# The Impact of Transitive Closure on the Boolean Expressiveness of Navigational Query Languages on Graphs

George H.L. Fletcher[1], Marc Gyssens[2], Dirk Leinders[2,⋆], Jan Van den Bussche[2], Dirk Van Gucht[3], Stijn Vansummeren[4], and Yuqing Wu[3]

[1] Eindhoven University of Technology
g.h.l.fletcher@tue.nl
[2] Hasselt University and Transnational University of Limburg
School for Information Technology
{marc.gyssens,dirk.leinders,jan.vandenbussche}@uhasselt.be
[3] Indiana University
{vgucht,yuqwu}@cs.indiana.edu
[4] Université Libre de Bruxelles
stijn.vansummeren@ulb.ac.be

**Abstract.** Several established and novel applications motivate us to study the expressive power of navigational query languages on graphs, which represent binary relations. Our basic language has only the operators union and composition, together with the identity relation. Richer languages can be obtained by adding other features such as other set operators, projection and coprojection, converse, and the diversity relation. In this paper, we show that, when evaluated at the level of boolean queries with an unlabeled input graph (i.e., a single relation), adding transitive closure to the languages with coprojection adds expressive power, while this is *not* the case for the basic language to which none, one, or both of projection and the diversity relation are added. In combination with earlier work [10], these results yield a complete understanding of the impact of transitive closure on the languages under consideration.

## 1 Introduction

In previous work [10], the present authors studied the relative expressive power of query languages on graphs (i.e., binary relations). They considered a basic language, consisting of union, composition, and the identity relation, to which one or more features can be added, such as intersection, set difference, projection, coprojection, converse, and the diversity relation. We refer to the basic language to which all the non-basic features have been added as the *relation algebra*.

A relation algebra expression can be seen as a function mapping the input binary relation to a binary relation. We call such queries *path queries* because the result can be interpreted as all the ways in which the input graph can be

⋆ This author carried out most of his research as a Senior Research Assistant of the Fund of Scientific Research – FWO Flanders.

T. Lukasiewicz and A. Sali (Eds.): FoIKS 2012, LNCS 7153, pp. 124–143, 2012.

navigated in accordance with the expression. By identifying nonemptiness with the boolean value *true* and emptiness with *false*, as is standard in database theory [3], we can also express yes/no queries within this framework. To distinguish them from general path queries, we shall refer to the latter as *boolean queries*.

The present authors were able to establish the complete Hasse diagram for the relative expressive power of the various relation algebra fragments, and this both at the levels of (1) path queries and (2) boolean queries, both for the cases where the input graph is (1) labeled (i.e., may represent multiple binary relations) and (2) is unlabeled (i.e., represents a single relation).

This study was motivated by similar work on the expressive power of XPath fragments as query languages for navigating on trees, which is now well understood (e.g., [7,13,20,21,25]). Motivated by data on the Web [2,11] and new applications such as dataspaces [12], Linked Data [8,15], and RDF [1], it is natural to look at similar navigational query languages for graphs. The languages we study are very natural and similar to languages already considered in the fields of description logics, dynamic logics, arrow logics, and relation algebras [6,9,14,17,19,23]. Moreover, graph query languages have a rich history in database theory, in particular in the context of object-oriented and semi-structured database systems. We refer to Angles and Gutiérrez [5] for a comprehensive review.

In addition to what has been described above, we also investigated whether adding transitive closure to a relation algebra fragment yields additional expressive power. At the level of path queries, this is obviously the case for all fragments, as the transitive closure of a binary relation is not expressible in FO [4], whereas the full relation algebra is known to be equivalent to $FO^3$ [24]. We were also able to show [10] that adding transitive closure does not result in a collapse at the level of boolean queries, provided the input graph is labeled (i.e., there may be several input relations). For boolean queries on unlabeled graphs (i.e., on a single input relation), several cases remained open, however.

The purpose of the present paper is to solve these cases, and thus to complete our understanding of whether or not the relation algebra fragments with transitive closure collapse to their counterparts without transitive closure at the level of boolean queries on unlabeled graphs.

To see the practical relevance of these results, consider the following example. Facebook is a large social network which maintains a graph of people that are connected via a friendship relationship. It is customary that people wish to communicate with their friends, navigate recursively to friends of friends etc. This navigation can be expressed with path expressions in a suitable relation algebra fragment, either with or without using transitive closure. In addition to navigation, certain topological properties of the Facebook graph can be discovered. For example, one can discover whether there are people whose friends are all friends of each other. Again, some of these topological properties can be formulated as boolean queries in a suitably chosen relation algebra fragment, either with or without using transitive closure. The proliferation of social networks is thus a real-world phenomenon to which our theory applies.

From this perspective, the collapse results are very meaningful.

With regard to the possible collapse of the various relation algebra fragments with transitive closure to their counterparts without transitive closure at the level of boolean queries on unlabeled graphs, it was already established [10] that adding transitive closure to a relation algebra fragment adds expressive power if either intersection, or converse, or both, can be expressed in that fragment. In particular, this is the case when set difference is in the fragment.

It thus remained to look at relation algebra fragments consisting of the basic language built from union, composition, and the identity relation, to which a subset of the features projection, coprojection, and the diversity relation has been added. It was also established [10] that adding transitive closure to (1) the basic language, (2) the basic language to which projection is added, and (3) the basic language to which the diversity relation is added does *not* increase the expressive power.

Taking into account that projection can be expressed as the coprojection of the coprojection, three cases remained open, however. Does adding transitive closure to (1) the basic language to which coprojection is added, (2) the basic language to which both coprojection and the diversity relation are added, and (3) the basic language to which both projection and the diversity relation are added increase the expressive power?

In this paper, we show that there is no collapse for the first two fragments, but that there is a collapse for the third fragment. The emphasis of this paper is on the proof technique used for establishing the collapse for this last fragment, which we think is interesting in its own right.

The paper is organized as follows. In Section 2, we define syntax and semantics of the class of languages studied in the paper. In Section 3, we show that for any relation algebra fragment in which coprojection can be expressed, adding transitive closure yields additional expressive power at the level of boolean queries, thus settling the previously open cases for (1) the basic language to which coprojection is added and (2) the basic language to which both coprojection and the diversity relation are added. In Section 4, we describe a two-step proof strategy to show that adding transitive closure to (3) the basic language to which both projection and the diversity relation are added does not increase the expressive power, and we deal with the first step. In Sections 5 to 8, we deal with the much more elaborate second step. We conclude in Section 9 by summarizing our understanding of the impact of adding transitive closure to relation algebra fragments, which has now been completed.

Finally, notice that, in this extended abstract, most proofs have either been omitted or only summarily sketched.

## 2 Graphs and Languages

In this paper, we are interested in navigating over graphs. For our purposes, a graph is a relational structure $G$, consisting of a set of nodes $V$ and a binary relation $R \subseteq V \times V$, the set of edges of $G$. In what follows, both $V$ and $R$ may be either finite or infinite.

An extension of this model consists of allowing multiple binary relations, by labeling the edges.[1] For comparison, we shall sometimes refer to labeled graphs, though the emphasis of this paper is on unlabeled graphs.

The most basic language for navigating over graphs we consider is the algebra $\mathcal{N}$ whose expressions are built recursively from the edge set symbol $R$, the primitive $\emptyset$, and the primitive $id$, using composition $(e_1 \circ e_2)$ and union $(e_1 \cup e_2)$.[2]

Semantically, each expression $e \in \mathcal{N}$ defines a path query. A path query takes as input a graph $G$ and returns a binary relation $e(G) \subseteq \mathrm{adom}(G) \times \mathrm{adom}(G)$, where $\mathrm{adom}(G)$ denotes the *active domain* of $G$, which is the set of all entries occurring in one of the relations of $G$, i.e.,

$$\mathrm{adom}(G) = \{v \mid \exists w : (v,w) \in R \vee (w,v) \in R\}.$$

In particular, the semantics of $\mathcal{N}$ is inductively defined as follows:

$$\begin{array}{ll} R(G) & = R\,; \\ \emptyset(G) & = \emptyset\,; \\ id(G) & = \{(v,v) \mid v \in \mathrm{adom}(G)\}\,; \\ e_1 \circ e_2(G) & = \{(v,w) \mid \exists z : (v,z) \in e_1(G)\ \&\ (z,w) \in e_2(G))\}\,; \\ e_1 \cup e_2(G) & = e_1(G) \cup e_2(G)\,. \end{array}$$

