Noname manuscript No.
(will be inserted by the editor)

The impact of transitive closure on the expressiveness of navigational query languages on unlabeled graphs

George H. L. Fletcher · Marc Gyssens · Dirk Leinders · Jan Van den Bussche · Dirk Van Gucht · Stijn Vansummeren · Yuqing Wu

the date of receipt and acceptance should be inserted later

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. The expressive power of the languages thus obtained can not only be evaluated at the level of path queries (queries returning binary relations), but also at the level of boolean or yes/no queries (expressed by the nonemptiness of an expression). While, in general, adding transitive closure always augments the expressive power of the language under consideration, this is no longer true in the context on unlabeled graphs (i.e., in the case where there is only one input relation). In this paper, we show that this is indeed 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 [10,9], this result yield a complete understanding of the impact of transitive closure on the languages under consideration.

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

George H. L. Fletcher
Eindhoven University of Technology, Department of Mathematics and Computer Science, P.O. Box 513, NL-5600 MB Eindhoven, the Netherlands.
E-mail: g.h.l.fletcher@tue.nl

Marc Gyssens · Dirk Leinders · Jan Van den Bussche
Hasselt University and Transnational University of Limburg, School for Information Technology, Agoralaan, Building D, B-3590 Diepenbeek, Belgium.
E-mail: {marc.gyssens,dirk.leinders,jan.vandenbussche}@uhasselt.be

Dirk Van Gucht · Yuqing Wu
Indiana University, School of Informatics and Computing, Lindley Hall, 150 S. Woodlawn Ave., Bloomington, Indiana 47405
E-mail: {vgucht,yuqwu}@cs.indiana.edu

Stijn Vansummeren
Université Libre de Bruxelles, Campus du Solbosch, CP165/15, av. F.D. Roosevelt 50, B-1050 Brussels, Belgium.
E-mail: stijn.vansummeren@ulb.ac.be

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 navigated in accordance with the expression. By identifying nonemptiness with the boolean value *true* and emptiness with *false*, as is standard in database theory [2], 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., [6,13,18,19,25]). Motivated by data on the Web [1,11] and new applications such as dataspaces [12], Linked Data [7,15], and RDF [22], 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 [5,8,14,16,20,23]. 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 [4] 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 [3], 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). The argument used to prove this could not be generalized to the boolean queries on unlabeled graphs (i.e., on a single input relation), however. With different arguments, we were able to show [10,9] that, for labeled graphs, there is still no collapse if the language to which transitive closure is added has one one of the operators set difference, intersection, coprojection, or converse.

The purpose of the present paper is to show that in the remaining cases, i.e., if the language under consideration is the basic language augmented with none, one, or all of the features projection and diversity, adding transitive closure does *not* yield more expressive power at the level of boolean queries on unlabeled graphs. This result completes 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.

The emphasis of this paper is on the proof technique used for establishing the collapse results, which we think is interesting in its own right.

The remainder paper is organized as follows. In Section 2, we define syntax and semantics of the class of languages studied in the paper. In Section **??**, 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 **??**, 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 8 to 13, we deal with the much more elaborate second step. We conclude in Section 14 by summarizing our understanding of the impact of adding transitive closure to relation algebra fragments, which has now been completed.

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

[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.

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). For future use, we put $e^0 := id$.

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.

We refer to the language $\mathcal{N}(\setminus, di, {}^{-1})$ 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 [16].

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 symbol, 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 8 and illustrated in Example 2.

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

[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.

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 $\mathrm{Sub}(e)$, is defined recursively, as follows:

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

An expression that is either "R", "id", "di", or a conditional is called *atomic*. 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 $|e|$ the number of occurrences of "R" in e.

3 Trace expressions

If we evaluate an expression e in $\mathcal{N}(\Gamma, \pi, di, {}^+)$, with Γ a set of conditionals, then, to validate that, for some nodes v and w in a graph G, $(v, w) \in e(G)$, we must in general make some choices to arrive at that result. In particular, when evaluating a subexpression $f_1 \cup f_2$, we must decide whether to evaluate f_1 or f_2, Similarly, if we encounter a subexpression f^+, we must decide how many times we are going to iterate over f. To formalize this, we introduce *trace expressions*.

Definition 3 Let e be an expression in $\mathcal{N}(\Gamma, di, {}^+)$, with Γ a set of conditionals. Then, $\mathcal{T}(e)$, the set of *trace expressions* of e, is defined recursively, as follows:

- if e is an atomic expression, then $\mathcal{T}(e) = \{e\}$;
- if for $i = 1, 2$, $\mathcal{T}(\pi_i(e)) = \{\pi_i(f) \mid f \in \mathcal{T}(e)\}$;
- $\mathcal{T}(e_1 \cup e_2) = \mathcal{T}(e_1) \cup \mathcal{T}(e_2)$;
- $\mathcal{T}(e_1 \circ e_2) = \{f_1 \circ f_2 \mid f_1 \in \mathcal{T}(e_1)\ \&\ f_2 \in \mathcal{T}(e_2)\}$; and
- $\mathcal{T}(e^+) = \bigcup_{k \geq 1} \{f_1 \circ \cdots \circ f_k \mid \forall i = 1, \ldots, k : f_i \in \mathcal{T}(e)\}$.

Notice that, indeed, trace expressions do not contain union and transitive closure. Our earlier intuition is now captured by Proposition 1, which follows from a straightforward structural induction argument.

Proposition 1 *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 $f \in \mathcal{T}(e)$ such that $(v,w) \in f(G)$.*

Notice that it is possible to formally link a trace expression with a particular way of evaluating the original expression (hence the name "trace"). Formally, this can be done by first marking the original expression. That is, we tag each symbol in that expression with its index in that expression, interpreted as a mathematical sequence. We can then define marked traces in much the same way as above, which allow us to associate symbols in the trace with symbols in the original expressions in an unambiguous manner.

Notice that it is possible that different marked traces define the same unmarked expression. In other words, it is not always possible to link the symbols of an unmarked trace with the symbols of the original expression in an unambiguous manner.

In this work, we shall not introduce marked traces formally, to avoid overloading the notation. Nevertheless, we shall assume implicitly for each trace we consider that a marking is available that links the symbols in the trace with symbols in the original expressions.

4 Describing the context

In this section, we describe some results [10, 9] necessary to understand the context of the results of the present paper.

