Data complexity of finite query entailment in description logics with transitive roles

Jakub Kuklis
jk.kuklis@gmail.com

University of Warsaw, 02-097 Warszawa, Poland

Abstract. We study the problem of finite ontology-mediated query answering (FOMQA), the variant of OMQA where the represented world is assumed to be finite, and thus only finite models of the ontology are considered. We adapt the setting of unions of conjunctive queries and ontologies expressed in description logics with transitive role declarations. FOMQA is already undecidable for $\mathcal{SHOIF}$, but was proved to be 2Exp-complete for $\mathcal{SOI}$, $\mathcal{SOF}$ and $\mathcal{SIF}$. The proof relies on the fact that any finite counter-model can be transformed to a counter-model of certain regularity. By adjusting this transformation, we are able to provide an upper bound on the size of the smallest finite counter-model for $\mathcal{SOI}$ and $\mathcal{SOF}$; the bound is linear in the size of the ABox. This lets us show that for these logics FOMQA is in coNP with respect to data complexity; as coNP-hardness follows from previous results on weaker logics, we deduce coNP-completeness. [1]

Keywords: FOMQA · Data complexity · Counter-models · $\mathcal{SOI}$· $\mathcal{SOF}$

1 Introduction

Evaluating queries in the presence of background knowledge has been extensively studied in several communities. A particularly prominent take on this problem is ontology mediated query answering (OMQA) where background knowledge represented by an ontology is leveraged to infer more complete answers to queries [3]. A widely accepted family of ontology languages with varying expressive power is offered by Description Logics (DLs) [2], while the most commonly studied query language is that of (unions of) conjunctive queries.

Identifying ontology languages, for which query evaluation scales to large amounts of instance data, is one of the key research goals, as data tends to dominate in size both the ontology used and the evaluated queries. An early reference on *data complexity* in DLs, that is, complexity in the settings with fixed TBox and query, is showing coNP-hardness of instance queries in $\mathcal{ALE}$ [13]. In the much more expressive DL $\mathcal{SHIQ}$, a coNP upper bound was obtained for instance queries [8] and conjunctive queries without transitive roles [10], and later

[1] Copyright © 2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

generalized to any CQs [5]. The classical approach to avoiding intractability is to replace $\mathcal{ALC}$ and $\mathcal{SHIQ}$ with less expressive Horn DLs. These often admit query evaluation in PTime, examples include a variety of logics from the $\mathcal{EL}$ [1] and DL-Lite families [4], as well as Horn-$\mathcal{SHIQ}$ [8]. An alternative approach is to study complexity with respect to fixed TBoxes only [9].

Often, the intended models of the ontology are finite and this additional assumption allows to infer more properties: *finite ontology mediated query answering (FOMQA)* is the variant of OMQA restricted to finite models. For some logics, like $\mathcal{ALC}$, the finite variant and the unrestricted variant of the problem coincide; we then say that OMQA is *finitely controllable.* Studying FOMQA is interesting in settings lacking finite controllability. This is the case not only for DLs lacking the *finite model property* (e.g., DLs allowing both inverse roles and number restrictions), but also for logics allowing *transitive role declarations*, like logics from the $\mathcal{S}$ family. FOMQA is undecidable for $\mathcal{SHOIF}$ ontologies [11], but recently its decidability has been shown for three proper $\mathcal{SOIF}$ fragments: $\mathcal{SOI}$, $\mathcal{SOF}$, and $\mathcal{SIF}$ [6]. For all three fragments, the problem turned out to be 2Exp-complete in terms of combined complexity, but its data complexity was not studied. The present paper aims to fill this gap, thus initiating the study of the data complexity of FOMQA.

In [6], FOMQA is reduced to query entailment over a certain class of tree-like models recognisable by tree automata. Here, we adjust and adapt this approach to obtain the *linear-size counter-model property* for $\mathcal{SOI}$ and $\mathcal{SOF}$ (with respect to fixed TBox and query). This lets us derive an algorithm for FOMQA, which is coNP with respect to data complexity. The matching lower bound can be obtained using finite controllability from the results on answering instance queries, which is coNP-hard even for the $\mathcal{ALE}$ fragment of $\mathcal{ALC}$ [13] (see also [2]).

2 Preliminaries

The DL $\mathcal{SOIF}$ extends the classical DL $\mathcal{ALC}$ with transitivity declarations on roles ($\mathcal{S}$), nominals ($\mathcal{O}$), inverses ($\mathcal{I}$), and role functionality declarations ($\mathcal{F}$) [2]. We assume a signature of countably infinite disjoint sets of *concept names* $\mathsf{N_C} = \{A_1, A_2, \dots\}$, *role names* $\mathsf{N_R} = \{r_1, r_2, \dots\}$ and *individual names* $\mathsf{N_I} = \{a_1, a_2, \dots\}$. *$\mathcal{SOIF}$-concepts* C, D are defined by the grammar:

$$C, D ::= \top \mid A \mid \neg C \mid C \sqcap D \mid \{a\} \mid \exists r.C\,,$$

where $r \in \mathsf{N_R} \cup \{s^- \mid s \in \mathsf{N_R}\}$ is a *role.* Roles of the form r^- are called *inverse roles.* A *$\mathcal{SOIF}$ TBox $\mathcal{T}$* is a finite set of *concept inclusions (CIs) $C \sqsubseteq D$*, *transitivity declarations* $\mathsf{Tr}(r)$, *functionality declarations* $\mathsf{Fn}(r)$, where C, D are $\mathcal{SOIF}$-concepts and r is a role. We assume that if the TBox contains $\mathsf{Tr}(r)$, then it contains neither $\mathsf{Fn}(r)$ nor $\mathsf{Fn}(r^-)$. With an appropriate extension of the signature, each $\mathcal{SOIF}$ TBox can be transformed into an equivalent TBox with CIs in the following normal forms:

$$\bigsqcap A_i \sqsubseteq \bigsqcup B_j\,, \quad A \equiv \{a\}\,, \quad A \sqsubseteq \forall r.B\,, \quad A \sqsubseteq \exists r.B\,,$$

where A, A_i, B, B_j are concept names, empty conjunction is equivalent to $\top$ and empty disjunction to $\bot$. We assume that all knowledge bases we work with are normalized; normalization yields at most linearly larger knowledge base, so the size estimations still hold. We also assume that for each concept name A used in $\mathcal{T}$ there is a *complementary* concept name $\bar{A}$ axiomatised using CIs $\top \sqsubseteq A \sqcup \bar{A}$ and $A \sqcap \bar{A} \sqsubseteq \bot$.

$\mathcal{SOI}$ and $\mathcal{SOF}$ TBoxes are restrictions of $\mathcal{SOIF}$ TBoxes. $\mathcal{SOI}$ TBoxes do not contain functionality declarations, whereas concept inclusions in $\mathcal{SOF}$ do not contain inverse roles. As the inverse of a transitive role is transitive anyway, for $\mathcal{SOI}$ we shall assume that if $\mathsf{Tr}(r)$ is present in the TBox, then so is $\mathsf{Tr}(r^-)$.

An *ABox* is a finite set of *concept* and *role* assertions of the form $A(a)$ and $r(a, b)$, where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$ and $\{a, b\} \subseteq \mathsf{N_I}$. A *knowledge base (KB)* is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. We write $|\mathcal{K}|$ for $|\mathcal{A}| + |\mathcal{T}|$. We use $\mathsf{CN}(\mathcal{K})$, $\mathsf{Rol}(\mathcal{K})$, $\mathsf{TRol}(\mathcal{K})$, $\mathsf{Nom}(\mathcal{K})$, and $\mathsf{Ind}(\mathcal{K})$ to denote, respectively, the set of *all concept names, roles, transitive roles, nominals, and individuals occurring in* $\mathcal{K}$. Each nominal is an individual, so $\mathsf{Nom}(\mathcal{K}) \subseteq \mathsf{Ind}(\mathcal{K})$.

A *unary type* is a subset of $\mathsf{CN}(\mathcal{K})$ that contains exactly one of the concept names $A, \bar{A}$ for each $A \in \mathsf{CN}(\mathcal{K})$. We let $\mathsf{Tp}(\mathcal{K})$ denote the set of all unary types.

The semantics is defined via interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ with a non-empty *domain* $\Delta^{\mathcal{I}}$ and an *interpretation function* $\cdot^{\mathcal{I}}$ assigning to each $A \in \mathsf{CN}(\mathcal{K})$ a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and to each role name r with $r \in \mathsf{Rol}(\mathcal{K})$, a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The interpretation of complex concepts and roles is defined as usual [2]. We only consider interpretations complying with the *standard name assumption* and set $a^{\mathcal{I}} = a$ for each individual name a used in ABox or TBox. We write $d \in \mathcal{I}$ as a shorthand for $d \in \Delta^{\mathcal{I}}$.

Interpretation $\mathcal{I}$ is a *subinterpretation* of interpretation $\mathcal{J}$, written as $\mathcal{I} \subseteq \mathcal{J}$, if $\Delta^{\mathcal{I}} \subseteq \Delta^{\mathcal{J}}$, $A^{\mathcal{I}} \subseteq A^{\mathcal{J}}$, and $r^{\mathcal{I}} \subseteq r^{\mathcal{J}}$ for all $A \in \mathsf{CN}(\mathcal{K})$, $r \in \mathsf{Rol}(\mathcal{K})$. An interpretation $\mathcal{I}$ is a subinterpretation of $\mathcal{J}$ *induced* by $\Delta_0 \subseteq \Delta^{\mathcal{J}}$, written as $\mathcal{I} = \mathcal{J} \restriction \Delta_0$, if $\Delta^{\mathcal{I}} = \Delta_0$, $A^{\mathcal{I}} = A^{\mathcal{J}} \cap \Delta_0$, and $r^{\mathcal{I}} = r^{\mathcal{J}} \cap \Delta_0 \times \Delta_0$ for all $A \in \mathsf{CN}(\mathcal{K})$, $r \in \mathsf{Rol}(\mathcal{K})$. We write $\mathcal{J} \setminus X$ for the subinterpretation of $\mathcal{J}$ induced by $\Delta^{\mathcal{J}} \setminus X$.

An interpretation $\mathcal{I}$ *satisfies* $\alpha \in \mathcal{T} \cup \mathcal{A}$, written as $\mathcal{I} \models \alpha$, if the following holds: if α is a CI $C \sqsubseteq D$ then $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, if α is a transitivity declaration $\mathsf{Tr}(r)$ then $r^{\mathcal{I}}$ is transitive, if α is a functionality declaration $\mathsf{Fn}(r)$ then $r^{\mathcal{I}}$ is a partial function, if α is an assertion $A(a)$ then $a \in A^{\mathcal{I}}$, and if α is an assertion $r(a, b)$ then $(a, b) \in r^{\mathcal{I}}$. Finally, $\mathcal{I}$ is a *model* of: a TBox $\mathcal{T}$, denoted $\mathcal{I} \models \mathcal{T}$, if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{T}$; an ABox $\mathcal{A}$, denoted $\mathcal{I} \models \mathcal{A}$, if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{A}$; and a KB $\mathcal{K}$ if $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$.

Let $\mathcal{I}$ and $\mathcal{J}$ be interpretations of $\mathcal{K}$. A *homomorphism* from $\mathcal{I}$ to $\mathcal{J}$, written as $h : \mathcal{I} \to \mathcal{J}$ is a function $h : \Delta^{\mathcal{I}} \to \Delta^{\mathcal{J}}$ that preserves roles, concepts, and individual names; that is, $(h(d), h(d')) \in r^{\mathcal{J}}$ whenever $(d, d') \in r^{\mathcal{I}}$, $r \in \mathsf{Rol}(\mathcal{K})$, $h(d) \in A^{\mathcal{J}}$ whenever $d \in A^{\mathcal{I}}$, $A \in \mathsf{CN}(\mathcal{K})$, and $h(a) = a$ for all $a \in \mathsf{Ind}(\mathcal{K})$. Note that $\mathcal{I} \subseteq \mathcal{J}$ iff the identity mapping id is a homomorphism $\mathrm{id} : \mathcal{I} \to \mathcal{J}$.