The basic algebra $\mathcal{N}$ can be extended by adding some of the following features: diversity ($di$), converse ($e^{-1}$), intersection ($e_1 \cap e_2$), difference ($e_1 \setminus e_2$), projections ($\pi_1(e)$ and $\pi_2(e)$), coprojections ($\overline{\pi}_1(e)$ and $\overline{\pi}_2(e)$), and transitive closure ($e^+$). We refer to the operators in the basic algebra $\mathcal{N}$ as *basic features*; we refer to the extensions as *nonbasic features*. The semantics of the extensions is as follows:

$$\begin{array}{ll} di(G) & = \{(v,w) \mid v,w \in \mathrm{adom}(G)\ \&\ v \neq w\}\,; \\ e^{-1}(G) & = \{(v,w) \mid (w,v) \in e(G)\}\,; \\ e_1 \cap e_2(G) & = e_1(G) \cap e_2(G)\,; \\ e_1 \setminus e_2(G) & = e_1(G) \setminus e_2(G)\,; \\ \pi_1(e)(G) & = \{(v,v) \mid v \in \mathrm{adom}(G)\ \&\ \exists w : (v,w) \in e(G)\}\,; \\ \pi_2(e)(G) & = \{(v,v) \mid v \in \mathrm{adom}(G)\ \&\ \exists w : (w,v) \in e(G)\}\,; \\ \overline{\pi}_1(e)(G) & = \{(v,v) \mid v \in \mathrm{adom}(G)\ \&\ \neg\exists w : (v,w) \in e(G)\}\,; \\ \overline{\pi}_2(e)(G) & = \{(v,v) \mid v \in \mathrm{adom}(G)\ \&\ \neg\exists w : (w,v) \in e(G)\}\,; \\ e^+(G) & = \bigcup_{k \geq 1} e^k(G)\,. \end{array}$$

Here, $e^k$ denotes $e \circ \cdots \circ e$ ($k$ times).

If $F$ is a set of nonbasic features, we denote by $\mathcal{N}(F)$ the language obtained by adding all features in $F$ to $\mathcal{N}$. For example, $\mathcal{N}(\cap)$ denotes the extension of $\mathcal{N}$ with intersection, and $\mathcal{N}(\cap, \pi, +)$ denotes the extension of $\mathcal{N}$ with intersection, both projections,[3] and transitive closure.

[1] In this case, the number of relation names is always finite.

[2] By abuse of notation, we shall use "$R$" both as a symbol in the algebra $\mathcal{N}$ and as the name of the corresponding edge relation in $G$.

[3] We do not consider extensions of $\mathcal{N}$ in which only one of the two projections, respectively one of the two coprojections, is present.

We refer to the language $\mathcal{N}(\backslash, di, ^{-1})$ as the *relation algebra*. For each set $F$ of nonbasic features considered above not containing transitive closure, all path queries expressible in $\mathcal{N}(F)$ are also expressible in the relation algebra [17].

For the purpose of showing the main result, we also consider *conditionals* as nonbasic atomic features in this paper. At the syntactic level, a conditional is an expression denoted by some constant, say $c$. The semantics of $c$ is given by some (implicit) mapping that associates to each directed graph $G$ a set $c(G)$ of identical pairs of $G$. Hence, $c(G) \subseteq id(G)$. Informally, $(v, v) \in c(G)$ means that node $v$ "satisfies" $c$ in $G$. In this paper, we shall use conditionals to eliminate projection subexpression temporarily, as explained in Section 5 and illustrated in Example 2.

Language expressiveness can be considered at the level of path queries and at the level of boolean queries.

**Definition 1.** *A* path query $q$ *is expressible in a language* $\mathcal{N}(F)$ *if there exists an expression* $e \in \mathcal{N}(F)$ *such that, for every graph* $G$*, we have* $e(G) = q(G)$*. Similarly, a* boolean query $q$ *is expressible in* $\mathcal{N}(F)$ *if there exists an expression* $e \in \mathcal{N}(F)$ *such that, for every graph* $G$*, we have that* $e(G)$ *is nonempty if and only if* $q(G)$ *is true. In both cases, we say that* $q$ is expressed by $e$*.*

In this paper, we are mainly interested in boolean queries. Compared to path queries, this means that we are not interested in the precise set of pairs returned by an expression on a given input graph, but rather in whether or not this set is empty. Hence, if we can establish that adding transitive closure to a language does not increase its expressive power at the level of path queries, this must necessarily also be the case at the level of boolean queries. The converse, however, need not be true. Therefore, studying expressiveness issues is considerably more difficult at the level of boolean queries than at the level of path queries.

To conclude these preliminaries, we formally define what we mean by a *subexpression* of a given expression.

**Definition 2.** *Let* $F$ *be a set of nonbasic features, and let* $e$ *be an expression in* $\mathcal{N}(F)$*. The set of all* subexpressions *of* $e$*, denoted* $Sub(e)$*, is defined recursively, as follows:*

1. *if* $e$ *is either* $R$*,* $\emptyset$*,* $id$*,* $di$*, or a conditional, then* $Sub(e) = \{e\}$*;*
2. *if "*$\diamond$*"is either composition or a set operation, and if, for some expressions* $e_1$ *and* $e_2$ *in* $\mathcal{N}(F)$*,* $e = e_1 \diamond e_2$*, then* $Sub(e) = Sub(e_1) \cup Sub(e_2) \cup \{e\}$*; and*
3. *if "*$\theta$*" is either projection, coprojection, converse, or transitive closure, and if, for some expression* $f$ *in* $\mathcal{N}(F)$*,* $e = \theta(f)$*, then* $Sub(e) = Sub(f) \cup \{e\}$*.*

An *atomic subexpression* of an expression is a subexpression that is either "$R$", "$id$", "$di$", or a conditional. For an expression $e$ in the relation algebra with or without transitive closure, we denote by $|e|$ the number of its atomic subexpressions and by $|e|_R$ the number of occurrences of "$R$" in $e$.

## 3 Relation Algebra Fragments with Coprojection

In this Section, we show that adding transitive closure to a relation algebra fragment in which coprojection can be expressed yields additional expressive power at the level of boolean queries, thus settling the previously open cases for (1) the basic language to which coprojection is added and (2) the basic language to which both coprojection and the diversity relation are added.

A node $v$ of a graph $G$ is a *sink node* if $v$ has no outgoing edges, i.e., if $(v, v) \in \overline{\pi}_1(R)(G)$. We now present an expressibility and an inexpressibility result for a particular graph property stated in terms of sink nodes.

**Proposition 1.** *The boolean query "There is a non-sink node from which no sink node can be reached" is expressible in* $\mathcal{N}(\overline{\pi}, ^+)$.

*Proof.* This query returns *true* if and only if $\overline{\pi}_1((R^+ \circ \overline{\pi}_1(R)) \cup \overline{\pi}_1(R)) \neq \emptyset$. □

Using an Ehrenfeucht-Fraïssé argument [16], we can show the following, however (proof omitted).

**Proposition 2.** *The boolean query "There is a non-sink node from which no sink node can be reached" is not expressible in FO.*

From Propositions 1 and 2 we can now conclude the following.

**Theorem 1.** *Let $F$ be a set of nonbasic features not containing transitive closure such that coprojection can be expressed in $\mathcal{N}(F)$. Then $\mathcal{N}(F, ^+)$ does* not *collapse to $\mathcal{N}(F)$ at the level of boolean queries.*

*Proof.* In Propositions 1 and 2, we identified a property of graphs that is expressible in $\mathcal{N}(F, ^+)$ but not in FO. It is well known [24] that the full relation algebra is equivalent to $\mathrm{FO}^3$, both at the level of path queries and at the level of boolean queries. Since $\mathrm{FO}^3$ is a fragment of FO, it follows that the aforementioned property is *not* expressible in $\mathcal{N}(F)$. □

As an immediate corollary, two open cases are now settled.

**Corollary 1.** *At the level of boolean queries, $\mathcal{N}(\overline{\pi}, ^+)$ does not collapse to $\mathcal{N}(\overline{\pi})$ and $\mathcal{N}(\overline{\pi}, di, ^+)$ does not collapse to $\mathcal{N}(\overline{\pi}, di)$.*

## 4 Relation Algebra Fragments with at Most Projection and Diversity

This Section is devoted to demonstrating that $\mathcal{N}(\pi, di, ^+)$ collapses to $\mathcal{N}(\pi, di)$ at the level of boolean queries. We start with an introductory example.

