{e \in E^+} q_e$. Since q fits E, we have $q \to q'$. Furthermore, q' fits E. Indeed, q' fits E^+ by construction, and if there was a homomorphism from a disjunct of q' to a negative example, then, since $q \to q'$, also q would fail to fit the same negative example. Thus, $q' \subseteq q$ and q fits E. Therefore, by the definition of "weakly most-specific", we have that $q \equiv q'$, which means that q and q' are homomorphically equivalent.

For the implication from 3 to 1, let $q' = \bigcup_i q'_i$ be any UCQ that fits E. We must show that $q \subseteq q'$. Consider any disjunct q_i of q.

Since q is homomorphically equivalent to $\bigcup_{e\in E^+} q_e$, we know that, for some $e \in E^+$, $q_e \to q_i$. Furthermore, since q' fits E^+, for some disjunct q'_j of q' we have $q'_j \to q_e$. Therefore, by composition, $q'_j \to q_i$, which means that $q_i \subseteq q'_j$. □

Proposition 4.3. *For all collections of labeled examples $E = (E^+, E^-)$ and UCQs $q = q_1 \cup \cdots \cup q_n$, the following are equivalent:*

(1) q is a strongly most-general fitting UCQ for E,
(2) q is a weakly most-general fitting UCQ for E,
(3) q fits E^+ and $(\{e_{q_1}, \ldots, e_{q_n}\}, E^-)$ is a homomorphism duality.

Proof. The implication from 1 to 2 is trivial.

For the implication from 2 to 3, assume that $q = \bigcup_i q_i$ is weakly most-general fitting for E, and let e be any data example. If $e_{q_i} \to e$, then $e \not\to E^-$, as otherwise, by transitivity, we would have that $e_{q_i} \to E^-$, which we know is not the case because q fits E. Conversely, if $e \not\to E^-$, then the UCQ $q' = q \cup q_e$ fits E. Since q is contained in q', it follows by the definition of "weakly most-general" that $q \equiv q'$, which means that q and q' are homomorphically equivalent. In particular, some q_i maps to q_e, and hence, $e_{q_i} \to e$.

Finally, for the implication from 3 to 1, suppose q' is a fitting UCQ for E. Consider any disjunct q'_i of q'. Then q'_i does not map to E^-. Hence, since $(\{e_{q_1}, \ldots, e_{q_n}\}, E^-)$ is a homomorphism duality, we have that some e_{q_j} maps to $e_{q'_i}$, and hence, $q_j \to q'_i$. This shows that q homomorphically maps to q'. □

Proposition 4.4. *For all collections of labeled examples $E = (E^+, E^-)$ and UCQs q, the following are equivalent:*

(1) q is a unique fitting UCQ for E,
(2) q fits E and the pair (E^+, E^-) is a homomorphism duality,
(3) q is homomorphically equivalent to $\bigcup_{e\in E^+} q_e$ and (E^+, E^-) is a homomorphism duality.

Proof. From 1 to 2, suppose q is a unique fitting UCQ for E. We must show that (E^+, E^-) is a homomorphism duality. Let e be any data example. If a positive example maps to e, then q maps to e. Since q does not map to any negative example, it follows that e does not map to any negative example either. If, on the other hand, no positive example maps to e, then e must map to a negative example. For, otherwise, $q' = q \cup q_e$ would be a fitting UCQ that is not homomorphically equivalent (and hence not logically equivalent) to q.

For the implication from 2 to 3, we must show that q is homomorphically equivalent to $q' = \bigcup_{e\in E^+} q_e$. The direction $q \to q'$ is immediate from the fact that q fits E^+. For the other direction, let q_i be any disjunct of q. Since q_i fits E^- and (E^+, E^-) is a homomorphism duality, we know that, for some positive example $e \in E^+$, $q_e \to q_i$. This shows that $q' \to q$.

For the implication from 3 to 1, it follows from Prop. 4.2 that q is a most-specific fitting UCQ for E, and from Prop. 4.3 that q is a most-general fitting UCQ for E. Hence, q is a unique fitting UCQ for E. □

Theorem 4.5.

(1) The existence problem for fitting UCQs (equivalently, for most-specific fitting UCQs) is coNP-complete; if a fitting UCQ exists, a most-specific fitting UCQ can be computed in PTime.
(2) The existence problem for most-general fitting UCQs is NP-complete; if a most-general fitting UCQ exists, one can be computed in 2ExpTime.
(3) The verification problem for fitting UCQs is DP-complete.
(4) The verification problem for most-specific fitting UCQs is DP-complete.

Proof. 1. By Prop. 4.2, it suffices to test that the UCQ $\bigcup_{e\in E^+} q_e$ fits. It fits the positive examples by definition. Therefore, it is enough to test the non-existence of a homomorphism to E^-, which can be done in coNP. The lower bound is by reduction from graph homomorphism: $G \to H$ holds if and only if there is no fitting UCQ for $(E^+ = \{G\}, E^- = \{H\})$.

2. The upper bound follows immediately from Prop. 4.3 together with Thm. 3.16 (one can choose p to be the single-element instance containing all possible facts). The NP-hardness follows directly from [35] (we can choose $E^+ = \emptyset$). Finally, it follows from Proposition 4.3 and Theorem 3.16(2) (again choosing p to be the single-element instance containing all possible facts) that the union of the canonical queries of all instances of size at most $2^{2^{O(poly(||E^-||))}}$ not homomorphic to any instance in E^- is a most-general fitting UCQ, provided it exists. Hence, such q can be constructed in 2ExpTime.

3. To test if q fits $E = (E^+, E^-)$, we test that (i) for each $e \in E^+$, some disjunct of q maps to it, and (ii) no disjunct of q maps to any $e \in E^-$. This clearly shows that the problem is in DP. For the lower bound, we reduce from exact 4-colorability [43]: a graph G is exact 4-colorable if and only if the canonical CQ of G fits $(E^+ = \{K_4\}, E^- = \{K_3\})$, where K_n is the n-element clique.

4. To verify that q is a most-specific fitting UCQ for $E = (E^+, E^-)$, by Prop. 4.2, it suffices to test that (i) q is homomorphically equivalent to $\bigcup_{e\in E^+} q_e$, and (ii) fits E^-. This clearly places the problem in DP. For the lowerbound, we reduce again from exact 4-colorability [43]: a graph G is exact 4-colorable if and only if the canonical CQ of G is a most-specific fitting UCQ for $(E^+ = \{K_4 \times G\}, E^- = \{K_3\})$. □

In order to prove the upper bound in Proposition 4.6 we shall need the algorithm given in the next lemma.

Lemma C.1. *There is polynomial time algorithm (arc consistency) that given instances e', e with possible distinguished elements as input determines whether it is true that for each c-acyclic t, $t \to e'$ implies $t \to e$.*

Proof. The constraint literature includes several sightly different algorithms under the name 'arc consistency' so we give a reference for the sake of concreteness [19]. It is well known that the arc-consistency algorithm, which has been defined only for instances without designated elements, verifies precisely the condition stated. To extend it to instances with distinguished elements, the only modification that is needed is to initialize the algorithm such that each designated element in e' is mapped to the corresponding designated element in e. □

Proposition 4.6. *HomDual is in ExpTime and NP-hard.*

Proof. We first prove the upper bound. We may assume that F consists of pairwise homomorphically incomparable instances: if not, then we can take a minimal subset $F' \subseteq F$ with the property

that for every $e \in F$, there is $e' \in F'$ such that $e' \to e$. Similarly, we can assume that D consists of pairwise homomorphic incomparable instances: again, we can take a minimal subset $D' \subseteq D$, with the property that for every $e \in D$, there is $e' \in D'$ such that $e \to e'$.

It is easy to see that (F, D) is a homomorphism duality if and only if (F', D') is. We may also assume that each $e \in F \cup D$ is a core.

We claim that, in order for (F, D) to be a homomorphism duality, each $e \in F$ must be c-acyclic. By Thm. 2.1, it suffices to show that, if (F, D) is a homomorphism duality, then each $e \in F$ has a frontier (since, core instances that have a frontier are c-acyclic). Indeed, if (F, D) is a homomorphism duality, then $Fr_e = \{e' \times e \mid e' \in D\}$ is a frontier for e: since $e \not\to e'$, we have that $e' \times e \to e$ and $e \not\to e \times e'$ (by Prop. A.1). Furthermore, suppose $e'' \to e$ and $e \not\to e''$. Since F consists of pairwise homomorphically incomparable data examples, it follows that there is no data example in F has a homomorphism to e''. Hence, $e'' \to e'$ for some $e' \in D$. Therefore, $e'' \to e' \times e$. The latter belongs to Fr_e by construction.

Next, we therefore test that F consists of c-acyclic instances. If this test succeeds, then, by Thm. 2.1, we can compute, for each $e \in F$, a finite set D_e such that $(\{e\}, D_e)$ is a homomorphism duality. Let $D' = \{e_1 \times \cdots \times e_n \mid (e_1, \ldots, e_n) \in \Pi_{e \in F} D_e\}$. It is straightforward to show that (F, D') is a homomorphism duality.

It follows that (F, D) is a homomorphism duality if and only D and D' are homomorphically equivalent, in the sense that (i) for each $e \in D$, there is $e' \in D'$ such that $e \to e'$, and (ii) vice versa: for each $e' \in D$, there is a $e \in D$, such that $e' \to e$.

Condition (i) can be tested in polynomial time since it is equivalent to the fact that $e \twoheadrightarrow e'$ for every $e \in F$ and $e' \in D$ (and e is guaranteed to be c-acyclic). For condition (ii) we first check whether there is some set F' of instances such that (F', D) is a homomorphism duality. It follows from Lemmas B.20 and B.21 that this can be done by verifying that every $e \in D$ satisfies condition (1) in Lemma B.21 (choosing P to be the instance with only one element and having all possible facts). This check can be done clearly in NP (and, indeed, in polynomial time if e is a core although this is not needed here).

Next, let $e' \in D'$ and $e \in D$. We shall show that $e' \to e$ is equivalent to the condition checked by the algorithm in Lemma C.1. We note that since D' has at most exponentially many instances and all of them have size bounded above exponentially then this implies that (ii) can be verified in ExpTime by an iterative application of arc-consistency.

Let us proof our claim. Let $e' \in D'$ and $e \in D$. If $e' \to e$, then, clearly (by composition of homomorphisms), $t \to e'$ implies $t \to e$, for all c-acyclic instances t. For the converse, assume that $t \to e'$ implies $t \to e$ for every c-acyclic instance t. Since e satisfies Lemma B.21(1) it follows that there exists some set T of instances such that $(T, \{e\})$ is a homomorphism duality. Moreover, by [1], there is such a set T consisting of c-acyclic instances. Then, for every $t \in T$, we have that $t \twoheadrightarrow e'$ since otherwise $t \to e$, which is impossible as $(T, \{e\})$ is a homomorphism duality. Again, using the fact that $(T, \{e\})$ is a homomorphism duality it follows that $e' \to e$.

For the lower bound, we use an argument that was also used in [35] to show that FO definability of a CSP is NP-hard: we reduce from 3-SAT. Fix a schema consisting of a single binary relation R. Let $F = \{P_{n+1}\}$ where P_{n+1} is the path of length $n+1$, and let $D = \{T_n\}$ where T_n is the transitive tournament (i.e., total linear order) of length n. It is well known that (F, D) is a homomorphism duality (this is known as the Gallai–Hasse–Roy–Vitaver theorem). Now, consider any 3-SAT input

$$\phi = \bigwedge_{i=1\ldots n} \bigvee_{j=1,2,3} L_{ij}$$

Let H be the instance with domain $\{1, \ldots, n\} \times \{1, 2, 3\}$, with an atom $R(\langle i, j\rangle, \langle i', j'\rangle)$ whenever $i < j$ and $L_{i'j'}$ is not the negation of L_{ij}. We claim that the following are equivalent:

(1) ϕ is satisfiable
(2) H is homomorphically equivalent to T_n
(3) $(F, \{H\})$ is a homomorphism duality

The equivalence of 2 and 3 is obvious. Therefore, it suffices only to show that 1 and 2 are equivalent. By construction, $H \to T_n$. From 1 to 2, a homomorphism from $T_n$to H may be constructed out of a satisfying assignment by mapping the i-th element of T_n to any true literal from the i-th clause of ϕ. Conversely, any homomorphism from T_n to H clearly induces a satisfying truth assignment for ϕ. □

Theorem 4.7. *The following problems are computationally equivalent (via polynomial conjunctive reductions) to* HomDual*:*

(1) The existence problem for unique fitting UCQs,
(2) The verification problem for unique fitting UCQs,
(3) The verification problem for most-general fitting UCQs.

Proof. Recall that a conjunctive reduction takes an instance of the first problem and produces one or more instances of the second problem, such that the input instance is a *Yes* instance for the first problem if and only if each output instance is a *Yes* instance of the second problem. It follows immediately from Prop. 4.4 and Prop. 4.3, together with the NP-hardness of HomDual that these problems polynomially conjunctively reduce to HomDual. The converse direction is immediate as well (where, for the verification problems, it suffices to choose $q = \bigcup_{e \in E^+} q_e$). □

D DETAILED PROOFS FOR SECT. 5

D.1 Preliminaries

We start with a basic lemma that pertains to simulations and unravelings.

Lemma D.1. *Let I, J be instances, $a \in adom(I)$, $b \in adom(J)$, and U the unraveling of I at a. Then $(I, a) \preceq (J, b)$ iff $(U, a) \preceq (J, b)$. Moreover, if I and J are finite, then this is the case iff $(U_m, a) \preceq (J, b)$ for all m-finite unravelings U_m of I at a.*

Proof. For the 'if' direction, assume that $(U, a) \preceq (J, b)$ is witnessed by simulation S. Then $S' = \{(a', b') \mid (p, b') \in S \text{ and } p \text{ ends in } a'\}$ is a simulation that witnesses $(I, a) \preceq (J, b)$. For the 'only if' direction, assume that $(I, a) \preceq (J, b)$ is witnessed by simulation S. Then $S' = \{(p, b') \mid (a', b') \in S \text{ and } p \text{ ends in } a'\}$ is a simulation that witnesses $(U, a) \preceq (J, b)$.

For the 'moreover' part, the 'only if' direction is clear. Thus assume that $(U_m, a) \preceq (J, b)$ for all m-finite unravelings U_m of I at a. Then also $U_m \to J$ for all $m \geq 1$ via homomorphism $h_1, h_2, \ldots$ with $h_i(a) = b$. It suffices to manipulate this sequence so that

$(*)$ $h_i(a) = h_j(a)$ whenever $h_i(a), h_j(a)$ are both defined,

as then $\bigcup_{i\geq 1} h_i$ is a homomorphism from the (unbounded) unraveling U of I at a to J. The first part of the lemma yields $(I, a) \preceq (J, b)$, as required.

To achieve $(*)$, we start with h_1 and observe that since U_1 and J are finite, there are only finitely many homomorphisms h from U_1 to J. Some such homomorphism must occur infinitely often in the restrictions of $h_1, h_2, \ldots$ to $\text{adom}(U_1)$ and thus we find a subsequence $h'_1, h'_{2+1}, \ldots$ of $h_1, h_2, \ldots$ such that the restriction of each h'_i to $\text{adom}(U_1)$ is identical. We may assume w.l.o.g. that each h'_i is a homomorphism from U_i to J and can thus replace $h_1, h_2, \ldots$ with $h'_1, h'_2, \ldots$. We proceed in the same way for the restrictions of the sequence $h_2, h_3, \ldots$ to $\text{adom}(U_2)$, then for the restrictions of the sequence $h_3, h_4, \ldots$ to $\text{adom}(U_3)$, and so on. In the limit, we obtain a sequence that satisfies $(*)$. □

D.2 Alternating Tree Automata

We introduce alternating tree automata, which are used in several of the subsequent upper bound proofs. A *tree* is a non-empty set $T \subseteq \mathbb{N}^*$ closed under prefixes. We say that T is *m-ary* if for every $x \in T$, the set $\{i \mid x \cdot i \in T\}$ is of cardinality at most m, and assume w.l.o.g. that all nodes in an m-ary tree are from the set $\{1, \ldots, m\}^*$. For an alphabet Γ, a Γ-*labeled tree* is a pair (T, L) with T a tree and $L : T \to \Gamma$ a node labeling function.