First, we observe that there exists the following interdependencies between the features introduced in Section 2:

$$\begin{aligned}
\pi_1(e) &= (e \circ e^{-1}) \cap id = (e \circ (id \cup di)) \cap id = \overline{\pi}_1(\overline{\pi}_1(e));\\
\pi_2(e) &= (e^{-1} \circ e) \cap id = ((id \cup di) \circ e) \cap id = \overline{\pi}_2(\overline{\pi}_2(e));\\
\overline{\pi}_1(e) &= id \setminus \pi_1(e);\\
\overline{\pi}_2(e) &= id \setminus \pi_2(e);\\
e_1 \cap e_2 &= e_1 \setminus (e_1 \setminus e_2).
\end{aligned}$$

For a set of nonbasic features F not containing transitive closure, let $\bar{F}$ be the set obtained by augmenting F with all nonbasic features that can be expressed in $\mathcal{N}(F)$ through a repeated application of the above equalities. For example, $\overline{\{\setminus, ^{-1}\}} = \{\setminus, ^{-1}, \cap, \pi, \overline{\pi}\}$.

The present authors have been able to show the following result.

Proposition 2 *[10] Let F_1 and F_2 be sets of nonbasic features not containing transitive closure. The language $\mathcal{N}(F_1)$ is at most as expressive as the language $\mathcal{N}(F_2)$ at the level of path queries if and only if $F_1 \subseteq \bar{F}_2$.*

For boolean queries, the situation is slightly more complicated. at turns out that, under certain conditions, converse can be eliminated.

Proposition 3 *[10] Let F be a set of nonbasic features not containing transitive closure. If $\bar{F}$ does not contain intersection, then $\mathcal{N}(F \cup \{^{-1}\})$ is at most as expressive as $\mathcal{N}(F \cup \{\pi\})$.*

So, in the presence of projection and in the absence of intersection, converse does not add expressive power at the boolean level. To accommodate this additional result, we define the following notion, given a set set of nonbasic features F not containing transitive closure:

$$\tilde{F} = \begin{cases} \bar{F} \cup \{^{-1}\} & \text{if } \pi \in \bar{F} \text{ and } \cap \notin \bar{F}; \\ \bar{F} & \text{otherwfise.} \end{cases}$$

For example, $\widetilde{\{\overline{\pi}, di\}} = \{^{-1}, \pi, \overline{\pi}, di\}$.

The present authors were able to establish the following analogue of Proposition 2for boolean queries.

Proposition 4 *[10] Let F_1 and F_2 be sets of nonbasic features not containing transitive closure. The language $\mathcal{N}(F_1)$ is at most as expressive as the language $\mathcal{N}(F_2)$ at the level of boolean queries if and only if $F_1 \subseteq \tilde{F}_2$.*

In particular, Proposition 4 establishes which relation algebra fragments not containing transitive closure are equivalent in expressive power at the level of boolean queries.

What happens if we add transtive closure to these fragments?

At the level of path queries, the answer is straightforward, as it is well-known that the expression R^+ represents a query not expressible in FO (see, e.g., [2]). Hence, adding transitive closure always strictly increases the expressive power at the level of path queries. At the level of boolean queries, the situation is more subtle. Indeed, the argument above is no longer applicable, as $R^+ \neq \emptyset$ if and only if $R \neq \emptyset$. Using a straightforward Ehrenfeucht-Fraïssé argument (see, e.g., [2]), it is nevertheless still possible to show that the boolean query represented by the expression $R \circ S^+ \circ R$ is not expressible in FO. However, this expression contains *two* relation names. Hence, also at the level of Boolean queries, adding transitive closure always strictly increases the expressive power, but only if labeled input graphs with at least two relation names are allowed. This begs the question whether this result still holds for *unlabeled* input graphs. Using ad-hoc arguments, the present authors were able to establish the following.

Proposition 5 *[10, 9] Let F be a set of nonbasic features such that $\bar{F}$ contains at least one of intersection, converse, or coprojection. Then, adding transitive closure to $\mathcal{N}(F)$ strictly increases the expressive power.*

Taking into account Proposition 4, four relation algebra fragments are not covered by Proposition 5: $\mathcal{N}$, $\mathcal{N}(\pi)$, $\mathcal{N}(di)$, and $\mathcal{N}(\pi, di)$. It is the purpose of the present paper to prove that adding transitive closure to these fragments does *not* increase their expressive power.

5 General proof strategy

In this section, we describe in very general terms the proof strategy we use to show that $\mathcal{N}(F \cup \{^+\})$ collapses to $\mathcal{N}(F)$ for all sets of nonbasic features F for which $F \subseteq \{\pi, di\}$.

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, for $F \subseteq \{\pi, di\}$, $\mathcal{N}(F \cup \{^+\})$ collapses to $\mathcal{N}(F)$ at the level of boolean queries. In this section, we outline this technique, and, in subsequent sections, we work it out in further detail. It consists of two steps. Given an expression e in $\mathcal{N}(F \cup \{^+\})$,

1. find an expression $\mathit{suff}_{F,e}$ $\mathcal{N}(F)$ such that, for every graph G, $\mathit{suff}_{F,e}(G) \neq \emptyset$ 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 $\mathit{suff}_{F,e}(G) = \emptyset$.

It then follows immediately that, on all graphs, e is equivalent to $\mathit{suff}_{F,e} \cup e'$ at the level of boolean queries, i.e., for every graph G, $\mathit{suff}_{F,e} \cup e'(G) \neq \emptyset$ if and only if $e(G) \neq \emptyset$. Intuitively, $\mathit{suff}_{F,e}(G) \neq \emptyset$ 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 $\mathit{suff}_{F,e}$ is well-chosen, then the latter condition will turn out to be sufficiently restrictive for our purposes.

6 The first step

The first step of the proof strategy described in Section 5 is, given $F \subseteq \{\pi, di\}$ and an expression e in $\mathcal{N}(F \cup \{^+\})$, finding an expression $\mathit{suff}_{F,e}$ in $\mathcal{N}(F)$ for which $\mathit{suff}_{F,e}(G) \neq \emptyset$ implies $e(G) \neq \emptyset$ for every input graph G. This first step is secured by a series of lemmas, summarized in Theorem 1.

We start with the following straightforward observations.

Lemma 1 *Let G_1 and G_2 be graphs, and let h be a homomorphism from G_1 to G_2.*

- *Let e be an expression in $\mathcal{N}(\pi, ^+)$. Then $(v, w) \in e(G_1)$ implies $(h(v), h(w)) \in e(G_2)$.*
- *Let e be an expression in $\mathcal{N}(\pi, di, ^+)$. If f is* injective*, then $(v, w) \in e(G_1)$ implies $(h(v), h(w)) \in e(G_2)$.*

Arguably, the simplest graphs we can consider in this contexts are *chains*. A chain of length m, denoted C_m, is a graph consisting of nodes $v_0, \ldots, v_m$ and edges between subsequent nodes. Lemma 1 can then help us to link the behavior of an expression on a such a chain to the behavior of that expression on the given input graph.