*Example 1.* Consider the expression $e := \pi_1(R^3) \circ R^+ \circ di \circ \pi_2(R) \circ R^2$ in $\mathcal{N}(\pi, di, ^+)$. Let $G$ be a graph. For $e(G)$ to be nonempty, the subexpressions to the right of "$di$" must return nonempty. Hence, there must exist a chain

$w_0 \to w_1 \to w_2 \to w_3$ in $G$. Unless, for each such chain, $w_1 = w_2 = w_3$, it is readily seen that this condition is also sufficient for $e(G) \neq \emptyset$. In the other case, there must exist an edge $v_0 \to v_1$ with a self-loop in $v_1$ for which $v_1 \neq w_1$ in order for $e(G)$ to be nonempty. It can now be readily verified that, in both cases, $e'(G) \neq \emptyset$, with $e' := \pi_1(R^3) \circ (R \cup R^2) \circ di \circ \pi_2(R) \circ R^2$ in $\mathcal{N}(\pi, di)$. As always $e'(G) \subseteq e(G)$, the converse implication also holds, so $e' \in \mathcal{N}(\pi, di)$ is equivalent to $e \in \mathcal{N}(\pi, di, ^+)$ at the level of boolean queries. □

The argument used to show that transitive closure can be eliminated from the expression in Example 1 is very ad-hoc. Moreover, the considered expression is very simple. We therefore need a general technique to show that $\mathcal{N}(\pi, di, ^+)$ collapses to $\mathcal{N}(\pi, di)$ at the level of boolean queries. In this section, we outline this technique, and, in subsequent sections, we work it out in further detail.

The technique we are about to describe actually works more generally for showing that $\mathcal{N}(F, ^+)$ collapses to $\mathcal{N}(F)$ for all sets of nonbasic features $F$ for which $F \subseteq \{\pi, di\}$. It consists of two steps. Given an expression $e$ in $\mathcal{N}(F, ^+)$,

1. find an expression $\mathit{suff}_e$ in $\mathcal{N}(F)$ such that, for every graph $G$, $\mathit{suff}_e(G) \neq \emptyset$ implies $e(G) \neq \emptyset$; and
2. find an expression $e'$ in $\mathcal{N}(F)$ that is equivalent to $e$ at the level of boolean queries on all graphs $G$ for which $\mathit{suff}_e(G) = \emptyset$.

It then follows immediately that, on all graphs, $e$ is equivalent to $\mathit{suff}_e \cup e'$ at the level of boolean queries, i.e., for every graph $G$, $\mathit{suff}_e \cup e'(G) \neq \emptyset$ if and only if $e(G) \neq \emptyset$. Intuitively, $\mathit{suff}_e(G) \neq \emptyset$ is a sufficient condition for $e(G)$ to be nonempty. It therefore suffices to show the collapse on graphs that do not satisfy this condition, i.e., for which $\mathit{suff}_e(G) = \emptyset$. If $\mathit{suff}_e$ is well-chosen, then the latter condition will turn out to be sufficiently restrictive for our purposes.

The first step of the proof procedure is secured by the following result.

**Theorem 2.** *Let $F \subseteq \{\pi, di\}$. Let $e$ be an expression in $\mathcal{N}(F, ^+)$. Let $\mathit{suff}_{F,e}$ in $\mathcal{N}(F)$ be as tabulated in Table 1, where $m_e = \max(1, |e|_R)$. Then, for every graph $G$, $\mathit{suff}_{F,e}(G) \neq \emptyset$ implies $e(G) \neq \emptyset$.*

**Table 1.** Expressions $\mathit{suff}_{F,e}$ in $\mathcal{N}(F)$ for which $\mathit{suff}_{F,e}(G) \neq \emptyset$ implies $e(G) \neq \emptyset$, $F \subseteq \{\pi, di\}$. In these expressions, $m_e = \max(1, |e|_R)$.

| $F$ | $\mathit{suff}_{F,e}$ |
|---|---|
| $\emptyset$ | $R^{m_e}$ |
| $\{\pi\}$ | $R^{m_e}$ |
| $\{di\}$ | $R^{m_e} \circ di \circ R^{m_e}$ |
| $\{\pi, di\}$ | $\pi_1(R^{m_e}) \circ \pi_2(R^{m_e}) \circ di \circ \pi_1(R^{m_e}) \circ \pi_2(R^{m_e})$ |

*Proof.* Here, we only sketch the proof for $F = \{\pi, di\}$. So, let $e$ be an expression in $\mathcal{N}(\pi, di, ^+)$ and let $G$ be a graph. We first observe that the condition $\pi_1(R^{m_e}) \circ \pi_2(R^{m_e}) \circ di \circ \pi_1(R^{m_e}) \circ \pi_2(R^{m_e})(G) \neq \emptyset$ is equivalent to the existence of two

sequences of not necessarily all different nodes $v_{-m_e}, \ldots, v_{-1}, v_0, v_1, \ldots, v_{m_e}$ and $w_{-m_e}, \ldots, w_{-1}, w_0, w_1, \ldots, w_{m_e}$ in $G$ such that, (1) for $i = -m_e, \ldots, m_e - 1$, $(v_i, v_{i+1}) \in R$ and $(w_i, w_{i+1}) \in R$ and (2) $v_0 \neq w_0$. By an inductive argument, we show that, for any union-free expression $f$ in $\mathcal{N}(\pi, di)$ with $|f|_R \leq |e|_R$, $f$ is nonempty on the subgraph of $G$ consisting of the nodes and edges singled out above, as a consequence of which $f(G) \neq \emptyset$. Finally, from $e$, we construct a union-free expression $e'$ in $\mathcal{N}(\pi, di)$ for which $|e'|_R \leq |e|_R$ and $e'(G) \subseteq e(G)$. It follows now that $e(G) \neq \emptyset$, as had to be shown. □

For the second step, we first of all observe that, for any graph $G$, and for any nonzero natural number $m$, $R^m(G) = \emptyset$ implies that $R^m \circ di \circ R^m(G) = \emptyset$, and that $R^m \circ di \circ R^m(G) = \emptyset$ implies that $\pi_1(R^m) \circ \pi_2(R^m) \circ di \circ \pi_1(R^m) \circ \pi_2(R^m) = \emptyset$. Any necessary condition on the graph $G$ for the last expression to return the empty set on $G$ is therefore also a necessary condition for the two other expressions to return the empty set on $G$.

For our purpose, we extend the notion of directed acyclic graph (DAG).

**Definition 3.** *An* extended directed acyclic graph *(EDAG) is a (not necessarily connected) DAG to which self-loops may be added provided each path in the DAG contains at most one node with a self-loop. The DAG obtained from an EDAG by removing all self-loops (but not the nodes in which these self-loops occur) is called the* underlying *DAG. The* depth *of an EDAG is the depth of the underlying DAG, i.e., the maximal length of a path in that DAG.*

We now have the following.

**Lemma 1.** *Let $m$ be a nonzero natural number, and let $G$ be a graph such that $\pi_1(R^m) \circ \pi_2(R^m) \circ di \circ \pi_1(R^m) \circ \pi_2(R^m)(G) = \emptyset$. Then $G$ is an EDAG of depth at most $2m$.*

*Proof.* If $\pi_1(R^m) \circ \pi_2(R^m) \circ di \circ \pi_1(R^m) \circ \pi_2(R^m)(G) = \emptyset$, then it is the case that, for any two sequences of nodes $v_{-m}, \ldots, v_{-1}, v_0, v_1, \ldots, v_m$ and $w_{-m}, \ldots, w_{-1}, w_0, w_1, \ldots, w_m$ in $G$ such that, for $i = -m, \ldots, m - 1$, $(v_i, v_{i+1}) \in R$ and $(w_i, w_{i+1}) \in R$, we have that $v_0 = w_0$ (cf. the proof of Theorem 2). Clearly, this is not the case if $G$ contains either one loop of length at least two; or two self-loops; or a non-selfintersecting path of length at least $2m + 1$. Hence, $G$ is an EDAG of depth at most $2m$. □

Notice that $G$ being an EDAG of depth at most $2m$ is not a sufficient condition for the expression in Lemma 1 to evaluate to the empty set. For instance, an EDAG may contain more than one self-loop in total (at most one on each path in the underlying DAG). Also, a DAG (which is a special case of an EDAG) of depth $2m$ may contain two paths of length $2m$ of which the middle nodes do not coincide. Hence, $G$ being an EDAG of depth at most $2m$ is only a necessary condition for $\pi_1(R^m) \circ \pi_2(R^m) \circ di \circ \pi_1(R^m) \circ \pi_2(R^m)(G) = \emptyset$. For our purposes, however, this is all we need.

