Relative Expressive Power of Navigational Querying on Graphs

George H. L. Fletcher
Eindhoven University of Technology
g.h.l.fletcher@tue.nl

Marc Gyssens
Hasselt University &
Transnational Univ. of Limburg
marc.gyssens@uhasselt.be

Dirk Leinders[*]
Hasselt University &
Transnational Univ. of Limburg
dirk.leinders@uhasselt.be

Jan Van den Bussche
Hasselt University &
Transnational Univ. of Limburg
jan.vandenbussche@uhasselt.be

Dirk Van Gucht
Indiana University
vgucht@cs.indiana.edu

Stijn Vansummeren
Université Libre de Bruxelles
stijn.vansummeren@ulb.ac.be

Yuqing Wu
Indiana University
yuqwu@cs.indiana.edu

ABSTRACT

Motivated by both established and new applications, we study navigational query languages for graphs (binary relations). The simplest language has only the two operators union and composition, together with the identity relation. We make more powerful languages by adding any of the following operators: intersection; set difference; projection; coprojection; converse; transitive closure; and the diversity relation. All these operators map binary relations to binary relations. We compare the expressive power of all resulting languages. We do this not only for general path queries (queries where the result may be any binary relation) but also for boolean or yes/no queries (expressed by the nonemptiness of an expression). For both cases, we present the complete Hasse diagram of relative expressiveness. In particular, the Hasse diagram for boolean queries contains nontrivial separations and a few surprising collapses.

Categories and Subject Descriptors

F.4.1 [**Mathematical Logic and Formal Languages**]: Model theory; H.2.3 [**Database Management**]: Query languages

General Terms

Languages, Theory

[*]Postdoctoral Fellow of the Research Foundation - Flanders (FWO)

Keywords

XPath, graphs, relation algebra, indistinguishability, expressive power, finite variable logic

1. INTRODUCTION

The expressive power of XPath as a query language for navigating on trees is well understood [6, 19, 20]. Motivated by data on the Web [2, 12] and new applications such as dataspaces [13], Linked Data [7], and RDF [1], it is natural to look at similar navigational query languages for graphs. Graph query languages have a rich history in database theory. Indeed, motivated by object-oriented and semistructured database systems, graph query languages have been investigated since the mid 80's. This is nicely reviewed in the survey of graph database models by Angles and Gutiérrez [4].

In the present paper, we consider a number of natural operators on binary relations (graphs): union; composition; intersection; set difference; projection; coprojection; converse; transitive closure; and the identity and diversity relations[1]. While some of these operators also appear in XPath, they are there evaluated on trees. The largest language that we consider has all operators, while the smallest language has only union, composition, and the identity relation. Just as in the relational algebra, expressions are built up from input relation names using these operators. Since each operator maps binary relations to binary relations, these query languages express queries from binary relations to binary relations: we call such queries *path queries*. 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 contribution of the present paper is providing a complete comparison of the expressiveness of all resulting lan-

[1]We assume that identity and diversity return all identical, respectively, nonidentical pairs of values from the active domain.

guages, and this both for general path queries and boolean queries. While establishing the relative expressiveness for general path queries did not yield particularly surprising results, the task for the case of boolean queries proved much more challenging. For example, consider the converse operator $R^{-1} = \{(y, x) \mid (x, y) \in R\}$. On the one hand, adding converse to a language not yet containing this feature sometimes adds boolean query power. This is, e.g., the case for the language containing all other features. The proof, however, is nontrivial and involves a specialized application of invariance under bisimulation known from arrow logics. On the other hand, adding converse to a language containing projection but not containing intersection and transitive closure does not add any boolean query power. We thus obtain a result mirroring similar results known for XPath on trees [6, 22, 27], where, e.g., downward XPath is known to be as powerful as full XPath for queries evaluated at the root. Marx also obtained a similar result for XPath with transitive closure [19], but we show here that this no longer holds on graphs.