Lemma 2 *Let e be an expression in $\mathcal{N}(\pi, ^+)$. Then, for $m \geq |e|$, $e(C_m) \neq \emptyset$.*

Proof The proof is a structural induction argument. The only non-straightforward case to consider is the induction step for composition. Thus, suppose that $e = e_1 \circ e_2$, and that e_1 and e_2 satisfy Lemma 2. In particular, $e_1(C_{|e_1|}) \neq \emptyset$ and $e_2(C_{|e_2|}) \neq \emptyset$. Let the chain $C_{|e_1|}$ consist of the nodes $v_0, \ldots, v_{|e_1|}$ and $C_{|e_2|}$ consist of the nodes $w_0, \ldots, w_{|e_2|}$. Let $(v_i, v_j) \in e_1(C_{|e_1|})$ and $(w_k, w_l) \in e_2(C_{|e_2|})$. Finally, for $m \geq |e| = |e_1| + |e_2|$, let C_m consist of the nodes $z_0, \ldots, z_m$. We now distinguish two cases.

1. $j \geq k$. Consider the homomorphism from $C_{|e_1|})$ to C_m mapping v_0 to z_0, and hence v_i to z_i and v_j to z_j. By Lemma 1, $(z_i, z_j) \in e_1(C_m)$. Since $j \geq k$, there exists a homomorphism from $C_{|e_2|})$ to C_m mapping w_k to z_j, and hence w_l to z_{j+l-k}. By Lemma 1, $(z_j, z_{j+l-k}))) \in e_2(C_m)$. Hence, $(z_i, z_{j+l-k}) \in e(C_m)$.
2. $j < k$. Consider the homomorphism from $C_{|e_2|})$ to C_m mapping $w_{|e_2|}$ to z_m, and hence w_k to $z_{k+m-|e_2|}$ and w_l to $z_{l+m-|e_2|}$. By Lemma 1, $(z_{k+m-|e_2|}, z_{l+m-|e_2|}) \in e_2(C_m)$. Since $j < k$, there exists a homomorphism from $C_{|e_1|})$ to C_m mapping v_j to $z_{k+m-|e_2|}$, and hence v_i to $z_{k+i-j+m-|e_2|}$. By Lemma 1, $(z_{k+i-j+m-|e_2|}, z_{k+m-|e_2|}) \in e_1(C_m)$. Hence, $(z_{k+i-j+m-|e_2|}, z_{l+m-|e_2|}) \in e(C_m)$.

In both cases, we find that $e(C_m) \neq \emptyset$.

Using the above lemmas, we can easily find an expression $suff_{F,e}$ if $F \subseteq \{pi\}$.

Lemma 3 *Let e be an expression in $\mathcal{N}(\pi, ^+)$, and let G be a graph. If $R^{|e|}(G) \neq \emptyset$, then $e(G) \neq \emptyset$.*

Proof The condition $R^{|e|}(G) \neq \emptyset$ is equivalent to the existence of a homomorphism from $C_{|e|}$ to G. By Lemma 2, $e(C_{|e|}) \neq \emptyset$. Hence, by Lemma 1, $e(G) \neq \emptyset$.

We now consider the case where $F = \{di\}$.

Lemma 4 *Let e be an expression in $\mathcal{N}(di, ^+)$, and let G be a graph. If $R^{|e|} \circ di \circ R^{|e|}(G) \neq \emptyset$, then $e(G) \neq \emptyset$.*

Proof We first consider an expression f in $\mathcal{N}(di)$ of the form $f := R^{m_1} \circ di \circ R^{m_2} \circ di \circ \cdots \circ diR^{m_n}$, $n \geq 1$, $1 \leq m_1, \ldots, m_n \leq |e|$, and prove that it returns a nonempty result on graphs satisfying $R^{|e|} \circ di \circ R^{|e|}(G) \neq \emptyset$.[4] Thereto, we distinguish two cases.

1. *There exist nodes v_1, w_1, v_2, and w in G such that $(v_1, w_1) \in R^{|e|}(G)$, $(v_2, w_2) \in R^{|e|}(G)$, and $v_1 \neq v_2$.* For $i = 1, \ldots, n$, let $f_i = R^{m_1} \circ diR^{m_2} \circ di \circ \cdots \circ di \circ R^{m_i}$. We prove by induction that there exist nodes $z_1, \ldots, z_n$ in G such that $(v_1, z_i) \in f_i(G)$. For the base case, $i = 1$, this follows from $m_1 \leq |e|$. Assume that we have already established that for some node z_i of G, $(v_1, z_i) \in f_i(g)$. Since

[4] Notice that this statement is voidly true if $|e| = 0$.

$v_1 \neq v_2$, $z_i \neq v_1$ or $z_i \neq v_2$. Without loss of generality, assumme the latter. Since $(v_2, w_2) \in R^{|e|}(G)$ and $m_{i+1} \leq |e|$, it follows that there exists a node z_{i+1} in G such that $(v_2, z_{i+1}) \in R^{m_{i+1}}(G)$. Hence, $(v_0, z_{i+1}) \in f_{i+1}(G)$, as was to be shown. We find in particular that $f(G) = f_n(G) \neq \emptyset$.

2. *There is only one node v in G such that, for some node w in G, $(v, w) \in R^{|e|}(G)$.* From $R^{|e|} \circ di \circ R^{|e|}(G) \neq \emptyset$, it follows that $v \neq w$.
   We distinguish two subcases.
   (a) $(v, v) \in R$. Notice that there must also exist a node z in G with $(v, z) \in R$ and $v \neq z$. Otherwise, it would be impossible that $(v, w) \in R^{|e|}$. From $(v, v) \in R$ and $(v, z) \in R$, it follows that, for all $m \geq 1$, $(v, z) \in R^m(G)$. Since $v \neq z$, we may conclude that also $(v, z) \in f(G)$. In particular, $f(G) \neq \emptyset$.
   (b) $(v, v) \notin R$. By assumption, there exists nodes $v = v_0, v_1, \ldots, v_{|e|} = w$, such that, for $m = 0, \ldots, |e| - 1$, $(v_m, v_{m+1}) \in R$. From the assumption in this subcase, it immediately follows that $v = v_0 \neq v_1$. Next, consider node v_m for $2 \leq m \leq |e|$. If $v = v_0 = v_m$, then there exists k, $0 \leq k \leq m-1$, such that $(v_1, v_k) \in R^{|e|}(G)$, contradicting the assumption that $v = v_0$ is the unique node for which there exist a node w such that $(v, w) \in R^{|e|}(G)$. Now, for $m = 1, \ldots, |e|$, $(v, v_m) \in R^m(G)$, and $v \neq v_m$. Therefore, we may conclude as in Subcase 2a that $(v, v_{m_n}) \in f(G)$. In particular, $f(G) \neq \emptyset$.