Using our earlier observation, we can bootstrap Lemma 1 as follows.

**Proposition 3.** *Let $F \subseteq \{\pi, di\}$, and let $e$ be an expression in $\mathcal{N}(F, ^+)$. Let $G$ be a graph such that $suff_{F,e}(G) = \emptyset$. Then $G$ is an EDAG of depth at most $2m_e$.*

Assume that we are given an expression $e$ in $\mathcal{N}(\pi, di, ^+)$ and an EDAG $G$ of depth at most $m$, with $m$ some nonzero natural number. The remainder of this paper is concerned with proving that there exists a nonzero natural number $s$ depending only on $m$ and $e$ such that $e(G) = \emptyset$ if and only $e'(G) = \emptyset$, where $e'$ is obtained from $e$ by exhaustively replacing each subexpression of the form $f^+$ by $\bigcup_{i=1}^{s} f^i$.

To achieve this, we intend to show (Proposition 12) that there exists a nonzero natural number $s$ such that, for any node $v$ of $G$, there exists a subgraph $G_v$ of $G$ containing $v$ which has at most $s$ nodes and satisfies the following property: there exists a node $w$ for which $(v, w) \in e(G)$ if and only if there exists a node $w'$ in $G_v$ for which $(v, w') \in e(G_v)$. To see that this is sufficient for our purposes, assume first that $e(G) = \emptyset$. Then $e'(G) = \emptyset$, since $e'(G) \subseteq e(G)$. Therefore, assume next that $e(G) \neq \emptyset$. Then, for some nodes $v$ and $w$ of $G$, $(v, w) \in e(G)$. Hence, there exists a node $w'$ in $G_v$ such that $(v, w') \in e(G_v)$. Since $G_v$ has at most $s$ nodes, $e(G_v) = e'(G_v)$. It follows that $e'(G_v) \neq \emptyset$. Since $e'(G_v) \subseteq e'(G)$, it also follows that $e'(G) \neq \emptyset$.

In the remaining sections, we shall establish that such subgraphs $G_v$ exist.

Finally, notice that, whenever $e$ is in $\mathcal{N}(F, ^+)$ with $F \subseteq \{\pi, di\}$, then $e'$ is in $\mathcal{N}(F)$. Hence, our efforts in the context of $\mathcal{N}(\pi, di, ^+)$ also serve to show that any of the languages $\mathcal{N}(F, ^+)$ with $F \subseteq \{\pi, di\}$ collapses to $\mathcal{N}(F)$ at the level of boolean queries.

## 5 Expressions with Conditionals

To facilitate achieving the goals set at the end of the previous section, we shall first simplify the expressions under consideration. In Section 2, we introduced conditionals, which are constants at the syntactical level, representing at the semantic level functions that associate to each graph a set of identical pairs of that graph. Now, notice that any subexpression of the form $\pi_1(f)$ or $\pi_2(f)$ of an expression in $\mathcal{N}(\pi, di, ^+)$ can be interpreted as a function defining the semantics of some conditional. Given an expression in $\mathcal{N}(\pi, di, ^+)$, we shall therefore as a first step replace all projection subexpressions which themselves do not occur within a projection subexpression by a conditional with the same semantics. In this way, projection is formally eliminated, which simplifies the further development considerably. Once we have a partial result for this case, we will reintroduce the projections and bootstrap the initial result to the desired result.

*Example 2.* Consider the expression $(R \circ \pi_1((R^3 \circ di \circ \pi_2(R^2) \circ R)^+))^+ \circ R^2$. If we associate a conditional $c$ to $\pi_1((R^3 \circ di \circ \pi_2(R^2) \circ R)^+)$, the expression can be rewritten as $(R \circ c_1)^+ \circ R^2$, i.e., the projection has formally been eliminated.

Therefore, we introduce a finite set of conditionals $\Gamma = \{c_1, \ldots, c_p\}$, and consider the language $\mathcal{N}(\Gamma, di, ^+)$, as well as some of its sublanguages. Later on, we

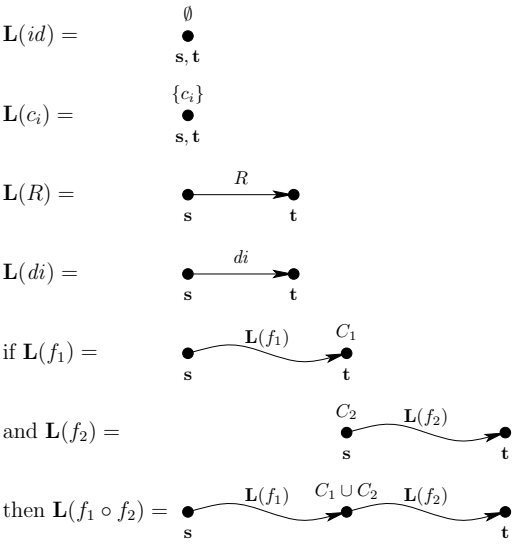


**Fig. 1.** Definition of the line pattern $\mathbf{L}(f)$ of a union-free expression in $\mathcal{N}(\Gamma, di)$

will choose $p$ as a function of the number of projection subexpressions in the expression under consideration.

A useful property is that, for union-free expressions in $\mathcal{N}(\Gamma, di)$, the presence of a particular pair of nodes of a graph in the output of the expression applied to the graph can be rephrased as the existence of a particular homomorphism from a chain-like directed graph, representing the expression, into the graph.

More concretely, let $f$ be a union-free expression in $\mathcal{N}(\Gamma, di)$. We shall associate a *line pattern* $\mathbf{L}(f)$ with $f$. This line pattern is a chain-like directed graph in which each edge is labeled with either "$R$" or "$di$" and each node is labeled by a (possibly empty) set of conditionals. In addition, each line pattern has one *source node*, labeled $\mathbf{s}$, and one *target node*, labeled $\mathbf{t}$, which may coincide. The precise, inductive, definition is given in Figure 1.

Line patterns are special cases of *graph patterns*. A graph pattern is a directed graph in which each edge is labeled with either "$R$" or "$di$" and each node is labeled by a (possibly empty) set of conditionals. At least one node is marked as source, and at least one node is marked as target.

Let $\mathbf{P}$ be a graph pattern, and let $G$ be a directed graph. A mapping $h$ from the nodes of $\mathbf{P}$ to the nodes of $G$ is called a *homomorphism* from $\mathbf{P}$ to $G$ if

1. for each node $\mathbf{v}$ of $\mathbf{P}$, all the conditionals by which $\mathbf{v}$ is labeled are satisfied by $h(\mathbf{v})$ in $G$;
2. for each edge $(\mathbf{v}, \mathbf{w})$ of $\mathbf{P}$ labeled by "$R$", $(h(\mathbf{v}), h(\mathbf{w}))$ is an edge of $G$; and
3. for each edge $(\mathbf{v}, \mathbf{w})$ of $\mathbf{P}$ labeled by "$di$", $h(\mathbf{v}) \neq h(\mathbf{w})$.

Notice that we use boldface characters for the nodes of line and graph patterns to distinguish them clearly from the nodes of the input graph.

General graph patterns will be put to use in Section 6 to construct, given an expression $e$ in $\mathcal{N}(\Gamma, di, ^+)$, a natural number $m$, an EDAG $G$ of depth at most $m$, and a node $v$ of $G$, a sequence of subgraphs of $G$. The number of nodes of these subgraphs can be bounded by a natural number depending only on $m$ and $e$. This last property is essential for our proof strategy to work, as explained at the end of Section 4.

Turning back to line patterns for now, the following result is obvious.

**Proposition 4.** *Let $G$ be a directed graph and let $f$ be a union-free expression in $\mathcal{N}(\Gamma, di)$. Let $\boldsymbol{L}(f)$ be the line pattern of $f$. Let $v$ and $w$ be nodes of $G$. Then $(v, w) \in f(G)$ if and only if there exists a homomorphism $h$ from $\boldsymbol{L}(f)$ to $G$ with $h(\boldsymbol{s}) = v$ and $h(\boldsymbol{t}) = w$.*

In order to put line patterns to use, we must link expressions in $\mathcal{N}(\Gamma, di, ^+)$ to union-free expressions in $\mathcal{N}(\Gamma, di)$. Thereto, we introduce *trace expressions*.