Let $\mathsf{N_V}$ be a countably infinite set of *variables*. An *atom* is an expression of the form $A(x)$ or $r(x, y)$ with $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, and $x, y \in \mathsf{N_V}$, referred to as *concept atoms* and *role atoms*, respectively. A *conjunctive query (CQ)* Q is an existentially quantified conjunction q of atoms, $\exists x_1 \cdots \exists x_n \, q$. For simplicity we restrict q to Boolean; that is, q uses only variables $x_1, \cdots, x_n$. This is without loss of generality since the case of non-Boolean CQs can be reduced to the case of Boolean queries; see e.g. [12].

Let *var* denote the function from queries to sets of variables, such that $var(Q)$ is the set of all the variables used in Q. A *match for Q in $\mathcal{I}$* is a total function $\pi : var(Q) \to \Delta^{\mathcal{I}}$ such that $\mathcal{I}, \pi \models q$ under the standard semantics of first-order logic. An interpretation $\mathcal{I}$ satisfies Q, written as $\mathcal{I} \models Q$, if there exists a match for Q in $\mathcal{I}$. A *partial match for Q in I* is a partial function $\pi : var(Q) \to \Delta^{I}$ such that $I, \pi \models q'$, where q' is obtained from q by dropping atoms which bind some variable not in the domain of π.

Note that we do not consider queries with constants (i.e., individual names): such queries can be viewed as non-boolean queries with a fixed valuation of free variables, and thus are covered by the reduction to the Boolean case. We do consider *unions of conjunctive queries (UCQs)*, which are disjunctions of CQs. An interpretation $\mathcal{I}$ satisfies a UCQ Q if it satisfies one of its disjuncts. It follows immediately that UCQs are *preserved under homomorphisms*; that is, if $\mathcal{I} \models Q$ and there is a homomorphism from $\mathcal{I}$ to $\mathcal{J}$, then also $\mathcal{J} \models Q$.

A query Q is *entailed by a KB $\mathcal{K}$*, denoted as $\mathcal{K} \models Q$, if every model of $\mathcal{K}$ satisfies Q. A model of $\mathcal{K}$ that does not satisfy Q is called a *counter-model*. The *query entailment problem* asks whether a KB $\mathcal{K}$ entails a (U)CQ Q. Moreover, this problem is equivalent to that of finding a counter-model. It is well known that the *query answering problem* can be reduced to query entailment.

In this paper, we address the problem of *finite query entailment*, which is a variant of query entailment where only finite interpretations are considered: an interpretation $\mathcal{I}$ is *finite* if $\Delta^{\mathcal{I}}$ is finite, and a query Q is *finitely entailed by $\mathcal{K}$*, denoted as $\mathcal{K} \models_{\mathsf{fin}} Q$, if every finite model of $\mathcal{K}$ satisfies Q.

3 Counter-model surgery

Later in this paper, we transform a tree-like counter-model into a finite counter-model of bounded size, using a variant of the *blocking principle*: a systematic policy of reusing elements. Before we dive into this construction, we show that we can add certain edges to a counter-model without satisfying the query. These edges must connect a pair of points, for which a different pair of similar points is already connected via an edge over the same role. Throughout this section, we assume no transitivity declarations are present; we will handle these later.

Sharing the same unary type is a natural similarity criterion. Without transitivity and functional declarations, concept inclusions rely only on the unary types of directly linked elements, so adding an r-edge between a pair of elements yields a correct model of the knowledge base if a pair of elements of the same unary types is already connected with an r-edge. However, that criterion is not sufficient, since this way we might accidentally satisfy the query, for example by creating a cycle of given length when the query asks for one. This happens to be a key difficulty to overcome: we cannot introduce cycles shorter than the size of the query.

The first step to enhance the criterion is to look at sufficiently large neighbourhoods, rather than just unary types. We adapt the definition of neighbourhoods from [6] and parameterize it with a set of *constants* $\mathsf{Const} \subseteq \Delta^{\mathcal{I}}$: elements which are considered neighbours of all the elements in the interpretation, regardless of the actual connections between them. Throughout the paper, we assume $\mathsf{Const} = \mathsf{Nom}(\mathcal{K})$ unless specified otherwise. By the *distance* between two elements we understand the length of the shortest undirected path over the union of all roles between these two elements.

Definition 1. *For d in $\mathcal{I} \setminus \mathsf{Const}$ and an integer n, the n-neighbourhood $N_n^{\mathcal{I}}(d)$ of d with respect to Const is the subinterpretation of $\mathcal{I}$ induced by Const and all elements e in $\mathcal{I} \setminus \mathsf{Const}$ within distance n from d in $\mathcal{I} \setminus \mathsf{Const}$, enriched with a concept D interpreted as $\{d\}$, and a concept interpreted as $\{a\}$ for each $a \in \mathsf{Const}$. For $a \in \mathsf{Const}$, $N_n^{\mathcal{I}}(a)$ is the subinterpretation induced by Const, enriched similarly.*

Replacing unary types with large neighbourhoods is still not enough, because nearby elements can have arbitrary large isomorphic neighbourhoods: in the integers with the successor relation, all n-neighbourhoods are isomorphic. The next step is to enrich the initial counter-model in such a way that overlapping neighbourhoods are not isomorphic, following an idea from [7].

Definition 2. *A* colouring with k colours *of an interpretation $\mathcal{I}$ is an extension $\mathcal{J}$ of $\mathcal{I}$ with $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}}$, such that $\mathcal{J}$ coincides with $\mathcal{I}$ in every element in the signature of $\mathcal{I}$, and interprets k fresh concept names $B_1, \ldots, B_k$ such that $B_1^{\mathcal{J}}, \ldots, B_k^{\mathcal{J}}$ is a partition of $\Delta^{\mathcal{J}}$. We say that $d \in B_i^{\mathcal{J}}$ has colour B_i. A colouring $\mathcal{J}$ of $\mathcal{I}$ is n-*proper *if for each $d \in \Delta^{\mathcal{J}}$ all elements of $N_n^{\mathcal{I}}(d)$ have different colours.*

Lemma 1. *If $\mathcal{I} \setminus \mathsf{Const}$ has bounded degree, then for all $n \geq 0$ there exists an n-proper colouring of $\mathcal{I}$ with at most $M + |\mathsf{Const}|$ colours, where M is the size of the largest $(2n)$-neighbourhood.*

We write $\mathcal{I}_n$ for an arbitrarily chosen n-proper colouring of $\mathcal{I}$.

Definition 3. *Let $i \leq n$ and let d, e be elements of $\mathcal{I}_n$. We say that (d, e) is an i-link along role r if either d has an r-successor e' in $\mathcal{I}_n$ such that $N_i^{\mathcal{I}_n}(e') \simeq N_i^{\mathcal{I}_n}(e)$, or e has an r-predecessor d' in $\mathcal{I}_n$ such that $N_i^{\mathcal{I}_n}(d') \simeq N_i^{\mathcal{I}_n}(d)$.*

Theorem 1 (Lifting Theorem). *For a conjunctive query with at most k binary atoms and $n \geq k$ and an interpretation $\mathcal{I}_n$ in which the query is not satisfied, adding n-links to $\mathcal{I}_n$ does not make the query satisfied.*

This theorem was established in [6] for $n \geq k^2$. We prove the stronger variant in the appendix. The strengthening is not necessary for our approach, but it tightens the resulting size estimations.

4 Clique-forest counter-models

In this section, we revisit prior work showing that for $\mathcal{SOI}$ and $\mathcal{SOF}$, the existence of a finite counter-model for Q is equivalent to the existence of a so called *counter-example*, a possibly infinite counter-model of a special form, which generalises tree-shaped models [6]. We impose tighter size constraints on counter-examples and construct automata recognizing them.

The special form is based on the notion of clique-forests.

Definition 4 ([6]). *Let us fix a KB $\mathcal{K}$. A clique-forest for an interpretation $\mathcal{I}$ is a forest whose each node v is labelled with a subinterpretation $\mathcal{I}_v$ of $\mathcal{I} \setminus \mathsf{Nom}(\mathcal{K})$ such that:*

- *the sets $\Delta^{\mathcal{I}_v}$ are a partition of $\Delta^{\mathcal{I} \setminus \mathsf{Nom}(\mathcal{K})}$;*
- *each $\mathcal{I}_v$ is either a single element with all roles empty (element node) or a clique over some transitive role with all other roles empty and no repetitions of unary types (clique node);*
- *apart from edges within cliques and between individuals, in $\mathcal{I} \setminus \mathsf{Nom}(\mathcal{K})$ there is exactly one edge between $\Delta^{\mathcal{I}_u}$ and $\Delta^{\mathcal{I}_v}$ for every two adjacent nodes u and v: assuming u is the parent of v, it is an r-edge from an element of $\Delta^{\mathcal{I}_u}$ to an element of $\Delta^{\mathcal{I}_v}$ for some $r \in \mathsf{Rol}(\mathcal{K})$. We call v the* origin *of $\mathcal{I}_v$.*

Definition 5. *Let $\mathcal{K}^*$ denote the KB obtained from $\mathcal{K}$ by dropping transitivity declarations and let $\mathcal{I}^*$ denote the interpretation obtained from $\mathcal{I}$ by closing transitively the interpretation of each transitive role.*

Note that each existential restriction satisfied in $\mathcal{I}$ is also satisfied in $\mathcal{I}^*$. The same holds for quantifier-free CI, and for universal restrictions involving non-transitive roles. For universal restrictions involving transitive roles, we ensure this property by preprocessing the interpretation: for each pair of $B \in \mathsf{CN}(\mathcal{K})$ and $r \in \mathsf{TRol}(\mathcal{K})$, we add a fresh concept name $P_{r,B}$, axiomatised as $P_{r,B} \sqsubseteq \forall r.(B \sqcap P_{r,B})$, and replace each CI of the form $A \sqsubseteq \forall r.B$ with $A \sqsubseteq \forall r.P_{r,B}$.

An interpretation is *safe* if it does not contain an infinite simple r-path for any transitive role r.

Definition 6. *A* counter-example *for Q is a safe interpretation $\mathcal{I}$ such that: $\mathcal{I} \models \mathcal{K}^*$, $\mathcal{I}^* \not\models Q$, and $\mathcal{I}$ admits a clique-forest with at most $|\mathsf{Ind}(\mathcal{K})| + |\mathsf{Nom}(\mathcal{K})| \cdot |\mathsf{CI}(\mathcal{K})|$ trees, nodes containing at most $|\mathsf{CN}(\mathcal{K})|$ elements, at most $|\mathsf{CI}(\mathcal{K})|$ children nodes connected to a single element, and each element of $\mathsf{Ind}(\mathcal{K}) \setminus \mathsf{Nom}(\mathcal{K})$ occurring in some root.*

If $\mathcal{I}$ is a counter-example for Q, then $\mathcal{I}^*$ is a counter-model for Q, thanks to the initial preprocessing.

Theorem 2. *Q has a finite counter-model iff Q has a counter-example.*

The theorem was originally stated in [6], where counter-examples by definition admitted clique-forests with at most $|\mathcal{K}|^2$ trees and branching at most $|\mathcal{K}|^2$, but the proof there is still valid for the present stronger statement. The bound on the branching of clique-forests is replaced with bounds on the size of clique-tree nodes and number of children nodes connected to a single element within any node; their precise values follow from the construction of a single clique-tree in the proof. The bound on the number of trees can be tightened, as the constructed clique-forests are built of trees rooted in non-nominal individuals and children nodes of nominals.

To obtain the final counter-model of bounded size, we are going to apply the blocking principle to a counter-example. Thanks to Theorem 2, we already know that a counter-example exists whenever Q has a finite counter-model. In the next section, we require that the clique-forests of counter-examples we work with are of certain regularity. We will show that in regular interpretations with a so-called *completeness* property, if two elements are connected with a path over a transitive role, then there is at least one short path over that role which connects them. To be able to focus on counter-examples with regular clique-forests, we shall construct tree automata recognizing the latter.

The following definitions refer to edges and paths in a fixed interpretation $\mathcal{I}$, either known from the context or explicitly specified.