Notice that it also follows from our reasoning that expressions of the form

$$\begin{aligned} &di \circ R^{m_1} \circ di \circ R^{m_2} \circ di \circ \cdots \circ di R^{m_n}\,; \\ &\quad R^{m_1} \circ di \circ R^{m_2} \circ di \circ \cdots \circ di R^{m_n} \circ di\,; \text{ and} \\ &di \circ R^{m_1} \circ di \circ R^{m_2} \circ di \circ \cdots \circ di R^{m_n} \circ di\,, \end{aligned}$$

$n \geq 1$, $1 \leq m_1, \ldots, m_n \leq |e|$, returns a nonempty result on graphs satisfying $R^{|e|} \circ di \circ R^{|e|}(G) \neq \emptyset$, since this condition implies that G contains at least two nodes.

Now, consider a trace expression $f \in \mathcal{T}(e)$ for which $|f| \leq |e|$. (All trace expressions obtained by iterating only once over all transitive closure subexpressions satisfy this condition.) First, superfluous occurences of "id" can be eliminated from f. Next, remember that a graph G satisfying $R^{|e|} \circ di \circ R^{|e|}(G) \neq \emptyset$ contains at least two nodes. If G contains extactly two nodes, then di^k is equivalent to id if k is even and to di if k is odd. Otherwise, di^k is equivalent to $id \cup di$. It follows that, if $|e| = 0$, then, on G, f is equivalent to either id, di, or $id \cup di$. In each case $f(G) \neq \emptyset$. If $|e| > 0$, then, on G, f is equivalent to a union of expressions of the type considered before. Hence, also in this case, we may conclude that $f(G) \neq \emptyset$.

From Proposition 1, it now immediately follows that, in all cases, $e(G) \neq \emptyset$.

Finally, we deal with the case where $F = \{\pi, di\}$.

Lemma 5 *Let e be an expression in $\mathcal{N}(\pi, di, ^+)$, and let G be a graph. If $\pi_2(R^{|e|}) \circ \pi_1(R^{|e|}) \circ di \circ \pi_2(R^{|e|}) \circ \pi_1(R^{|e|}) \neq \emptyset$, then $e(G) \neq \emptyset$.*

Proof We first observe that the condition $\pi_1(R^{|e|}) \circ \pi_2(R^{|e|}) \circ di \circ \pi_1(R^{|e|}) \circ \pi_2(R^{|e|})(G) \neq \emptyset$ is equivalent to the existence of two sequences of not necessarily all different nodes $v_{-|e|}, \ldots, v_{-1}, v_0, v_1, \ldots, v_{|e|}$ and $w_{-|e|}, \ldots, w_{-1}, w_0, w_1, \ldots, w_{|e|}$ in G such that, (1) for $i = -|e|, \ldots, |e| - 1$, $(v_i, v_{i+1}) \in R$ and $(w_i, w_{i+1}) \in R$ and (2) $v_0 \neq w_0$.

Let H be the subgraph of G consisting of the nodes and edges singled out above. Let f be in $\mathcal{N}(\pi, di)$ with $|f| \leq |e|$. We show by a nested inductive argument that,

1. for all $i = 0 \ldots, |e| - |f|$, there exists j with $0 \leq j \leq i + |f|$ such that $(v_i, v_j) \in f(H)$ or $(v_i, w_j) \in f(H)$;
2. for all $i = -|e| \ldots, 0$, there exists j with $i \leq j \leq |f|$ such that $(v_i, v_j) \in f(H)$ or $(v_i, w_j) \in f(H)$;
3. for all $i = 0, \ldots, |e| - |f|$, there exists j with $0 \leq j \leq i + |f|$ such that $(w_i, v_j) \in f(H)$ or $(w_i, w_j) \in f(H)$;
4. for all $i = -|e| \ldots, 0$, there exists j with $i \leq j \leq |f|$ such that $(w_i, v_j) \in f(H)$ or $(w_i, w_j) \in f(H)$;
5. for all $i = -|e| + |f|, \ldots, 0$, there exists j with $i - |f| \leq j \leq 0$ such that $(v_j, v_i) \in f(H)$ or $(w_j, v_i) \in f(H)$;
6. for all $i = 0, \ldots, |e|$, there exists j with $-|f| \leq j \leq i$ such that $(v_j, v_i) \in f(H)$ or $(w_j, v_i) \in f(H)$;
7. for all $i = -|e| + |f|, \ldots, 0$, there exists j with $i - |f| \leq j \leq 0$ such that $(v_j, w_i) \in f(H)$ or $(w_j, w_i) \in f(H)$; and
8. for all $i = 0, \ldots, |e|$, there exists j with $-|f| \leq j \leq i$ such that $(v_j, w_i) \in f(H)$ or $(w_j, w_i) \in f(H)$.

We first show the first statement for the case that f is projection-free, i.e., that f is in $\mathcal{N}(di)$. First, observe that, always, $(v_i, v_i) \in id(H)$. We can view each union-free expression in $\mathcal{N}(di)$ as a composition of "id" with none, one, or more factors "R" or "di". Thus, assume that f is a union-free expression in $\mathcal{N}(di)$ with $|f| \leq |e|$, and let $f = g \circ R$, where g satisfies the first statement above. Let $0 \leq i \leq |e| - |f|$. Since $|f| = |g| + 1$, $0 \leq i \leq |e| - |g|$. By the first statement of the induction hypothesis, there exists j with $0 \leq j \leq i + |g|$ such that $(v_i, v_j) \in f(H)$ or $(v_i, w_j) \in f(H)$. Observe that $j \leq i + |g| \leq |e| - |f| + |g| = |e| - 1$. Hence, $(v_j, v_{j+1}) \in R$ and $(w_j, w_{j+1}) \in R$, as a consequence of which $(v_i, v_{j+1}) \in f(H)$ or $(v_i, w_{j+1}) \in f(H)$. Finally, notice that $j+1 \leq i+|g|+1 = i+|f|$. Alternatively, assume that $f = g \circ di$, where g satisfies the first statement above. Let $0 \leq i \leq |e| - |f|$. Since $|f| = |g|$, $0 \leq i \leq |e| - |g|$. By the induction hypothesis, there exists j with $0 \leq j \leq i + |g|$ such that $(v_i, v_j) \in f(H)$ or $(v_i, w_j) \in f(H)$. Without loss of generality, assume the latter. Since $v_0 \neq w_0$, $w_j \neq v_0$ or $w_j \neq w_0$. Again without loss of generality, assume the latter. Then $(v_i, w_0) \in f(H)$. We have thus shown that the first statement holds for all union-free expressions f in $\mathcal{N}(di)$ with $|f| \leq |e|$. The other statements this case are shown analagously.