For any set X, let $\mathcal{B}^+(X)$ denote the set of all positive Boolean formulas over X, i.e., formulas built using conjunction and disjunction over the elements of X used as propositional variables, and where the special formulas true and false are admitted as well. An *infinite path* P in a tree T is a prefix-closed set $P \subseteq T$ such that for every $i \geq 0$, there is a unique $x \in P$ with $|x| = i$, where $|x|$ denotes the length of word x.

Definition D.2 (TWAPA). A *two-way alternating parity automaton (TWAPA) on finite m-ary trees* is a tuple $\mathfrak{A} = (S, \Gamma, \delta, s_0, c)$ where S is a finite set of *states*, Γ is a finite alphabet, $\delta : S \times \Gamma \to \mathcal{B}^+(\text{tran}(\mathfrak{A}))$ is the *transition function* with $\text{tran}(\mathfrak{A}) = \{\langle i\rangle s,\ [i]s \mid -1 \leq i \leq m \text{ and } s \in S\}$ the set of *transitions* of $\mathfrak{A}$, $s_0 \in S$ is the *initial state*, and $c : S \to \mathbb{N}$ is the *parity condition* that assigns to each state a *priority*.

Intuitively, a transition $\langle i\rangle s$ with $i > 0$ means that a copy of the automaton in state s is sent to the i-th successor of the current node, which is then required to exist. Similarly, $\langle 0\rangle s$ means that the automaton stays at the current node and switches to state s, and $\langle -1\rangle s$ indicates moving to the predecessor of the current node, which is then required to exist. Transitions $[i]s$ mean that a copy of the automaton in state s is sent to the relevant successor/predecessor if it exists, which is then not required.

Definition D.3 (Run, Acceptance). Let $\mathfrak{A} = (S, \Gamma, \delta, s_0, c)$ be a TWAPA on finite m-ary trees. A *run* of $\mathfrak{A}$ on a finite Γ-labeled m-ary tree (T, L) is a $T \times S$-labeled tree (T_r, r) such that the following conditions are satisfied:

(1) $r(\varepsilon) = (\varepsilon, s_0)$;
(2) if $y \in T_r$, $r(y) = (x, s)$, and $\delta(s, L(x)) = \varphi$, then there is a (possibly empty) set $S \subseteq \text{tran}(\mathfrak{A})$ such that S (viewed as a propositional valuation) satisfies φ as well as the following conditions:
   (a) if $\langle i\rangle s' \in S$, then $x \cdot i$ is defined and there is a node $y \cdot j \in T_r$ such that $r(y \cdot j) = (x \cdot i, s')$;
   (b) if $[i]s' \in S$ and $x \cdot i$ is defined and a node in T, then there is a node $y \cdot j \in T_r$ such that $r(y \cdot j) = (x \cdot i, s')$.

We say that (T_r, r) is *accepting* if on all infinite paths $\varepsilon = y_1 y_2 \cdots$ in T_r, the maximum priority that appears infinitely often is even. A finite Γ-labeled m-ary tree (T, L) is *accepted* by $\mathfrak{A}$ if there is an accepting run of $\mathfrak{A}$ on (T, L). We use $L(\mathfrak{A})$ to denote the set of all finite Γ-labeled m-ary trees accepted by $\mathfrak{A}$.

We remark that most of our automata use the acceptance condition in a trivial way, that is, every state has priority 0 and thus every run is accepting. In fact, we shall typically remain silent about the acceptance condition, then meaning that every state has priority 0. But we shall also apply complementation to TWAPAs, which gives rise to other types of acceptance. The following properties of TWAPAs are well-known [40, 45].

Theorem D.4.

(1) *Given a TWAPA $\mathfrak{A} = (S, \Gamma, \delta, s_0, c)$, it can be decided in time single exponential in $|S|$ and the maximum priority used by c, and polynomial in $|\mathfrak{A}|$, whether $L(\mathfrak{A})$ is empty.*
(2) *Given a TWAPA $\mathfrak{A} = (S, \Gamma, \delta, s_0, c)$, one can compute in polynomial time a TWAPA $\mathfrak{A} = (S, \Gamma, \delta', s_0, c')$ such that $L(\mathfrak{A}') = \overline{L(\mathfrak{A})}$.*
(3) *Given TWAPAs $\mathfrak{A}_i = (S_i, \Gamma, \delta_i, s_{0,i}, c_i)$, $i \in \{1, 2\}$, one can compute in polynomial time a TWAPA $\mathfrak{A} = (S_1 \uplus S_2 \uplus \{s_0\}, \Gamma, \Delta, s_0, c)$ with $L(\mathfrak{A}) = L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$.*
(4) *Given a TWAPA $\mathfrak{A} = (S, \Gamma, \delta, s_0, c)$, one can compute in single exponential time an NTA $\mathfrak{A}' = (S', \Gamma, \Delta, F)$ with $\mathfrak{A} = \mathfrak{A}'$.*
(5) *Given a TWAPA $\mathfrak{A} = (S, \Gamma, \delta, s_0, c)$ on m-ary trees with $L(\mathfrak{A}) \neq \emptyset$, we can compute in single exponential time a succinct representation (in the form of a directed acyclic graph) of a tree with a minimal number of nodes accepted by $\mathfrak{A}$. The number of nodes in that tree is at most $m^{2^{p(|S|)}}$, p a polynomial.*
(6) *Given a TWAPA $\mathfrak{A} = (S, \Gamma, \delta, s_0, c)$ on m-ary trees, it can be decided in time single exponential in $|S|$ and the maximum priority used by c, and polynomial in $|\mathfrak{A}|$, whether $L(\mathfrak{A})$ is infinite. Moreover, if $L(\mathfrak{A})$ is finite, then the size of every tree in it is at most The number of nodes in that tree is at most $m^{2^{p(|S|)}}$, p a polynomial.*

Note that Point (5) follows from Point (4) together with Point 2 of Theorem B.5. To obtain the decision procedure promised by Point (6), one would first convert the TWAPA $\mathfrak{A}$ to an equivalent NTA $\mathfrak{A}'$ via Point (4). Let the state set of $\mathfrak{A}'$ be Q and let m be the cardinality of Q. We then use the fact that $L(\mathfrak{A}')$ is infinite iff $L(\mathfrak{A}')$ contains a tree whose depth exceeds the number m of states of $\mathfrak{A}'$. More concretely, we convert $\mathfrak{A}'$ into an NTA $\mathfrak{A}''$ that accepts exactly the trees in $L(\mathfrak{A}')$ that are of depth exceeding m by using as the new states $\{0, \ldots, m\} \times Q$ and implementing a counter on the first component of the states and simulation $\mathfrak{A}'$ on the second component. It remains to decide the non-emptiness of $L(\mathfrak{A}'')$ via Point (1) of Theorem B.5.

D.3 Tree CQs as Γ-labeled trees

For the automata constructions in the subsequent sections, we wish to view tree CQs as Γ-labeled trees, for some suitable alphabet Γ.

Since our automata run on trees of bounded degree, however, it is convenient to bound the degree of fitting tree CQs. The following lemma shows that this can be done in many relevant cases. The remaining cases are covered by Lemma D.8 below.

Lemma D.5. *Let (E^+, E^-) be a collection of labeled examples. If there is a tree CQ q that is a fitting of (E^+, E^-), then there is such a q of degree at most $||E^-||$. The same is true for weakly and strongly most-general fittings, for unique fittings, and for all these kinds of fittings with a minimal number of variables.*

Proof. Take any tree CQ $q(x_0)$ that is a fitting for (E^+, E^-). We consider the answer variable x_0 to be the root of q, imposing a direction on it and allowing us to speak about successors, predecessors, etc.

Assume that q contains a variable x with more than $||E^-||$ successors $y_1, \ldots, y_m$. For $1 \leq i \leq m$, let $q|^i_x$ denote the tree CQ obtained from the subquery of q rooted at x by deleting all successors $y_i, \ldots, y_m$ and the subtrees below them. We define S_i to be the set of all values a such that for some $e \in E^-$, there is a homomorphism h from $q|^i_x$ to e with $h(x) = a$. Clearly, $S_1 \supseteq S_2 \cdots$. Consequently, $S_j = S_{j+1}$ for some $j \leq ||E^-||$. Let q' be obtained from q by removing the successor y_{j+1} of x and the subtree below it.

We show in the following that q' is a fitting for (E^+, E^-). It must then be weakly/strongly most-general resp. unique if q is since q' may only be more general than q (and q' must be equivalent to q if q is a unique fitting). Likewise, q' cannot have more variables than q and thus if q' had a minimal number of variables, then so does q'. In fact, q' has less variables than q, so showing that q' is a fitting for (E^+, E^-) establishes a contradiction to our assumption that q has degree exceeding $||E^-||$.

It is also clear that q' fits all positive examples. For the negative examples, assume to the contrary that there is an $(I, c) \in E^-$ and a homomorphism h' from q' to I with $h'(x_0) = c$. We can construct from h' a homomorphism h from q to I with $h(x_0) = c$, yielding a contradiction. Clearly, h' is also a homomorphism from $q|^j_x$ to I and since $S_j = S_{j+1}$, this means that we also find a homomorphism g from $q|^{j+1}_x$ to I such that $h'(x) = g(x)$. We can plug g into h in an obvious way to find the desired homomorphism h from q to I with $h(x_0) = c$.

If we apply the above argument repeatedly, we thus find a fitting of degree at most $||E^-||$ and this fitting is weakly/strongly most-general and has a minimal number of variables if this was the case for the original fitting. □

Now for the encoding of tree CQs as Γ-labeled trees. Assume that we are concerned with a collection of labeled examples over some binary schema $\mathcal{S}$. Then the symbols in Γ are the sets σ that contain at most one $\mathcal{S}$-role (binary relation symbol from $\mathcal{S}$ or converse thereof) and any number of unary relation symbols from $\mathcal{S}$. A Γ-labeled tree is *proper* if the symbol σ that labels the root contains no $\mathcal{S}$-role while any other label used contains exactly one $\mathcal{S}$-role. Nodes in the tree correspond to variables in the tree CQ and the $\mathcal{S}$-role in a node/variable label determines how the predecessor links to the variable. If, for example, (T, L) is a Γ-labeled tree, $122 \in T$ with $R^- \in L(122)$, x is the variable that corresponds to node 122 in the tree CQ and y its predecessor (corresponding to node 12), then the tree CQ contains the atom $R(x, y)$. In this way, tree CQs correspond to proper Γ-labeled trees in an obvious way, and vice versa. In the following, we do often not explicitly distinguish between tree CQs and their encoding as a Γ-labeled tree and say, for example, that an automaton *accepts* a tree CQ. Note that properness can always be ensured by intersecting with a trivial two-state TWAPA which ensures that the input tree is proper.

D.4 Arbitrary Tree CQ Fittings

Theorem 5.4. *If any tree CQ fits a collection of labeled examples $E = (E^+, E^-)$, then we can produce a DAG representation of a fitting tree CQ with a minimal number of variables in single exponential time and the size of such a tree CQ is at most double exponential.*

To prove Thm. 5.4, we use an automata-based approach that also reproves the ExpTime upper bound from Theorem 5.3 and will be reused to prove results in subsequent sections.

Let (E^+, E^-) be a collection of labeled examples over schema $\mathcal{S}$. We construct a TWAPA $\mathfrak{A}$ that accepts exactly the fittings for (E^+, E^-) of degree at most $||E^-||$ and whose number of states is only polynomial in $||E^+ \cup E^-||$. We may then use an emptiness check on $\mathfrak{A}$ to reprove the ExpTime upper bound from Thm. 5.3. What is more important, we might use Point 5 of Thm. D.4 to extract from $\mathfrak{A}$ in single exponential time the DAG representation of a fitting for (E^+, E^-) with a minimal number of variables, and to show that the number of variables in that fitting is at most $2^{2^{p(||E^+ \cup E^-||)}}$, p a polynomial. By Lemma D.5, the number of variables in the resulting fitting is minimal not only among the fittings of degree at most $||E^-||$, but also among all fittings.

To obtain the desired TWAPA $\mathfrak{A}$, we construct one TWAPA $\mathfrak{A}_e$ for each example $e \in E^+ \cup E^-$, ensuring that it accepts exactly the tree CQs of degree at most m that admit a simulation (equivalently: a homomorphism) to e, then complement the TWAPA $\mathfrak{A}_e$ if $e \in E^-$, and finally take the intersection of all obtained TWAPAs. Building the TWAPA $\mathfrak{A}_e$ is very simple, we only give a sketch. Let $e = (I, c_0)$. The set of states S of $\mathfrak{A}_e$ contains the pairs (c, R) such that I contains a fact of the form $R(c', c)$ or $c \in \text{adom}(I)$ and R is the special symbol '$-$'. The initial state is $(c_0, -)$. All that $\mathfrak{A}_e$ does is repeatedly sending a copy of itself to every successor node in the input tree, guessing a homomorphism target in I for that node. Since the connecting role is only made explicit at that successor, we also guess that role and then verify that the guess was correct once that we are at the successor. More precisely, for $(c, R) \in S$ and $\sigma \in \Gamma$, we set

$$\delta((c, R), \sigma) = \bigwedge_{1 \leq i \leq m} \bigvee_{R' \ \mathcal{S}\text{-role and } R'(c,d) \in I} [i](d, R')$$

if $R \in \sigma \cup \{-\}$ and $A \in \sigma$ implies $A(c) \in I$, and otherwise put $\delta((c, R), \sigma) = \text{false}$.

D.5 Most-Specific Fitting Tree CQ

The equivalence between Points 2 and 3 of the following proposition has already been observed in [33].

Proposition 5.5. *For all tree CQs q and collections of labeled examples $E = (E^+, E^-)$, the following are equivalent:*

(1) q is a weakly most-specific fitting for E,

(2) q is a strongly most-specific fitting for E,
(3) q fits E and $\Pi_{e\in E^+}(e) \preceq q$.

Proof. "2 $\Rightarrow$ 1" is immediate. For "3 $\Rightarrow$ 2", assume that q fits E and $\Pi_{e\in E^+}(e) \preceq q$. Further, let p be a tree CQ that fits E. We have to show that $q \subseteq p$. Since p fits E, we have $p \preceq (I, a)$ for all $(I, a) \in E^+$ and thus also $p \preceq \Pi_{e\in E^+}(e)$. From $\Pi_{e\in E^+}(e) \preceq q$, we obtain $p \preceq q$, thus $q \subseteq p$ as required.

For "1 $\Rightarrow$ 3", assume that q is a weakly most-specific fitting for E. Then $q \preceq e$ for all $e \in E^+$ and thus also $q \preceq \Pi_{e\in E^+}(e)$. Let m be the depth of q. Then also $q \preceq U_m$ where U_m is the m-finite unraveling of $\Pi_{e\in E^+}(e)$ at the tuple $\bar{e}$ that consists of the distinguished elements of the examples in E^+. Since q is a fitting for E, $q \not\preceq e$ for all $e \in E^-$ and thus $q \preceq U_m$ implies that $U_m \not\preceq e$ for all $e \in E^-$. Consequently, U_m is a fitting for E. It easily follows that the m'-finite unraveling $U_{m'}$ of $\Pi_{e\in E^+}(e)$ at $\bar{e}$ is also a fitting for E, for all $m' \geq m$. Since q is weakly most-specific and $q \preceq U_{m'}$, we must have $U_{m'} \preceq q$ for all $m' \geq m$ (for, otherwise, the canonical CQ of $U_{m'}$ would a fitting tree CQ that is strictly more specific than q). Then clearly also $U_i \preceq q$ for all $i \geq 1$. Thus Lemma D.1 yields $\Pi_{e\in E^+}(e) \preceq q$, as required. □

In [33] it is shown that verification and existence of a most-specific fitting tree CQ are in ExpTime and PSpace-hard when there are only positive examples, but no negative examples. Here, we consider the case with negative examples and show ExpTime-completeness.