Definition 7. *Fix an ordered pair of elements a, b. We say that (a, b) is an r-edge if there is an edge over r from a to b. We say that (a, b) is an r^*-edge, if either r is transitive and there is a path over r from a to b, or (a, b) is an r-edge.*

Definition 8. *An interpretation $\mathcal{J}$ is* individual-complete*, if in $\mathcal{J}$ each r^*-edge over two individuals used in $\mathcal{K}$ is an r-edge. An interpretation $\mathcal{J}$ is* nominal-complete*, if in $\mathcal{J}$ each r^*-edge with a nominal as one of the endpoints is an r-edge. An interpretation is* complete*, if it is individual- and nominal-complete.*

Note the difference between the individual- and nominal-completeness definitions: in the former, both endpoints of the edges considered have to be individuals, whereas in the latter, only one endpoint has to be a nominal. Each counter-model can be extended to a complete one in a natural way so that the transitive closure stays the same. We can thus restate Theorem 2: Q has a finite counter-model iff Q has a complete counter-example.

Theorem 3. *For a union Q of CQs, with each CQ of size at most m, and a complete interpretation $\mathcal{M}$ with $\Delta^{\mathcal{M}} = \mathsf{Ind}(\mathcal{K})$, there exists an automaton with at most $2^{2 \cdot 3^{4m}|Q||\mathcal{T}|^{3m}}$ states recognizing clique-forests of complete counter-examples for Q, for which the subinterpretation induced by $\mathsf{Ind}(\mathcal{K})$ is equal to $\mathcal{M}$.*

See Appendix for the automaton construction. Theorem 3 is inspired by a similar theorem in [6]. We add the completeness requirement and restrict the size of the automaton state-space, rather than its overall size. This way, we are able to get a bound which is independent of the ABox; it depends only on the TBox and query sizes. The automaton size still depends on the ABox, since its description has to contain all the initial lists of states, each state corresponding to one tree, and we consider clique-forest with at least one tree per individual occurring in $\mathcal{K}$.

These tweaks result in significant differences between our construction of the automaton and the construction presented in [6]. Most importantly, we cannot transfer the information about the edges between individuals to the TBox, as that would introduce a dependency of the automaton state-space on the ABox. This influences especially the automaton component responsible for verifying that the query cannot be satisfied.

5 Bounded counter-models

We say that DL has the *linear-size counter-model property* if whenever there exists a counter-model for a given query Q and a knowledge base $\mathcal{K} = (\mathcal{A}, \mathcal{T})$ expressed in that DL, there exists a counter-model for Q of size bounded by $|\mathcal{A}| \cdot B(Q, \mathcal{T})$ for some function B. The goal of the larger part of this section is to prove the following theorem.

Theorem 4. *$\mathcal{SOI}$ and $\mathcal{SOF}$ have the linear-size counter-model property.*

Clique-forests of complete counter-examples can be recognized by a family of automata by Theorem 3. Suppose $\mathcal{K}$ and Q admit a finite counter-model. As discussed in the last section, they admit then a complete counter-example, which in turn can be recognized by an automaton. Any automaton accepting some clique-forest must also accept some regular forest, which has only finitely many non-isomorphic subtrees. The number of these subtrees can be bounded by the number of states of the automaton. Following the bound in Theorem 3, we pick a complete counter-example $\mathcal{I}$ admitting a regular clique-forest with at most $p = 2^{2 \cdot 3^{4m} |Q| |\mathcal{T}|^{3m}}$ non-isomorphic subtrees.

We shall turn $\mathcal{I}$ into a finite counter-model for Q, using the blocking principle. The main obstacle is that Q uses transitive roles, which are not fully represented. To solve that, we replace Q with a different query. In [6], this was done by exploiting a bound on the length of simple r-paths for transitive roles r, guaranteed by the regularity of clique-forests of the considered counter-examples. Here, we require that the counter-example we process is complete and consider the distance between elements that are connected by r-paths, which will let us derive better bounds on the size of the obtained finite counter-models.

Definition 9. *A pair of elements a, b is at r-*distance k *if the shortest r-path from a to b is of length k. An interpretation is* ℓ-ranged *if for each r^*-edge, its endpoints are at r-distance at most ℓ.*

For transitive role atoms in queries, the path that leads from one variable to another does not play any role in the query evaluation, only the unary types of the connected variables matter (and the fact that they are connected by some path). We use this intuition to deduce that in ℓ-ranged interpretations, any query with transitive binary atoms can be rewritten so that closing the roles by transitivity does not influence the query evaluation.

Let Q^* be obtained from Q by replacing each transitive atom $s(x, y)$ by the disjunction $\bigvee_{i \leq \ell} s^i(x, y)$, where $s^i(x, y)$ is the conjunctive query expressing the

i-fold composition of s. If each disjunct of Q contains at most k binary atoms, Q^* can be rewritten as a union of conjunctive queries, each using at most $k \cdot \ell$ binary atoms.

Lemma 2. *For all ℓ-ranged $\mathcal{J}$, $\mathcal{J}^* \models Q$ iff $\mathcal{J} \models Q^*$.*

Lemma 3. *$\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ is $(4p)$-ranged.*

Proof. Let r be a transitive role in $\mathcal{K}$. Each r-path going down the clique-forest of $\mathcal{I}$ contains at most p nodes. Indeed, if there were a longer r-path, then a subtree would occur twice on that path, which immediately leads to an infinite simple r-path in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$, contradicting the safety of $\mathcal{I}$. For each clique node, at most one additional step is needed to traverse it with an r-edge. Additionally, if $\mathcal{K}$ is a $\mathcal{SOI}$ KB, an r-path can be composed of two parts, one going down the tree and the other consisting of r^{-1} edges going upwards. If follows that, for $\mathcal{SOI}$ and $\mathcal{SOF}$, any r-reachable pair in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ is at r-distance at most $4p$. $\square$

The following lemma, regarding a bound on the range of an interpretation, is a stronger version of the analogous lemma presented in [6], regarding a bound on the length of simple transitive paths. The bound on the range has to be multiplied by a constant to get the bound on the range in the full interpretation, whereas in [6], the size of the set of nominals also comes into play. We draw upon the individual-completeness of the interpretations we consider to achieve this improvement.

Lemma 4. *For each individual-complete interpretation $\mathcal{J}$, if $\mathcal{J} \setminus \mathsf{Ind}(\mathcal{K})$ is ℓ-ranged, then $\mathcal{J}$ is $(2\ell + 3)$-ranged.*

Proof. Take two elements a, b, such that (a, b) is an r^*-edge in $\mathcal{J}$, but not an r^*-edge in $\mathcal{J} \setminus \mathsf{Ind}(\mathcal{K})$; it follows that any r-path from a to b visits an individual. If r is non-transitive, then (a, b) is an r-edge and a, b are at distance 1, otherwise consider such r-path π. It starts in one of the trees in $\mathcal{J} \setminus \mathsf{Ind}(\mathcal{K})$ (or in an individual, in which case the tree degenerates to an empty one), and reaches a nominal connected to this tree or the tree root. Eventually, π enters the tree containing its other endpoint, again via a nominal connected to that tree or its root. Let c and d be the first and last individuals on π. As $\mathcal{J}$ is individual-complete, (c, d) is an r-edge, if $c \neq d$. We can reduce π to a path segment starting in a in the first tree, a single edge to c, possibly a single r-edge (c, d), a single edge from d, and a path segment ending in b in the last visited tree. As the first and last path segments are in $\mathcal{J} \setminus \mathsf{Ind}(\mathcal{K})$, each pair of elements in them is at r-distance at most ℓ. This gives a $(2\ell + 3)$ bound on r-distance in $\mathcal{J}$. $\square$

We know that $\mathcal{I}$ is ℓ^*-ranged for $\ell^* = (8p + 3)$ by Lemmas 3 and 4. We conclude that $\mathcal{I} \not\models Q^*$ by Lemma 2.

The degree of elements in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ is bounded by $|\mathsf{CN}(\mathcal{K})| + |\mathsf{CI}(\mathcal{K})|$ (elements are connected within clique-nodes of size at most $|\mathsf{CN}(\mathcal{K})|$ and with at most $|\mathsf{CI}(\mathcal{K})| + 1$ elements from other nodes). Non-nominal individuals used in $\mathcal{K}$ can be connected with any number of other individuals, so even though their degree is bounded, the bound is relatively large. Nominals can be connected with an arbitrary number of elements.

We aim to use the blocking principle. We consider an n-proper colouring $\mathcal{I}'_n$ of $\mathcal{I}' = \mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$, for some fixed n, instead of a proper colouring of $\mathcal{I}$; we do so to avoid the influence of the individuals on the size of the neighbourhoods (and therefore the required number of colours). $\mathcal{I}'$ is a clique-forest too: the children nodes of non-nominal individuals become new roots. On each branch π in $\mathcal{I}'_n$, let D_π be the first node for which some earlier node E_π satisfies $N_n^{\mathcal{I}_n}(d_\pi) \simeq N_n^{\mathcal{I}_n}(e_\pi)$, where $d_\pi \in D_\pi$ and $e_\pi \in E_\pi$ are the endpoints of the edges connecting D_π and E_π to their parent nodes. The new interpretation $\mathcal{F}_n$ is obtained by including the branch π up to the predecessor of node D_π, with the edge originally leading to d_π redirected to e_π. Individuals are copied from $\mathcal{I}$, together with their adjacent edges. Note that for $\mathcal{SOF}$, all functionality declarations are satisfied, as each redirected edge is a forward edge.

As each node in $\mathcal{I}'_n$ has finitely many children and we cut all the infinite paths off, $\mathcal{F}_n$ is finite by Kőnig's lemma. As we start with $\mathcal{I} \models \mathcal{K}^*$ and the unary types of edge endpoints are preserved in $\mathcal{F}_n$, we get $\mathcal{F}_n \models \mathcal{K}^*$. By the initial preprocessing, $(\mathcal{F}_n)^* \models \mathcal{K}$. The conclusions drawn so far are valid for any n.

Lemma 5. *$\mathcal{F}_n \setminus \mathsf{Ind}(\mathcal{K})$ is $(4p)$-ranged for $n \geq 4p$.*

Proof. During the construction of $\mathcal{F}_n$ from $\mathcal{I}$, we add only one link for each infinite downwards path in $\mathcal{I}$ starting in a tree root. Each of these links goes up the tree. By *link origins* we shall understand the elements that had an edge added during the construction of $\mathcal{F}_n$. We can visualize $\mathcal{F}_n \cup \mathcal{I}$ as the core $\mathcal{F}_n$ with additional hanging subtrees, each attached to $\mathcal{F}_n$ with a single edge to a link origin. There is no way of accessing these subtrees from $\mathcal{F}_n \setminus \mathsf{Ind}(\mathcal{K})$ other than via the corresponding link origin.

Let $\ell = 4p$ and fix some $n \geq \ell$. We prove the following claim by induction on m: if there is a path of length $\ell + 1$ from a in $\mathcal{F}_n \setminus \mathsf{Ind}(\mathcal{K})$ to b in $(\mathcal{F}_n \cup \mathcal{I}) \setminus \mathsf{Ind}(\mathcal{K})$ over a transitive role r using at most m links, then there is an r-path in $(\mathcal{F}_n \cup \mathcal{I}) \setminus \mathsf{Ind}(\mathcal{K})$ from a to b over r of length at most ℓ.

If we prove the claim for all m, we can deduce that $\mathcal{F}_n \setminus \mathsf{Ind}(\mathcal{K})$ is ℓ-ranged. Indeed, take a transitive role s and an s-path π in $\mathcal{F}_n \setminus \mathsf{Ind}(\mathcal{K})$ of length $\ell + 1$, with endpoints (a, b) and m links (possibly none). Applying the claim, we get an s-path ρ in $(\mathcal{F}_n \cup \mathcal{I}) \setminus \mathsf{Ind}(\mathcal{K})$ of length at most ℓ, connecting a and b. To show that ρ has to be in $\mathcal{F}_n \setminus \mathsf{Ind}(\mathcal{K})$, we take advantage of the tree structure: if a path goes down a given branch, it has to use a link to come back to an element higher up in the tree. There are no links from elements in $\mathcal{I} \setminus \mathcal{F}_n$, so ρ must only visit nodes in $\mathcal{F}_n \setminus \mathsf{Ind}(\mathcal{K})$; otherwise ρ would get stuck in $\mathcal{I} \setminus \mathcal{F}_n$ and have its endpoint there.