We conclude this introduction by a short discussion of some of the methods we use. In many cases where we separate a language $\mathcal{L}_1$ from a language $\mathcal{L}_2$, we can do this in a strong sense: we are able to give a single counterexample, consisting of a pair (A, B) of finite binary relations such that A and B are distinguishable by an expression from $\mathcal{L}_1$ but indistinguishable by any expression from $\mathcal{L}_2$. Notice that in general, separation is established by providing an infinite sequence of relation pairs such that some expression from $\mathcal{L}_1$ distinguishes all pairs but no single expression of $\mathcal{L}_2$ distinguishes all pairs. Existence of a single counterexample pair is therefore nonobvious, and we do not really know whether there is a deeper reason why in our setting this strong form of separation can often be established. Strong separation is desirable as indistinguishability of a pair of finite binary relations can in principle be checked by computer, as the number of possible binary relations on a finite domain is finite. In some cases, however, a brute-force approach is not feasible within a reasonable time. Fortunately, by applying invariance under bisimulation for arrow logics [21], we can alternatively check a sufficient condition for indistinguishability in polynomial time. We have applied this alternative approach in our computer checks. Finally, we developed two methods to establish ordinary separation for the cases where we could not establish strong separation. The first method, applicable to languages that fall in the class of conjunctive queries, is based on homomorphism techniques. The second method, applicable to languages with transitive closure, is based on proofs in the classical finite model theory style, involving winning strategies in bisimulation pebble games for longer and longer rounds on pairs of larger and larger finite structures.

Our work is certainly not an endpoint in this line of research, as many questions remain to be answered. It is interesting to investigate, for example, the decidability of satisfiability or containment of expressions. Indeed, the decidability of these problems is very important for query optimization. The language with all operators except transitive closure is equivalent in expressive power to FO^3 (for path as well as for boolean queries [26]), where these problems are known to be undecidable; we come back to these issues in Section 8. Other interesting questions that can be guided by our comparison are, e.g., the complexity of query evaluation; and the complexity of the containment problem.

Related Work. The languages considered here are very natural and similar to languages already considered in the fields of description logics, dynamic logics, arrow logics, and relation algebras [5,8,16,17,21]. Thus, our results also yield some new insight into those logics. The investigation of expressive power as in the present paper is very natural from a database theory perspective. In the above-mentioned fields, however, one is primarily interested in other questions, such as computational complexity of model checking, decidability of satisfiability, and axiomatizability of equivalence. We must point out that, also in the database field, graph query languages have been investigated intensively [4]. There is, for example, the work on conjunctive regular path queries (CRPQs) [9, 10]. The focus there, however, is not so much on expressiveness either but on the complexity of the containment problem.

Organization. The paper is organized as follows. In Section 2, we define the class of languages studied in the paper. In Section 3, we describe the techniques we use to separate one language from another. Then we establish the complete Hasse diagram of relative expressiveness. We do so for path queries in Section 4, and for boolean queries in Section 5. In Section 6, we show there are a few additional collapses in the Hasse diagram when one is interested in unlabeled graphs only. Finally, we conclude and discuss future research directions in Section 7 and Section 8.

In this extended abstract, the proofs are either omitted or only summarily sketched.

2. PRELIMINARIES

In this paper, we are interested in navigating over graphs whose edges are labeled by symbols from a finite, nonempty set of labels Λ. For our purposes, a graph is a relational structure G, consisting of a set of nodes V and, for every $R \in \Lambda$, a relation $G(R) \subseteq V \times V$, the set of edges with label R. In what follows, both V and $G(R)$ may be infinite, unless explicitly stated otherwise.[2]

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

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

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

$$\begin{aligned} R(G) &= G(R)\,; \\ \emptyset(G) &= \emptyset\,; \\ id(G) &= \{(m, m) \mid m \in \mathrm{adom}(G)\}\,; \\ e_1 \circ e_2(G) &= \{(m, n) \mid \exists p\, ((m, p) \in e_1(G) \ \&\ (p, n) \in e_2(G))\}\,; \\ e_1 \cup e_2(G) &= e_1(G) \cup e_2(G)\,. \end{aligned}$$

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

[2]Notwithstanding, all inexpressibility results in this paper already hold over finite graphs.

$\pi_2(e)$), 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:

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

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 and both projections.[3]

Note that the language $\mathcal{N}(\setminus, di, ^{-1})$ is better known as the *relation algebra.* It is known that for every set of nonbasic features F not containing transitive closure, all path queries expressible in $\mathcal{N}(F)$ are also expressible in the relation algebra [17].