**Definition 4.** *Let $e$ be an expression in $\mathcal{N}(\Gamma, di, ^+)$. Then, $\mathcal{T}(e)$, the set of* trace expressions *of $e$, is defined recursively, as follows:*

- *if $e$ is an atomic expression, then $\mathcal{T}(e) = \{e\}$;*
- $\mathcal{T}(e_1 \cup e_2) = \mathcal{T}(e_1) \cup \mathcal{T}(e_2)$;
- $\mathcal{T}(e_1 \circ e_2) = \{\tau_1 \circ \tau_2 \mid \tau_1 \in \mathcal{T}(e_1)\ \&\ \tau_2 \in \mathcal{T}(e_2)\}$; *and*
- $\mathcal{T}(e^+) = \bigcup_{n>0}\{\tau_1 \circ \cdots \circ \tau_n \mid \forall i = 1, \ldots, n : \tau_i \in \mathcal{T}(e)\}$.

Notice that trace expressions do not contain "$\cup$" and "$^+$".

By a straightforward structural induction, one can show the following.

**Proposition 5.** *Let $e$ be an expression in $\mathcal{N}(\Gamma, di, ^+)$. Let $G$ be a graph and $v$ and $w$ nodes of $G$. Then, $(v, w) \in e(G)$ if and only if there exists a trace expression $f \in \mathcal{T}(e)$ such that $(v, w) \in f(G)$.*

The problem with trace expressions is that they may contain a lot of redundancy. Therefore, we define the following notions.

**Definition 5.** *Let $n$ be a nonzero natural number. An expression $g$ in $\mathcal{N}(\Gamma)$ is $n$-normal if (1) $g$ is union-free, (2) $|g|_R \leq n$, and (3) a subexpression of $g$ consisting only of "id" conditionals, and composition has at most one occurrence of "id" and at most one occurrence of every conditional.*

Observe that, for all $n$, "$id$" is always $n$-normal. We denote the $n$-normal expressions of $\mathcal{N}(\Gamma)$ by $\mathcal{N}_n^{\text{norm}}(\Gamma)$.

**Definition 6.** *Let $n$ be a nonzero natural number. An expression $f$ in $\mathcal{N}(\Gamma, di)$ is $n$-normal if it is of the form $g_1 \circ di \circ g_2 \circ di \circ \cdots \circ g_{k-1} \circ di \circ g_k$, with $g_1, \ldots, g_k \in \mathcal{N}_n^{\text{norm}}(\Gamma)$.*

In particular, all $n$-normal expressions of $\mathcal{N}(\Gamma)$ are also $n$-normal expressions of $\mathcal{N}(\Gamma, di)$. We denote the $n$-normal expressions of $\mathcal{N}(\Gamma, di)$ by $\mathcal{N}_n^{\text{norm}}(\Gamma, di)$. The interest of normal expressions lays in the following proposition.

**Proposition 6.** *Let $\Gamma = \{c_1, \ldots, c_p\}$ be a set of conditionals, and let $n$ be a nonzero natural number. Then,*

1. *the number of atomic subexpressions of an expression of $\mathcal{N}_n^{\mathrm{norm}}(\Gamma)$ can be bounded by a number depending only on $n$ and $p$; and*
2. *the number of expressions in $\mathcal{N}_n^{\mathrm{norm}}(\Gamma)$ is finite, and can be bounded by a number depending only on $n$ and $p$.*

Given a nonzero natural number $n$, we now define the set $\mathcal{T}_n^{\mathrm{norm}}(e)$ of *$n$-normal trace expressions* as the set of all expressions $\mathcal{N}_n^{\mathrm{norm}}(\Gamma, di)$ for which there exists an equivalent expression in $\mathcal{T}(e)$ at the level of path queries.

The following result states that trace expressions can be normalized provided the input graph is an EDAG of bounded depth.

**Proposition 7.** *Let $m$ be a nonzero natural number, and let $e$ be an expression in $\mathcal{N}(\Gamma, di, ^+)$. Then, there exists a nonzero natural number $n$ depending only on $m$ and $e$ such that, for every EDAG $G$ of depth at most $m$, and for every pair of nodes $v$ and $w$ of $G$, $(v, w) \in e(G)$ if and only if there exists an $n$-normal trace expression $f$ in $\mathcal{T}_n^{\mathrm{norm}}(e)$ for which $(v, w) \in f(G)$.*

## 6 Canonical Subgraphs

Given a set of conditionals $\Gamma = \{c_1, \ldots, c_p\}$, a natural number $n$, a directed graph $G$, and a node $v$ of $G$, we shall now define a sequence of so-called $n$-canonical subgraphs $G_0^v, G_1^v, G_2^v, \ldots$ of order $0, 1, 2, \ldots$. (In the notation, we shall leave $\Gamma$ and $n$ implicit.)

Important for our purpose is that it will turn out that the number of nodes of each of these $n$-canonical subgraphs depends only on its order and on $p$ and $n$, but *not* on the particular graph $G$ or the particular node $v$ under consideration.

We start by defining $G_0^v$.

Thereto, let $g$ be an expression in $\mathcal{N}_n^{\mathrm{norm}}(\Gamma)$. We define $\mathfrak{P}(g)$ to be the set of graph patterns that can be obtained from $\mathbf{L}(g)$ in the following way:

1. Start with one, two, three, or four pairwise disjoint copies of $\mathbf{L}(g)$.
2. Optionally, merge some of the source nodes of these copies.
3. Optionally, merge some of the target nodes of these copies.
4. Optionally, connect some of the remaining source nodes by "*di*" edges.
5. Optionally, connect some of the remaining target nodes by "*di*" edges.

Observe that the line pattern $\mathbf{L}(g)$ itself is always in $\mathfrak{P}(g)$.

Figure 2 shows a more representative example of a graph pattern that belongs to $\mathfrak{P}(g)$.

Now, let $\mathbf{P}$ be a graph pattern in $\mathfrak{P}(g)$, and let $v$ be a node of $G$. With $\mathbf{P}$, we associate a minimal (in number of elements) set $\mathfrak{H}_v(\mathbf{P})$ of homomorphisms from $\mathbf{P}$ to $G$ satisfying the following conditions:

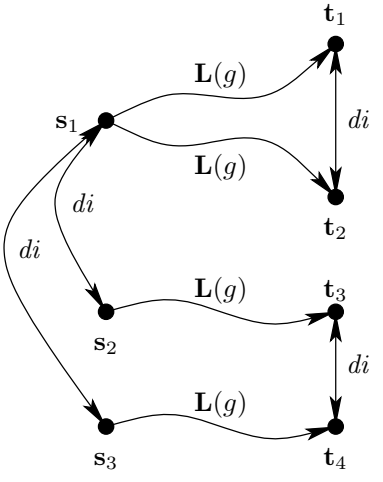


**Fig. 2.** Example of a graph pattern in $\mathfrak{P}(g)$

1. if there exists a homomorphism from $\mathbf{P}$ to $G$, then $\mathfrak{H}_v(\mathbf{P}) \neq \emptyset$;
2. if, for an arbitrary node $\mathbf{v}$ of $\mathbf{P}$, there exist two homomorphisms from $\mathbf{P}$ to $G$ mapping $\mathbf{v}$ to different nodes of $G$, then $\mathfrak{H}_v(\mathbf{P})$ contains two homomorphisms from $\mathbf{P}$ to $G$ mapping $\mathbf{v}$ to different nodes of $G$;
3. if $\mathbf{P}$ has a single source node $\mathbf{s}$ and there exists a homomorphism from $\mathbf{P}$ to $G$ mapping $\mathbf{s}$ to $v$, then $\mathfrak{H}_v(\mathbf{P})$ contains such a homomorphism;
4. if $\mathbf{P}$ has a single target node $\mathbf{t}$ and there exists a homomorphism from $\mathbf{P}$ to $G$ mapping $\mathbf{t}$ to $v$, then $\mathfrak{H}_v(\mathbf{P})$ contains such a homomorphism;

For a good understanding, we first observe the following:

- Given $\mathbf{P}$, $G$, and $v$, we *choose* a minimal set of homomorphisms $\mathfrak{H}_v(\mathbf{P})$ satisfying the above conditions. In other words, it is to be expected that, in general, several sets of homomorphisms satisfy the above conditions. From these, we pick one arbitrarily, and denote it by $\mathfrak{H}_v(\mathbf{P})$.
- The definition of $\mathfrak{H}_v(\mathbf{P})$ refers explicitly to $v$ only if $\mathbf{P}$ has either a single source node, or a single target node, or both. In all other cases, we may therefore choose $\mathfrak{H}_v(\mathbf{P})$ independent of $v$.