Let us approach the induction proof. The induction base for $m = 0$ is almost immediate. A path without links in $(\mathcal{F}_n \cup \mathcal{I}) \setminus \mathsf{Ind}(\mathcal{K})$ is also a path in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$, which is ℓ-ranged, as shown in Lemma 3.

Assume the inductive hypothesis holds for m. Suppose that for a transitive role s there is an s-path π in $(\mathcal{F}_n \cup \mathcal{I}) \setminus \mathsf{Ind}(\mathcal{K})$ of length $l + 1$ from a to b, such that it uses at most $m+1$ links. If π uses fewer than $m+1$ links, we simply use the inductive hypothesis; thus we can assume π uses exactly $m+1$ links.

As $m \geq 0$, there is at least one link in π. Let (d, e) be the last link in π, with the witnessing element e' in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$. Note that the node containing e' is not an ancestor of the node containing a in the clique-tree, as otherwise a would lie in a subtree rooted in e' and would be removed while constructing $\mathcal{F}_n$. Let $\pi_{a,d}$ and $\pi_{e,b}$ denote the parts of π from a to d and from e to b. The path $\pi_{e,b}$ is of length at most ℓ. We have chosen $n \geq \ell$, so the n-neighbourhood of e' in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ contains a path $\pi'_{e',b'}$ isomorphic to $\pi_{e,b}$. The path consisting of $\pi_{a,d}$, the edge from d to e' and $\pi_{e',b'}$ is of length $\ell+1$, but uses only m links, so there is an s-path ρ in $(\mathcal{F}_n \cup \mathcal{I}) \setminus \mathsf{Ind}(\mathcal{K})$ from a to b' of length at most ℓ.

Due to the construction of $\mathcal{F}_n$, each path to e' starting in $\mathcal{F}_n \setminus \mathsf{Ind}(\mathcal{K})$ must visit d. Let $\rho_{a,d}$ and $\rho_{e',b'}$ denote the parts of ρ from a to d and from e' to b'. Note that $\rho_{e',b'}$ lies in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$, so it does not use any links. At least one of the following inequalities holds: $|\rho_{a,d}| < |\pi_{a,d}|$ or $|\rho_{e',b'}| < |\pi_{e,b}|$. If the first one holds, we can take the path consisting of $\rho_{a,d}$, the edge from d to e and of $\pi_{e,b}$ as an s-path from a to b of length at most ℓ. If the second inequality holds, we use the fact that $\rho_{e',b'}$ lies in the n-neighbourhood of e' in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ and find an isomorphic path $\rho'_{e,b}$ from e to b. The path consisting of $\pi_{a,d}$, an edge from d to e and of $\rho'_{e,b}$ is an s-path from a to b of length at most ℓ. This proves the inductive hypothesis for $m+1$. □

Recall that $p = 2^{2\cdot 3^{4m}|Q||\mathcal{T}|^{3m}}$ and $l^* = 8p+3$. For $n \geq 4p$, $\mathcal{F}_n$ is ℓ^*-ranged by Lemmas 4 and 5. The size of the expanded query Q^* is bounded by $k \cdot \ell^*$. Furthermore, $\mathcal{F}_n \not\models Q^*$ for $n \geq k \cdot \ell^*$ by Theorem 1. We conclude that $\mathcal{F}_n^* \not\models Q$ for $n \geq k \cdot \ell^*$ by Lemma 2.

We can now approach the estimation of the size of $\mathcal{F}_n$. The interpretation $\mathcal{I}$ is a counter-example, so in its clique-forest, each node is of size at most $|\mathsf{CN}(\mathcal{K})|$ and each element is connected to at most $|\mathsf{Cl}(\mathcal{K})|$ children nodes. Children nodes of individuals in $\mathcal{I}$ form the roots of clique-trees in $\mathcal{I}'$, so the number of trees in $\mathcal{I}'_n$ is bounded by $|\mathsf{Ind}(\mathcal{K})| \cdot |\mathsf{Cl}(\mathcal{K})|$. As described before, the degree of elements in $\mathcal{I}'$ is bounded by $D = |\mathsf{CN}(\mathcal{K})| + |\mathsf{Cl}(\mathcal{K})|$, so the neighbourhoods in $\mathcal{I}'$ of radius m consist of at most $O(D^m)$ elements. It is enough to use $C = O(D^{2n}) + |\mathsf{Nom}(\mathcal{K})|$ colours to obtain a proper n-colouring of $\mathcal{I}'$ by Lemma 1. There are $2^{|\mathsf{CN}(\mathcal{K})|}$ unary types, so there are $L = 2^{|\mathsf{CN}(\mathcal{K})|} \cdot C$ ways of labelling a single element with its unary type and colour. The neighbourhoods of radius n are of size $O(D^n)$, which means there are $N = L^{O(D^n)}$ different neighbourhoods in $\mathcal{I}'_n$. On a downward path in $\mathcal{F}_n$, we cannot come across the same neighbourhood centred in an element node or a clique origin, as otherwise a link upwards would be created higher up on this path; thus N bounds the depth of clique-trees in $\mathcal{F}_n$. The degree of the clique-forest of $\mathcal{I}'_n$ is bounded by $D' = |\mathsf{CN}(\mathcal{K})| \cdot |\mathsf{Cl}(\mathcal{K})|$, so the number of nodes in each tree is bounded by $(D')^N$.

The overall size of $\mathcal{F}_n$ is bounded by $M \cdot S(Q, \mathcal{T})$, where $M = |\mathsf{Ind}(\mathcal{K})| \cdot |\mathsf{Cl}(\mathcal{K})|$ bounds the number of clique-trees in $\mathcal{I}'$ and $S(Q, \mathcal{T})$ is a bound on the size of a single tree. The size estimation of a single tree is independent of the ABox size; it is only governed by the TBox and query. This finalizes the proof of Theorem 4.

Corollary 1. *Finite entailment of UCQs by $\mathcal{SOI}$ and $\mathcal{SOF}$ KBs is coNP-complete with respect to data complexity.*

Proof. If we fix the sizes of the TBox and query, then $S(Q, \mathcal{T})$ is fixed and the size of $\mathcal{F}_n$ is linear with respect to the ABox size. We consider the following non-deterministic approach to disproving query entailment: we simply guess a transitively closed counter-model of size at most $M \cdot S(Q, \mathcal{T})$ and verify all the required properties. First, we need to check that it is indeed transitively closed. This can be done in polynomial time with respect to the interpretation size. Next, we need to verify ABox and TBox constraints; as the interpretation is transitively closed, for each element each constraint can be verified by looking at that element's immediate neighbours. Finally, we need to check that the query is not satisfied. This comes down to verifying a first-order logic formula, which can be done in time polynomial of $|\mathsf{Ind}(\mathcal{K})|$, where the degree of the polynomial depends only on the sizes of the TBox and query. Altogether, this constitutes a coNP algorithm for deciding query entailment for fixed-size TBox and query.

We have shown that finite query entailment is in coNP with respect to data complexity. It is known that (unrestricted) query entailment is coNP-hard (with respect to data complexity) in $\mathcal{ALE}$ [13]. By finite controllability, so is finite query entailment. Since $\mathcal{ALE}$ is a fragment of both $\mathcal{SOI}$ and $\mathcal{SOF}$, NP-hardness carries over. $\square$

6 Conclusions

By enhancing the approach to finite query entailment presented in [6], we were able to show the linear-size counter-model property for $\mathcal{SOI}$ and $\mathcal{SOF}$ (for fixed TBox and query). This let us show that finite query entailment is in coNP with respect to data complexity for these DLs. Basing on previous work, we deduced that it is a coNP-complete problem. To improve the final size estimations, we also established a stronger version of the Lifting Theorem.

The techniques used to construct the final counter-models were not efficient enough to provide asymptotically tight size estimations for arbitrary (not fixed) TBoxes and queries. One limiting factor is the doubly-exponential bound on the range of the considered counter-models, stemming from the automata state-space size. It would be interesting to see if this could be improved. Another natural question is whether our techniques are applicable to $\mathcal{SIF}$.

Acknowledgements. The author was supported by Poland's National Science Centre grant 2018/30/E/ST6/00042. The whole work has been prepared with huge help from Filip Murlak, the supervisor of the author's Master thesis. We thank the anonymous reviewers of DL 2020 for their helpful comments.

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Proc. Int. J. C. on Artif. Int. pp. 364–369 (2005)

2. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, New York, NY, USA, 2nd edn. (2010)
3. Bienvenu, M., Ortiz, M.: Ontology-mediated query answering with data-tractable description logics. In: Reasoning Web. LNCS, vol. 9203, pp. 218–307. Springer (2015)
4. Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., , Riccardo: Tractable reasoning and efficient query answering in description logics: The DL-Lite family. J. of Autom. Reasoning **39(3)**, 385–429 (2007)
5. Glimm, B., Lutz, C., Horrocks, I., Sattler, U.: Conjunctive query answering for the description logic SHIQ. J. Artif. Intell. Res. **31**, 157–204 (2008)
6. Gogacz, T., Ibáñez-García, Y.A., Murlak, F.: Finite query answering in expressive description logics with transitive roles. In: Proc. KR 2018. pp. 369–378 (2018)
7. Gogacz, T., Marcinkowski, J.: On the BDD/FC conjecture. In: PODS. pp. 127–138 (2013). https://doi.org/10.1145/2463664.2463668
8. Hustadt, U., Motik, B., Sattler, U.: Reasoning in description logics by a reduction to disjunctive datalog. J. Autom. Reasoning **39(3)**, 351–384 (2007)
9. Lutz, C., Wolter, F.: The data complexity of description logic ontologies. Log. Met. in Comp. Sci. **13(4:7)**, 1–46 (2017)
10. Ortiz, M., Calvanese, D., Eiter, T.: Characterizing data complexity for conjunctive query answering in expressive description logics. In: Proc. Nat. C. on Artif. Int. pp. 275–280. AAAI Press (2006)
11. Rudolph, S.: Undecidability results for database-inspired reasoning problems in very expressive description logics. In: KR. pp. 247–257. AAAI Press (2016)
12. Rudolph, S., Glimm, B.: Nominals, inverses, counting, and conjunctive queries or: Why infinity is your friend! J. Artif. Intell. Res. **39**, 429–481 (2010)
13. Schaerf, A.: On the complexity of the instance checking problem in concept languages with existential quantification. J. of Intel. Inf. Systems **2**, 265–278 (1993)

A Proof of Lemma 1

The following proof is taken from [6], where the lemma was formulated with no mention of the number of required colours. Let $n \geq 0$. Because $\mathcal{I} \setminus \mathsf{Nom}(\mathcal{K})$ has bounded degree, $2n$-neighbourhoods in $\mathcal{I}$ have size bounded by some m. We colour the elements of $\mathcal{I}$ one by one, with m colours. Pick an uncoloured element d. At most $m-1$ colours are already used in $N^{\mathcal{I}}_{2n}(d)$. Assign to d any colour that is not yet used in $N^{\mathcal{I}}_{2n}(d)$. This procedure gives an n-proper colouring. Indeed, consider different e, e' from $N^{\mathcal{I}}_{n}(d)$ for some $d \in \mathcal{I}$. Without loss of generality we can assume that e was coloured before e'. But e belongs to $N^{\mathcal{I}}_{2n}(e')$, so the colours of e and e' are different by construction.

B Lifting Theorem

B.1 Formulation

Definition 10. *For $i \leq n$, let $\mathcal{I}^i_n$ be the interpretation obtained from $\mathcal{I}_n$ by including into the interpretation of each role r all i-links along r; that is, for every role r and every i-link (d, e) along r, $(d, e) \in r^{\mathcal{I}^i_n}$.*

Notice that for $i < j$, each j-link is also an i-link. Note also that (d, e) is an i-link along role r if and only if (e, d) is an i-link along r^-. We have

$$\mathcal{I}_n \subseteq \mathcal{I}^n_n \subseteq \mathcal{I}^{n-1}_n \subseteq \cdots \subseteq \mathcal{I}^1_n \subseteq \mathcal{I}^0_n\,,$$

but the domains of all these interpretations coincide. We keep referring to the edges present in $\mathcal{I}^i_n$ but not in $\mathcal{I}_n$ as i-links, even though they are ordinary edges now.