We will actually compare language expressiveness at the level of both path queries and 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 what follows, we write $\mathcal{N}(F_1) \leq^{\mathrm{path}} \mathcal{N}(F_2)$ if every path query expressible in $\mathcal{N}(F_1)$ is also expressible in $\mathcal{N}(F_2)$. Similarly, we write $\mathcal{N}(F_1) \leq^{\mathrm{bool}} \mathcal{N}(F_2)$ if every boolean query expressible in $\mathcal{N}(F_1)$ is also expressible in $\mathcal{N}(F_2)$. Note that $\mathcal{N}(F_1) \leq^{\mathrm{path}} \mathcal{N}(F_2)$ implies $\mathcal{N}(F_1) \leq^{\mathrm{bool}} \mathcal{N}(F_2)$, but not necessarily the other way around. We write $\not\leq^{\mathrm{path}}$ and $\not\leq^{\mathrm{bool}}$ for the negation of $\leq^{\mathrm{path}}$ and $\leq^{\mathrm{bool}}$.

For the technical development, it will be useful to consider the following variants of $\not\leq^{\mathrm{path}}$ and $\not\leq^{\mathrm{bool}}$.

Definition 2. *The language $\mathcal{N}(F_1)$ is* strongly separable from *the language $\mathcal{N}(F_2)$* at the level of path queries *if there exists a path query q expressible in $\mathcal{N}(F_1)$ and a finite graph G, such that, for every expression $e \in \mathcal{N}(F_2)$, we have $q(G) \neq e(G)$. We write $\mathcal{N}(F_1) \not\leq^{\mathrm{path}}_{\mathrm{strong}} \mathcal{N}(F_2)$ in this case. Similarly, $\mathcal{N}(F_1)$ is* strongly separable from *$\mathcal{N}(F_2)$* at the level of boolean queries *if there exists a boolean query q expressible in $\mathcal{N}(F_1)$ and two finite graphs G_1 and G_2, with $q(G_1)$ true and $q(G_2)$ false, such that, for every expression $e \in \mathcal{N}(F_2)$, $e(G_1)$ and $e(G_2)$ are both empty, or both nonempty. We write $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}}_{\mathrm{strong}} \mathcal{N}(F_2)$ in this case.*

[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.

3. TOOLS TO ESTABLISH SEPARATION

Our results in Sections 4 and 5 will use the following tools to separate a language $\mathcal{N}(F_1)$ from a language $\mathcal{N}(F_2)$, i.e., to obtain that $\mathcal{N}(F_1) \not\leq^{\mathrm{path}} \mathcal{N}(F_2)$, or $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}} \mathcal{N}(F_2)$.

3.1 Path separation

Since $\mathcal{N}(F_1) \leq^{\mathrm{path}} \mathcal{N}(F_2)$ implies $\mathcal{N}(F_1) \leq^{\mathrm{bool}} \mathcal{N}(F_2)$, also $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}} \mathcal{N}(F_2)$ implies $\mathcal{N}(F_1) \not\leq^{\mathrm{path}} \mathcal{N}(F_2)$ by contraposition. In most instances, we can therefore establish separation at the level of general path queries by establishing separation at the level of boolean queries. In the cases where $\mathcal{N}(F_1) \not\leq^{\mathrm{path}} \mathcal{N}(F_2)$ although $\mathcal{N}(F_1) \leq^{\mathrm{bool}} \mathcal{N}(F_2)$, we either rely on the well-known result of the inexpressibility of transitive closure in FO (see, e.g., [3]) or identify a finite graph G and an expression e_1 in $\mathcal{N}(F_1)$ and show that, for each expression e_2 in $\mathcal{N}(F_2)$, $e_1(G) \neq e_2(G)$. Notice that, in the latter case, we actually establish strong path separation.

3.2 Boolean separation

To establish separation at the level of boolean queries, we use the following techniques.

3.2.1 Brute-force approach

Two graphs G_1 and G_2 are said to be distinguishable at the boolean level in a language $\mathcal{N}(F)$ if there exists a boolean query q expressible in $\mathcal{N}(F)$ such that exactly one of $q(G_1)$ and $q(G_2)$ is true, and the other is false. If such a query does not exists, G_1 and G_2 are said to be indistinguishable in $\mathcal{N}(F)$.