Theorem 5.7. *Verification and existence of most-specific fitting tree CQs is in ExpTime.*

Proof. By Prop. 5.5, we may verify whether a tree CQ q is a most-specific fitting for some $E = (E^+, E^-)$ by checking whether q fits E based on Thm. 5.2, then constructing $\Pi_{e\in E^+}(e)$ and deciding in PTime whether $\Pi_{e\in E^+}(e) \preceq q$. This gives the desired ExpTime upper bound.

Now for existence. An ExpTime upper bound is proved in [33] for the case where there are only positive examples, but no negative examples. We extend this to negative examples. Given a collection of labeled examples $E = (E^+, E^-)$, we may first decide whether E has a fitting tree CQ based on Thm. 5.3, answer 'no' if this is not the case, and then use the algorithm from [33] to decide whether $(E^+, \emptyset)$ has a strongly most-specific fitting tree CQ and return the result.

We have to argue that this is correct. This is clearly the case if E has no fitting tree CQ. Thus assume that there is such a CQ q_0. First assume that the answer returned by the second check is 'no'. Assume to the contrary of what we have to prove that E has a most-specific fitting tree CQ q. Then Point 3 of Prop. 5.5 yields $\Pi_{e\in E^+}(e) \preceq q$. Since q fits also $(E^+, \emptyset)$, again from Point 3 we obtain that q is also a most-specific fitting for $(E^+, \emptyset)$, a contradiction. Now assume that the answer returned by the second check is 'yes'. It suffices to show that any (strongly) most-specific fitting tree CQ q^+ for $(E^+, \emptyset)$ satisfies $q^+ \not\preceq e$ for all $e \in E^-$. But this follows from the existence of q_0 since we know that $q_0 \preceq q^+$ (since q_0 also fits $(E^+, \emptyset)$ and q^+ is strongly most-specific for $(E^+, \emptyset)$) and $q_0 \not\prec e$ for all $e \in E^-$. □

We next establish an upper bound on the size of most-specific fitting tree CQs.

Theorem 5.8. *If a collection of labeled examples $E = (E^+, E^-)$ admits a most-specific tree CQ fitting, then we can construct a DAG representation of such a fitting with a minimal number of variables in single exponential time and the size of such a tree CQ is at most double exponential.*

To prove Thm. 5.8, we again use an automata-based approach that also reproves the ExpTime upper bound in Thm. 5.7 and will be reused to prove results in subsequent section. It relies on a characterization of most-specific fittings via complete initial pieces of the unraveling of $\Pi_{e\in E^+}(e)$. We next make this precise. It is related to the decision procedure for the existence of weakly most-specific fitting tree CQs with only positive examples given in [33].

Let $E = (E^+, E^-)$ be a collection of labeled examples, and let U be the unraveling of $\Pi_{e\in E^+}(e)$. An *initial piece* U' of U is a connected instance that is obtained as the restriction of U to some finite non-empty subset $S \subseteq \mathrm{adom}(U)$. We say that U' is *complete* if for all paths $pRa \in \mathrm{adom}(U)$ with $p \in \mathrm{adom}(U')$ and $pRa \notin \mathrm{adom}(U')$, there is an $R(p, c) \in U'$ with $(U, pRa) \preceq (U, c)$.

Lemma D.6. *If U' is a complete initial piece of U, then $U \preceq U'$.*

Proof. Let U' be a complete initial piece of U. It can be verified that the following is a simulation from U to U': $S = \{(p, p') \mid p \in \mathrm{adom}(U), p' \in \mathrm{adom}(U')$ and $(U, p) \preceq (U, p')\}$. □

In the following, we view an initial piece U' of U as a CQ. We then take the unique path of length 1 in $\mathrm{adom}(U')$ to be the answer variable. The following proposition links complete initial pieces tightly to most-specific fittings. In particular, it implies that if there is a most-specific fitting, then there is a complete initial piece that is a most-specific fitting.

Proposition D.7. *Let $E = (E^+, E^-)$ be a collection of labeled examples. Then*

(1) any complete initial piece of the unraveling U of $\Pi_{e\in E^+}(e)$ that fits E is a most-specific fitting for E and conversely,
(2) any most-specific fitting of E is simulation equivalent to some and every complete initial piece of U.

Proof. For Point 1, let U' be a complete initial piece of U that fits E. By Lemma D.6, $U \preceq U'$, and thus also $\Pi_{e\in E^+}(e) \preceq U'$. By Prop. 5.5, U' is thus a most-specific fitting for E.

For Point 2, let q be a most specific fitting for E. Then the fact that $q \preceq \Pi_{e\in E^+}(e)$ and Prop. 5.5 imply that q is simulation equivalent to $\Pi_{e\in E^+}(e)$, thus to U. By Lemma D.6, U is simulation equivalent to any complete initial piece of U. It therefore remains to show that a complete initial piece of U exists.

We first observe that there is an $m \geq 1$ such that $U \preceq U_m$ with U_m the m-finite unraveling of U. In fact, simulation equivalence of U and q implies that there is a homomorphism h_1 from U to q and h_2 from q to U, but since q is finite the composition $h_2 \circ h_1$ is a homomorphism from U into some U_m.

Since $U \preceq U_m$, there is a homomorphism h from U to U_m that is the identity on the root of U (the unique path in U of length 1). Let U' be the initial piece of U such that $\mathrm{adom}(U')$ is the range of h. We argue that U' is complete. Take any path $pRa \in \mathrm{adom}(U)$ with

$p \in \text{adom}(U')$ and $pRa \notin \text{adom}(U')$. Then $h(pRa) \in \text{adom}(U')$ and $R(p, h(pRa)) \in U'$. Moreover, h witnesses that $(U, pRa) \preceq (U', h(pRa))$, thus $(U, pRa) \preceq (U, h(pRa))$ as desired. □

Next, we establish a version Lemma D.5 for most-specific fittings and make some additional observations. In contrast to Lemma D.5, however, we can only bound the degree exponentially instead of linearly. This shall not be a problem in what follows and in particular using tree automata on trees of exponential outdegree does not stand in the way of obtaining ExpTime upper bounds.

Lemma D.8. *Let (E^+, E^-) be a collection of labeled examples. If there is a tree CQ q that is a most-specific fitting of (E^+, E^-), then there is such a q of degree at most $2^{||E^+||}$. The same is true with a minimal number of variables. Moreover, any most-specific fitting with a minimal number of variables must be isomorphic to a complete initial piece of the unraveling of $\Pi_{e \in E^+}(e)$.*

Proof. The case where we disregard the number of variables is immediate. Prop. D.7 tells us that if there exists a most-specific fitting q, then some initial piece p of the unraveling of $\Pi_{e \in E^+}(e)$ is a fitting of the same kind. Clearly, the degree of p is bounded by $2^{||E^+||}$ and we are done.

Let us now add the requirement that the number of variables be minimal. It suffices to show that $q(x)$ has an injective homomorphism h to $p(x)$ with $h(x) = x$ because this means that q is actually a subquery of p (can be obtained from it by dropping atoms), and thus the degree of q cannot be larger than that of p. Since p and q are simulation equivalent, we find homomorphisms h_1 from p to q and h_2 from q to p, both the identity on x. The composition $h_1 \circ h_2$ is a homomorphism from q to q and must be surjective as otherwise it identifies a strict subquery of q (with fewer variables!) that is homomorphically equivalent to q, and thus simulation equivalent. But the composition can only be surjective if h_2 is injective, so h_2 is the desired injective homomorphism from q to p.

For the 'moreover part', we note that we have already shown above that a most-specific fitting q with a minimal number of variables is isomorphic to a subtree p' of some initial piece p of the unraveling U of $\Pi_{e \in E^+}(e)$, but that subtree p' is an initial piece of U itself, and thus it remains to prove that p' is complete. However, any incomplete initial piece p' of U does not admit a simulation from U as otherwise the simulation would witness completeness of p'. Consequently, p' must be complete as by Prop. 3.5 it is simulation equivalent to U. □

We now give the NTA construction for Thm. 5.8. Let (E^+, E^-) be a collection of labeled examples over schema $\mathcal{S}$ and set $m = 2^{||E^+||}$. We aim to build an NTA $\mathfrak{A}$ with single exponentially many states that accepts exactly the most-specific tree CQ fittings for (E^+, E^-) which have degree at most m and are isomorphic to a complete initial piece of the unraveling U of $\Pi_{e \in E^+}(e)$. Note that the latter conditions are without loss of generality due to Prop. D.7 and Lemma D.8, no matter whether we want to decide the existence of most-specific fittings or construct a most-specific fitting with a minimal number of variables.

We may use an emptiness check on $\mathfrak{A}$ to reprove the ExpTime upper bound from Thm. 5.7. Moreover, we might use Point 2 of Thm. B.5 to extract from $\mathfrak{A}$ in single exponential time the DAG representation of a most-specific fitting for (E^+, E^-) with a minimal number of variables, and to show that the number of variables in that fitting is at most $2^{2^{p(||E^+ \cup E^-||)}}$, p a polynomial.

The NTA $\mathfrak{A}$ is constructed as the intersection of two NTAs $\mathfrak{A}_1$ and $\mathfrak{A}_2$, c.f. Point 4 of Thm. B.5. The first NTA checks that the input tree CQ is a fitting for (E^+, E^-). It is obtained by converting the TWAPA used in the proof of Thm. 5.3 into an NTA as per Point 4 of Thm. D.4. The second NTA verifies that the input tree CQ is a complete initial piece of the unraveling U of $\Pi_{e \in E^+}(e)$ and thus, by Prop. D.7, a most-specific fitting.

We define $\mathfrak{A}_2 = (Q, \Gamma, \Delta, F)$ where Γ is the alphabet for tree CQs over schema $\mathcal{S}$. The states in Q take the form (aR, b) with $a, b \in \text{adom}(\Pi_{e \in E^+}(e))$ and R an $\mathcal{S}$-role. As a special case, aR can be '$-$'. Informally, state (aR, b) means that we are currently visiting a path in U that ends with aRb. In the transition relation, we verify that all successors required for the initial piece to be complete are present. It is convenient to view $\mathfrak{A}_2$ as a top-down automaton and F as a set of initial states. We set $F = \{(-, e_0)\}$ where $e_0 \in \text{adom}(\Pi_{e \in E^+}(e))$ is the root of the unraveling U of $\Pi_{e \in E^+}(e)$, that is, the tuple that consists of all the selected points in the data examples in E^+. We then include in Δ all transitions

$$\langle q_1, \dots, q_m \rangle \xrightarrow{\sigma} (aR, b),$$

where each q_i can also be $\bot$, such that the following conditions are satisfied:

(1) $R \in \sigma$ (unless $aR = -$);
(2) $A \in \sigma$ iff $A(b) \in \Pi_{e \in E^+}(e)$, for all unary $A \in \mathcal{S}$;
(3) if $q_i = (bS_i, c_i)$, then $S_i(b, c_i) \in \Pi_{e \in E^+}(e)$ for $1 \le i \le m$;
(4) if b has an S-successor c in U, then one of the following holds:
   - there is an i such that $q_i = (bS, c_i)$ and $(\Pi_{e \in E^+}(e), c) \preceq (\Pi_{e \in E^+}(e), c_i)$;
   - $R = S^-$ and $(\Pi_{e \in E^+}(e), c) \preceq (\Pi_{e \in E^+}(e), a)$;
(5) all of $q_1, \dots, q_m$ that are not $\bot$ are pairwise distinct.

It can be verified that the automaton recognizes precisely the intended language. Note that to construct the NTA, we need to know about simulations between values in $\Pi_{e \in E^+}(e)$. We determine these by first constructing $\Pi_{e \in E^+}(e)$ in single exponential time and then computing in PTime the maximal simulation on it.

D.6 Weakly Most-General Fitting Tree CQs

Theorem 5.11. *Verification of weakly most-general fitting tree CQs is in PTime.*

Based on Prop. 5.10, Thm. 5.11 is easy to prove. Given a CQ q and a collection of labeled examples (E^+, E^-), we may first verify in polynomial time that q fits E. We then compute in polynomial time a frontier $\mathcal{F}$ of q w.r.t. tree CQs and then check that none of the CQs in $\mathcal{F}$ simulates into a negative example. For the latter, we can use the frontier construction presented in Sect. B.1 which yields a frontier not only w.r.t. tree CQs, but w.r.t. unrestricted CQs.

Prop. 5.10 also serves as the basis for a decision procedure for the existence of weakly most-general fitting tree CQs. In principle, we could use the frontier construction from Sect. B.1 and NTAs, as in that section. However, the construction presented there is already complex and we would have to replace homomorphisms with simulations, which results in additional complications. This led

us to working with a different frontier construction tailored towards tree CQs, presented in [11], and with alternating tree automata.

Theorem 5.12. *Existence of weakly most-general fitting tree CQs is in ExpTime. Moreover, if a collection of labeled examples $E = (E^+, E^-)$ admits a weakly most-general tree CQ fitting, then we can construct a DAG representation of such a fitting with a minimal number of variables in single exponential time and the size of such a tree CQ is at most double exponential.*

Let (E^+, E^-) be a collection of labeled examples over schema $\mathcal{S}$ and set $m := ||E^-|$. We aim to construct a TWAPA $\mathfrak{A}$ with polynomially many states that accepts exactly the weakly most-general tree CQ fittings for (E^+, E^-) which have degree at most m. The latter restriction is justified by Lemma D.5. By Prop. 5.10, we may construct $\mathfrak{A}$ as the intersection of two TWAPAs $\mathfrak{A}_1$ and $\mathfrak{A}_2$ where $\mathfrak{A}_1$ verifies that the q fits (E^+, E^-) and $\mathfrak{A}_2$ that every element of the frontier $\mathcal{F}$ for q w.r.t. tree CQs simulates to an example in E^-. For $\mathfrak{A}_1$, we can use the TWAPA from Sect. D.4.

We next describe the frontier construction.

Step 1: Generalize. For each variable x in q, define a set $\mathcal{F}_0(x)$ that contains all tree CQs which can be obtained by starting with the subquery of q rooted at x and then doing one of the following:

(1) choose an atom $A(x)$ and remove it;
(2) choose a successor y of x, with $R(x, y) \in q_x$, and then
   (a) remove $R(x, y)$ and the subtree rooted at y and
   (b) for each $q'(y) \in \mathcal{F}_0(y)$, add a disjoint copy $\widehat{q}'$ of q' and the role atom $R(x, y'')$ with y'' the copy of y in $\widehat{q}'$.

Every variable x in the resulting tree CQs may be associated in an obvious way with a variable from q that it derives from. We denote that original variable with $x^{\downarrow}$.

Step 2: Compensate. We construct the frontier $\mathcal{F}$ of $q(x_0)$ by including, for each $p \in \mathcal{F}_0(x_0)$, the tree CQ obtained from p by adding, for every atom $R(x, y)$ in p directed away from the root, an atom $R(z, y)$, z a fresh variable, as well as a disjoint copy $\widehat{q}$ of q and glue the copy of $x^{\downarrow}$ in $\widehat{q}$ to z.