Definition 11. *We say that functions $f, f' : X \to \Delta^{\mathcal{I}}$ are* d-indistinguishable *if for all $x \in X$:*

$$N^{\mathcal{I}}_d(f(x)) \simeq N^{\mathcal{I}}_d(f'(x)).$$

Throughout the paper, we consider neighbourhoods with respect to $\mathsf{Const} = \mathsf{Nom}(\mathcal{K})$, however, all the definitions reliant on the notion of neighbourhoods are correct for every choice of the set of constants Const. Let us note that for every pair of d-indistinguishable functions f, f', if for some $x \in X$ and $c \in \mathsf{Const}$ either $f(x) = c$ or $f'(x) = c$, then $f(x) = f'(x)$. This is because whenever we consider any neighbourhoods, we add a nominal-like concept for each constant.

Theorem 5. *Let P be a CQ with at most k binary atoms and let $n \geq k$. For each homomorphism $h : P \to \mathcal{I}^n_n$ there exists a homomorphism $h' : P \to \mathcal{I}_n$ which is $(n-k)$-indistinguishable from h.*

Theorem 5 holds for any interpretation $\mathcal{I}$ of any $\mathcal{SOIF}$ KB. It was originally stated in [6] for $n \geq k^2$ and regarded a pair of $(n-k^2)$-indistinguishable homomorphisms. Furthermore, it was proved only for $\mathsf{Const} = \mathsf{Nom}(\mathcal{K})$, and here we consider an arbitrary choice of the set of constants Const. We prove the enhanced version of the theorem in the next subsection; let us now see that it implies the Lifting Theorem (Theorem 1).

Proof (Lifting Theorem). Interpretation $\mathcal{J}_n$ we obtain in the process satisfies $\mathcal{J}_n \subseteq \mathcal{I}_n^n$. Consequently, if there was a homomorphism $h : P \to \mathcal{J}_n \subseteq \mathcal{I}_n^n$ for some CQ P constituting Q, Theorem 5 would yield a homomorphism $h' : P \to \mathcal{I}_n$, contradicting $\mathcal{I} \not\models Q$. □

B.2 Proof of Theorem 5

We start with the following observation.

Lemma 6. *All links involving elements of* Const *are ordinary edges.*

Proof. Assume (d,e) is an i-link for d in Const (the symmetrical case is analogous). Assume that d has a successor f such that $N_i^{\mathcal{I}_n}(f) \simeq N_i^{\mathcal{I}_n}(e)$. Then, the edge (d,e) must also exist in $N_i^{\mathcal{I}_n}(e)$, which shows that (d,e) is an ordinary edge. Assume now that e has a predecessor f such that $N_i^{\mathcal{I}_n}(f) \simeq N_i^{\mathcal{I}_n}(d)$. Then, f must be equal to d, due to the concepts used to mark constants in the definition of a neighbourhood, so (d,e) is an ordinary edge. □

Let $h(P)$ denote the subinterpretation of $\mathcal{I}_n^n$ obtained by restricting the domain to $h(\mathit{var}(P))$, and only keeping in each role r edges $(h(x),h(y))$ such that $r(x,y)$ is an atom from P. We say that h *uses* an r-edge of $\mathcal{I}_n^n$ if this r-edge is present in $h(P)$. We say that h *uses* a link over r of $\mathcal{I}_n^n$ if this link is an edge of $h(P)$, but not an edge of $\mathcal{I}_n$.

Definition 12. *By $\mathcal{J}' \cap \mathcal{J}''$ we mean the interpretation $\mathcal{J}$ such that $\Delta^{\mathcal{J}} = \Delta^{\mathcal{J}'} \cap \Delta^{\mathcal{J}''}$, $A^{\mathcal{J}} = A^{\mathcal{J}'} \cap A^{\mathcal{J}''}$ for all concept names A, and $r^{\mathcal{J}} = r^{\mathcal{J}'} \cap r^{\mathcal{J}''}$ for all role names r.*

Let us analyze the structure of $h(P)$. Let $\mathcal{I}_n' = \mathcal{I}_n \setminus \mathsf{Const}$ and let $\mathcal{C}$ denote the set of connected components in $h(P) \cap \mathcal{I}_n'$, understood as graphs. Each component in $\mathcal{C}$ contains at most k edges, as P contains at most k binary atoms. We shall now prove that h is *far-linked*; that is, h does not use links that join two elements within the same connected component in $\mathcal{C}$. This is a general property of such homomorphisms, which stems from the fact that each link connects elements that are far apart, as expressed in the following lemma.

Lemma 7. *Elements forming a link are at distance at least $2n$ in $\mathcal{I}_n'$.*

Proof. Let (a,b) be a link along r. Suppose that the distance between a and b in $\mathcal{I}_n'$ is smaller than $2n$. Take the shortest path π in $\mathcal{I}_n'$ between a and b; π is of length at most $2n-1$. As (a,b) is a link, either a has an r-successor f in

$\mathcal{I}'_n$ with the same colour as b, or b has an r-predecessor f in $\mathcal{I}'_n$ with the same colour as a. In the first case, there is an element on π, which is at distance at most n from both b and f, in the second case, there is an element on π, which is at distance at most n from both a and f. Since a, b, f share the same colour, are pairwise unequal and are in an n-neighbourhood of another element, we obtain a contradiction. Thus, elements forming a link must be at distance at least $2n$ in $\mathcal{I}'_n$. □

Lemma 8. *Each homomorphism $h'' : P \to \mathcal{I}^i_n$ is far-linked.*

Proof. Let $r(x, y)$ be an atom of the query P such that $h''(x)$ and $h''(y)$ are in the same connected component $C \in h''(P) \cap \mathcal{I}'_n$. As P has at most k binary atoms, elements in C are connected with at most k edges. Since $h''(x)$ and $h''(y)$ are both in C, they at distance at most $k < 2n$ in $\mathcal{I}'_n$ (unless $k = 0$; we verify this case separately). We use Lemma 7 to deduce that $(h''(x), h''(y))$ is an r-edge in $\mathcal{I}'_n$. □

We see that all the links used by h join elements from different connected components in $\mathcal{C}$. We will construct a series of homomorphisms $(h_0, h_1, ..., h_l)$, with $l \leq k$, $h_j : P \to \mathcal{I}^{n_j}_n$, $n_j < n_{j-1}$ for each $j > 0$, starting with $h_0 = h$ and $n_0 = n$, satisfying the following constraints. First, we require that h_{j+1} uses strictly fewer links than h_j. Second, we require that h_{j+1} and h_j are (n_{j+1})-indistinguishable. Third, we require that h_l uses no links and that $n_l \geq n - k$. If we manage to satisfy these constraints, h_l treated as $h' : P \to \mathcal{I}_n$ is the homomorphism postulated in the theorem statement.

Let $\mathcal{C}_j$ denote the connected components of $h_j(P) \cap \mathcal{I}'_n$. We keep the invariant that $(\mathcal{C}_j \cap \mathcal{C})$ is non-empty for $j < l$. We construct the *components graph* of $\mathcal{C}_j$: the graph with elements of $\mathcal{C}_j$ as vertices and edges between vertices representing connected components from $\mathcal{C}_j$ that are joined by a link between their elements. The second invariant is that for $j > 0$, there is exactly one connected component in $\mathcal{C}_j$ that is not present in $\mathcal{C}_{j-1}$; let D'_j denote that component. If $j > 0$ and there is a component $D_j \in (\mathcal{C}_j \cap \mathcal{C})$ joined by a link to D'_j, set $C_j = D_j$ and $C'_j = D'_j$. Otherwise, either there are $C_j, C'_j \in (\mathcal{C}_j \cap \mathcal{C})$ joined by a link, or h_j uses no links and we finish the construction. In case C_j, C'_j exist, homomorphism h_{j+1} is obtained from h_j by *pulling* C_j closer to C'_j by the link that joins these components; as a result, C_j, C'_j are combined into one component, namely D'_{j+1}. The details of that construction are presented in the next subsection.

We will later show that h_{j+1} uses links for a subset of these binary atoms of P, for which h_j uses links and which do not join C_j with C'_j; the links joining C_j and C'_j in h_j are replaced with regular edges of $\mathcal{I}_n$ in h_{j+1}. We compensate for eliminating these links by paying a certain cost: homomorphism h_{j+1} is less strict than h_j when it comes to preserving the neighbourhoods of the elements joined with a link, which is why we have to set $n_{j+1} = n_j - |C_j|$.

We will now prove that $n_j \geq n - k$ for all j. This will entail that h' is indeed $(n-k)$-indistinguishable from h. We use the following invariant: altogether there are at least $n - n_j$ edges in connected components from $(\mathcal{C}_j \setminus \mathcal{C})$. This can be proved by induction on j. The base step for $j = 0$ reduces to the statement that an empty graph has no edges. In the inductive step, we pull the component C_j towards C'_j, merging the two within a larger connected component. We have that $C_j \in \mathcal{C}_j$ is in $\mathcal{C}$ and the new connected component $D'_{j+1} \in \mathcal{C}_{j+1}$ is not in $\mathcal{C}$. This means that the number of edges in connected components from $\mathcal{C}_{j+1} \setminus \mathcal{C}$ must be larger than the number of edges in connected components from $\mathcal{C}_j \setminus \mathcal{C}$. The total increase is equal to at least $|C_j|$; this stems from incorporating $|C_j| - 1$ edges within C_j, and from reducing at least one link. Using the inductive hypothesis, we get that there are at least $n - n_j$ edges in connected components from $\mathcal{C}_j \setminus \mathcal{C}$, and at least $n - n_j + |C_j|$ edges in connected components from $\mathcal{C}_{j+1} \setminus \mathcal{C}$. As $n_{j+1} = n_j - |C_j|$, we get $n - n_j + |C_j| = n - n_{j+1}$, which proves the inductive step. If there was j such that $n_j < n - k$, the number of edges in connected components from $(\mathcal{C}_j \setminus \mathcal{C})$ would be at least $n - n_j$; as $n - n_j > n - (n - k) = k$, this would contradict the fact that $h_j(P)$ has at most k edges.

B.3 Definition of h_{j+1}

In this subsection, we mark the names of interpretation elements with bold colour fonts. Whenever we mention distance between elements, we refer to distance in $\mathcal{I}'_n$.

We are going to define h_{j+1} based on h_j and $C_j, C'_j \in \mathcal{C}_j$.

Let $(\mathbf{d}, \mathbf{e})$ be a link along s used by h_j and joining C_j with C'_j. We only consider the case when $\mathbf{d} \in C'_j$ and $\mathbf{e} \in C_j$; the proof for the other case is symmetrical. Either $\mathbf{d}$ has an s-successor $\mathbf{e}'$ such that $N^{\mathcal{I}_n}_{n_j}(\mathbf{e}) \simeq N^{\mathcal{I}_n}_{n_j}(\mathbf{e}')$, or $\mathbf{e}$ has an s-predecessor $\mathbf{d}'$ such that $N^{\mathcal{I}_n}_{n_j}(\mathbf{d}) \simeq N^{\mathcal{I}_n}_{n_j}(\mathbf{d}')$; once again we take advantage of almost full symmetry and only present the proof for the first case.

Let $g : N^{\mathcal{I}_n}_{n_j}(\mathbf{e}) \to N^{\mathcal{I}_n}_{n_j}(\mathbf{e}')$ be the witnessing isomorphism for the $(\mathbf{d}, \mathbf{e})$ link. We define $h_{j+1} : P \to \mathcal{I}^{n_{j+1}}_n$ for $n_{j+1} = n_j - |C_j|$ as follows: for each $z \in \mathit{var}(P)$, let $h_{j+1}(z) = g(h_j(z))$ if $h_j(z) \in C_j$, and $h_{j+1}(z) = h_j(z)$ otherwise. To ensure that the definition of h_{j+1} is correct, we need to show that $n_{j+1} \geq 0$. Altogether, components in $\mathcal{C}$ have at most n edges (more precisely, at most $k \leq n$ edges), and we have proven an invariant that there are at least $n - n_j$ edges in connected components from $(\mathcal{C}_j \setminus \mathcal{C})$. As C_j is in $(\mathcal{C}_j \cap \mathcal{C})$, and $(\mathbf{d}, \mathbf{e})$ is a link, C_j can have at most $n - 1 - (n - n_j) = n_j - 1$ edges. Thus, $n_{j+1} = n_j - |C_j| \geq 0$.