We now consider the general case, and use the case above as the basis for an induction on the number of projection subexpressions in the expression under consideration. We focus again of the first statement. Thus, assume that f is in $\mathcal{N}(\pi, di)$ with $|f| \leq |e|$, and that $0 \leq i \leq |e| - |f|$. If f is not projection-free, we can write $f = f_1 \circ \pi_1(f_2) \circ f_3$ or $f = f_1 \circ \pi_2(f_2) \circ f_3$, with f_1 projection-free, and f_2 and f_3 containing fewer projection subexpressions than f. By the first statement of the basis of this induction, there exists j, $0 \leq j \leq i + |f_1|$, such that $(v_i, v_j) \in f_1(H)$ or $(v_i, w_j) \in f_1(H)$. Without loss of generality, assume that latter. Clearly, $j \leq |e_R| - (|f_2| + |f_3|)$, in particular, $j \leq |e_R| - |f_2|$ and $j \leq |e_R| - |f_3|$. By the latter condition and the third statement of the induction hypothesis, there exists k, $0 \leq k \leq j + |f_3| \leq i + |f_1| = |f_3| \leq i + |f|$ such that $(w_j, v_k) \in f_3(H)$ or $(w_j, w_k) \in f_3(H)$. We now distinguish the two cases.

1. $f = f_1 \circ \pi_1(f_2) \circ f_3$. As above, we can derive from the the third statement of the induction hypothesis that there exists l, $0 \leq l \leq j + |f_2|$ such that $(w_j, v_l) \in f_2(H)$ or $(w_j, w_l) \in f_2(H)$. In particular, $(w_j, w_j) \in \pi_1(f_2)(H)$. Combining

Table 1 Expressions $\mathit{suff}_{F,e}$ in $\mathcal{N}(F)$ for which $\mathit{suff}_{F,e}(G) \neq \emptyset$ implies $e(G) \neq \emptyset$, $F \subseteq \{\pi, di\}$.

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

everything together, we find that $(v_i, v_k) \in f(H)$ or $(v_i, w_k) \in f(H)$, with k in the desired range.'

2. $f = f_1 \circ \pi_2(f_2) \circ f_3$. By the last statement of the induction hypothesis, it follows that there exists l, $-|f_2| \leq l \leq j$, such that $(v_l, w_j) \in f_2(H)$ or $(w_l, w_j) \in f_2(H)$. In particular, $(w_j, w_j) \in \pi_2(f_2)(H)$. Combining everything together, we find that, also in this case, $(v_i, v_k) \in f(H)$ or $(v_i, w_k) \in f(H)$, with k in the desired range.

The induction step for the other seven statements is analagous.

We thus shown that, for every union-free expression f in $\mathcal{N}(\pi, di)$ with $|f| \leq |e|$, that $f(H) \neq \emptyset$, and, hence, by Lemma 1, that $f(G) \neq \emptyset$. Since all trace expressions $f \in \mathcal{T}(e)$ obtained by iterating only once over transitive closure subexpressions satisfy $|f| \leq |e|$, it follows from Proposition 1 that also $e(G) \neq \emptyset$.

Theorem 1 below summarizes Lemmas 3, 4, and 5.

Theorem 1 *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. Then, for every graph G, $\mathit{suff}_{F,e}(G) \neq \emptyset$ implies $e(G) \neq \emptyset$.*

7 Proof strategy for the second step

In Section 6, we established, for $F \subseteq \{\pi, di\}$ and e an expression in $\mathcal{N}(F \cup \{^+\})$, the existence of an expression $\mathit{suff}_{F,e}$ in $\mathcal{N}(F)$ such that, for every graph G, $\mathit{suff}_{F,e}(G) \neq \emptyset$ implies $e(G) \neq \emptyset$.

The second step in our general proof strategy requires finding an expression e' in $\mathcal{N}(F)$ such that, for every graph G satisfying $\mathit{suff}_{F,e}(G) \neq \emptyset$, $e'(G) \neq \emptyset$ if and only if $e(G) \neq \emptyset$. (As explained before, we may then conclude that e is equivalent to $\mathit{suff}_{F,e}(G) \cup e'$ at the level of Boolean queries.)

For that purpose, we need to know some information on how a graph G satisfying $\mathit{suff}_{F,e}(G) \neq \emptyset$ looks like.

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

Definition 4 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 6 *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 Lemma 5). 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 6 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.

We are now ready to bootstrap Lemma 6, as follows.

Proposition 6 *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 $2|e|$.*

Proof We first observe that $R^{|e|}(G) = \emptyset$ implies that $R^{|e|}(G) \circ di \circ R^{|e|}(G) = \emptyset$ and that $R^{|e|}(G)\circ di\circ R^{|e|}(G) = \emptyset$ implies that $\pi_1(R^{|e|})\circ\pi_2(R^{|e|})\circ di\circ\pi_1(R^{|e|})\circ\pi_2(R^{|e|}) = \emptyset$. Proposition 6 now follows from Lemma 6.

Now assume that we are given an expression e in $\mathcal{N}(\pi, di, ^+)$ and an EDAG G of depth at most $2|e|$. The remainder of this paper is concerned with proving that there exists a nonzero natural number m_e depending only on 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 $\bigcup_{i=1}^{m_e} f^i$.

Notice that this expression is in $\mathcal{N}(F)$, $F \subseteq \{\pi, di\}$, whenever e is in $\mathcal{N}(F\cup\{tc\})$. Hence, there is no need to threat the cases $F = \emptyset$, $F = \{\pi\}$ and $F = \{di\}$ separately.

To achieve our goal, we intend to show (Proposition 14) that there exists a nonzero natural number m_e such that, for any node v of G, there exists a subgraph G_v of G containing v which has at most m_e 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 property is sufficient for our purposes, assume first that $e(G) = \emptyset$. Then $e'(G) = \emptyset$, since, by construction, $e'(G) \subseteq e(G)$. 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 m_e nodes, $e(G_v) = e'(G_v)$. It follows that $e'(G_v) \neq \emptyset$. By Lemma **??**, $e'(G_v) \subseteq e'(G)$, and, hence, we also have that $e'(G) \neq \emptyset$.

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

8 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 con-

ditionals, 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.

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.

Once we have a partial result for this case, we will reintroduce the projections and bootstrap the initial result to the desired result.

To this end, 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 will choose p as a function of the number of projection subexpressions in the expression under consideration.

9 Line patterns and graph patterns

Let $\Gamma = \{c_1, \ldots, c_p\}$ be a finite set of conditionals. A useful property for union-free expressions in $\mathcal{N}(\Gamma, di)$ is that 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.

From a straightforward inductive argument, we can derive the following result.