Using this terminology, two languages $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ are strongly separable if there exist two finite graphs G_1 and G_2 that are distinguishable in $\mathcal{N}(F_1)$, but indistinguishable in $\mathcal{N}(F_2)$.

For two finite graphs G_1 and G_2, (in)distinguishability in a language $\mathcal{N}(F)$ can easily be machine-checked through the Brute-Force Algorithm described below.

First observe that $\mathrm{adom}(G_1)$ and $\mathrm{adom}(G_2)$ are finite since G_1 and G_2 are finite. Moreover, for any e in $\mathcal{N}(F)$, $e(G_1) \subseteq \mathrm{adom}(G_1) \times \mathrm{adom}(G_1)$ and $e(G_2) \subseteq \mathrm{adom}(G_2) \times \mathrm{adom}(G_2)$. Hence, $e(G_1)$ and $e(G_2)$ are finite and the set $\{(e(G_1), e(G_2)) \mid e \in \mathcal{N}(F)\}$ is also finite. Clearly, G_1 is indistinguishable from G_2 if this set contains only pairs that are both empty or both nonempty.

The Brute-Force Algorithm computes the above set by first initializing the set

$$\begin{aligned}B = \{(id(G_1), id(G_2))\} &\cup \{(di(G_1), di(G_2))\}\\ &\cup \{(G_1(R), G_2(R)) \mid R \in \Lambda\}\end{aligned}$$

(where $\{(di(G_1), di(G_2))\}$ is omitted if $di \notin F$). It then adds new pairs (R_1, R_2) to B by closing B pair-wise under the features in $\mathcal{N}(F)$. That is, for every binary operator $\otimes$ in $\mathcal{N}(F)$ and all pairs $(R_1, R_2), (S_1, S_2)$ in B the algorithm adds $(R_1 \otimes S_1, R_2 \otimes S_2)$ to B, and similarly for the unary operators. Since there are only a finite number of pairs, the algorithm is guaranteed to end. Of course, the worst-case complexity of this brute-force algorithm is exponential. Nevertheless, we have successfully checked indistinguishability using this Brute-Force Algorithm in many of the cases that follow.

3.2.2 Bisimulation

We will not always be able to use the methodology above to separate two languages. In particular, to establish that $\mathcal{N}(^{-1}, \cap) \not\leq^{\text{bool}} \mathcal{N}(\setminus, di)$ and $\mathcal{N}(^{-1}, ^{+}) \not\leq^{\text{bool}} \mathcal{N}(\setminus, di, ^{+})$ we will employ invariance results under the notion of bisimulation below. In essence, this notion is based on the notion of bisimulation known from arrow logics [21]. Below, we adapt this notion to the current setting.

We require the following preliminary definitions. Let $\mathbf{G} = (G, a, b)$ denote a *marked graph*, i.e., a graph G with $a, b \in \text{adom}(G)$. The *degree* of an expression e is the maximum depth of nested applications of composition, projection and coprojection in e. For example, the degree of $R \circ R$ is 1, while the degree of both $R \circ (R \circ R)$ and $\pi_1(R \circ R)$ is 2. Intuitively, the depth of e corresponds to the quantifier rank of the standard translation of e into FO^3. For a set of features F, $\mathcal{N}(F)_k$ denotes the set of expressions in $\mathcal{N}(F)$ of degree at most k.

In what follows, we are only concerned with bisimulation results regarding $\mathcal{N}(\setminus, di)$. The following is an appropriate notion of bisimulation for this language.