By definition, h_j and h_{j+1} are (n_{j+1})-indistinguishable: $N^{\mathcal{I}_n}_{n_j}(\mathbf{e}) \simeq N^{\mathcal{I}_n}_{n_j}(\mathbf{e}')$ and $n_{j+1} = n_j - |C_j|$, so g preserves (n_{j+1})-neighbourhoods of elements within distance $|C_j|$ from $\mathbf{e}$. We need to verify that h_{j+1} is indeed a homomorphism.

Let $r(x, y)$ be an atom of the query P and let $\mathbf{p} = h_j(x), \mathbf{q} = h_j(y)$, $\mathbf{p}' = h_{j+1}(x)$, $\mathbf{q}' = h_{j+1}(y)$. We have to consider the following cases:

1. $\mathbf{p}, \mathbf{q} \notin C_j$,
2. $\mathbf{p}, \mathbf{q} \in C_j$,
3. exactly one of $\mathbf{p}, \mathbf{q}$ is in C_j; we take advantage of the symmetry and only consider the case $\mathbf{p} \notin C_j, \mathbf{q} \in C_j$, with the following subcases:
   (a) $\mathbf{p} \in C'_j$,
   (b) $\mathbf{p} \in C''_j$ for $C''_j \in \mathcal{C}_j$ such that $C''_j \neq C'_j$.
   (c) $\mathbf{p} \in \mathsf{Const}$,

Case 1. By definition, $(\mathbf{p}', \mathbf{q}') = (\mathbf{p}, \mathbf{q})$. As h_j is a homomorphism into $\mathcal{I}_n^{n_j} \subseteq \mathcal{I}_n^{n_{j+1}}$, we have that $(\mathbf{p}', \mathbf{q}')$ is an r-edge in $\mathcal{I}_n^{n_{j+1}}$. Clearly, h_{j+1} uses no new links for such atoms.

Case 2. By definition, $(\mathbf{p}', \mathbf{q}') = (g(\mathbf{p}), g(\mathbf{q}))$. We assumed that $\mathbf{p}$ and $\mathbf{q}$ are both in C_j, and connected components in $\mathcal{C}_j$ contain at most $k-1$ edges (we subtract one, because $(\mathbf{d}, \mathbf{e})$ is a link). This means $\mathbf{p}, \mathbf{q}$ are at distance at most $k - 1 < 2n$, and thus $(\mathbf{p}, \mathbf{q})$ is an r-edge in $\mathcal{I}_n$ by Lemma 7. As $g(\mathbf{p})$ and $g(\mathbf{q})$ are both in $D'_{j+1} \in \mathcal{C}_{j+1}$, they are at distance at most k. This means that $(\mathbf{p}', \mathbf{q}')$ is an r-edge in $\mathcal{I}_n$.

Case 3. By definition, $(\mathbf{p}', \mathbf{q}') = (\mathbf{p}, g(\mathbf{q}))$. As g preserves (n_{j+1})-neighbourhoods of elements within distance $|C_j|$ from $\mathbf{e}$ and the distance between $\mathbf{q}$ and $\mathbf{e}$ is at most $|C_j|$, we see that $N^{\mathcal{I}_n}_{n_{j+1}}(\mathbf{q}) \simeq N^{\mathcal{I}_n}_{n_{j+1}}(g(\mathbf{q}))$. Let us analyze the three subcases.

Subcase 3 (a). As $\mathbf{p} \in C'_j$ and $\mathbf{q} \in C_j$, and C'_j and C_j are different connected components in $\mathcal{C}_j$, we have that $(\mathbf{p}, \mathbf{q})$ is a link along r. Since C'_j and C_j are merged together into D'_{j+1}, both $\mathbf{p}$ and $g(\mathbf{q})$ are in D'_{j+1}. This means they are at distance at most k from each other, which allows us to use Lemma 7 to deduce that $(\mathbf{p}', \mathbf{q}')$ is an r-edge in $\mathcal{I}_n$.

Subcase 3 (b). Like above, $\mathbf{p}$ and $\mathbf{q}$ are in different connected components from $\mathcal{C}_j$, so it must be that $(\mathbf{p}, \mathbf{q})$ is a link along r. We need a different line of reasoning than in the previous case, since we cannot assume C_j and C''_j are merged together. Since $(\mathbf{p}, \mathbf{q})$ is an n_j-link, it is also an (n_{j+1})-link. We have two subcases to consider.

Suppose that $\mathbf{p}$ has an r-successor $\mathbf{f}$ in $\mathcal{I}'_n$ such that $N^{\mathcal{I}_n}_{n_{j+1}}(\mathbf{f}) \simeq N^{\mathcal{I}_n}_{n_{j+1}}(\mathbf{q}) \simeq N^{\mathcal{I}_n}_{n_{j+1}}(g(\mathbf{q}))$. We see that then $(\mathbf{p}, g(\mathbf{q}))$ is an (n_{j+1})-link. If $\mathbf{f} = g(\mathbf{q}))$, then $(\mathbf{p}', \mathbf{q}')$ is an r-edge in $\mathcal{I}_n$. Furthermore, the connected component C''_j of $\mathbf{p}$ merges with C'_j; all three components C_j, C'_j and C''_j form subgraphs of D'_{j+1}. As elements in D'_{j+1} are at distance at most k from each other, there can be no links between them by Lemma 7. This means that all the links between C''_j and C_j used by h_j become regular edges in h_{j+1}. We also see that h_j cannot use any links between C''_j and C'_j, as elements of these components are at distance at most k from each other in $\mathcal{I}'_n$. The other possibility is that $\mathbf{f} \neq g(\mathbf{q}))$, in which case $(\mathbf{p}', \mathbf{q}')$ is a link along r. By analogous reasoning, this also entails that h_{j+1} uses links for all the links used by h_j which join C_j and the connected component of $\mathbf{f}$.

Suppose now that $\mathbf{q}$ has an r-predecessor $\mathbf{f}$ in $\mathcal{I}'_n$ such that $N^{\mathcal{I}_n}_{n_{j+1}}(\mathbf{f}) \simeq N^{\mathcal{I}_n}_{n_{j+1}}(\mathbf{p})$. We prove in the same way as above that either $(\mathbf{p}', \mathbf{q}')$ is an r-edge in $\mathcal{I}'_n$ and the connected component of $\mathbf{p}$ and C'_j merge, or $(\mathbf{p}', \mathbf{q}')$ is a link along r.

Subcase 3 (c). $(\mathbf{p}, \mathbf{q})$ is an r-edge in $\mathcal{I}_n$ by Lemma 6. As $N^{\mathcal{I}_n}_{n_{j+1}}(\mathbf{q}) \simeq N^{\mathcal{I}_n}_{n_{j+1}}(g(\mathbf{q}))$, $(\mathbf{p}, g(\mathbf{q}))$ must be an r-edge in $N^{\mathcal{I}_n}_{n_{j+1}}(g(\mathbf{q}))$. Thus, $(\mathbf{p}', \mathbf{q}')$ is an r-edge in $\mathcal{I}_n$.

This proves that h_{j+1} is a homomorphism, which uses links for a subset of these binary atoms of P, for which h_j uses links and which do not join C_j with C'_j.

C Proof of Theorem 3

C.1 Construction

C.1.1 Outline The authors of [6] present a product of tree automata that independently verify the following acceptance conditions:

- consistency of the forest encoding,
- satisfying the constraints from the knowledge base $\mathcal{K}^* = (\mathcal{T}, \mathcal{A})$,
- not satisfying the query Q,
- path safety.

We enhance their construction by adapting these components and adding verification of individual- and nominal-completeness. Our automaton is parametrized by a fixed subinterpretation $\mathcal{M}$ induced by individuals (note that it is not the case in [6]).

We describe the construction with $\mathcal{SOI}$ in mind, but it adapts easily to $\mathcal{SOF}$. The only differences are that for $\mathcal{SOF}$, we do not validate parts referring to inverse roles, and we add another component verifying functionality declarations, which does not affect our size estimations. This component is decribed in the last subsection.

C.1.2 Preprocessing The interpretations we work with are preprocessed to cater for transitivity: for an interpretation $\mathcal{I}$, if $\mathcal{I} \models \mathcal{K}$, then also $\mathcal{I}^* \models \mathcal{K}$. This is achieved by supplementing the interpretation with auxiliary concepts expressing universal restrictions on transitive roles, as described under Definition 5: for each pair of $B \in \mathsf{CN}(\mathcal{K})$ and a transitive role r, we add a fresh concept name $P_{r,B}$, axiomatised as $P_{r,B} \sqsubseteq \forall r.(B \sqcap P_{r,B})$, and replace each CI of the form $A \sqsubseteq \forall r.B$ with $A \sqsubseteq P_{r,B}$.

The information about the edges to and from nominals is also stored in auxiliary concepts: for each $a \in \mathsf{Nom}(\mathcal{K})$ and $r \in \mathsf{Rol}(\mathcal{K})$ fresh concept names $N_{r,a}$ and $N_{r^-,a}$ are added to the knowledge base, axiomatised as $N_{r,a} \equiv \exists r.\{a\}$ and $\{a\} \equiv \forall r.N_{r^-,a}$. These concepts do not substantially increase the size of the automaton.

We aim to accept only these counter-examples which are complete. First, we tackle nominal-completeness. For each $a \in \mathsf{Nom}(\mathcal{K})$ and $r \in \mathsf{TRol}(\mathcal{K})$ we add the following concept inclusions to TBox: $\bar{N}_{r,a} \sqsubseteq \forall r.\bar{N}_{r,a}$ and $N_{r^-,a} \sqsubseteq \forall r.N_{r^-,a}$.

This ensures that whenever for a transitive role r there is no edge r from some element to a nominal (or vice versa), then there is also no r-path from that element to the nominal.

Recall that we assume the completeness of $\mathcal{M}$ (the subinterpretation induced by individuals). Consequently, any path along a transitive role between two individuals not connected with a single edge must enter some clique-tree and lead through a nominal. Additional CIs described in the last paragraph ensure that whenever there is a path from an individual to a nominal in the clique-tree associated with that individual, these elements are also connected with a single edge. If there was a path breaking individual-completeness of the whole forest, it would entail that also nominal-completeness does not hold. Thus, individual-completeness follows from nominal-completeness.

C.1.3 Automaton To make clique-forests accessible to automata, we encode them as finitely labelled forests. Let $\mathsf{TRol}(\mathcal{K})$ be the set of transitive roles from $\mathsf{Rol}(\mathcal{K})$, and let $[X]^{\leq k}$ be the family of subsets of X of size at most k. In the encoding, nodes are labelled with elements of the alphabet

$$\Sigma = \mathsf{Tp}(\mathcal{K}) \cup \Big(\mathsf{TRol}(\mathcal{K}) \times [\mathsf{Tp}(\mathcal{K})]^{\leq |\mathsf{CN}(\mathcal{K})|}\Big)$$

and edges are labelled with elements of the alphabet

$$\Gamma = \mathsf{Tp}(\mathcal{K}) \times \mathsf{Rol}(\mathcal{K}) \times \mathsf{Tp}(\mathcal{K})\,.$$

We build an automaton parametrized by the subinterpretation $\mathcal{M}$. Let $\mathcal{I}$ denote the interpretation whose clique-forest is encoded in the input forest. The automaton runs on the encoding of the clique-forest of $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ (not of $\mathcal{I}$, like in [6]; this change is motivated by the way we modify the query component). We say that an individual a is the *super-root* of a clique-tree t in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$, if the root of t is a successor of a in $\mathcal{I}$; we also say that t is *super-rooted* in a.

For a complete counter-example $\mathcal{I}$, we obtain the encoding of the clique-forest for $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ as follows. We fix some order on individuals and order the trees in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ so that they are sorted with respect to the order on their super-roots. We add a separating tree stub (represented with a fresh symbol added to Σ) between contiguous blocks of trees super-rooted in the same individual; it is only used as a marker for the automaton so that it can recognize which trees are associated with a given individual. All the components of the automaton always accept these tree stubs.