Proposition 7 *Let $\Gamma = \{c_1, \ldots, c_p\}$ be a finite set of conditionals and let f be a union-free expression in $\mathcal{N}(\Gamma, di)$, and let G be a graph. There exist nodes v and w in G such that $(v, w) \in f(G)$ if and only if there exists a homomorphism h from $\boldsymbol{L}(f)$ to G such that $h(\boldsymbol{s}) = v$ and $h(\boldsymbol{t}) = w$.*

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

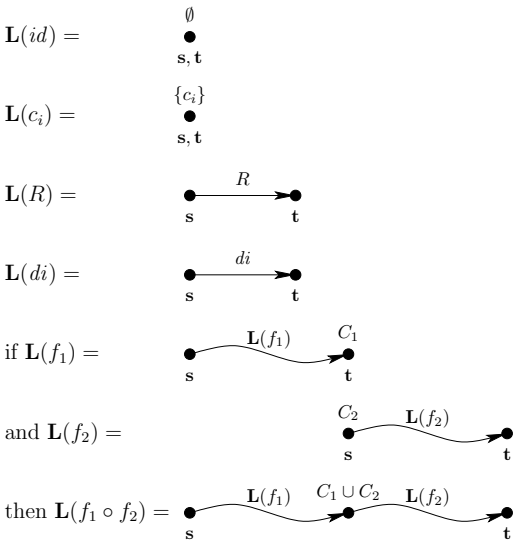

Fig. 1 Definition of the line pattern $\mathbf{L}(f)$ of a union-free expression in $\mathcal{N}(\Gamma, di)$.

3. for each edge $(\mathbf{v}, \mathbf{w})$ of $\mathbf{P}$ labeled by "di", $h(\mathbf{v}) \neq h(\mathbf{w})$.

An example of graph pattern that is not a line pattern can be seen in Figure 2.

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 11 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 natural numbers depending only on m and e. One of these subgraphs will turn out to be the subgraph G_v mentioned at the end of Section 7, for appropriate choices of the conditionals and their semantics, and for $m = 2|e|$.

10 Normalizing trace expressions

In Section 9, we associated line patterns with union-free expression in $\mathcal{N}(\Gamma, di)$, with $\Gamma = \{c_1, \ldots, c_p\}$ a set of conditionals. These contain the trace expressions of expressions in $\mathcal{N}(\Gamma, di, ^+)$.

Not all trace expressions will be useful for our purposes, and, in addition, trace expressions may contain a lot of redundany. redundancy. Therefore, we define the following notions.

Definition 5 Let $\Gamma = \{c_1, \ldots, c_p\}$ be a set of conditionals, and let $n \geq 0$. An expression g in $\mathcal{N}(\Gamma)$ is n-normal if (1) g is union-free, (2) $|g| \leq n$, and (3) a subexpression of g consisting only of "id" conditionals, and composition is either "id" or does not contain "id" and contains 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 $\Gamma = \{c_1, \ldots, c_p\}$ be a set of conditionals, and let $n \geq 0$. 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 $\mathcal{N}_n^{\text{norm}}(\Gamma, di)$.

We now define the set $\mathcal{T}_n^{\text{norm}}(e)$ of *n-normal trace expressions* as the set of all expressions in $\mathcal{N}_n^{\text{norm}}(\Gamma, di)$ for which there exists an equivalent expression in $\mathcal{T}(e)$ at the level of path queries. The following results links expressions in $\mathcal{N}(\Gamma, di, ^+)$ to normalized trace expressions in $\mathcal{N}(di, \Gamma)$ in the context of an EDAG of bounded depth.

Proposition 8 *Let e be an expression in $\mathcal{N}(di, \Gamma, ^+)$, let $m \geq 0$, and let G be an EDAG of depth at most m. Then, there exists a number M only dependent on e and m such that, for all nodes v and w of G, $(v, w) \in e(G)$ if and only if there exists an M-normal trace expression f in $\mathcal{T}_M^{\text{norm}}(e)$ for which $(v, w) \in f(G)$.*

Proof By Proposition 1, $(v, w) \in e(G)$ if and only if there exists a trace expression f in $\mathcal{T}(e)$ for which $(v, w) \in f(G)$. In particular, this settles the "if". For the "only if", assume that f is a trace expression of minimal length for which $(v, w) \in f(G)$. It remains to show that we can "normalize" f.

By Proposition 7, there exists a homomorphism h from $\mathbf{L}(f)$ to G with $h(\mathbf{s}) = v$ and $h(\mathbf{t}) = w$. Now consider a "di"-free subexpression g of f of maximal length. Consider the path in G defined by $h(\mathbf{L}(g))$. Notice that the length of this path, measured in the DAG underlying G, is at most m. Hence, if this path does not contain a node with a self-loop, then there are also at most m occurrences of the symbol "R" in g. Thus assume that on this path there is a node with a self loop, and hence precisely one (cf. Definition 4), say, z.

Now, assume there is a subexpression f_1 of f that is a trace of k subsequent iterations of e_1, with e_1^+ a transitive-closure subexpression of e, such that the following conditions are satisfied:

1. the first "R" symbol in g mapped by h to the self-loop in z corresponds in e to an "R" symbol in the first of the k iterations under consideration of e_1;
2. the last "R" symbol in g mapped by h to the self-loop in z corresponds in e to an "R" symbol in the last of the k iterations under consideration of e_1.

Consequently, f_1 need not be a maximal subexpression of f that is a trace of consecutive iterations of e_1 in e.

Suppose, for the sake of contradiction, that $k > 2$. Let g_1 be the subexpression of g corresponding to the k iterations under consideration of e_1 in e, except for the first and the last one.[5] Obviously, h maps all nodes of the subpattern $\mathbf{L}(g_1)$ of $\mathbf{L}(f)$ to z. Hence, we can omit g_1 from f and still retain a trace expression $\hat{f}$ for which h is a homomorphism mapping $\mathbf{L}(\hat{f})$ to G such that $h(\mathbf{s}) = v$ and $h(\mathbf{t}) = w$, contradicting our assumption that f has minimal length. Hence, $k \leq 2$.

Now, let $\hat{e}$ be the expression in $\mathcal{N}(di, \pi)$ obtained from e by recursively substituting each subexpression of the form e_1^+ in e by $e_1 \cup e_1^2$. By the above argument,

[5] In other words, g_1 is a trace of $k - 2$ iterations of e_1.

it follows that the minimal subexpression of f containing all "R" symbols of g mapped to the self-loop in z by h is also a subexpression of a trace of $\hat{e}$. Hence, the number of these "R" symbols is bounded by $|\hat{e}|$, the length of $\hat{e}$. Notice that this number solely depends on e. The number of "R" symbols of g *not* mapped to the self-loop in z is bounded by m, by the same argument as before. We may therefore conclude that, in all cases, the total number of "R" symbols in g is bounded by $M := m + |\hat{e}|$.