Definition 3 (Bisimilarity). *Let k be a natural number, and let $\mathbf{G}_1 = (G_1, a_1, b_1)$ and $\mathbf{G}_2 = (G_2, a_2, b_2)$ be marked graphs. We say that $\mathbf{G}_1$ is bisimilar to $\mathbf{G}_2$ up to depth k, denoted $\mathbf{G}_1 \simeq_k \mathbf{G}_2$, if the following conditions are satisfied:*

Atoms *$a_1 = b_1$ if and only if $a_2 = b_2$; and $(a_1, b_1) \in G_1(R)$ if and only if $(a_2, b_2) \in G_2(R)$, for every $R \in \Lambda$;*

Forth *if $k > 0$, then, for every c_1 in $\text{adom}(G_1)$, there exists some c_2 in $\text{adom}(G_2)$ such that both $(G_1, a_1, c_1) \simeq_{k-1} (G_2, a_2, c_2)$ and $(G_1, c_1, b_1) \simeq_{k-1} (G_2, c_2, b_2)$;*

Back *if $k > 0$, then, for every c_2 in $\text{adom}(G_2)$, there exists some c_1 in $\text{adom}(G_1)$ such that both $(G_1, a_1, c_1) \simeq_{k-1} (G_2, a_2, c_2)$ and $(G_1, c_1, b_1) \simeq_{k-1} (G_2, c_2, b_2)$.*

Expressions in $\mathcal{N}(\setminus, di)$ of depth at most k are invariant under bisimulation:

Proposition 4. *Let k be a natural number; let e be an expression in $\mathcal{N}(\setminus, di)_k$; and let $\mathbf{G}_1 = (G_1, a_1, b_1)$ and $\mathbf{G}_2 = (G_2, a_2, b_2)$ be marked graphs. If $\mathbf{G}_1 \simeq_k \mathbf{G}_2$ then $(a_1, b_1) \in e(G_1) \Leftrightarrow (a_2, b_2) \in e(G_2)$.*

In other words, if $\mathbf{G}_1 \simeq_k \mathbf{G}_2$, then any expression of degree at most k either both selects (a_1, b_1) in G_1 and (a_2, b_2) in G_2, or neither of them. As such, the marked graphs $\mathbf{G}_1$ and $\mathbf{G}_2$ are *indistinguishable* by expressions in $\mathcal{N}(\setminus, di)_k$. The proof of Proposition 4 is by a straightforward induction on e.

The following proposition states how we can use Proposition 4 to show that some boolean query is not expressible in $\mathcal{N}(\setminus, di)_k$.

Proposition 5. *Let k be a natural number. A boolean query q is not expressible in $\mathcal{N}(\setminus, di)_k$ if there exist graphs G_1 and G_2 such that $q(G_1)$ is true and $q(G_2)$ is false, and, for each pair $(a_1, b_1) \in \text{adom}(G_1)^2$, there exists $(a_2, b_2) \in \text{adom}(G_2)^2$ such that $(G_1, a_1, b_1) \simeq_k (G_2, a_2, b_2)$.*

As Proposition 5 is standard in finite model theory, we omit the proof.

3.2.3 Homomorphism approach

To show that $\mathcal{N}(\pi) \not\leq^{\text{bool}} \mathcal{N}(^{-1}, di, ^{+})$, we used an entirely different technique, which is based on the following observation: let $Q = H \leftarrow B$ be a conjunctive query and let G be a graph; if $Q(G)$ is nonempty, then there exists a homomorphism from B to G. Notice that queries in $\mathcal{N}(\pi)$ are conjunctive. Let $Q_1 = H_1 \leftarrow B_1$ be a particular query in $\mathcal{N}(\pi)$. If we can somehow reduce the equivalence of Q_1 to a query in $\mathcal{N}(^{-1}, di, ^{+})$ at the boolean level to the equivalence of conjunctive queries at the boolean level, we may be able to use the above property to establish the existence of a nontrivial endomorphism of B_1. If no such endomorphism exists, we have obtained the required contradiction.

4. PATH QUERIES

In this section, we characterize the order $\leq^{\text{path}}$ of relative expressiveness for path queries by Theorem 7 below.

Towards the statement of this characterization, first notice the following interdependencies between features:

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

For a set of nonbasic features F, let $\overline{F}$ 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, $\overline{\{\setminus, ^{-1}\}} = \{\setminus, ^{-1}, \cap, \pi, \overline{\pi}\}$.