Let $E^- = \{(I_1, \widehat{c}_1), \ldots, (I_n, \widehat{c}_n)\}$. We assume w.l.o.g. that the negative counterexamples have pairwise disjoint domains. Set $I = I_1 \cup \cdots \cup I_n$ and $\mathsf{adom} = \mathsf{adom}(I)$. The TWAPA $\mathfrak{A}_2$ starts in state s_0 and universally branches over all queries in the frontier, choosing for each of them a negative example that it simulates into. Since the queries in the frontier are tree CQs, we can actually verify the existence of a homomorphism in place of a simulation (which we consider slightly more intuitive).

Assume that the input tree represents the tree CQ $q(x_0)$. Then by construction, $\mathcal{F}$ contains a query q_A for every atom $A(x_0) \in q$ and a query q_y for every successor y of x in q. We set for all $\sigma \in \Gamma$:

$$\delta(s_0, \sigma) = \bigwedge_{A \in \sigma} \bigvee_{1 \le \ell \le n} s^A_{\widehat{c}_\ell} \wedge \bigwedge_{1 \le i \le m} ([i]\bot \vee \bigvee_{1 \le \ell \le n} s^i_{\widehat{c}_\ell}).$$

So the TWAPA is now located at the root of the input tree, in state $s^A_{\widehat{c}_\ell}$ to verify that the query q_A in $\mathcal{F}$ maps to $\widehat{c}_\ell$, and in state $s^i_{\widehat{c}_\ell}$ to verify that the query q_y in $\mathcal{F}$, where y is the variable in q represented by the i-th successor of the root in the input tree, maps to $\widehat{c}_\ell$.

The queries q_A are easy to deal with. We can essentially use the same automaton as in Sect. D.4, except that we must ignore the atom $A(x_0)$ and also take into account the subqueries added in the compensate step. More precisely, for all $c \in \mathsf{adom}$ and $\sigma \in \Gamma$ we set

$$\delta(s^A_c, \sigma) = \bigwedge_{1 \le i \le m} \bigvee_{R \ \mathcal{S}\text{-role and } R(c,d) \in I} [i] s_{d,R}$$

if $B \in \sigma \setminus \{A\}$ implies $B(c) \in I$, and $\delta(s^A_c, \sigma) = \text{false}$ otherwise. For all $c \in \mathsf{adom}$, $\mathcal{S}$-roles R, and $\sigma \in \Gamma$ we also set

$$\delta(s_{c,R}, \sigma) = t^0_{c,R} \wedge \bigwedge_{1 \le i \le m} \bigvee_{R' \ \mathcal{S}\text{-role and } R'(c,d) \in I} [i] s_{d,R'}$$

if $R \in \sigma$ and $A \in \sigma$ implies $A(c) \in I$, and $\delta(s_{c,R}, \sigma) = \text{false}$ otherwise. The state $t^0_{c,R}$ is for verifying the subqueries added in the compensate step. For all $\sigma \in \Gamma$, set

$$\delta(t^0_{c,R}, \sigma) = \bigvee_{R' \ \mathcal{S}\text{-role and } R(d,c) \in I} \langle -1 \rangle t_{d,R'}$$

and for all $c \in \mathsf{adom}$, $\mathcal{S}$-roles R, and $\sigma \in \Gamma$, set

$$\delta(t_{c,R}, \sigma) = s_{c,R} \wedge \bigvee_{R' \ \mathcal{S}\text{-role and } R(d,c) \in I} [-1] t_{d,R'}$$

if $R \in \sigma$ and $A \in \sigma$ implies $A(c) \in I$, and $\delta(s_{c,R}, \sigma) = \text{false}$ otherwise.

It remains to deal with the states s^i_c. Remember that their purpose is to verify that the query q_y in $\mathcal{F}$, where y is the variable in q represented by the i-th successor of the root in the input tree, maps to $\widehat{c}_\ell$. Also recall that in the generalize step of the construction of q_y, the successor y of x_0 is replaced with one successor for each query in $\mathcal{F}_0(y)$. For $1 \le i \le m$ and all $c \in \mathsf{adom}$, set

$$\delta(s^i_c, \sigma) = \bigvee_{R \ \mathcal{S}\text{-role}} \langle i \rangle u_{c,R} \wedge \bigwedge_{\substack{1 \le j \le m \\ j \ne i}} \bigvee_{R \ \mathcal{S}\text{-role and } R(c,d) \in I} [j] s_{d,R}$$

State $u_{c,R}$ expresses that variable y of the input tree CQ q that the TWAPA is currently visiting is replaced with each of the queries in $\mathcal{F}_0(y)$, and that it is an R-successor of its predecessor, which is mapped to c. There is one such query for each atom $A(y)$ in q and every successor of y in q. We are thus in a very similar, though not identical, situation as in the beginning. For all $c \in \mathsf{adom}$, $\mathcal{S}$-roles R, and $\sigma \in \Gamma$ set

$$\delta(u_{c,R}, \sigma) = \bigwedge_{A \in \sigma} \bigvee_{R(c,d) \in I} s^A_d \wedge \bigwedge_{1 \le i \le m} ([i]\bot \vee \bigvee_{R(c,d) \in I} s^i_d).$$

if $R \in \sigma$ and $\delta(u_{c,R}, \sigma) = \bot$ otherwise. This finishes the construction of the automaton.

D.7 Unique Fitting Tree CQ

Clearly, a fitting tree CQ is a unique fitting if and only if it is both a most-specific fitting and a weakly most-general fitting. We may thus obtain results for unique fitting tree CQs by combining the results from Sections D.5 and D.6.

Theorem 5.13. *Verification and existence of unique fitting tree CQs is in ExpTime.*

The upper bound for verification in Thm. 5.13 is a direct consequence of the upper bounds in Theorems 5.7 and 5.12.

The upper bound for existence can be established as follows. In the proof of Thm. 5.8, we have constructed, given a collection of labeled examples (E^+, E^-), an NTA $\mathfrak{A}_1$ with single exponentially many states that accepts exactly the fitting tree CQs for (E^+, E^-) that are most-specific, have degree at most $2^{||E^+||}$, and are isomorphic to a complete initial piece of the unraveling of $\Pi_{e\in E^+}(e)$. Recall that the latter two conditions are w.l.o.g. in the sense that (i) if a most-specific fitting exists, then there exists one that satisfies the conditions and (ii) if there is any most-specific fitting at all, then the most-specific fitting with the minimal number of variables also satisfies the conditions. Clearly, the same is true for unique fittings as every unique fitting is a most-specific fitting. We can easily modify $\mathfrak{A}_1$ so that it runs on trees of degree $m := \max\{2^{||E^+||}, ||E^-||\}$.

In the proof of Thm. 5.12, we have constructed a TWAPA $\mathfrak{A}_2$ with polynomially many states that accepts exactly the fitting tree CQs for (E^+, E^-) that are weakly most-general and have degree at most $||E^-||$. We can easily modify $\mathfrak{A}_2$ so that it runs on tree of degree m. Moreover, we can convert it into an equivalent NTA with single exponentially many states and then intersect with $\mathfrak{A}_1$ to obtain an NTA $\mathfrak{A}$ with still single exponentially many states that accepts exactly the unique fitting tree CQs for (E^+, E^-) of degree at most m. It remains to check emptiness of $\mathfrak{A}$ in polynomial time.

D.8 Basis of Most-General Fitting Tree CQ

Proposition 5.15. *For all collections of labeled examples $E = (E^+, E^-)$, the following are equivalent, for $p = \Pi_{e\in E^+}(e)$:*

(1) $\{q_1, \ldots, q_n\}$ is a basis of most-general fitting tree CQs for E,
(2) each q_i fits E and $(\{q_1, \ldots, q_n\}, E^-)$ is a simulation duality relative to p.

Proof. From 1 to 2: By assumption, each q_i fits E. Let e be any data example such that $e \to p$ (and, hence, e fits E^+). We need to show that $q_i \preceq e$ for some $i \leq n$ iff $e \not\preceq e'$ for all $e' \in E^-$. First, assume $q_i \preceq e$, and assume for the sake of a contradiction that $e \preceq e' \in E^-$. Then, by transitivity, $q_i \preceq e'$, and hence (since q_i is a tree CQ), $q_i \to e'$, contradicting the fact that q_i fits E. For the converse direction, assume $e \not\preceq e'$ for all $e' \in E^-$. Since $e \preceq p$, this means that there is some finite unraveling e^* of e, such that canonical CQ q_{e^*} of e^* is a tree CQ that fits E. Hence, $q_i \preceq q_{e^*}$ for some $i \leq n$.

From 2 to 1: let q' be any tree CQ that fits (E^+, E^-). Then $q' \to p$ and hence $q' \preceq p$. Furthermore, for all $e \in E^-$, $q' \not\to e$ and hence $q' \not\preceq e$. It follows that $q_i \preceq q'$ for some $i \leq n$, and therefore $q' \subseteq q_i$. □

Proposition 5.17. *Let D be a finite set of data examples and $\widehat{e}$ a data example. Then the following are equivalent:*

(1) there is a finite set of tree data examples F such that (F, D) is a simulation duality relative to $\widehat{e}$,
(2) there is a finite number of critical tree obstructions q for D that satisfy $q \to \widehat{e}$ (up to isomorphism).

Proof. "(1) ⇒ (2)". Assume that (F, D) is a simulation duality relative to $\widehat{e}$ with F a set of tree examples. Let n be the maximum number of variables of any example in F. To show that there are only finitely many critical tree obstructions q for D that satisfy $q \preceq \widehat{e}$, it suffices to show that each such q has at most n variables. So take a critical tree obstruction q for D with $q \preceq \widehat{e}$. Then $q \not\preceq e$ for all $e \in D$ and thus there is an $e' \in F$ with $e' \preceq q$. Since e' is a tree, this implies $e' \to q$. Now, the homomorphism witnessing the latter must be surjective as otherwise it gives rise to a tree CQ q' that can be obtained from q by dropping subtrees and that still satisfies $q' \not\preceq e$ for all $e \in D$ (because $q' \preceq e$ would yield $e' \preceq e$ by composition of simulations) which contradicts the fact that q is a critical tree obstruction for D. Consequently, the number of variables in q is bounded by the number of variables in e', thus by n.

"(2) ⇒ (1)". Assume that there is a finite number of critical tree obstructions q for D that satisfy $q \to \widehat{e}$, up to isomorphism. Let F be a set of tree CQs that contains one representative for every isomorphism class. Then (F, D) is a simulation duality relative to $\widehat{e}$.

To see this, first take a data example e such that $e \preceq e'$ for some $e' \in D$ and $e \preceq \widehat{e}$. Then $e'' \not\preceq e$ for all $e'' \in F$ because otherwise we obtain $e'' \preceq e'$ be composing simulations, which contradicts the fact that e'' satisfies the first condition of critical tree obstructions.

Now take a data example e with $e' \not\preceq e$ for all $e' \in F$ and $e \preceq \widehat{e}$. Assume to the contrary of what we have to show that $e \not\preceq e''$ for all $e'' \in D$. By Lemma D.1, there is then some m-finite unraveling u of e such that $u \not\preceq e''$ for all $e'' \in D$. Moreover, $u \preceq \widehat{e}$. Let u' be obtained from u by dropping subtrees as long as $u' \not\preceq e''$ for all $e'' \in D$. Clearly, u' is a critical tree obstruction. But then F contains a CQ that is isomorphic to u', contradicting our assumption that $e' \not\preceq e$ for all $e' \in F$.

□

Theorem 5.16. *The verification problem for bases of most-general fitting tree CQs is in ExpTime.*

Proof. Let $E = (E^+, E^-)$ and $\{q_1, \ldots, q_n\}$ be given. We first verify in ExpTime that each q_i fits E.

By Thm. 2.1, we can compute, in single exponential time, for each q_i, a set of data examples D_{q_i}, such that $(\{e_{q_i}\}, D_{q_i})$ is a homomorphism duality. Let $D = \{e_1 \times \cdots \times e_n \mid e_i \in D_{q_i}\}$. It is easy to see (using Prop. A.1) that $(\{e_{q_1}, \ldots, e_{q_n}\}, D)$ is a homomorphism duality. Since the elements of $\{e_{q_1}, \ldots, e_{q_n}\}$ are trees, it is also a simulation duality.

We claim that the following are equivalent:

(1) $\{q_1, \ldots, q_n\}$ is a basis of most-general fitting tree CQs for E,
(2) $(\{e_{q_1}, \ldots, e_{q_n}\}, E^-)$ is a simulation duality relative to p, where $p = \Pi_{e\in E^+}(e)$,
(3) For each $e \in D$, there is $e' \in E^-$ such that $e \times p \preceq e'$.

The equivalence of 1 and 2 is given by Prop. 5.15.

(2 ⇒ 3) Let $e \in D$. Since $(\{e_{q_1}, \ldots, e_{q_n}\}, D)$ is a homomorphism duality and $e \in D$, we have $e_{q_i} \not\preceq e$ for all $i \leq n$. Hence, by Prop. A.1, also $e_{q_i} \not\preceq e \times p$. Therefore, since $e \times p \preceq p$, we have that $e \times p \preceq e'$ for some $e' \in E^-$.

(3 ⇒ 2) Let e be any data example such that $e \preceq p$. If some $e_{q_i} \preceq e$, then, since q_i fits E, we know that $e \not\preceq e'$ for all $e' \in E^-$. If, on the other hand, $e_{q_i} \not\preceq e$ for all e_{q_i}, then $e \preceq e'$ for some $e' \in D$. Hence, since $e \preceq p$, by Prop. A.1, we have that $e \preceq e' \times p$, and therefore $e \preceq e''$ for some $e'' \in E^-$.

This concludes the proof since (3) can be tested in ExpTime. □

We now turn to the existence problem. Let (E^+, E^-) be a collection of labeled examples. A fitting tree CQ q for (E^+, E^-) is *critical* if no tree CQ that can be obtained from q by removing subtrees is a fitting. The following is a consequence of Propositions 5.15 and 5.17.

Lemma D.9. *A collection (E^+, E^-) of labeled examples has a basis of most-general fitting tree CQs iff it has a finite number of critical fitting tree CQs (up to isomorphism).*

Lemma D.10. *If q is a critical fitting tree CQ for a collection of labeled examples (E^+, E^-), then the degree of q is bounded by $||E^-||$.*

Proof. The proof is similar to that of Lemma D.5. Let $q(x_0)$ be a critical fitting tree CQ for (E^+, E^-). Assume to the contrary of what we want to show that q contains a variable x with more than $||E^-||$ successors $y_1, \ldots, y_m$. For $1 \leq i \leq m$, let $q|^i_x$ denote the tree CQ obtained from the subquery of q rooted at x by deleting all successors $y_i, \ldots, y_m$ and the subtrees below them. We define S_i to be the set of all values a such that for some $e \in E^-$, there is a homomorphism h from $q|^i_x$ to e with $h(x) = a$. Clearly, $S_1 \supseteq S_2 \cdots$. Consequently, $S_j = S_{j+1}$ for some $j \leq ||E^-||$. Let q' be obtained from q by removing the successor y_{j+1} of x and the subtree below it. We can show as in Lemma D.5 that q' is fitting for (E^+, E^-). This, however, contradicts the fact that q is critical. □

Theorem 5.18. *The existence problem for bases of most-general fitting tree CQs is in ExpTime. Moreover, if a collection of labeled examples E has a basis of most-general fitting tree CQs, then it has such a basis in which every tree CQ has size at most double exponential in $||E||$.*

Assume that we are given a collection of labeled examples $E = (E^+, E^-)$. We construct a TWAPA $\mathfrak{A}$ with polynomially many states that accepts exactly the critical fitting tree CQs for E. By Lemma D.10, $\mathfrak{A}$ may run on tree of degree at most $m := ||E^-||$.