We are now ready to define $G_0^v$, the basic $n$-canonical subgraph of order 0:

$$G_0^v = \bigcup_{g \in \mathcal{N}_n^{\mathrm{norm}}(\Gamma)} \quad \bigcup_{\mathbf{P} \in \mathfrak{P}(g)} \quad \bigcup_{h \in \mathfrak{H}_v(\mathbf{P})} h(\mathbf{P}).$$

In the above formula, $h(\mathbf{P})$ must be understood as the subgraph of $G$ the set of nodes of which is $\{h(\mathbf{v}) \mid \mathbf{v} \text{ is a node of } \mathbf{P}\}$ and the set of edges of which is $\{(h(\mathbf{v}), h(\mathbf{w})) \mid (\mathbf{v}, \mathbf{w}) \text{ is an } R\text{-labeled edge of } \mathbf{P}\}$. The basic $n$-canonical subgraph of order 0 is then defined as a union of some of these subgraphs, where

this union must be interpreted componentwise, i.e., the set of nodes and the set of edges of this union are the union of the sets of nodes and the union of the sets of edges of the subgraphs involved.

We point out at this stage that if a node $v$ of $G$ satisfies a conditional $c$, and $G'$ is a subgraph of $G$ containing $v$, then a priori $v$ does not have to satisfy $c$ in $G'$. We shall therefore avoid evaluating expressions over subgraphs of $G$ (in particular, the canonical ones), until we reinterpret conditionals as the projection subexpressions for which they actually stand, in Section 8.

At this point, several aspects of the definition of the basic $n$-canonical subgraph of order 0 have been left unexplained, in particular,

- the definition of the set of graph patterns $\mathfrak{P}(g)$ for $g \in \mathcal{N}_n^{\mathrm{norm}}(\Gamma)$, and, more specifically, why up to four copies of the line pattern $\mathbf{L}(g)$ are allowed in such a graph pattern; and
- the definition of the set of homomorphisms $\mathfrak{H}_v(\mathbf{P})$ for $\mathbf{P} \in \mathfrak{P}(g)$.

These definitions are tailored to make the key results Propositions 10 and 11 in Section 7 work, as is explained in that section. The essence is that, given an $n$-normal trace expression $f$ in $\mathcal{T}_n^{\mathrm{norm}}(e)$ and a homomorphism $h$ from $\mathbf{L}(f)$ to $G$, we wish to show via an inductive process that there also exists such a homomorphism of which the image is fully contained in one of the $n$-canonical subgraphs of order 0. On the one hand, we must ensure that the $n$-canonical subgraphs of order 0 are sufficiently large for this process to work, but, on the other hand, we must also ensure that their size can be bounded by a bound not depending on the size of $G$ (see Proposition 9, below). Obtaining this delicate balance is what led to the definition above.

However, Propositions 10 and 11 are but the first albeit important step in proving the collapse of $\mathcal{N}(\pi, di, {}^+)$ to $\mathcal{N}(\pi, di)$. Indeed, the conditionals represent projection conditions, and the operands of these projections may in turn contain projection conditions.

To accommodate this, we next define $G_1^v, G_2^v, \ldots$, the $n$-canonical subgraphs of $G$ of order $1, 2, \ldots$, with the following inductive rule. For $i > 0$,

$$G_i^v = G_0^v \cup \left( \bigcup_{w \text{ node of } G_0^v} G_{i-1}^w \right).$$

The following property of $n$-canonical subgraphs is straightforward.

**Proposition 8.** *Given a set of conditionals $\Gamma = \{c_1, \ldots, c_p\}$, a nonzero natural number $n$, and a directed graph $G$, we have, for every node $v$ of $G$ and for $i = 0, 1, 2, \ldots$, that $G_i^v$ is a subgraph of $G_{i+1}^v$.*

The $n$-canonical subgraphs of $G$ of higher order are put to use in Section 8, more in particular in Proposition 12.

For the remainder of the exposition, it is important that we can also provide bounds on the sizes of the $n$-canonical subgraphs of $G$.

**Proposition 9.** *Given a set of conditionals $\Gamma = \{c_1, \ldots, c_p\}$, a nonzero natural number $n$, and a directed graph $G$, we have, for every node $v$ of $G$ and for $i = 0, 1, 2, \ldots$, that the number of nodes in $G_i^v$ can be bounded by a number depending only on $n$, $p$, and $i$.*

## 7 The Key Result

Let $\Gamma = \{c_1, \ldots, c_p\}$ be a set of conditionals. The key results on which the second step in our proof strategy for the collapse of $\mathcal{N}(\Gamma, di, ^+)$ to $\mathcal{N}(\Gamma, di)$ at the boolean level (cf. item 2 on p. 130 and the concluding paragraphs of Section 4) rely, are the following.

**Proposition 10.** *Let $m$ be a nonzero natural number, and let $e$ be an expression in $\mathcal{N}(\Gamma, di, ^+)$. Then, there exists a nonzero natural number $n$ depending only on $m$ and $e$ such that, for every EDAG $G$ of depth at most $m$, and for every node $v$ of $G$, if there exists a node $w$ in $G$ such that $(v, w) \in e(G)$, then there exists an $n$-normal trace expression $f$ in $\mathcal{T}_n^{\mathrm{norm}}(e)$ and a homomorphism $h$ from $\boldsymbol{L}(f)$ to $G$ such that $h(\boldsymbol{s}) = v$ and $h(\boldsymbol{L}(f))$ is contained in $G_0^v$, with $\boldsymbol{s}$ the source node of the line pattern $\boldsymbol{L}(f)$ and $G_0^v$ the basic $n$-canonical subgraph of $G$.*

**Proposition 11.** *Let $m$ be a nonzero natural number, and let $e$ be an expression in $\mathcal{N}(\Gamma, di, ^+)$. Then, there exists a nonzero natural number $n$ depending only on $m$ and $e$ such that, for every EDAG $G$ of depth at most $m$, and for every node $w$ of $G$, if there exists a node $v$ in $G$ such that $(v, w) \in e(G)$, then there exists an $n$-normal trace expression $f$ in $\mathcal{T}_n^{\mathrm{norm}}(e)$ and a homomorphism $h$ from $\boldsymbol{L}(f)$ to $G$ such that $h(\boldsymbol{t}) = w$ and $h(\boldsymbol{L}(f))$ is contained in $G_0^w$, with $\boldsymbol{t}$ the target node of the line pattern $\boldsymbol{L}(f)$ and $G_0^w$ the basic $n$-canonical subgraph of $G$.*

It is important to notice here that the homomorphism $h$ in Propositions 10 and 11 need not be a homomorphism from $\mathbf{L}(f)$ to $G_0^v$, respectively $G_0^w$. If this were the case, then, by Proposition 4, $(v, w) \in e(G_0^v)$, respectively $(v, w) \in e(G_0^w)$, and we would have found the subgraphs $G_v$ of $G$ we set out to find at the end of Section 4 to achieve the second step of our proof strategy. However, this is in general not the case, the reason being that conditionals are in general not preserved under taking subgraphs. Indeed, if $z$ is a node of $G$ such that $(z, z) \in c_i(G)$, $1 \leq i \leq p$, then it does not follow that, necessarily, $(z, z) \in c_i(G_0^v)$. As mentioned, the case that we are interested in is the case where the conditionals are in fact projection conditions. These have the property of being monotone. To guarantee the above implication, we will therefore have to extend the subgraph $G_0^v$, and that is where the normal subgraphs of higher order come in play, at a later stage of our development, in Section 8.

Because of the strong analogy between both Propositions, we shall focus here on the proof of Proposition 10. It can be easily seen that Proposition 10 follows from Propositions 7 and 4, provided we can prove the following lemma.