Notice that, if $F_1 \subseteq \overline{F}_2$, we can always rewrite an expression $e \in \mathcal{N}(F_1)$ into an equivalent expression in $\mathcal{N}(F_2)$ using the equalities above. Therefore, we obtain

Proposition 6. *If $F_1 \subseteq \overline{F}_2$, then $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2)$.*

We will actually show that the converse also holds, whence

Theorem 7. *$\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2)$ if and only if $F_1 \subseteq \overline{F}_2$.*

The "only if" direction of Theorem 7, however, requires a detailed analysis. For the clarity of presentation, we divide the languages under consideration into the following four classes:

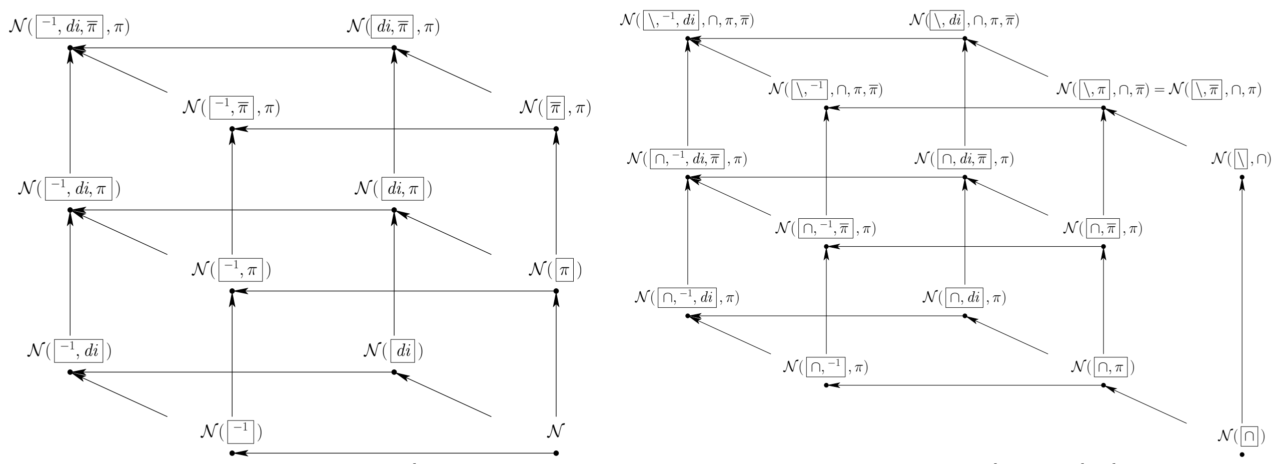
$$\begin{aligned}
\mathcal{C} &= \{\mathcal{N}(F) \mid \cap \notin \overline{F},\ ^{+} \notin \overline{F}\},\\
\mathcal{C}[\cap] &= \{\mathcal{N}(F) \mid \cap \in \overline{F},\ ^{+} \notin \overline{F}\},\\
\mathcal{C}[^{+}] &= \{\mathcal{N}(F) \mid \cap \notin \overline{F},\ ^{+} \in \overline{F}\},\\
\mathcal{C}[\cap, ^{+}] &= \{\mathcal{N}(F) \mid \cap \in \overline{F},\ ^{+} \in \overline{F}\}.
\end{aligned}$$

We first establish the "only if" direction for the cases where $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ belong to the same class. We do so for each class separately in Sections 4.1–4.4. Finally, in Section 4.5, we consider the case where $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ belong to distinct classes.

4.1 Languages without $\cap$ and without $^{+}$

In this subsection, we show the "only if" direction of Theorem 7, restricted to $\mathcal{C}$, the class of languages without $\cap$ and without $^{+}$.

Proposition 8. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be in $\mathcal{C}$. If $F_1 \not\subseteq \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$.*