Next, we label each element node with the single unary type it realises and each clique node with its single nonempty role and the set of unary types it realises. We do not represent nominals explicitly in the encoding, but thanks to the initial preprocessing, all relevant information about them is contained in the unary types of the remaining elements. Finally, if in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ there is an r-edge from an element of type τ in some parent node to an element of type σ in some child node, then in the encoding the edge from the parent node to the child node is labelled with (τ, r, σ). Because unary types do not repeat within cliques, this uniquely determines the endpoints.

There are at most $N = |\mathsf{Ind}(\mathcal{K})| \cdot |\mathsf{CI}(\mathcal{K})|$ trees in the input forests, with branching bounded by $D = |\mathsf{CN}(\mathcal{K})| \cdot |\mathsf{CI}(\mathcal{K})|$. Transition relation has the form

$$\delta \subseteq Q \times \Sigma \times (\Gamma \times Q)^{\leq D} ,$$

where Q is the set of states. The automaton processes the forests top down. Whenever a node is reached after an edge labelled (τ, r, σ), we refer to the unique element of type τ in that node as the *current element*, and to the unique element of type σ in the parent node as the *parent element*. The initial states are specified for each tree separately: the automaton has a set $I \subseteq Q^{\leq N}$ of sequences of initial states. A run is a labelling of the input forest with states in such a way that the sequence of states in the roots belongs to I, and if a node has state q, label α, and its children are connected via edges with labels $\beta_1, \beta_2, \ldots, \beta_n$ and have states $q_1, q_2, \ldots, q_n$, then

$$\Big(q, \alpha, \big((\beta_1, q_1), \ldots, (\beta_n, q_n)\big)\Big) \in \delta .$$

We use *Büchi acceptance condition*: we specify a set $F \subseteq Q$ of marked states that need to be revisited, and consider a run accepting if on each branch marked states occur infinitely often. A forest is *accepted* by the automaton if there exists an accepting run over it.

An automaton has *trivial acceptance condition* if $F = Q$. Then, each run is accepting but the automaton may still reject some forests, because there may be no run for them: a branch of the computation can get stuck if no transition is consistent with the current state, label and edge labels. An automaton is *weak* if on each branch of each run, once a marked state is visited, all subsequent states are marked. Notice that all automata with trivial acceptance condition are weak. Given a weak automaton and an arbitrary Büchi automaton it is particularly easy to construct an automaton recognising trees accepted by both input automata: it suffices to take the standard (synchronous) product automaton and mark all states that contain a marked states on both coordinates.

C.1.4 Automata components The final automaton is obtained as a product of automata verifying independently various parts of the condition.

The first thing to check is the consistency of the encoding: if an edge has label (τ, r, σ), then τ must occur in the label of the parent node, and σ must occur in the label of the child node. To check this, it suffices to examine for each node the labels of all edges incident to it plus the label of the node itself. When a transition is made, all these are available except the label on the edge to the parent: it must be stored in the state. The automaton has $|\Gamma|$ states and trivial acceptance condition.

The second thing to check is that the clique-forest is a model of $\mathcal{K}^*$. If $\mathcal{M}$ does not satisfy the restrictions from the ABox $\mathcal{A}$ of $\mathcal{K}^*$, the final automaton rejects everything.

To verify that the TBox is satisfied we need to check each CI. For CIs of the form

$$\bigsqcap_i A_i \sqsubseteq \bigsqcup_j B_j$$

we have a two-state automaton with trivial acceptance condition that simply tests that each type used in the encoding satisfies this CI. If the type of some individual violates this CI, the final automaton rejects everything. To verify CIs of the form

$$A \sqsubseteq \forall r.B$$

for elements in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$, it suffices to check that in the input interpretation there is no r-edge from an element whose unary type contains A to an element whose unary type contains $\bar{B}$. This amounts to verifying that none of the following are used in the encoding:

- node labels (r, T) such that $A \in \tau \in T$ and $\bar{B} \in \sigma \in T$ for some τ, σ;
- edge labels (τ, r, σ) with $A \in \tau$ and $\bar{B} \in \sigma$;
- edge labels (σ, r^-, τ) with $A \in \tau$ and $\bar{B} \in \sigma$;
- unary types containing both A and $N_{r,b}$ for some b such that $b \in \bar{B}^{\mathcal{M}}$;
- unary type containing both $\bar{B}$ and $N_{r^-,b}$ for some b such that $b \in A^{\mathcal{M}}$.

These conditions simply disallow certain labels; they can be checked by a two-state automaton with a trivial acceptance condition.

Let us take a CI of the form

$$A \sqsubseteq \exists r.B\,.$$

For elements in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$, this condition can be tested in a similar way as above, except that one needs access to the label of the current node and all edges incident to it. Like for the initial consistency check, it suffices to store in the state the label of the edge to the parent. To ensure the existential restrictions are met for nominals, for each nominal a such that $a \in A^{\mathcal{M}}$, if that CI is not met in $\mathcal{M}$, we construct a two-state weak automaton looking for a label that uses a type τ such that $B \in \tau$ and $N_{r^-,a} \in \tau$. Note that this automaton has a non-trivial acceptance condition, but it is weak: as soon as it finds an appropriate label, it loops in a marked state. The initial lists of states for that weak consist of a single unmarked state and all the remaining initial states already marked.

We also have to verify that all existential and universal restrictions are met for non-nominal individuals. The former are ensured by constraining initial lists of states so that all the existential restrictions for an individual not satisfied in $\mathcal{M}$ are satisfied by roots of clique-trees super-rooted in that individual. If the universal restrictions are not met in $\mathcal{M}$, the final automaton rejects everything; otherwise, we ensure that the restrictions hold for non-individual elements connected to individuals by constraining initial lists of states so that the roots of clique-trees super-rooted in a given individual do not break universal restrictions imposed on that individual.

Summing up, the total size of the state-space of the TBox component is not larger than $2^{|\mathsf{CI}(\mathcal{K})|} \cdot 2^{|\mathsf{CI}(\mathcal{K})|} \cdot |\Gamma|^{|\mathsf{CI}(\mathcal{K})|} \cdot 2^{|\mathsf{Nom}(\mathcal{K})| \cdot |\mathsf{CI}(\mathcal{K})|}$. The first three factors stem from the sizes of automata used to verify on $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$ all the CIs of the forms $\bigsqcap_i A_i \sqsubseteq \bigsqcup_j B_j$, $A \sqsubseteq \forall r.B$, and $A \sqsubseteq \exists r.B$, respectively. The last factor stems from the verification of existential restrictions for nominals.

Next, we verify that the query cannot be satisfied. We need an automaton recognizing counter-examples such that the number of states in the automaton does not depend on the number of individuals in $\mathcal{K}$. This is achieved by making sure each tree in the forest is processed without explicitly referring to the knowledge about individuals.

In [6], all the information about edges between individuals is transferred to the TBox. In particular, this means that the query parts are self-contained in single trees, as individuals are not connected (and the edges to nominals are hidden from the automata perspective). It lets the query component build partial query matchings independently for each tree, but at the cost of enlarging the TBox.

Here, we cannot use the same approach: the number of states depends on the number of unary types in particular, which would be tied to the ABox size if we were to store the information about edges between individuals in concepts. Matchings of connected parts of the query are no longer contained in single trees: they might span over multiple trees. We have to accommodate for that by enhancing the query component.

We start by replacing query Q with a query Q' such that for each model $\mathcal{I}$ of $\mathcal{K}$: $\mathcal{I}^* \models Q$ iff $(\mathcal{I} \setminus \mathsf{Nom}(\mathcal{K}))^* \models Q'$. The query Q' is obtained in two steps.

In the first step, we account for the possibility that binary atoms of the form $r(x, y)$ for transitive r can be satisfied with paths that cross different trees. Take such path π, and let a and b denote respectively the first and the last individual on π (possibly $a = b$). Thanks to the individual-completeness property, π can be reduced to a path consisting of: a path segment ending in a and contained in a single tree, a single edge from a to b (if $a \neq b$), and another path segment contained in a single tree. To detect such paths, for each CQ P constituting Q, we add to Q each CQ that can be obtained from P by subdividing some transitive atoms; that is, by replacing some atoms of the form $r(x, y)$ either with atoms $r(x, z)$ and $r(z, y)$ for a fresh variable z, or with atoms $r(x, z_1)$, $r(z_1, z_2)$, $r(z_2, y)$ for fresh variables z_1, z_2.

In the second step, we account for the influence of nominals on the query satisfaction. For each CQ P of the modified Q, we add to Q each CQ that can be obtained from P by performing the following operation any number of times. Let $\mathsf{tp}(x)$ be the set of all A such that P contains $A(x)$. Choose $x \in \mathit{var}(P)$ and $a \in \mathsf{Nom}(\mathcal{K})$ such that $a \in A^{\mathcal{M}}$ whenever $A \in \mathsf{tp}(x)$. Drop all atoms of the form $A(x)$ from P. Replace in P each atom of the form $r(y, x)$ by $N_{r,a}(y)$ and each atom of the form $r(x, y)$ by $N_{r^-,a}(y)$.

The resulting query Q' satisfies the condition $\mathcal{I}^* \models Q$ iff $(\mathcal{I} \setminus \mathsf{Nom}(\mathcal{K}))^* \models Q'$. After the first step, the number of CQs grows by the factor 3^m and their size is at most $3m$. After the second step, the number of CQs grows by the factor $|\mathsf{Nom}(\mathcal{K})|^{3m}$ and their size is still at most $3m$. Thus, the size of the resulting query is at most $|Q| \cdot 3^m \cdot |\mathsf{Nom}(\mathcal{K})|^{3m}$, and its CQs have size at most $3m$.

For each CQ P of the preprocessed union of queries Q', we construct an automaton that ensures that $\mathcal{I}^* \not\models P$. Its states consist of the parent edge label β and a set F representing all partial matchings of P to elements represented in the subtree rooted in the currently processed node.

The set F consists of partial functions $f : \mathit{var}(P) \to L$, with $L = \{\mathsf{succ}, \mathsf{other}\}$, representing different ways variables can be assigned in relation to the parent element. Each such function represents a partial matching, that is a partial assignment of variables such that the restricted query induced by these variables is satisfied (not all the collected partial matchings can potentially be extended to a total match; discussion on that part is presented in the next subsection). The identifier succ is used for each variable mapped to an element in the current subtree that is an r-successor of the parent element in $\mathcal{I} \setminus \mathsf{Ind}(\mathcal{K})$. If r is non-transitive, this simply means the current element (the clique origin, if we are processing a clique node). If r is transitive, it means any element r-reachable from the current element. The identifier other is used for variables assigned to any other elements in the current subtree.

The rationale behind the automaton is that it constructs all the possible partial matchings across the forest. The trick here is that the states reached before traversing a given node already contain the information about all the partial matchings that will be constructed in the subtree rooted at that node. The automaton will have at most one accepting run on each forest—the states in the initial list of states in that run will contain all the possible partial matchings across the forest.

We define the transition relation in the next subsection. The definition will ensure the intended semantics of the states; when processing a given node, sets F in the automaton state will represent all partial matchings of P to elements represented in the subtree rooted in that node.

All states are accepting. Apart from the partial matchings collected in trees rooted in children nodes of individuals, we also compute all partial matchings in $\mathcal{M} \setminus \mathsf{Nom}(\mathcal{K})$ (the possibility that some variables are assigned to nominals has already been accounted for in the second step of the query preprocessing). As lists of initial states, we take all lists of states such that a total match over the whole forest cannot be constructed by fusing together a partial matching in $\mathcal{M} \setminus \mathsf{Nom}(\mathcal{K})$ and one partial matching per initial state. We know how to stitch these partial matchings together, because we know the edges from super-roots to new clique-tree roots and the way we collect partial matchings in a clique-tree accounts for the label on that edge. This sums up the query automaton construction. This extra information about the edges from super-roots motivates the approach of traversing clique forests built of trees rooted in children nodes of individuals, rather than trees rooted in individuals.

The last component of the automaton checks that the input interpretation is safe. Observe that it is unsafe if in the input clique-forest there is a branch with consecutive node and edge labels $\alpha_1\beta_1\alpha_2\beta_2\ldots$ such that for some transitive r and all i large enough, $\beta_i = (\tau_i, r, \sigma_i)$ and either $\sigma_i = \tau_{i+1}$ (consecutive edges are incident in the clique-forest) or $\alpha_{i+i} = (r, T_i)$ (consecutive edges are incident with an r-clique). An automaton can easily check that there is no such branch. Each time it sees a transitive role it moves to an unmarked state, storing the role. It moves to a marked state as soon as the condition above is broken. The automaton has $|\mathsf{TRol}(\mathcal{K})|$ states.