Finally, we can rewrite f as $f' = g_1 \circ di \circ g_2 \circ di \circ \cdots \circ g_{n-1} \circ di \circ g_n$, with $g_1, \ldots, g_n \in \mathcal{N}(\Gamma)$, by inserting "$id$" primitives where needed. By our previous argument, the number of "R" symbols in g_i, $1 \leq i \leq n$, is bounded by M. Without loss of generality, we may also assume that subexpressions of f' consisting solely of "id", conditionals, and composition are either "id" or do not contain "id", and contain each conditional occurs at most once, by removing superfluous occurrences of "id" and repetitions of conditionals. Clearly, $f' \in \mathcal{T}^{\mathrm{norm}}_{M_e}(e)$, and $(v, w) \in f'(G)$.

Proposition 8 becomes interesting in conjunction with Proposition 9, below.

Proposition 9 *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 $\mathcal{N}^{\mathrm{norm}}_n(\Gamma)$ can be bounded by a number depending only on n and p; and*
2. *the number of expressions in $\mathcal{N}^{\mathrm{norm}}_n(\Gamma)$ is finite, and can be bounded by a number depending only on n and p.*

Proof First, consider item (1). Let g be an expression of $\mathcal{N}^{\mathrm{norm}}_m(\Gamma)$. We know that g contains "R" at most n times. Unless g is "id", we know that before the first "R", in between subsequent "R"s, and after the last "R", we can have a sequence of conditionals, in which each of these occurs at most once. Hence, the number of atomic subexpression in g is at most $\max(1, n+(n+1)p)$. Item (2) now immediately follows.

11 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 $G^v_0, G^v_1, G^v_2, \ldots$ of order $0, 1, 2, \ldots$. (In the notation, we shall leave Γ and n implicit.)

In doing so, we have two opposite concerns:

1. For some $n \geq 0$ and some order $i \geq 0$, G^v_i, the n-canonical subgraph of order i of G, will be the subgraph G_v of G mentioned at the end of Section 7 satisfying $e(G_v) \neq \emptyset$ if and only if $e(G) \neq \emptyset$. In order to work towards that goal, we must define $G^v_0, G^v_1, G^v_2, \ldots$ sufficiently large to ensure that we can simulate the behavior of e on G on these subgraphs of G.
2. For our proof strategy to work, it is at same important that there is a bound on the number of nodes of G_v that only depends on e, and not on G or v. Therefore, we may define $G^v_0, G^v_1, G^v_2, \ldots$ not too large either. In particular, we shall ensure that the number of nodes of each of these n-canonical subgraphs depends only on its order and on p and n.

Balancing these two concerns is the motivatation behind the definitions that will follow.

We start by defining G_0^v.

Thereto, let g be an expression in $\mathcal{N}_n^{\text{norm}}(\Gamma)$. 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 $\mathfrak{P}(g)$.

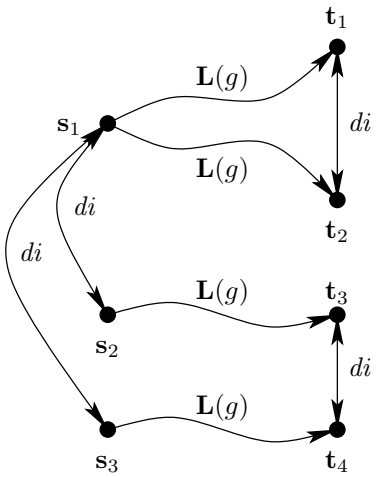

Fig. 2 Example of a graph pattern in $\mathfrak{P}(g)$.

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:

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 minimal 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 n-canonical subgraph of order 0:

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

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 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 13.

At this point, several aspects of the definition of the 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 $\mathbf{P} \in \mathfrak{P}(g)$.

The only answer we can give at this point is that these definitions are tailored to make some of the key results in Section 12 work (in particular, Lemma **??**), 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. As argued before, we must ensure on the one hand 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 11, below). Obtaining this delicate balance is what led to the definition above.

However, the results in Section 12 are only a 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 $i > 0$,

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

We note the following properties of n-canonical subgraphs.

Proposition 10 *Let $\Gamma = \{c_1, \ldots, c_p\}$ be a set of conditionals, let $n \geq 0$, and let G be a directed graph. For every node v of G, and for $i = 0, 1, 2, \ldots$, we have that (1) G_i^v is a subgraph of G, and (2) G_i^v is a subgraph of G_{i+1}^v.*

Proof By construction, G_0^v is a subgraph of G for every node v of G. This is the basis for a straightforward induction argument to show that, for $i = 1, 2, \ldots$, G_i^v is also a subgraph of G. G_i^x. This settles the first statement.

The second statement can also be shown by induction. The base case, that G_0^v is a subgraph of G_1^v, follows immediately from the definition of G_1^v. As induction hypothesis, assume that, for some $i > 0$, we have already established, for all nodes v of G, that G_{i-1}^v is a subgraph of G_i^v. As induction step, we now show that G_i^v is a subgraph of G_{i+1}^v. We have that

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

By the induction hypothesis, we know that, for each node w of G_0^v, G_{i-1}^w is a subgraph of G_i^w. Hence, G_i^v is a subgraph of

$$G_0^v \cup \left(\bigcup_{w \text{ node of } G_0^v} G_i^w \right),$$

which by definition is G_{i+1}^v.

The n-canonical subgraphs of G of higher order are put to use in Section 13, more in particular in Proposition 14.

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 11 *Let $\Gamma = \{c_1, \ldots, c_p\}$ be a set of conditionals, let $n \geq 0$, and let G be a directed graph. For every node v of G, and for $i = 0, 1, 2, \ldots$, the number of nodes in G_i^v can be bounded by a number depending only on p, n, and i.*

Proof Let us first focus on G_0^v. From Proposition 9, it can easily be inferred that both the number of graph patterns involved in the construction of G_0^v as the number of nodes they contain are bounded by numbers depending only on p and n. Let us call these numbers P and N, respectively. Given a graph pattern $\mathbf{P}$, the number of homomorphisms from $\mathbf{P}$ to G in $\mathfrak{H}_v(\mathbf{P})$ is bounded by 2^N. To see this, select a set of homomorphisms from $\mathbf{P}$ to G, as follows. Consider an arbitrary node $\mathbf{v}$ of $\mathbf{P}$.