The TWAPA $\mathfrak{A}$ is the intersection of three TWAPAs $\mathfrak{A}_1$, $\mathfrak{A}_2$, and $\mathfrak{A}_3$ such that

- $\mathfrak{A}_1$ accepts those tree CQs that have no homomorphism to any negative example;
- $\mathfrak{A}_2$ accepts those tree CQs that have a homomorphism to some negative example once any subtree is dropped;
- $\mathfrak{A}_3$ accepts those tree CQs that have a homomorphism to all positive examples.

Already in the context of arbitrary fittings, we have seen that the automata $\mathfrak{A}_1$ and $\mathfrak{A}_3$ are easy to construct. The TWAPA $\mathfrak{A}_2$ is a straightforward variation, we only sketch the idea. The automaton first sends a copy of itself to every node in the input tree except the root. It then verifies that, when the subtree rooted at the node that it currently visits is dropped, then the remaining input tree maps to some negative example. It does this by traveling upwards one step to the predecessor and memorizing the successor that it came from. It also uses disjunction to guess an $(I, c) \in E^-$ and an $a \in \mathsf{adom}(I)$ that the predecessor maps to. It then behave essentially like $\mathfrak{A}_3$, verifying the existence of a homomorphism to I, but avoiding the subtree at the memorized successor. Once the root of the input tree is reached, the automaton verifies that the constructed homomorphism uses c as the target.

Since $\mathfrak{A}$ accepts exactly the critical fitting tree CQs for E, by Lemma D.9 it remains to solve the infinity problem for $\mathfrak{A}$. By Point (6) of Theorem D.4, we obtain an ExpTime upper bound.

We next observe that if a collection of labeled examples E has a basis of most-general fitting tree CQs, then it has such a basis in which every tree CQ has size at most double exponential in $||E||$. If, in fact, E has a basis of most-general fitting tree CQs, then we might assume w.l.o.g. that the basis contains only critical fitting tree CQs. By Lemma D.9, E has only finitely many critical fitting tree CQs, and by the construction of $\mathfrak{A}$ above and Point (6) of Theorem D.4, every critical fitting tree CQ has size at most double exponential in $||E||$.

D.9 The Product Simulation Problem into Trees

We prove the following result, which underlies almost all our lower bounds for the verification and existence of fitting tree CQs.

Theorem 5.19. *The product simulation problem into trees is* ExpTime*-hard, even for a fixed schema.*

The proof is by reduction of the product $\downarrow$-simulation problem on instances with a fixed schema Σ and fixed target instance I and target value $c \in \mathsf{adom}(I)$, proved ExpTime-hard in [24]. We may assume w.l.o.g. that $\mathsf{adom}(I)$ contains at least two values.

Assume that we are given as input Σ-instances $I_1, \ldots, I_n$ and a value $\bar{c} \in \mathsf{adom}(\prod_{1 \leq i \leq n} I_i)$. We refer to $I_1, \ldots, I_n$ as the source instances and to $\bar{c}$ as the source value. The aim is to decide whether $(\prod_{1 \leq i \leq n} I_i, \bar{c}) \preceq^{\downarrow} (I, c)$.

Let Γ be the schema that contains all unary relations from Σ, a single binary relation R, as well as fresh unary relations From_c and To_c for all $c \in \mathsf{adom}(I)$. Note that since Σ and I are fixed, the schema Γ is finite and fixed. For the product simulation problem into trees that we are reducing to, we use schema Γ. As in the problem that we are reducing from, it shall suffice to use a fixed target (tree) instance I'.

We convert every instance I_i, $1 \leq i \leq n$, into a Γ-instance I'_i by replacing every fact $S(c, c') \in I_i$ with a collection of paths of length 6. More precisely, we introduce one path for every value $d \in \mathsf{adom}(I)$ as follows, where each $a_{c,c',S,d,i}$ is a fresh value:

- $R(a_{c,c',S,d,1}, c), R(a_{c,c',S,d,2}, a_{c,c',S,d,1})$, $R(a_{c,c',S,d,3}, a_{c,c',S,d,2}), R(a_{c,c',S,d,3}, a_{c,c',S,d,4})$, $R(a_{c,c',S,d,4}, a_{c,c',S,d,5}), R(a_{c,c',S,d,5}, c')$;
- $\mathsf{From}_d(a_{c,c',S,d,4})$;
- $\mathsf{To}_{d'}(a_{c,c',S,d,5})$ for all $S(d, d') \in I$;
- $\mathsf{Out}(a_{c,c',S,d,1})$ and $\mathsf{In}(a_{c,c',S,d,5})$.

This is illustrated in Figure 2 where we only show the unary relations on one path and only the 'i' component of values $a_{c,c',S,d,i}$. Informally, the labels From_d and $\mathsf{To}_{d'}$ identify the edge $S(d, d')$ in I that an edge $S(\bar{c}, \bar{c}')$ in $\prod_{1 \leq i \leq n} I_i$ is mapped to. The labels In and Out serve to deal with the issue that we are reducing from a problem defined in terms of $\downarrow$-simulations to a problem defined in terms of $\uparrow\downarrow$-simulations. We shall refer to this as the $\downarrow\!\!\not\uparrow\downarrow$*-issue.*

As the source instances for the product simulation problem into trees, we use $I'_1, \ldots, I'_n$ plus an additional instance I'_0 illustrated in Figure 3. For all $d \in \mathsf{adom}(I)$, it contains the following facts:

- $P(c_0)$ for every unary relation symbol $P \in \Sigma$;
- $R(c_{d,1}, c_0), R(c_{d,2}, c_{d,1}), R(c_{d,3}, c_{d,2}), R(c_{d,3}, c_{d,4})$;

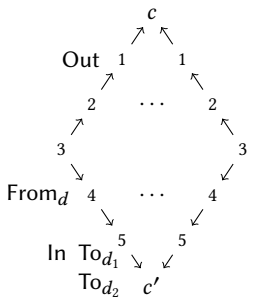

Figure 2: Gadget in I'_i replacing $S(c, c') \in I_i$.

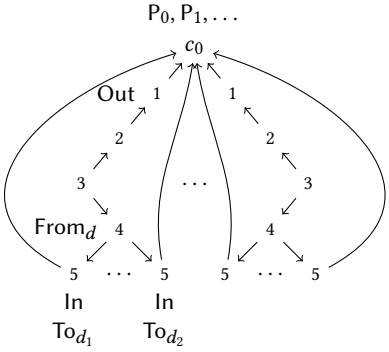

Figure 3: The additional instance I'_0

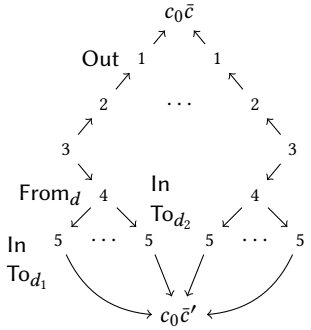

Figure 4: Gadget in $\prod_{0\leq i\leq n} I'_i$ replacing a fact $S(\bar{c}, \bar{c}')$ in $\prod_{1\leq i\leq n} I_i$.

- $\mathsf{Out}(c_{d,1}), \mathsf{From}_d(c_{d,4})$;
- for all $d' \in \mathsf{adom}(I)$:
  - $R(c_{d,4}, c_{d,d',5}), R(c_{d,d',5}, c_0)$;
  - $\mathsf{In}(c_{d,d',5}), \mathsf{To}_{d'}(c_{d,d',5})$.

The source value is $\bar{c}' = c_0\bar{c}$.

Analyzing the interplay of the above gadgets, the reader may verify that every edge $S(\bar{c}, \bar{c}')$ in the product $\prod_{1\leq i\leq n} I_i$ is replaced in the product $\prod_{0\leq i\leq n} I'_i$ by a gadget of the form shown in Figure 4. There are actually some additional 'improper' paths from $\bar{c}$ to $\bar{c}'$ not shown in the figure that carry no From_d label and/or no $\mathsf{To}_{d'}$ label, but these map homomorphically into the properly labeled paths and can be disregarded.

For any instance I' and $c' \in \mathsf{adom}(I')$, we have $(\prod_{0\leq i\leq n} I'_i, \bar{c}') \preceq^{\uparrow\downarrow} (I', c')$ iff there is a homomorphism h' from the unraveling U' of $\prod_{0\leq i\leq n} I'_i$ to I' with $h(\bar{c}') = c'$. We prefer to think in terms of this latter presentation. Let us discuss the shape of U'. Recall that the values in unravelings are paths. We are most interested in the paths $p \in \mathsf{adom}(U')$ that end in a value of the form $c_0\bar{d}$ with $\bar{d} \in \mathsf{adom}(\prod_{1\leq i\leq n} I_i)$. Note that this is the case for the root of U', that is, the path of length 1 that consists of only the source value. Then for every fact $S(\bar{d}, \bar{d}')$ in $\prod_{1\leq i\leq n} I_i$ and any path in U' that ends in $\bar{d}$, we find in U' a subtree rooted at p that can be obtained from the gadget in Figure 4 by duplicating the point $\bar{c}'$ sufficiently many times so that a tree is obtained. This subtree contains, for every fact $S(d, d') \in I$, a path of length 6 that starts at p and ends at a $p' \in \mathsf{adom}(U')$ that in turn ends with a value $c_0\bar{d}'$. The first value on the path is labeled with Out, the fourth with From_d, and the fifth with In and with $\mathsf{To}_{d'}$. Informally, each of the paths represents the choice for h' to map edge $S(\bar{c}, \bar{c}')$ to $S(d, d') \in I$. Note that there is an *and/or-issue* arising here: h' needs to map $S(\bar{c}, \bar{c}')$ only to a *single* $S(d, d') \in I$, but have paths for *all* possible choices.

Unsurprisingly, the $\downarrow/\uparrow\downarrow$-issue shows up in U'. There may in fact be other successors of p in U' than the ones described above. For every fact $S(\bar{c}', \bar{c})$ in $\prod_{1\leq i\leq n} I_i$, there is a gadget in $\prod_{0\leq i\leq n} I'_i$ of the form shown in Figure 4 with $c_0\bar{d}$ in place of $c_0\bar{c}'$. This leads to undesired successors of p in U' that are labeled with In rather than with Out, the latter being the case for the desired successors.

We next define a Γ-instance I' that replaces I. We start with a tree of depth three that branches only at the root b_0 and has one leaf for every pair of values in I. It contains the following facts for all $c, c' \in \mathsf{adom}(I)$:

- $R(b_0, b_{c,c',1}), R(b_{c,c',1}, b_{c,c',2}), R(b_{c,c',2}, b_{c,c',3})$;
- $\mathsf{From}_c(b_{c,c',1}), \mathsf{To}_{c'}(b_{c,c',2})$;
- $P(b_{c,c',3})$ for all $P(c') \in I$.

To address the $\downarrow/\uparrow\downarrow$-issue and the and/or issue, we include in I' additional gadgets. Note that both issues pertain to additional, undesired successors in U'. The additional gadgets, which we refer to as *sinks*, can accommodate these surplus successors and the subtrees below them. There are sinks of three types:

(I) when $S(\bar{c}, \bar{c}')$ is mapped to $S(d, d') \in I$, a sink that takes paths labeled From_e with $e \neq d$;
(II) when $S(\bar{c}, \bar{c}')$ is mapped to $S(d, d') \in I$, a sink that takes paths labeled From_d and To_e with $e \neq d'$;
(III) sinks that deal with additional successors in U due to the $\downarrow/\uparrow\downarrow$-issue.

Each sink takes the form of a path. For a word $w = \sigma_1 \cdots \sigma_k \in \{R, R^-\}^*$, a *full w-path* is a path in which the i-th edge is a forward R-edge if $\sigma_i = R$ and a backward R-edge if $\sigma_i = R^-$. Moreover, every value on the path except the starting value is labeled with all unary relation symbols from Γ.

Let $c, c' \in \mathsf{adom}(I)$. The sink of Type (I) is attached to value $b_{c,c',1}$. We add the following facts:

- $R(s_{c,c',I,3}, b_{c,c',1}), R(s_{c,c',I,3}, s_{c,c',I,4})$;
- $\mathsf{From}_e(s_{c,c',I,4})$ for all $e \in \mathsf{adom}(I) \setminus \{d\}$;

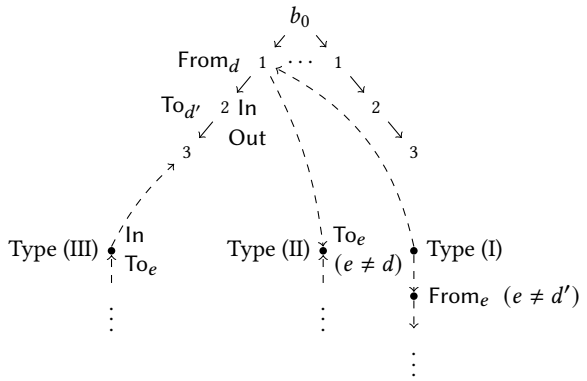

Figure 5: The instance I' and the three types of sinks

- a full $RRR^-R^-R^-RRR$-path attached to $s_{c,c',I,4}$.

The sink of Type (II) is also attached to value $b_{c,c',1}$. We add the following facts:

- $R(b_{c,c',1}, s_{c,c',II,5})$;
- $\mathsf{To}_e(s_{c,c',II,5})$ for all $e \in \mathsf{adom}(I) \setminus \{d'\}$;
- a full $RR^-R^-R^-RRR$-path attached to $s_{c,c',II,5}$.

The sink of Type (III) is attached to value $b_{c,c',3}$. We add the following facts:

- $R(s_{c,c',III,1}, b_{c,c',3})$;
- $\mathsf{In}(s_{c,c',III,1})$ and $\mathsf{To}_e(s_{c,c',III,1}))$ for all $e \in \mathsf{adom}(I)$;
- a full $R^-R^-RRRR^-R^-R^-RRR$-path attached to $s_{c,c',III,1}$.

This finishes the construction of I', which is illustrated in Figure 5. We make a few remark on sinks. It is important to realize which labels are *missing* on the first node in the sink. For sinks of Type I, the missing label is From_d, for sinks of Type II it is $\mathsf{To}_{d'}$, and for sinks of Type III it is Out. The latter, for example, is important to make sure that sinks of Type III will only accommodate the undesired successors that are due to the $\downarrow/\uparrow\downarrow$-issue (these are labeled with In), but not the desired successors (which are labeled with Out). The length and shape of the full paths that we attach in the third items is determined by which nodes in the gadget in Figure 4 we intend to map to the sink. These are nodes of depth 3 for sinks of Type (I), nodes of depth 5 for sinks of Type (II), and nodes of depth 1 for sinks of Type (III). A path of length 6 is not enough since, as described, the first node in the sink is missing some labels. But note that all attached full paths end in a path of length 6 with the pattern $R^-R^-R^-RRR$, and these are the actual sinks in the sense that anything can be mapped there.

We also note that the structure of I is not reflected by I', but rather by (the labels in) $\prod_{0\leq i\leq n} I'_i$. Instead, I' is merely a navigation gadget that is traversed by any simulation of $\prod_{0\leq i\leq n} I'_i$ in a systematic way. This is made more precise in the proof of the subsequent lemma, which asserts correctness of the reduction.

Lemma D.11. $(\prod_{1\leq i\leq n} I_i, \bar{c}) \preceq^{\downarrow} (I, c)$ *iff* $(\prod_{0\leq i\leq n} I'_i, \bar{c}') \preceq^{\uparrow\downarrow} (I', b_{c,c,3})$.[6]

[6]We could use any $b_{c',c,3}$ in place of $b_{c,c,3}$.