**Lemma 2.** *Let $G$ be a directed graph, let $n$ be a nonzero natural number, and let $f$ be an $n$-normal expression in $\mathcal{N}_n^{\mathrm{norm}}(\Gamma, di)$. Let $v$ be a node of $G$. If there exists a homomorphism $h$ from $\boldsymbol{L}(f)$ to $G$ such that $h(\boldsymbol{s}) = v$, with $\boldsymbol{s}$ the source node of $\boldsymbol{L}(f)$, then there exists a homomorphism $h'$ from $\boldsymbol{L}(f)$ to $G$ such that $h'(\boldsymbol{s}) = v$ and $h'(\boldsymbol{L}(f))$ is contained in $G_0^v$, with $G_0^v$ the basic $n$-canonical subgraph of $G$.*

If we write $f = g_1 \circ di \circ g_2 \circ di \circ \cdots \circ g_{n-1} \circ di \circ g_n$, with $g_1, \ldots, g_n \in \mathcal{N}_m^{\mathrm{norm}}(\Gamma)$, a sensible way to prove Lemma 2 is to consider the expressions $f_i = g_1 \circ di \circ g_2 \circ di \circ \cdots \circ g_{i-1} \circ di \circ g_i$, for $i = 1, \ldots, n$, and to prove the Lemma by induction on $i$. The basis of the induction, $i = 1$, is straightforward from the construction of the subgraph $G_0^v$. Thus suppose that, for $1 < i \leq n$, we have established the existence of a homomorphism $h'_{i-1}$ from $\mathbf{L}(f_{i-1})$ to $G$ such that $h'_{i-1}(\mathbf{s}) = v$ ($\mathbf{s}$ being the source node of $\mathbf{L}(f_{i-1})$) and $h'_{i-1}(\mathbf{L}(f_{i-1}))$ is contained in $G_0^v$. We would like to extend $h'_{i-1}$ to a homomorphism $h'_i$ from $\mathbf{L}(f_i)$ to $G$ such that $h'_i(\mathbf{L}(f_i))$ is contained in $G_0^v$. Thus, consider $\mathbf{L}(g_i)$, which is a subpattern of $\mathbf{L}(f_i)$. The restriction of $h$ to the nodes of $\mathbf{L}(g_i)$ is a homomorphism from $\mathbf{L}(g_i)$ to $G$. Hence, $\mathfrak{H}_v(\mathbf{L}(g_i))$ contains a homomorphism $h_{\mathbf{L}(g_i)}$ from $\mathbf{L}(g_i)$ to $G$, and, by construction of $G_0^v$, $h_{\mathbf{L}(g_i)}(\mathbf{L}(g_i))$ is contained in $G_0^v$. Now, let $\mathbf{t}_{i-1}$ be the target node of $\mathbf{L}(f_{i-1})$ and $\mathbf{s}_i$ the source node of $\mathbf{L}(g_i)$. If $h'_{i-1}(\mathbf{t}_{i-1}) \neq h_{\mathbf{L}(g_i)}(\mathbf{s}_i)$, the extension is straightforward. However, we cannot exclude that $h'_{i-1}(\mathbf{t}_{i-1}) = h_{\mathbf{L}(g_i)}(\mathbf{s}_i)$. If this is the case, it may even be so that $h_{\mathbf{L}(g_i)}$ is the *only* homomorphism mapping $\mathbf{L}(g_i)$ to $G$. Then, we cannot even consider an alternative homomorphism from $\mathbf{L}(g_i)$ to $G$ to make our extension strategy work.

However, we can avoid this pitfall by proving a slightly stronger statement.

**Lemma 3.** *Let $G$ be a directed graph, let $n$ be a nonzero natural number, and let $f$ be an $n$-normal expression in $\mathcal{N}_n^{\mathrm{norm}}(\Gamma, di)$. Let $v$ be a node of $G$, and let $G_0^v$ be the basic $n$-canonical subgraph of $G$. Then,*

1. *if there exist homomorphisms $h_1$ and $h_2$ from $\boldsymbol{L}(f)$ to $G$ such that $h_1(\boldsymbol{s}) = h_2(\boldsymbol{s}) = v$ and $h_1(\boldsymbol{t}) \neq h_2(\boldsymbol{t})$, with $\boldsymbol{s}$ and $\boldsymbol{t}$ the source and target nodes of $\boldsymbol{L}(f)$, then there exist homomorphisms $h'_1$ and $h'_2$ from $\boldsymbol{L}(f)$ to $G$ such that $h'_1(\boldsymbol{s}) = h'_2(\boldsymbol{s}) = v$, $h'_1(\boldsymbol{t}) \neq h'_2(\boldsymbol{t})$, and $h'_1(\boldsymbol{L}(f))$ and $h'_2(\boldsymbol{L}(f))$ are both contained in $G_0^v$;*
2. *otherwise, if there exists a homomorphism $h$ from $\boldsymbol{L}(f)$ to $G$ such that $h(\boldsymbol{s}) = v$, with $\boldsymbol{s}$ the source node of $\boldsymbol{L}(f)$, then there exists a homomorphism $h'$ from $\boldsymbol{L}(f)$ to $G$ such that $h'(\boldsymbol{s}) = v$ and $h(\boldsymbol{L}(f))$ is contained in $G_0^v$.*

The proof goes along the lines of the sketch we gave of the (failed) proof for Lemma 2. In the induction step, we may be in Case 1 or Case 2 of Lemma 2, and to carry out the inductive argument, we may be in Case 1 or Case 2 as far as the induction hypothesis is concerned, giving rise to four possible combinations we need to consider. However, when we are in Case 2 as far as the induction hypothesis is concerned, then, compared to our naive attempt to prove Lemma 2 directly, we can make use of the additional information that *all* homomorphisms from the line pattern under consideration map the target node to the same node of $G$, for, otherwise, we would be in Case 1. This additional information will prevent us from getting stuck in this case.

Each time we get a conflict of the sort described in the failed direct proof for Lemma 2, we will create a graph pattern by combining the given information on the existence of homomorphisms from the line segment under consideration to $G$ with the (not directly usable) homomorphisms from this line segment to $G$ of which the image is fully contained in $G_0^v$. We will reflect our knowledge on the equality or distinctness of nodes in the images of the various homomorphism by merging the corresponding nodes in the graph pattern (in the case of equality) or connecting these nodes by "$di$" edges (in the case of distinctness). This will result in a graph pattern such as the one shown in Figure 2. As, by construction, this graph pattern can be mapped homomorphically to $G$, it can also be mapped homomorphically to $G$ in such a way that the image is contained in $G_0^v$, provided the graph pattern does not contain more than four pairwise disjoint copies of the line segment under consideration. It turns out that, in each of the cases we must consider, this is indeed so. The richer information we obtain from the existence of a homomorphism mapping the graph pattern within $G_0^v$ as opposed to the existence of a homomorphism just mapping the line pattern within $G_0^v$ turns out to be sufficient to carry out the inductive step successfully.

## 8 The Collapse

We are now ready to deal with expressions in $\mathcal{N}(\pi, di, ^+)$ and bootstrap Propositions 10 and 11 by considering that conditionals stand for projection subexpressions. We recall that the homomorphism $h$ in the statements of these propositions is a homomorphism from $\mathbf{L}(f)$ to $G$ such that $h(\mathbf{s}) = v$ and $h(\mathbf{L}(f))$ is contained in $G_0^v$, but not necessarily a homomorphism from $\mathbf{L}(f)$ to $G_0^v$, the reason being that a node of $G_0^v$ satisfying a particular conditional within $G$ does not have to satisfy the same conditional within $G_0^v$. Using that the conditionals stand for projection subexpressions, and using the monotonicity of the projection operator, we are able to establish that $G_0^v$ can be extended to a higher-order canonical subgraph of $G$, say $G_i^v$, such that $h$ is also a homomorphism from $\mathbf{L}(f)$ to $G_i^v$. Only then will we be able to conclude that $(v, h(\mathbf{t})) \in e(G_i^v)$, with $\mathbf{t}$ the target node of $\mathbf{L}(f)$ and can we complete our argument.

For this purpose, we first define the *$\pi$-nesting depth* $\mathrm{depth}_\pi(e)$ of an expression $e$ in $\mathcal{N}(\pi, di, ^+)$ as follows, inductively:

- if $e$ is in $\mathcal{N}(di, ^+)$, then $\mathrm{depth}_\pi(e) = 0$;
- $\mathrm{depth}_\pi(\pi_1(e)) = \mathrm{depth}_\pi(\pi_2(e)) = \mathrm{depth}_\pi(e) + 1$;
- $\mathrm{depth}_\pi(e_1 \cup e_2) = \max(\mathrm{depth}_\pi(e_1), \mathrm{depth}_\pi(e_2))$;
- $\mathrm{depth}_\pi(e_1 \circ e_2) = \max(\mathrm{depth}_\pi(e_1), \mathrm{depth}_\pi(e_2))$; and
- $\mathrm{depth}_\pi(e^+) = \mathrm{depth}_\pi(e)$.

With every subexpression $\pi_i(f)$, $i = 1, 2$, of $e$, we can associate a conditional the semantics of which is precisely described by this subexpression $\pi_i(f)$. We denote the set of all these conditionals by $\mathit{\Pi}(e)$.