1. If all homomorphisms from $\mathbf{P}$ to G map $\mathbf{v}$ to the same node of G, then select one such homomorphism arbitrarily. Regardless of whether or not $\mathbf{v}$ may be the unique source or target node of $\mathbf{P}$, we see that conditions 2–4 of the definition of $\mathfrak{H}_v(\mathbf{P})$ are satisfied for that particular node.
2. Otherwise, not all homomorphisms from $\mathbf{P}$ to G map $\mathbf{v}$ to the same node of G. If, in addition, no homomorphism from $\mathbf{P}$ to G maps $\mathbf{v}$ to v, then select two such homomorphisms arbitrarily provided they map $\mathbf{v}$ to different nodes of G. Regardless of whether or not $\mathbf{v}$ may be the unique source or target node of $\mathbf{P}$, we see that conditions 2–4 of the definition of $\mathfrak{H}_v(\mathbf{P})$ are satisfied for that particular node.
3. Otherwise, there is a homomorphism from $\mathbf{P}$ to G mapping $\mathbf{v}$ to v and there is a homomorphism from $\mathbf{P}$ to G not mapping $\mathbf{v}$ to v. Then select arbitrarily one homomorphism from the first category and one homomorphism from the second category. Regardless of whether or not $\mathbf{v}$ may be the unique source or target node of $\mathbf{P}$, we see that conditions 2–4 of the definition of $\mathfrak{H}_x(\mathbf{P})$ are satisfied for that particular node.

By construction, the set of homomorphisms from $\mathbf{P}$ to G selected above contains at most 2^N members. It clearly satisfied condition 1 as well as conditions 2–4 for all nodes of $\mathbf{P}$. Since $\mathfrak{H}_x(\mathbf{P})$ is such as set of minimal size, we may finally conclude that $\mathfrak{H}_x(\mathbf{P})$ contains at most 2^N homomorphisms.

Consequently, the number of nodes of G_0^v is bounded by $2^N NP$, which again depends only on p and n. Let us denote this last number as B. Then, a straightforward induction reveals that, for all $i \geq 0$, the number of nodes of G_i^x is bounded by $B(B^i + B^{i-1} + \cdots + B + 1)$.

12 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. 8 and the concluding paragraphs of Section **??**) rely, are the following.

Proposition 12 *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 13 *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 12 and 13 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 7, $(v, w) \in e(G_0^v)$, 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 **??** 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 13.

Because of the strong analogy between both Propositions, we shall focus here on the proof of Proposition 12. It can be easily seen that Proposition 12 follows from Propositions **??** and 7, provided we can prove the following lemma.

Lemma 7 *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 7 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 8 *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 7. In the induction step, we may be in Case 1 or Case 2 of Lemma 7, 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 7 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 7, 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.

13 The collapse

We are now ready to deal with expressions in $\mathcal{N}(\pi, di, ^+)$ and bootstrap Propositions 12 and 13 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 *π-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 $\text{depth}_\pi(e) = 0$;
– $\text{depth}_\pi(\pi_1(e)) = \text{depth}_\pi(\pi_2(e)) = \text{depth}_\pi(e) + 1$;
– $\text{depth}_\pi(e_1 \cup e_2) = \max(\text{depth}_\pi(e_1), \text{depth}_\pi(e_2))$;
– $\text{depth}_\pi(e_1 \circ e_2) = \max(\text{depth}_\pi(e_1), \text{depth}_\pi(e_2))$; and
– $\text{depth}_\pi(e^+) = \text{depth}_\pi(e)$.

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 $\Pi(e)$.

We can now show the following.

Proposition 14 *Let m be a nonzero natural number, and let e be an expression in $\mathcal{N}(\pi, di, ^+)$. Let $\ell := \text{depth}_\pi(e)$. 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_ℓ such that $(v, w') \in e(G^v_\ell)$, with G^v_ℓ the n-canonical subgraph of G of order ℓ for the set of conditionals $\Gamma := \Pi(e)$.*

Proposition 14 is shown by proving that an extended version of it holds for every subexpression of e, by induction on its π-nesting depth. Propositions 12 and 13 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 12 already yields that, for some n-normal trace expression f in $\mathcal{T}^{\text{norm}}_n(e)$, 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_ℓ, the n-canonical subgraph of G of order ℓ, 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_ℓ, 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_ℓ, and we can then invoke Propositions **??** and 7 to obtain the conclusion of Proposition 14.

Now, from Proposition 11, it immediately follows that we can bound the number of nodes in G^v_ℓ 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 **??**, and we may thus conclude the following.

Theorem 2 *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_ℓ in Proposition 14, is of very high complexity in m, it may require very large graphs G before the difference between G and its subgraphs G^v_ℓ becomes significant.[6]

Combining Theorems 1 and 2, 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

[6] For the same reason, it was not possible to "discover" Proposition 14 and the ensuing Theorem 2 by looking at simple examples.

in Theorem 2 is more specifically in $\mathcal{N}(F)$. From our proof, we may therefore also conclude the following.

Corollary 1 *Let $F \subseteq \{\pi, di\}$. Then $\mathcal{N}(F, ^+)$ collapses to $\mathcal{N}(F)$ at the level of boolean queries.*

14 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 [3], 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 [21] is similar to the standard relational division operation in databases, and corresponds to the set containment join [17].

References

1. Abiteboul, S., Buneman, P., Suciu, D.: Data on the Web: From Relations to Semistructured Data and XML. Morgan Kaufmann (1999)
2. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison Wesley, Reading, MA (1995)
3. 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, January 1979., pp. 110–120 (1979)
4. Angles, R., Gutiérrez, C.: Survey of graph database models. ACM Comput. Surv. **40**(1), 1–39 (2008)
5. Baader, F., Calvanese, D., McGuiness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook. Cambridge University Press (2003)
6. Benedikt, M., Fan, W., Kuper, G.M.: Structural properties of XPath fragments. In: ICDT, pp. 79–95 (2003)
7. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. Int. J. Semantic Web Inf. Syst. **5**(3), 1–22 (2009)
8. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. Cambridge University Press (2001)
9. Fletcher, G.H.L., Gyssens, M., Leinders, D., den Bussche, J.V., Gucht, D.V., Vansummeren, S., Wu, Y.: The impact of transitive closure on the boolean expressiveness of navigational query languages on graphs. In: T. Lukasiewicz, A. Sali (eds.) FoIKS, *Lecture Notes in Computer Science*, vol. 7153, pp. 124–143. Springer (2012)
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: T. Milo (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: S. Vansummeren (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, *Synthesis Lectures on the Semantic Web: Theory and Technology*, vol. 1, 1st edn. Morgan & Claypool Publishers (2011)
16. Maddux, R.D.: Relation Algebras. Elsevier, Amsterdam (2006)
17. Mamoulis, N.: Efficient processing of joins on set-valued attributes. In: Proceedings ACM SIGMOD International Conference on Management of Data, pp. 157–168 (2003)
18. Marx, M.: Conditional XPath. ACM Trans. Database Syst. **30**(4), 929–959 (2005)
19. Marx, M., de Rijke, M.: Semantic characterizations of navigational XPath. SIGMOD Record **34**(2), 41–46 (2005)
20. Marx, M., Venema, Y.: Multi-Dimensional Modal Logic. Springer (1997)
21. 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)
22. RDF primer (2004). `http://www.w3.org/TR/rdf-primer/`
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)