Proof. "if". Let S' be a $\uparrow\downarrow$-simulation that witnesses $(\prod_{0\leq i\leq n} I'_i, \bar{c}') \preceq^{\uparrow\downarrow} (I', b_{c,c,3})$. Define relation S by setting

$$S := \{(\bar{d}, d) \in \mathsf{adom}(\textstyle\prod_{1\leq i\leq n} I_i) \times \mathsf{adom}(I) \mid (c_0\bar{d}, b_{e,d,3}) \in S' \text{ for some } e\}.$$

It can be verified that S is a $\downarrow$-simulation from $\prod_{1\leq i\leq n} I_i$ to I. In fact, it is easy to use the definitions of the instances I'_i and I' to verify that Condition 1 of such simulations is satisfied.

For Condition 2, take an edge $R(\bar{d}, \bar{d}')$ in $\prod_{1\leq i\leq n} I_i$ and let $(\bar{d}, e) \in S$. Then $(c_0\bar{d}, b_{f,e,3}) \in S'$ for some f. The edge $R(\bar{d}, \bar{d}')$ gives rise to a corresponding gadget in $\prod_{0\leq i\leq n} I'_i$ that starts at $c_0\bar{d}$ and ends at $c_0\bar{d}'$, as shown in Figure 4. The gadget has multiple paths that branch at the end, one for each value in $\mathsf{adom}(I)$. Consider the path associated with $e \in \mathsf{adom}(I)$ and let us analyze the value in I' that S' simulates this path into, starting from the root that S' simulates into $b_{f,e,3}$.

The first node on the path is labeled Out and S' cannot simulate it into the sink of Type III attached in I' to $b_{f,e,3}$ since the first node in that sink is missing the Out label. Thus, the node is simulated into $b_{f,e,2}$. The second node on the path is then simulated into $b_{f,e,1}$. The third and fourth node, the latter labeled From_e, are best considered together. They cannot be simulated into the sink of Type I attached to $b_{f,e,1}$ because the fourth node is labeled From_e, but the second node in the sink is not. Consequently, the third node is simulated into b_0 and the fourth one into some $b_{e,e',1}$ as only such nodes are labeled From_e.

At the fourth node, the 'path' that we are following branches. We are interested in further following the branch on which the fifth node is labeled $\mathsf{To}_{e'}$. This node cannot be simulated into the sink of Type II attached to $b_{e,e',1}$ because the first node in that sink is not labeled $\mathsf{To}_{e'}$. It must consequently be mapped to $b_{e,e',2}$ which leaves S' with the only option of simulating the ending node of the gadget $c_0\bar{d}'$ to $b_{e,e',3}$. The definition of S thus yields $(\bar{d}', e') \in S$. Since the gadget contains a path that has labels From_e and $\mathsf{To}_{e'}$, by construction of the instances $I'_1, \ldots, I'_n$ we know that $R(e, e') \in I$. We have just shown that Condition 2 of $\downarrow$-simulations is satisfied.

"only if". Assume that $(\prod_{1\leq i\leq n} I_i, \bar{c}) \preceq^{\downarrow} (I, c)$ and let S be a witnessing $\downarrow$-simulation. To prove that $(\prod_{0\leq i\leq n} I'_i, \bar{c}') \preceq^{\uparrow\downarrow} (I', b_{c,c,3})$, it suffices to show that there is a homomorphism h' from the unraveling of $\prod_{0\leq i\leq n} I'_i$ at $\bar{c}'$ to I' with $h'(\bar{c}') = b_{c,c,3}$.

We define h' step by step, obtaining the desired homomorphism in the limit. Start with setting $h'(\bar{c}') = b_{c,c,3}$ and note that $b_{c,c,3}$ satisfies the same unary relations in I' that c satisfies in I.

We call a $p \in \mathsf{adom}(U')$ a *frontier point* if $h'(p)$ is defined and U' contains an edge $R(p, pRa)$ with $h'(pRa)$ undefined (the construction will ensure that then $h'(pRa)$ is undefined for *all* edges $R(p, pRa)$ in U'). We will maintain the invariant that if $h'(p)$ is defined, then

(∗) p (which is a path, c.f. the definition of unravelings) ends in a value that takes the form $c_0\bar{d}$ with $\bar{d} \in \mathsf{adom}(\prod_{1\leq i\leq n} I_i)$ and there is an $e \in \mathsf{adom}(I)$ with $h'(p) = b_{f,e,3}$ (for some f) and $(\bar{d}, e) \in S$.

We extend h' by repeatedly selecting frontier points p whose length as a path is as short as possible (for fairness). Let p end at b. By Invariant (∗), b takes the form $c_0\bar{d}$ with $\bar{d} \in \mathsf{adom}(\prod_{1\leq i\leq n} I_i)$ and

there is an $e \in \text{adom}(I)$ with $h'(p) = b_{f,e,3}$ (for some f) and $(\bar{d}, e) \in S$. Now consider every edge $R(\bar{d}, \bar{d}') \in \prod_{1\leq i\leq n} I_i$. The edge gives rise to a corresponding gadget in $\prod_{0\leq i\leq n} I'_i$ that starts at $c_0\bar{d}$ and ends at $c_0\bar{d}'$, as shown in Figure 4. This in turns gives rise to a subtree U in U' obtained from the gadget from Figure 4 by duplicating the point $\bar{c}'$ sufficiently many times so that a tree is obtained. In fact, we shall view this subtree as several subtrees, one for each successor of p or, in other words, one subtree for every label From_d. Since $(\bar{d}, e) \in S$, we find an $e' \in \text{adom}(I)$ such that $(\bar{d}', e') \in S$. We now extend h' to the mention subtree in the following way:

- h' maps the initial path in the subtree with the From_e label upwards in I' from $b_{f,e,3}$, to the root b_0, and then downwards again one step to $b_{e,e',1}$. At this point, the subtree branches. The branch labeled $\mathsf{To}_{e'}$ is mapped downwards the remaining two steps to $b_{e,e',3}$ and all other branches are mapped to the sink of Type II attached to $b_{e,e',1}$, and so are the entire subtrees of U' below them.
- h' maps all path with a From_g label, $g \neq e$, upwards from $b_{f,e,3}$ for two steps, and then down into the sink of Type I attached to $b_{f,e,1}$; the entire subtrees of U' below them are also mapped into that sink.

This does not yet complete the extension of h' as we have not necessarily covered all successors of p in U' with the above construction. We now treat the remaining ones. Consider every edge $R(\bar{d}', \bar{d}) \in \prod_{1\leq i\leq n} I_i$. The edge gives rise to a successor $pR^-c_0\bar{d}'$ of p in U'. We map this successor, as well as the entire subtree of U' below it, to the sink of Type III attached to $b_{f,e,3}$. □

D.10 Lower Bounds

We start with the proof of Theorem 5.20, which we split up into several theorems.

Theorem D.12. *Verification and existence of unique fitting tree CQs and bases of strongly most-general fitting tree CQs are ExpTime-hard. This holds already when the tree CQ / all tree CQs in the basis are promised to fit resp. when a fitting tree CQ is promised to exist.*

Proof. We prove the hardness results in Thm. D.12 simultaneously, by reduction from the product simulation problem into trees, see Thm. 5.19. Assume that we are given finite pointed instances $(I_1, a_1), \ldots, (I_n, a_n)$ and (J, b) with J a tree, and that we are to decide whether $\Pi_{1\leq i\leq n}(I_i, a_i) \preceq (J, b)$. Let $I_1, \ldots, I_n, J$ be formulated in schema $\mathcal{S}$.

We construct a collection of labeled examples $E = (E^+, E^-)$, as follows. Assume w.l.o.g. that $\text{adom}(I_i) \cap \text{adom}(J) = \emptyset$ for $1 \leq i \leq n$. Let R be a binary relation symbol that does not occur in $\mathcal{S}$. E^+ contains the instances $(I'_1, a'_1), \ldots, (I'_n, a'_n)$ where (I'_i, a'_i) is obtained by starting with (I_i, a_i) and adding the following facts:

(1) $R(a'_i, a_i)$;
(2) $R(a'_i, b)$ and all facts from J (we refer to this as the copy of J in I'_i).

Since (J, b) is a tree, we may view it as a tree CQ $q(x)$. Let $q'(x')$ be the tree CQ obtained from q by making x' the answer variable and adding the atom $R(x', x)$. It was shown in [11] that every tree CQ has a frontier w.r.t. tree CQs which can be computed in polynomial time. We may thus compute such a frontier $\mathcal{F} = \{p_1(x), \ldots, p_k(x)\}$ for q'. E^- contains the instances $(p_1, x), \ldots, (p_k, x)$. It is easy to verify that q' is a fitting for E.

To establish all of the results in the theorem, it remains to show the following:

(a) If $\Pi_{1\leq i\leq n}(I_i, a_i) \preceq (J, b)$, then q' is a unique fitting tree CQ for E;
(b) If $\Pi_{1\leq i\leq n}(I_i, a_i) \not\preceq (J, b)$, then E has no basis of strongly most-general fitting tree CQs.

For Point (a), assume that $\Pi_{1\leq i\leq n}(I_i, a_i) \preceq (J, b)$ and let $\widehat{q}(x)$ be a fitting tree CQ for E. We have to show that $\widehat{q} \preceq q'$ and $q' \preceq \widehat{q}$. The latter actually follows from the former since $\widehat{q} \preceq q'$ and $q' \not\preceq \widehat{q}$ would imply that $\widehat{q}$ simulates into some element of the frontier of q', thus into a negative example, in contradiction to $\widehat{q}(x)$ being a fitting for E.

To obtain $\widehat{q} \preceq q'$, in turn, it suffices to show $\Pi_{1\leq i\leq n}(I'_i, a'_i) \preceq q'$ since $\widehat{q}$ is a fitting and thus $\widehat{q} \preceq \Pi_{1\leq i\leq n}(I'_i, a'_i)$. Let S be a simulation witnessing $\Pi_{1\leq i\leq n}(I_i, a_i) \preceq (J, b)$. We obtain S' from S as follows:

(1) add $(\bar{a}', x')$ for $\bar{a}' = (a'_1, \ldots, a'_n)$;
(2) for all $\bar{a} \in \text{adom}(\Pi_{1\leq i\leq n}(I'_i, a'_i))$ that contain an element c of $\text{adom}(J)$, add $(\bar{a}, c)$.

It can be verified that S' is a simulation of $\Pi_{1\leq i\leq n}(I'_i, a'_i)$ in q'. We have thus shown that $\Pi_{1\leq i\leq n}(I'_i, a'_i) \preceq q'$, as desired.

For Point (b), let $\Pi_{1\leq i\leq n}(I_i, a_i) \not\preceq (J, b)$ and assume to the contrary of what we have to show that E has a complete basis of strongly most-general fitting tree CQs. By Lemma D.1, there is then also an $m \geq 1$ such that $(U_m, \bar{a}) \not\preceq (J, b)$ where U_m is the m-finite unraveling of $\Pi_{1\leq i\leq n}(I_i, a_i)$ and $\bar{a} = a_1 \cdots a_n$. Since $(U_m, \bar{a})$ is a tree, we may view it as a tree CQ $p'(z)$. For all $i \geq 0$, let $p'_i(y_0)$ be obtained from p' by making y_0 the answer variable and adding an initial R-zig-zag path, that is:

$$R(y_0, z_0), R(y_1, z_0), R(y_1, z_1), R(y_2, z_1), R(y_2, z_2), \ldots, R(y_i, z_{i-1}), R(y_i, z)$$

where $y_0, \ldots, y_i$ and $z_0, \ldots, z_{i-1}$ are fresh variables. We argue that each p'_i is a fitting for E. By construction, p'_i fits the positive examples. It does not fit any of the negative examples because any such example simulates into q' and thus we would obtain $p'_i \preceq q'$ and any simulation witnessing this would also show $U_m \preceq (J, b)$.

Since E has a basis of strongly most-general fitting tree CQs, there is some CQ $\widehat{q}$ that maps into infinitely many of the tree CQs p'_i. Since $\widehat{q}$ is connected and the length of the initial R-zig-zag path gets longer with increasing i, it follows that the query $\widehat{q}$ simulates into an R-zig-zag path, and thus into the simple CQ $q_0(x) \text{ :- } \exists y\, R(x, y)$. But then $\widehat{q}$ clearly simulates into a negative example, which is a contradiction. □

Theorem D.13. *Verification and existence of most-specific fitting tree CQs is ExpTime-hard. This holds already when the tree CQ / all tree CQs in the basis are promised to fit resp. when a fitting tree CQ is promised to exist.*

Proof. We again reduce from the product simulation problem into trees Assume that we are given finite pointed instances $(I_1, a_1), \ldots, (I_n, a_n)$ and (J, b) with J a tree, and that we are to decide whether $\Pi_{1\leq i\leq n}(I_i, a_i) \preceq (J, b)$.

We construct a collection of labeled examples $E = (E^+, E^-)$, as follows. Assume w.l.o.g. that $\text{adom}(I_i) \cap \text{adom}(J) = \emptyset$ for $1 \leq i \leq n$.

Let R be a fresh binary relation symbol and A_1, A_2, B_1, B_2 fresh unary relation symbols. E^+ contains instances $(I'_1, a'_1), \ldots, (I'_n, a'_n)$ where (I'_i, a'_i) is obtained by starting with (I_i, a_i) and adding the following facts:

(1) $R(a'_i, a_i)$;
(2) $R(a'_i, b)$ and all facts from J (we refer to this as the copy of J in I'_i);
(3) $R(a'_i, c_1), R(a'_i, c_3), R(c_2, c_1), R(c_2, c_3), A_1(a'_i), B_1(c_1), B_2(c_3), A_2(c_2)$ with c_1, c_2, c_3 fresh values;
(4) $R(b, d), A_1(b), A_2(b), B_1(d), B_2(d)$ with b, d fresh values.

The purpose of Point 3 above is to create, in the unraveling of I_i at a'_i, an infinite path that starts at a'_i. The path is obtained by traveling a'_i, c_2, c_4, c_3, a'_i, ad infinitum. It alternates between forward and backwards R-edges and the labeling of its nodes with $A_1, B_1, A_2, B_2, A_1, \ldots$ ensures that the path does not simulate into a finite prefix of itself. Also note that there is a simulation of the subinstance created in Point 3 in the subinstance created in Point 4.

Since (J, b) is a tree, we may view it as a tree CQ $q(x)$. It was shown in [11] that every tree CQ has a frontier w.r.t. tree CQs which can be computed in polynomial time. We may thus compute such a frontier $\mathcal{F} = \{p_1(x), \ldots, p_k(x)\}$ for q. E^- contains instances $(L_1, x'), \ldots, (L_k, x')$ where (L_i, x) is obtained by starting with (p_i, x) and adding the following facts:

(1) $R(x', x)$;
(2) $R(x, d), A_1(b), A_2(b), B_1(d), B_2(d)$ with b, d fresh values.

Note that Point 2 above creates the same gadget as Point 4 in the definition of I'_i.

Let $q'(x')$ be the tree CQ obtained from q by making x' the answer variable and adding the atoms

(1) $R(x', x)$;
(2) $R(x, y), A_1(y), A_2(y), B_1(y), B_2(y)$ with y a fresh variable.

Observe that q' is a fitting for E. In fact, it is clear by construction of E^+ that $q' \preceq (I'_i, a'_i)$ for all $(I'_i, a'_i) \in E^+$. Moreover, since $q \not\preceq p$ for any $p \in \mathcal{F}$ and the relation symbols in Point 2 of the definition of (L_i, x') do not occur in q, we also have $q' \not\preceq (L_i, x')$ for all $(L_i, x') \in E^-$.

To establish the theorem, it remains to show the following:

(a) If $\Pi_{1 \leq i \leq n}(I_i, a_i) \preceq (J, b)$, then q' is a strongly most-specific fitting for E;
(b) If $\Pi_{1 \leq i \leq n}(I_i, a_i) \not\preceq (J, b)$, then E has no most-specific fitting.