The automaton recognising safe counter-examples can be obtained from these components by the simple product construction described above, because only the last component is not weak.

C.2 Transition relation of the query component

Below we describe the set of transitions of the query satisfaction component of the product automaton.

A transition

$$((\beta, F), \tau, ((\beta_i, (\beta_i, F_i))_{i=1}^{n})$$

over an element node of unary type τ, with $\beta = (\sigma, r, \tau)$ and $\beta_i = (\tau, r_i, \tau_i)$, is a transition of the automaton if F is the set of all functions $f : \mathit{var}(P) \to L$ which are both upward- and downward-compatibile with some witnessing functions $f_1 \in F_1, f_2 \in F_2, ..., f_n \in F_n$; we define these conditions below.

Fix a function $f : \mathit{var}(P) \to L$ and a sequence of domain-disjoint functions $(f_1, f_2, ..., f_n)$, with $f_i : \mathit{var}(P) \to L$.

f is *upward-compatibile* with functions f_i if for each variable $x \in \mathit{var}(P)$:

- if $\exists i : f_i(x) = \mathsf{other}$, then $f(x) = \mathsf{other}$,
- if $\exists i : f_i(x) = \mathsf{succ}$, then $f(x) = \mathsf{succ}$ if r is transitive and $r_i = r$, otherwise $f(x) = \mathsf{other}$,
- otherwise either $x \notin \mathsf{dom}(f)$, or $f(x) = \mathsf{succ}$ and $\mathsf{tp}(x) \subseteq \tau$, which we interpret as matching x to the current element. In the latter case x is in $(\mathsf{dom}(f) \setminus \bigcup \mathsf{dom}(f_i))$.

f is *downward-compatibile* with functions f_i if for each atom $s(x, y)$ of P:

- if $x \in (\mathsf{dom}(f) \setminus \bigcup \mathsf{dom}(f_i))$ and $y \in \mathsf{dom}(f_i)$, then $r_i = s$ and $f_i(y) = \mathsf{succ}$,
- if $y \in (\mathsf{dom}(f) \setminus \bigcup \mathsf{dom}(f_i))$ and $x \in \mathsf{dom}(f_i)$, then $r_i = s^-$ and $f_i(x) = \mathsf{succ}$,
- if $x \in \mathsf{dom}(f_i)$ and $y \in \mathsf{dom}(f_j)$ for $i \neq j$, then s is transitive, $r_i = r_j^- = s$, and $f_i(x) = f_j(y) = \mathsf{succ}$.

Note that we only seem to notice binary atoms for which both variables are already present in the subtree. When a variable gets the identifier other and there are unsatisfied binary atoms containing that variable, we know that such a matching cannot be extended to a total matching. We still gather such matchings in the automaton, which might be unintuitive. For clarity, we can forget such unextendable matchings in the automaton runs by adding the following condition to the downward-compatibility definition:

- if $f_i(x) = \mathsf{other}$ or $f_i(y) = \mathsf{other}$, then $x, y \in \mathsf{dom}(f_i)$.

The transitions over clique-nodes are described using analogous compatibility concepts, adapted to the clique-node setting. A transition

$$((\beta, F), (r', T), ((\beta_i, (\beta_i, F_i))_{i=1}^{n})$$

over a clique node encoded by a role r' and a set of unary types T, with $\beta = (\sigma, r, \tau)$ and $\beta_i = (\sigma_i, r_i, \tau_i)$, is a transition of the automaton if F is the set of all functions $f : \mathit{var}(P) \to L$ which are both upward- and downward-compatibile with some witnessing functions $f_1 \in F_1, f_2 \in F_2, ..., f_n \in F_n$; we define these conditions below.

Fix a function $f : \mathit{var}(P) \to L$ and a sequence of domain-disjoint functions $(f_1, f_2, ..., f_n)$, with $f_i : \mathit{var}(P) \to L$.

f is *upward-compatibile* with functions f_i if for each variable $x \in \mathit{var}(P)$:

- if $\exists i : f_i(x) = \mathsf{other}$, then $f(x) = \mathsf{other}$,
- if $\exists i : f_i(x) = \mathsf{succ}$, then $f(x) = \mathsf{succ}$ if r is transitive, $r_i = r$, and either $\sigma_i = \tau$ or $r' = r$, otherwise $f(x) = \mathsf{other}$,
- otherwise one of the following holds:
  - $x \notin \mathsf{dom}(f)$,
  - $f(x) = \mathsf{succ}$ and $\mathsf{tp}(x) \subseteq \tau$, which we interpret as matching x to the current element,
  - $f(x) = \mathsf{other}$ and $\mathsf{tp}(x) \subseteq \tau' \in T \setminus \{\tau\}$, which we interpret as matching x to an element in the current clique-node.

f is *downward-compatibile* with functions f_i if for each atom $s(x, y)$ of P:

- if $x \in (\mathsf{dom}(f) \setminus \bigcup \mathsf{dom}(f_i))$ and $y \in \mathsf{dom}(f_i)$, then $r_i = s$, $f_i(y) = \mathsf{succ}$, and either $\mathsf{tp}(x) \subseteq \sigma_i$, or s is transitive and $r' = s$,
- if $y \in (\mathsf{dom}(f) \setminus \bigcup \mathsf{dom}(f_i))$ and $x \in \mathsf{dom}(f_i)$, then $r_i = s^-$, $f_i(x) = \mathsf{succ}$, and either $\mathsf{tp}(y) \subseteq \sigma_i$, or s is transitive and $r' = s$,
- if $x \in \mathsf{dom}(f_i)$ and $y \in \mathsf{dom}(f_j)$ for $i \neq j$, then s is transitive, $r_i = r_j^- = s$, $f_i(x) = f_j(y) = \mathsf{succ}$, and either $\sigma_i = \sigma_j$ or $r' = s$.

Once again, we can enhance the downward-compatibility definition to get rid of unextendable partial matchings; it is enough to add the same condition as before.

To see that this transition relation ensures the intended semantics of the states one needs to argue that each partial matching is accurately represented. This can be done by induction on the size of the image. For size one, the matching will be accounted for based solely on the labels. For larger images, use the inductive hypothesis for restrictions of the match to variables mapped to the trees rooted at the children of the current node.

C.3 Size estimation

Let us approach the size estimation of the automaton state space. We start by analyzing the factors that contribute to the size of automata components. Note that k and m denote, respectively, the maximum number of binary atoms and the maximum size of a single conjunctive query in the union of CQs Q.

Measured entity	Notated as	Size estimation
number of unary types	$\lvert\mathsf{Tp}(\mathcal{K})\rvert$	$2^{\lvert\mathsf{CN}(\mathcal{K})\rvert+\lvert\mathsf{Nom}(\mathcal{K})\rvert\cdot\lvert\mathsf{TRol}(\mathcal{K})\rvert}$
size of node alphabet	$\lvert\Sigma\rvert$	$\mathcal{O}(\lvert\mathsf{TRol}(\mathcal{K})\rvert\cdot\lvert\mathsf{Tp}(\mathcal{K})\rvert^{\lvert\mathsf{CN}(\mathcal{K})\rvert})$
size of edge alphabet	$\lvert\Gamma\rvert$	$\lvert\mathsf{Tp}(\mathcal{K})\rvert^2\cdot\lvert\mathsf{TRol}(\mathcal{K})\rvert$
number of trees in a forest	N	$\lvert\mathsf{Ind}(\mathcal{K})\rvert\cdot\lvert\mathsf{CI}(\mathcal{K})\rvert$
clique-trees branching	D	$\lvert\mathsf{CI}(\mathcal{K})\rvert\cdot\lvert\mathsf{CN}(\mathcal{K})\rvert$
size of preprocessed query	$\lvert Q'\rvert$	$\lvert Q\rvert\cdot 3^m\cdot\lvert\mathsf{Nom}(\mathcal{K})\rvert^{3m}$

The bound of the size of the unary types set is not just $2^{\lvert\mathsf{CN}(\mathcal{K})\rvert}$ due to the encoding of neighbouring nominals, which requires $\mathcal{O}(\lvert\mathsf{Nom}(\mathcal{K})\rvert\cdot\lvert\mathsf{Rol}(\mathcal{K})\rvert)$ new concepts and concept inclusions.

The size estimation of the automata components is presented below.

Component	States count upper bound
consistency of encoding	$\lvert\Gamma\rvert$
TBox constraints:	
- CIs of form: $\sqcap A_i \sqsubseteq \sqcup B_j$	$2^{\lvert\mathsf{CI}(\mathcal{K})\rvert}$
- CIs of form: $A \sqsubseteq \forall r.B$	$2^{\lvert\mathsf{CI}(\mathcal{K})\rvert}$
- CIs of form: $A \sqsubseteq \exists r.B$	$\lvert\Gamma\rvert^{\lvert\mathsf{CI}(\mathcal{K})\rvert}\cdot 2^{\lvert\mathsf{Nom}(\mathcal{K})\rvert\cdot\lvert\mathsf{CI}(\mathcal{K})\rvert}$
path safeness	$\lvert\mathsf{TRol}(\mathcal{K})\rvert$
query unsatisfaction	$2^{3^{3m}\cdot\lvert Q'\rvert}$

By taking a product of the sizes of automata components, we bound the number of states of the product automaton by $2^{2\cdot 3^{4m}\cdot\lvert Q\rvert\cdot\lvert T\rvert^{3m}}$, where $\lvert T\rvert = \lvert\mathsf{CI}(\mathcal{K})\rvert + \lvert\mathsf{CN}(\mathcal{K})\rvert + \lvert\mathsf{Nom}(\mathcal{K})\rvert\cdot\lvert\mathsf{TRol}(\mathcal{K})\rvert$. The factor 2 in the exponent is introduced to eliminate summands of lower order. The query component makes the size of the state-space of the product automaton double exponential. This finalizes the proof of Theorem 3.

As a side note, an automaton with k states has total size $\mathcal{O}(k\cdot\lvert\Sigma\rvert\cdot(k\cdot\lvert\Gamma\rvert)^N + k^N)$, which in our case can be expanded to $2^{\lvert Q\rvert\cdot\lvert\mathcal{K}\rvert^{\mathcal{O}(m)}}$.

C.4 Functionality component for $\mathcal{SOF}$

We include an additional component for each functionality declaration $\mathsf{Fn}(r)$. To check functionality of r for ordinary nodes it suffices to examine the label of the node and the labels on all incident edges, which only requires storing in the state the label of the edge to the parent. Additionally, for all $a \in \mathsf{Nom}(\mathcal{K})$: if $a \in (N_{r,b})^{\mathcal{M}}$ and $a \in (N_{r,b'})^{\mathcal{M}}$ for some $b \neq b'$, the final automaton rejects everything; if $a \in (N_{r,b})^{\mathcal{M}}$, the automaton checks that no type used in the input forest contains $N_{r^-,a}$; if $a \notin (N_{r,b})^{\mathcal{M}}$ for each $b \in \mathsf{Nom}(\mathcal{K})$, the automaton checks that a type with $N_{r^-,a}$ occurs at most once in the input forest. The total number of states in the described component is $\lvert\Gamma\rvert^{\lvert\mathsf{CI}(\mathcal{K})\rvert}\cdot 2^{\lvert\mathsf{Nom}(\mathcal{K})\rvert\cdot\lvert\mathsf{CI}(\mathcal{K})\rvert}$, so including it does not affect the overall upper bound.

D Proof of Lemma 2

For each transitive role s, all the s-atoms in Q are replaced with disjunctions of s-chains of lengths from 1 to ℓ in Q^*. If such an atom can be satisfied in $\mathcal{J}^*$ for elements a and b, then there is an s-path from a to b of length at most ℓ. This shows that if $\mathcal{J} \models Q^*$, then $\mathcal{J}^* \models Q$. On the other hand, if $\mathcal{J}^* \models Q$, then each transitive role atom must be satisfied by two endpoints of some s-path from $\mathcal{J}$, of length at most ℓ, which satisfies the subquery replacing this atom in Q^*. This proves the converse implication.