We can now show the following.

**Proposition 12.** *Let $m$ be a nonzero natural number, and let $e$ be an expression in $\mathcal{N}(\pi, di, ^+)$. Let $\ell := depth_\pi(e)$. Then, there exists a nonzero natural number $n$ depending only on $m$ and $e$ such that, for every EDAG $G$ of depth at most $m$, and, for every node $v$ of $G$, if there exists a node $w$ in $G$ such that $(v, w) \in e(G)$, then there exists a node $w'$ in $G^v_\ell$ such that $(v, w') \in e(G^v_\ell)$, with $G^v_\ell$ the $n$-canonical subgraph of $G$ of order $\ell$ for the set of conditionals $\Gamma := \Pi(e)$.*

Proposition 12 is shown by proving that an extended version of it holds for every subexpression of $e$, by induction on its $\pi$-nesting depth. Propositions 10 and 11 play a key role in this, where the former is needed to deal with the first projection and the latter to deal with the second projection. Notice that, for the expression $e$ itself, Proposition 10 already yields that, for some n-normal trace expression $f$ in $\mathcal{T}^{\mathrm{norm}}_n(e)$, there exists a homomorphism $h$ from the line pattern $\mathbf{L}(f)$ to $G$ such that $h(\mathbf{s}) = v$ and $h(\mathbf{L}(f))$ is contained in $G^v_0$, with $\mathbf{s}$ the source node of $\mathbf{L}(f)$ and $G^v_0$ the basic $n$-canonical subgraph of $G$. It now turns out that the $G^v_\ell$, the $n$-canonical subgraph of $G$ of order $\ell$, is an extension of $G^v_0$ for which each node of $G^v_0$ satisfying some conditional of $\Pi(e)$ in $G$ also satisfies this conditional in $G^v_\ell$, but not necessarily in $G^v_0$! Hence, $h$, while in general not a homomorphism from $\mathbf{L}(f)$ to $G^v_0$, is a homomorphism from $\mathbf{L}(f)$ to $G^v_\ell$, and we can then invoke Propositions 7 and 4 to obtain the conclusion of Proposition 12.

Now, from Proposition 9, it immediately follows that we can bound the number of nodes in $G^v_\ell$ by a number $s$ depending only on $m$ and $e$. Hence, we have all the ingredients needed to complete the second step of our proof strategy as explained at the end of Section 4, and we may thus conclude the following.

**Theorem 3.** *Let $m$ be a nonzero natural number, and let $e$ be an expression in $\mathcal{N}(\pi, di, ^+)$. Then, there exists a nonzero number $s$ depending only on $m$ and $e$ such that, for every EDAG $G$ with depth at most $m$, $e(G) \neq \emptyset$ if and only if $e'(G) \neq \emptyset$, where $e'$ is the expression in $\mathcal{N}(\pi, di)$ obtained from $e$ by exhaustively replacing each subexpression of the form $f^+$ by $\bigcup_{i=1}^{s} f^i$.*

Since the parameter $s$, the bound on the size of the graphs $G^v_\ell$ in Proposition 12, is of very high complexity in $m$, it may require very large graphs $G$ before the difference between $G$ and its subgraphs $G^v_\ell$ becomes significant.[4]

Combining Theorems 2 and 3, we see that $\mathcal{N}(\pi, di, ^+)$ collapses to $\mathcal{N}(\pi, di)$ at the level of boolean queries. Furthermore, if $F$ is a subset of $\{\pi, di\}$ and $e$ is more specifically an expression of $\mathcal{N}(F, ^+)$, then it follows that the expression $e'$ defined in Theorem 3 is more specifically in $\mathcal{N}(F)$. From our proof, we may therefore also conclude the following.

**Corollary 2.** *Let $F \subseteq \{\pi, di\}$. Then $\mathcal{N}(F, ^+)$ collapses to $\mathcal{N}(F)$ at the level of boolean queries.*

[4] For the same reason, it was not possible to "discover" Proposition 12 and the ensuing Theorem 3 by looking at simple examples.

## 9 Conclusions and Future Work

We now have a complete understanding of the impact of adding transitive closure to the relation algebra fragments considered. While it is well-known that transitive closure adds expressive power to all fragments at the level of path queries [4], and the same was established in previous work of the present authors [10] at the level of boolean queries on labeled graphs (multiple input relations), we have now established, in contrast, that, while adding transitive closure adds expressive power to most relation algebra fragments at the level of boolean queries on unlabeled graphs (a single input relation), it does *not* add expressive power to $\mathcal{N}(F)$, with $F$ a set of nonbasic features, if and only if $F \subseteq \{\pi, di\}$.

Towards future work, one may investigate similar problems for other logics. An operation we did not consider, for instance, is residuation. Residuation [22] is similar to the standard relational division operation in databases, and corresponds to the set containment join [18].

## References

1. RDF primer (2004), `http://www.w3.org/TR/rdf-primer/`
2. Abiteboul, S., Buneman, P., Suciu, D.: Data on the Web: From Relations to Semistructured Data and XML. Morgan Kaufmann (1999)
3. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison Wesley, Reading (1995)
4. Aho, A.V., Ullman, J.D.: The universality of data retrieval languages. In: Conference Record of the Sixth Annual ACM Symposium on Principles of Programming Languages, San Antonio, Texas, pp. 110–120 (January 1979)
5. Angles, R., Gutiérrez, C.: Survey of graph database models. ACM Comput. Surv. 40(1), 1–39 (2008)
6. Baader, F., Calvanese, D., McGuiness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook. Cambridge University Press (2003)
7. Benedikt, M., Fan, W., Kuper, G.M.: Structural Properties of XPath Fragments. In: Calvanese, D., Lenzerini, M., Motwani, R. (eds.) ICDT 2003. LNCS, vol. 2572, pp. 79–95. Springer, Heidelberg (2002)
8. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. Int. J. Semantic Web Inf. Syst. 5(3), 1–22 (2009)
9. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. Cambridge University Press (2001)
10. Fletcher, G.H.L., Gyssens, M., Leinders, D., Van den Bussche, J., Van Gucht, D., Vansummeren, S., Wu, Y.: Relative expressive power of navigational querying on graphs. In: Milo, T. (ed.) ICDT, pp. 197–207. ACM (2011)
11. Florescu, D., Levy, A., Mendelzon, A.: Database techniques for the World-Wide Web: A survey. SIGMOD Record 27(3), 59–74 (1998)
12. Franklin, M.J., Halevy, A.Y., Maier, D.: From databases to dataspaces: a new abstraction for information management. SIGMOD Record 34(4), 27–33 (2005)
13. Gyssens, M., Paredaens, J., Van Gucht, D., Fletcher, G.H.L.: Structural characterizations of the semantics of XPath as navigation tool on a document. In: Vansummeren, S. (ed.) PODS, pp. 318–327. ACM (2006)
14. Harel, D., Kozen, D., Tiuryn, J.: Dynamic Logic. MIT Press (2000)

15. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space, 1st edn. Synthesis Lectures on the Semantic Web: Theory and Technology, vol. 1. Morgan & Claypool Publishers (February 2011)
16. Libkin, L.: Elements of Finite Model Theory. Springer, Berlin (2004)
17. Maddux, R.D.: Relation Algebras. Elsevier, Amsterdam (2006)
18. Mamoulis, N.: Efficient processing of joins on set-valued attributes. In: Proceedings ACM SIGMOD International Conference on Management of Data, pp. 157–168 (2003)
19. Marx, M., Venema, Y.: Multi-Dimensional Modal Logic. Springer, Heidelberg (1997)
20. Marx, M.: Conditional XPath. ACM Trans. Database Syst. 30(4), 929–959 (2005)
21. Marx, M., de Rijke, M.: Semantic characterizations of navigational XPath. SIGMOD Record 34(2), 41–46 (2005)
22. Pratt, V.R.: Origins of the calculus of binary relations. In: Proceedings 7th Annual IEEE Symposium on Logic in Computer Science, pp. 248–254 (1992)
23. Tarski, A.: On the calculus of relations. J. of Symbolic Logic 6(3), 73–89 (1941)
24. Tarski, A., Givant, S.: A Formalization of Set Theory without Variables. American Mathematical Society (1987)
25. Wu, Y., Van Gucht, D., Gyssens, M., Paredaens, J.: A study of a positive fragment of Path queries: Expressiveness, normal form and minimization. Comput. J. 54(7), 1091–1118 (2011)