For Point (a), assume that $\Pi_{1 \leq i \leq n}(I_i, a_i) \preceq (J, b)$ and let $\widehat{q}$ be a fitting for E. We have to show that $\widehat{q} \preceq q'$. We first argue that $\Pi_{1 \leq i \leq n}(I'_i, a'_i) \preceq q'$. Let S be a simulation witnessing $\Pi_{1 \leq i \leq n}(I_i, a_i) \preceq (J, b)$. We obtain S' from S as follows:

(1) add $(\bar{a}', x')$ for $\bar{a}' = (a'_1, \ldots, a'_n)$;
(2) add $(\bar{a}, x)$ for all $\bar{a} \in \text{adom}(\Pi_{1 \leq i \leq n}(I'_i, a'_i))$ that contain only values a_i, b, and c_2;
(3) add $(\bar{a}, y)$ for all $\bar{a} \in \text{adom}(\Pi_{1 \leq i \leq n}(I'_i, a'_i))$ that contain only values c_1, c_3, and d;
(4) for all $\bar{a} \in \text{adom}(\Pi_{1 \leq i \leq n}(I'_i, a'_i))$ that contain an element c of $\text{adom}(J)$, add $(\bar{a}, c)$.

It can be verified that S' is a simulation of $\Pi_{1 \leq i \leq n}(I'_i, a'_i)$ in q'. In particular, every tuple $\bar{a} \in \text{adom}(\Pi_{1 \leq i \leq n}(I'_i, a'_i))$ that is reachable in $\Pi_{1 \leq i \leq n}(I'_i, a'_i)$ from $(a'_1, \ldots, a'_n)$ and contains any of the values c_1, c_2, c_3, d must be of one of the forms treated in Points 2 and 3 above. We have thus shown that $\Pi_{1 \leq i \leq n}(I'_i, a'_i) \preceq q'$. Together with $\widehat{q} \preceq \Pi_{1 \leq i \leq n}(I'_i, a'_i)$, which holds since $\widehat{q}$ is a fitting of E, we obtain $\widehat{q} \preceq q'$, as desired.

For Point (b), assume that $\Pi_{1 \leq i \leq n}(I_i, a_i) \not\preceq (J, b)$. By Lemma D.1, there is then also an $m \geq 1$ such that $(U_m, \bar{a}) \not\preceq (J, b)$ with U_m the m-finite unraveling of $\Pi_{1 \leq i \leq n} I_i$ at $\bar{a} = (a_1, \ldots, a_n)$. Let U'_m be U_m extended with fact $R(x, \bar{a})$ and let $\widehat{q}(x)$ be U'_m viewed as a tree CQ. By construction of E^+, $\widehat{q} \preceq (I'_i, a'_i)$ for all $(I'_i, a'_i) \in E^+$. Moreover, $\widehat{q} \not\preceq (L_i, x')$ for all $(L_i, x') \in E^-$ because otherwise from the composition of simulations witnessing $\widehat{q} \preceq (L_i, x')$ and $p_i \preceq (J, b)$ we may obtain a simulation witnessing $(U_m, \bar{a}) \not\preceq (J, b)$, a contradiction. Thus, $\widehat{q}$ is a fitting for E. For every $i \geq 1$, let $\widehat{q}_i$ be $\widehat{q}$ extended with a path on variables $\bar{b}, x_1, \ldots, x_i$ that alternates between forwards and backwards R-edges (starting with forwards) and is additionally labeled with atoms

$$A_1(x_0), B_1(x_1), A_2(x_2), B_2(x_3), A_1(x_4) \ldots$$

to achieve that it does not map into a finite prefix of itself, as described above. It is easy to verify that $\widehat{q}_i$ is a fitting for all $i \geq 1$. Clearly, $\widehat{q}_i \prec \widehat{q}_{i+1}$ for all $i \geq 1$. To finish the proof, it thus remains to show that there is no fitting p for E such that $\widehat{q}_i \preceq p$ for all $i \geq 1$. Assume to the contrary that there is such a $p(x)$. Then $p \preceq (I'_1, a'_1)$. Let h be a homomorphism from p to (I'_1, a'_1) with $h(x) = a'_1$. We must find a $R(x, y) \in p$ such that for infinitely many i, there is a homomorphism h_i from $\widehat{q}_i$ to p with $h_i(x) = x$ and $h_i(\bar{b}) = y$. Then $h(y) = a_i$ as the only other option $h(y) = b$ implies that $(U_m, \bar{a}) \preceq (J, b)$, which is not the case. Consequently, each $h_i \circ h$ maps the path $\bar{b}, x_1, \ldots, x_i$ in $\widehat{q}_i$ to the subinstance of I'_1 induced by the values a_i, c_1, c_2, c_3. But clearly there is no tree instance (resp. tree CQ) that admits a homomorphism from all paths $\bar{b}, x_1, \ldots, x_i$, $i \geq 1$, and also a homomorphism to this subinstance. □

Next, we prove a matching lower bound. The proof applies simultaneously to tree CQs and to arbitrary CQs.

Theorem D.14. *The existence problem is ExpTime-hard for weakly most-general fitting CQs and for weakly most-general fitting tree CQs.*

Proof. We adapt a proof of ExpTime-hardness of the simulation problem for concurrent transition systems from [28]. Specifically, we reduce from the word problem for alternating, linear space bounded Turing machines (TMs). It is well known that there is a fixed such TM whose word problem is ExpTime-complete [17]. Given a word w, we thus construct a collection of labeled examples (E^+, E^-) such that (E^+, E^-) permits a weakly most-general fitting CQ iff M does not accept w. The weakly most-general fitting CQ of (E^+, E^-) is always a tree CQ, therefore this reduction shows hardness of the existence problem for CQs as well as of the existence problem for tree CQs.

In [24], a similar adaptation of the reduction in [28] is used to show that deciding the existence of an arbitrary fitting $\mathcal{EL}$ concept / directed tree CQ is ExpTime-hard. In our adaptation, we need to be more careful since we are interested in the existence of weakly most-general fittings.

For our purposes, an *alternating Turing machine (ATM) M* is a tuple $M = (\Gamma, Q_\forall, Q_\exists, \mapsto, q_0, F_{\text{acc}}, F_{\text{rej}})$ consisting of a finite set of tape symbols Γ, a set of universal states $Q_\forall$, a set of existential states $Q_\exists$, a set of accepting states F_{acc}, a set of rejecting states F_{rej} an initial

state $q_0 \in Q_\forall$ and a transition relation $\mapsto \subseteq Q \times \Gamma \times Q \times \Gamma \times \{-1, 0, +1\}$. The last component of the transition relation that is either -1, 0 or $+1$ indicates the head of the TM moving to the left, staying at the same tape cell, and moving to the right, respectively. We assume that the sets $Q_\forall$, $Q_\exists$, F_{acc}, F_{rej} partition Q and refer to states in $F_{\text{acc}} \cup F_{\text{rej}}$ as *final* states. A configuration of M is *universal* if its state is universal, and likewise for *existential* configurations and *final* configurations. In our model of alternation, every existential or universal configuration has exactly two successor configurations and every final configuration has no successor configurations. Hence, we write $(q, a) \mapsto ((q_\ell, b_\ell, \Delta_\ell), (q_r, b_r, \Delta_r))$ to indicate that when M is in state $q \in Q_\forall \cup Q_\exists$ reading symbol a, it branches to "the left" with $(q_\ell, b_\ell, \Delta_\ell)$ and to "the right" with (q_r, b_r, Δ_r). These directions are not related to the movement of the head on the tape. Furthermore, we assume that $\mapsto$ alternates between existential states and universal states, that q_0 is a universal state, and that M always reaches a final state.

With each configuration that is reached by an alternating TM M on an input w, we associate an acceptance value of 1 or 0 as follows. Final configurations with an accepting state have acceptance value 1 and final configurations with a rejecting state have acceptance value 0. The acceptance value of a universal configuration is the minimum of the acceptance value of its two successors. The acceptance value of an existential configuration is the maximum of the acceptance value of its two successors. An alternating TM *accepts* input w if the initial configuration $q_0 w$ of M on w has acceptance value 1 and *rejects* w otherwise.

Let $M = (\Gamma, Q_\forall, Q_\exists, \mapsto, q_0, F_{\text{acc}}, F_{\text{rej}})$ be a fixed alternating TM with linear space bound $s(n)$. Given a word w with $|w| = n$, we construct pointed instances (I_i, c_i) for all i with $1 \le i \le s(n)$ to be used as positive examples and a pointed instance (J, c) to be used as the only negative example. As the schema, we use the unary relation symbols Reject, Accept and the binary relation symbols $r_{q,a,i}$ and $\ell_{q,a,i}$ for all $q \in Q$, $a \in \Gamma$ and i with $1 \le i \le s(n)$. What we want to achieve is that

(1) if M accepts w, then $(\prod_{1 \le i \le s(n)} I_i, c_1 \dots c_{s(n)}) \to (J, c)$ and thus there is no fitting CQ;
(2) if M rejects w, then the computation tree of M on w, defined in the usual way, describes a fitting tree CQ q; moreover, we can extract from q a weakly most-general CQ by dropping subtrees.

We start with the pointed instances (I_i, c_i). Each I_i uses the values a and (q, a) for all $q \in Q$ and $a \in \Gamma$ to represent the i-th tape cell of M. The value a represents that the head of M is not on cell i and that cell i contains the symbol a. The value (q, a) represents that the head of M is on cell i, that M is in state q and that the cell i contains the symbol a. The facts in each I_i ensure that $r_{q,a,i}(e, e')$ is true in the part of $\prod_{1 \le i \le s(n)} I_i$ that is reachable from the value $c_1 \dots c_{s(n)}$ iff in state q, reading symbol a and head at tape cell i, M branches right from the configuration represented by e to the configuration represented by e'. The same is true for the facts $\ell_{q,a,i}(e, e')$ and branching left.

For each transition $(q, a) \mapsto ((q_\ell, b_\ell, \Delta_\ell), (q_r, b_r, \Delta_r))$ of M, I_i contains the following facts:

(1) Facts that correspond to the head moving away from cell i:
$$\ell_{q,a,i}((q, a), b_\ell) \text{ if } \Delta_\ell \neq 0,$$
$$r_{q,a,i}((q, a), b_r) \text{ if } \Delta_r \neq 0.$$
(2) Facts that correspond to the head staying on cell i:
$$\ell_{q,a,i}((q, a), (q_\ell, b_\ell)) \text{ if } \Delta_\ell = 0,$$
$$r_{q,a,i}((q, a), (q_r, b_r)) \text{ if } \Delta_r = 0.$$
(3) Facts that correspond to the head moving onto cell i from cell $i-1$ or $i+1$. For all $b \in \Gamma$:
$$\ell_{q,a,i-1}(b, (q_\ell, b)) \text{ if } \Delta_\ell = +1,$$
$$r_{q,a,i-1}(b, (q_r, b)) \text{ if } \Delta_r = +1,$$
$$\ell_{q,a,i+1}(b, (q_\ell, b)) \text{ if } \Delta_\ell = -1,$$
$$r_{q,a,i+1}(b, (q_r, b)) \text{ if } \Delta_r = -1.$$
(4) Facts that correspond to the transition not modifying the cell i. For all $j \neq i$ with $1 \le j \le s(n)$:
$$\ell_{q,a,j}(b, b) \text{ if } \Delta_\ell = +1 \text{ and } j \neq i - 1,$$
$$r_{q,a,j}(b, b) \text{ if } \Delta_r = +1 \text{ and } j \neq i - 1,$$
$$\ell_{q,a,j}(b, b) \text{ if } \Delta_\ell = -1 \text{ and } j \neq i + 1,$$
$$r_{q,a,j}(b, b) \text{ if } \Delta_r = -1 \text{ and } j \neq i + 1,$$
$$\ell_{q,a,j}(b, b) \text{ if } \Delta_\ell = 0,$$
$$r_{q,a,j}(b, b) \text{ if } \Delta_r = 0.$$

Additionally, I_i includes the following unary atoms for all $a \in \Gamma$ to mark accepting and rejecting final configurations:

$$\text{Reject}((q, a)), \text{ for all } q \in F_{\text{rej}},$$
$$\text{Reject}(a),$$
$$\text{Accept}((q, a)), \text{ for all } q \in F_{\text{acc}},$$
$$\text{Accept}(a).$$

Note that $\prod_{1 \le i \le s(n)} I_i$ contains the fact $\text{Accept}(e)$ iff e represents a configuration in an accepting state, similarly for $\text{Reject}(e)$ and rejecting states. We do not treat the cases $i = 1$ and $i = s(n)$ in a special way since we can assume that M does not move its head beyond tape cell 1 or $s(n)$. This completes the description of the instances I_i. We choose the values c_i such that the value $c_1 \dots c_{s(n)} \in \text{adom}(\prod_{1 \le i \le s(n)} I_i)$ represents the initial configuration of M on w. For input $w = a_1 \dots a_n$ and all i with $1 \le i \le s(n)$ we choose

$$c_i = \begin{cases} (q_0, a_1) & \text{if } i = 1 \\ a_i & \text{if } 2 \le i \le n \\ \beta & \text{otherwise} \end{cases}$$

where $\beta \in \Gamma$ is the symbol for an empty tape cell.

Next, we describe the negative example (J, c). Informally, the instance J together with the choice of $c \in \text{adom}(J)$ encodes that a computation is accepting. For that, $\text{adom}(J)$ contains the two values 0 and 1 for final configurations, the values $(\forall, 0, 0, 0)$, $(\forall, 1, 0, 0)$, $(\forall, 0, 1, 0)$, $(\forall, 1, 1, 1)$ $(\exists, 0, 0, 0)$, $(\exists, 1, 0, 1)$, $(\exists, 0, 1, 1)$, $(\exists, 1, 1, 1)$ as well as the two "sink"-values s_1 and s_2. A value of the form $(\forall, \ell, r, v)$ represents that a configuration is in a universal state, that the left successor configuration has acceptance value ℓ and that the right successor configuration has acceptance value r and the configuration hence has acceptance value v, and similarly for values of

the form $(\exists, \ell, r, v)$. Reflecting this intuition, J includes the following facts for all $q \in Q$, $a \in \Gamma$, i with $1 \leq i \leq s(n)$, and $(*, \ell, r, v), (*', \ell', r', v') \in \mathrm{adom}(J)$ with $* \neq *'$:

- $r_{q,a,i}((*, \ell, r, v), (*', \ell', r', v'))$ if $v' = r$, and
- $\ell_{q,a,i}((*, \ell, r, v), (*', \ell', r', v'))$ if $v' = \ell$.

Reflecting the acceptance behavior of final configurations, J additionally includes the facts

$$\mathsf{Reject}(0), \mathsf{Accept}(1)$$

as well as the following facts, for all $q \in Q, a \in \Gamma, i$ with $1 \leq i \leq s(n)$, and $(*, \ell, r, v) \in \mathrm{adom}(J)$:

$$r_{q,a,i}((*, \ell, r, v), r) \text{ and } \ell_{q,a,i}((*, \ell, r, v), \ell).$$

At this point, we have completely described the computational behavior of M. If we stopped here, however, then a weakly most-general fitting CQ would *never* exist, no matter whether M accepts w or not. We thus extend J with the following facts which ensure that if there is a fitting CQ at all, then there is a weakly most-general fitting CQ:

- $r_{q,a,i}(s_1, e)$ and $\ell_{q,a,i}(s_1, e)$ for all $e \in \mathrm{adom}(J)$,
- $r_{q,a,i}(e, s_2)$ and $\ell_{q,a,i}(e, s_2)$ for all $e \in \mathrm{adom}(J) \setminus \{s_1, s_2\}$,
- $r_{q,a,i}((\exists, 1, 0, 1), s_1)$, $\ell_{q,a,i}((\exists, 0, 1, 1), s_1)$,
- $r_{q,a,i}((\exists, 1, 0, 1), (\forall, 1, 1, 1))$, $\ell_{q,a,i}((\exists, 0, 1, 1), (\forall, 1, 1, 1))$, and
- $\mathsf{Accept}(s_1)$, $\mathsf{Reject}(s_1)$

for all $q \in Q$, $a \in \Gamma$ and i with $1 \leq i \leq s(n)$. This completes the construction of J. We choose $c = (\forall, 1, 1, 1)$ and set $E = (E^+, E^-)$ with $E^+ = \{(I_i, c_i) \mid 1 \leq i \leq s(n)\}$ and $E^- = \{(J, c)\}$.

It remains to show that the reduction is correct, that is:

$$M \text{ rejects } w \text{ iff } E \text{ has a weakly most-general fitting CQ.}$$

First, assume that M accepts w. To show that E has no weakly most-general fitting CQ, it suffices to show that $(\prod_{1 \leq i \leq s(n)} I_i, c_1 \dots c_{s(n)}) \to (J, c)$.

Let $I = \prod_{1 \leq i \leq s(n)} I_i$ and $\bar{c} = c_1 \dots c_{s(n)}$. We define a homomorphism h from I to J with $h(\bar{c}) = c$. If $e \in \mathrm{adom}(I)$ is not reachable from $\bar{c}$, set $h(e) = s_1$. If $e \in \mathrm{adom}(I)$ is reachable from $\bar{c}$, then it describes a configuration of M that appears in the computation of M on w. In the following, we will not distinguish values reachable from $\bar{c}$ in I and configurations of M. Let v be the acceptance value associated with e. If e is a final configuration, set $h(e) = v$. If e is an existential or a universal configuration, then e must have a left successor and a right successor. Let ℓ be the acceptance value of the left successor and r the acceptance value of the right successor of e. Set $h(e) = (\forall, \ell, r, v)$ if e is universal and $h(e) = (\exists, \ell, r, v)$ if e is existential.

To verify that h is as required, first note that $h(\bar{c}) = (\forall, 1, 1, 1) = c$ as M accepts w and $\bar{c}$ is a universal configuration. Then, let $r_{q,a,i}(e, e')$ be a fact in I that is reachable from $\bar{c}$ with e a universal configuration. The case of facts $\ell_{q,a,i}$ and of existential configurations is similar. Then $h(e) = (\forall, \ell, r, v)$ and $h(e') = (\exists, \ell', r', v')$ for some $\ell, r, v, \ell', r', v'$ with $v' = r$ by definition of computations of M and definition of h. Hence, $r_{q,a,i}(h(e), h(e')) \in J$ by construction. Thus, h is a homomorphism as required.

For the other direction, assume that M rejects w. From the computation of M on w we construct a CQ q that is a weakly most-general fitting of E. Informally, q will be a smallest subset of the unraveling of the computation of M on w that still witnesses that M rejects w.

For defining q formally, we first introduce the notion of a minimal path of the computation of M on w. A *path* of the computation of M on w is a sequence $p = e_1 d_1 \dots d_{n-1} e_n$ where e_1 is the initial configuration of M on w and for all i, $d_i = r$ if e_i has right successor e_{i+1} and $d_i = l$ if e_i has left successor e_{i+1}. We define $\mathrm{tail}(e_1 d_1 \dots d_{n-1} e_n) = e_n$. A path in the computation of M on w is *minimal* if for all i, e_i has acceptance value 0 and if e_i is a universal configuration and has a left successor with acceptance value 0, then $d_i = \ell$.

Now, let $q(e_1)$ be the unary CQ that contains the following atoms for all minimal paths p, p' of the computation of M on w:

- $r_{q,a,i}(p, p')$ if $p' = pre$ and $\mathrm{tail}(p)$ is a configuration with state q and head at tape cell i, which contains a.
- $\ell_{q,a,i}(p, p')$ if $p' = ple$ and $\mathrm{tail}(p)$ is a configuration with state q and head at tape cell i, which contains a.
- $\mathsf{Reject}(p)$ if $\mathrm{tail}(p)$ is a configuration in a rejecting state

Note that q is finite due to the assumption that M always terminates. By construction, it is a tree CQ and fg-connected. By Definition B.14 and Proposition B.16, q therefore has a frontier consisting of a single query $F(q)$. To prove that $q(e_1)$ is a weakly most-general fitting CQ for E, it thus remains to show that $q(e_1) \to (I, \bar{c})$, $q(e_1) \not\to (J, c)$ and $F(q) \to (J, c)$.

We begin with $q(e_1) \to (I, \bar{c})$. Recall that by construction of I, the values that are reachable from $\bar{c}$ represent configurations of the computation of M on w and that $\bar{c}$ corresponds to the initial configuration of M on w. We can thus construct a homomorphism h from q to I with $h(e_1) = \bar{c}$ by setting $h(p) = \mathrm{tail}(p)$ for all $p \in \mathrm{var}(q)$.

Next, we show that $q(e_1) \not\to (J, c)$. Recall that $c = (\forall, 1, 1, 1)$. For all $p \in \mathrm{var}(q)$, we use $q_p(p)$ to denote the restriction of q to all paths that start with p and has answer variable p. We show that $q_p(p) \not\to (J, c)$ for all $p \in \mathrm{var}(q)$ if $\mathrm{tail}(p)$ is universal or final, by induction on the depth of tree CQ $q_p(p)$. The desired result $q(e_1) \not\to (J, c)$ then follows for $p = e_1$. In the induction start, let $q_p(p)$ be of depth 0. Then $\mathrm{tail}(p)$ must be final by construction of q and hence $q_p(p) = \mathsf{Reject}(p)$. It follows that $q_p(p) \not\to (J, c)$.

Next, let $q_p(p)$ have depth > 0, with $\mathrm{tail}(p)$ universal, and assume that the statement holds for all $q'_p(p')$ of smaller depth. By construction of q $\mathrm{tail}(p) = e$ has acceptance value 0. Thus, there must be an atom $r_{q,a,i}(p, pre')$ or $\ell_{q,a,i}(p, ple')$ in q_p and e' must be existential or final. Assume that $r_{q,a,i}(p, pre')$ is in q_p, the other case is similar. If e' is final, it must be rejecting and hence q_p contains $\mathsf{Reject}(pre')$, implying that $q_p(p) \not\to (J, c)$. If e' is in an existential state, then by construction q_p must contain both atoms $r_{q,a,i}(pre', p'_r)$ for $p'_r = pre're''$ and $\ell_{q,a,i}(pre', p'_\ell)$ for $p'_\ell = pre'le'''$. For both $p' = p'_r$ and $p' = p'_\ell$, we have $q_{p'} \not\to (J, 1)$, $q_{p'} \not\to (J, s_2)$ since 1 and s_2 do not have outgoing edges in J, and $q_{p'} \not\to (J, c)$ by the induction hypothesis. Consequently, $q_{pre'}(pre') \not\to (J, (\exists, 1, 1, 1))$, $q_{pre'}(pre') \not\to (J, (\exists, 1, 0, 1))$, and $q_{pre'}(pre') \not\to (J, (\exists, 0, 1, 1))$, implying that $q_p(p) \not\to (J, (\forall, 1, 1, 1))$, as required.

It remains to show $F(q) \to (J, c)$. For that, recall that by Definition B.14, the answer variable of $F(q)$ is e_1 and the existential variables of $F(q)$ are u_{e_1} and $u_{(p,f)}$ for all minimal paths p and

atoms $f \in q$ such that p occurs in f. A variable of the latter kind is a *replica* of p. We construct a homomorphism h from $F(q)$ to J with $h(e_1) = c = (\forall, 1, 1, 1)$. Start by setting $h(e_1) = (\forall, 1, 1, 1)$ and $h(u_{e_1}) = s_1$. Now let $u_{(p,f)}$ be a replica of the variable p of q.

If $\text{tail}(p)$ is rejecting, then p occurs in exactly two atoms in q: $f_1 = d_{q,a,i}(p', p)$ for some $d \in \{r, \ell\}$ and $f_2 = \text{Reject}(p)$. Set $h(u_{(p,f_1)}) = 0$ and $h(u_{(p,f_2)}) = s_2$.

If $\text{tail}(p)$ is existential, then p occurs in exactly three atoms in q: $f_1 = d_{q,a,i}(p', p)$ for some $d \in \{r, \ell\}$, $f_2 = r_{q',a',i'}(p, pre_r)$ and $f_3 = \ell_{q',a',i'}(p, ple_\ell)$. Set $h(u_{(p,f_1)}) = s_1$, $h(u_{(p,f_2)}) = (\exists, 0, 1, 1)$ and $h(u_{(p,f_3)}) = (\exists, 1, 0, 1)$.

If $\text{tail}(p)$ is universal and not e_1, then p occurs in exactly two atoms in q: $f_1 = d_{q,a,i}(p', p)$ for some $d \in \{r, \ell\}$ and $f_2 = d'_{q',a',i'}(p, pd'e)$ for some $d' \in \{r, \ell\}$. Set $h(u_{(p,f_1)}) = s_1$ and $h(u_{(p,f_2)}) = (\forall, 1, 1, 1)$.

To verify that h is a homomorphism, consider an atom $r_{q,a,i}(p, p')$ in q, let $u_{(p,f)}$ be a replica of p in $F(q)$ and let $u_{(p',f')}$ be a replica of p' in $F(q)$. The case for $\ell_{q,a,i}$ atoms is symmetrical.

If $\text{tail}(p)$ is a universal configuration, then by definition of h, $h(u_{(p,f)}) \in \{s_1, (\forall, 1, 1, 1)\}$ and $h(u_{(p',f')}) \in \{0, s_1, s_2, (\exists, 1, 0, 1), (\exists, 0, 1, 1)\}$, since $\text{tail}(p')$ must be existential or final. By construction of J, $r_{q,a,i}(h(u_{(p,f)}), h(u_{(p',f')})) \notin J$ implies that $h(u_{(p,f)}) = (\forall, 1, 1, 1)$ and $h(u_{(p',f')}) = s_1$ or $h(u_{(p',f')}) = 0$. In both cases the definitions of h and q imply $f = f'$ and hence $r_{q,a,i}(u_{(p,f)}, u_{(p',f')}) \notin F(q)$ by construction of $F(q)$. Note that this case also applies to $p = e_1$, where $u_{(p,f)}$ is either p or u_p and the fact f is uniquely determined.

If $\text{tail}(p)$ is an existential configuration, then by definition of h, $h(u_{(p,f)}) \in \{s_1, (\exists, 1, 0, 1), (\exists, 0, 1, 1)\}$ and $h(u_{(p',f')}) \in \{0, s_1, s_2, (\forall, 1, 1, 1)\}$, since $\text{tail}(p')$ must be universal or final. By construction of J, $r_{q,a,i}(h(u_{(p,f)}), h(u_{(p',f')})) \notin J$ implies that $h(u_{(p,f)}) = (\exists, 0, 1, 1)$ and $h(u_{(p',f')}) = s_1$ or $h(u_{(p',f')}) = 0$. In both cases the definitions of h and q imply $f = f'$ and hence $r_{q,a,i}(u_{(p,f)}, u_{(p',f')}) \notin F(q)$ by construction of $F(q)$.

Hence, h is a homomorphism as required. □

Theorem 5.21. *For all $n \geq 0$, there is a collection of labeled examples of combined size polynomial in n such that a fitting tree CQ exists and the size of every fitting tree CQ is at least 2^{2^n}. This even holds for a fixed schema.*

Proof. The following construction extends the one used in the proof of Thm. 3.25 with *branching* to force every fitting tree CQ to have double exponential size. Let A be the unary relation of the schema and R, L the binary relations.

First, we describe the positive examples which each will consist of a cycle of prime length where two facts, an R fact and an L fact, connect an element of the cycle to the next one. Formally, for $j \geq 1$, let D_j denote the instance with domain $\{0, \ldots, j-1\}$ and the following facts:

- $R(k, k+1), L(k, k+1)$ for all $k < j-1$,
- and $R(j-1, 0), L(j-1, 0), A(j-1)$.

For $i \geq 1$, let p_i denote the i-th prime number (where $p_1 = 2$). Note that by the prime number theorem, D_{p_i} is of size $O(i \log i)$.

For the negative examples, construct the instance I with domain $\{00, 01, 10, 11, b\}$ and the following facts:

- $R(00, a), L(00, a)$ for all $a \in \{00, 01, 10\}$,
- $L(10, 11)$, and $R(10, a)$ for all $a \in \{00, 01, 10\}$,
- $R(01, 11)$, and $L(01, a)$ for all $a \in \{00, 01, 10\}$,
- $R(b, b), L(b, b), A(b)$, and $R(b, a), L(b, a)$ for all $a \in \{00, 01, 10\}$,
- $L(11, 11), R(11, 11), A(11)$.

To establish the result of the theorem, we will show that there is a tree CQ that fits the examples $E_n^+ = \{D_{p_i} \mid i = 1, \ldots, n\}$ and $E_n^- = \{(I, a) \mid a \in \{00, 01, 10\}\}$, and that every fitting CQ has size at least 2^{2^n}. For this we will talk about a tree CQ q as if it were a tree, i.e. use the notions of successors and predecessors of a variable as well as subtree below a variable. Additionally, we will refer to a binary tree where A is holds at every leaf and every non-leaf has exactly one direct L successor and one direct R successor as an L, R, A-tree. We say that a tree CQ $q(x)$ contains an L, R, A-tree if there is a subset of atoms of q that is an L, R, A-tree rooted at x. The following claim holds for the negative examples:

Claim. Let q be a tree CQ over the schema $\{L, R, A\}$. If q does not contain an L, R, A-tree, then $q \preceq (I, a)$ for some $a \in \{00, 10, 01\}$.

Proof of the claim. We show this claim by induction on the height of $q(x)$. In the induction start q has height 0. If q contains no L, R, A tree, then q does not contain $A(x)$ and therefore $q \preceq (I, a)$ for any $a \in \{00, 10, 01\}$. Now let the claim hold for all tree CQs of height at most i and let q be a tree CQ of height $i+1$. If q contains no L, R, A-tree, then there is a $P \in \{L, R\}$ such that there is no direct P successor x' of x that contains an L, R, A-tree. Consider the case $P = L$, the case $P = R$ is analogous. Then there is a simulation S that witnesses $q \preceq (I, 01)$, that can be constructed as follows: Start with $(x, 01) \in S$. If q contains the atom $R(x, x')$ for some x', map x' and the entire subtree below x' to 11. If q contains the atom $L(x, x')$, then, by the induction hypothesis, there is an $a \in \{00, 01, 10\}$ and a simulation from the subtree below x' to (I, a). Extend S to the subtree below x' according to this simulation. If q contains $R(x', x)$ or $L(x', x)$ map x' and the entire subtree below x' to b. This completes the construction of S and the proof of the claim.

Now, let q be the full binary L, R, A-tree of depth $(\prod_{i=1}^n p_i) - 1$. Observe that $\prod_{i=1}^n D_{p_i}$ is a double-linked cycle of size $\prod_{i=1}^n p_i > 2^n$ and A is only true at the *last* element. Therefore, $q \preceq D_{p_i}$ for all $i = 1, \ldots, n$ and it can be shown that $q \not\preceq (I, a)$ for all $a \in \{00, 01, 10\}$.. Thus, q is a fitting tree CQ. The query q is even a weakly most-general fitting of E^+ and E^-, as every element of its frontier no longer contains an L, R, A-tree.

Let p be any fitting tree CQ over the schema. By the property of I shown in the claim, p must contain a L, R, A-tree, but since $p \preceq D_{p_i}$ for all $i = 1, \ldots, n$ every A in this L, R, A-tree must have distance $(\prod_{i=1}^n p_i) - 1 > 2^n$ from the root. Hence, p must at least have size 2^{2^n}. □