$\mathcal{N}(\boxed{{}^{-1}, di, \overline{\pi}}, \pi)$
$\mathcal{N}(\boxed{di, \overline{\pi}}, \pi)$
$\mathcal{N}(\boxed{{}^{-1}, \overline{\pi}}, \pi)$
$\mathcal{N}(\boxed{\overline{\pi}}, \pi)$
$\mathcal{N}(\boxed{{}^{-1}, di, \pi})$
$\mathcal{N}(\boxed{di, \pi})$
$\mathcal{N}(\boxed{{}^{-1}, \pi})$
$\mathcal{N}(\boxed{\pi})$
$\mathcal{N}(\boxed{{}^{-1}, di})$
$\mathcal{N}(\boxed{di})$
$\mathcal{N}(\boxed{{}^{-1}})$
$\mathcal{N}$
$\mathcal{N}(\boxed{\setminus, {}^{-1}, di}, \cap, \pi, \overline{\pi})$
$\mathcal{N}(\boxed{\setminus, di}, \cap, \pi, \overline{\pi})$
$\mathcal{N}(\boxed{\setminus, {}^{-1}}, \cap, \pi, \overline{\pi})$
$\mathcal{N}(\boxed{\setminus, \pi}, \cap, \overline{\pi}) = \mathcal{N}(\boxed{\setminus, \overline{\pi}}, \cap, \pi)$
$\mathcal{N}(\boxed{\cap, {}^{-1}, di, \overline{\pi}}, \pi)$
$\mathcal{N}(\boxed{\cap, di, \overline{\pi}}, \pi)$
$\mathcal{N}(\boxed{\setminus}, \cap)$
$\mathcal{N}(\boxed{\cap, {}^{-1}, \overline{\pi}}, \pi)$
$\mathcal{N}(\boxed{\cap, \overline{\pi}}, \pi)$
$\mathcal{N}(\boxed{\cap, {}^{-1}, di}, \pi)$
$\mathcal{N}(\boxed{\cap, di}, \pi)$
$\mathcal{N}(\boxed{\cap, {}^{-1}}, \pi)$
$\mathcal{N}(\boxed{\cap, \pi})$
$\mathcal{N}(\boxed{\cap})$

(a) Hasse diagram of $\leq^{\text{path}}$ for $\mathcal{C}$.

(b) Hasse diagram of $\leq^{\text{path}}$ and $\leq^{\text{bool}}$ for $\mathcal{C}[\cap]$.

Figure 1: For each language, the boxed features are a minimal set of nonbasic features defining the language, while the other features can be derived from them in the sense of Theorem 7 (using the appropriate interdependencies).

Propositions 6 and 8 combined yield the Hasse diagram of $\leq^{\text{path}}$ for $\mathcal{C}$, shown in Figure 1(a). It is indeed readily verified that for any two languages $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ in $\mathcal{C}$, there is a path from $\mathcal{N}(F_1)$ to $\mathcal{N}(F_2)$ in Figure 1(a) if and only if $F_1 \subseteq \overline{F}_2$.

Towards a proof of Proposition 8, we first establish the following. For later use, we sometimes prove results that are stronger than strictly needed for this purpose.

PROPOSITION 9. *Let F_1 and F_2 be sets of nonbasic features.*

1. *If $di \in \overline{F}_1$ and $di \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2)$.*
2. *If $\overline{\pi} \in \overline{F}_1$, $\overline{\pi} \notin \overline{F}_2$, and $\setminus \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2)$.*
3. *If $\pi \in \overline{F}_1$ and $F_2 \subseteq \{{}^{-1}, di, {}^{+}\}$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}} \mathcal{N}(F_2)$.*
4. *If ${}^{-1} \in \overline{F}_1$ and ${}^{-1} \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{path}}_{\text{strong}} \mathcal{N}(F_2)$.*

PROOF. For (1), consider a graph consisting of two self-loops, and a graph consisting of a single self-loop, all with the same label. It can be verified by the Brute-Force Algorithm of Section 3.2.1 that these graphs are not distinguishable in $\mathcal{N}(F_2)$. The graphs, however, are distinguishable by the boolean query expressed by di.

For (2), it can be verified by the Brute-Force Algorithm that the graphs shown in Figure 2 (a) are not distinguishable in $\mathcal{N}(F_2)$. The graphs, however, are distinguishable by the boolean query expressed by $\overline{\pi}_2(R)$.

For (3), we use the homomorphism approach outlined in Section 3.2.3. The proof is omitted.

For (4), we establish strong path separation at the level of path queries as explained in Section 3.1. Thereto, we consider the graph G shown in Figure 4. It is easily verified that G^{-1} cannot be obtained from G using an expression in $\mathcal{N}(F_2)$. $\square$

Proposition 9 is now used to show that for every pair F_1 and F_2 of sets of nonbasic features for which $F_1 \not\subseteq \overline{F}_2$ (i.e.,

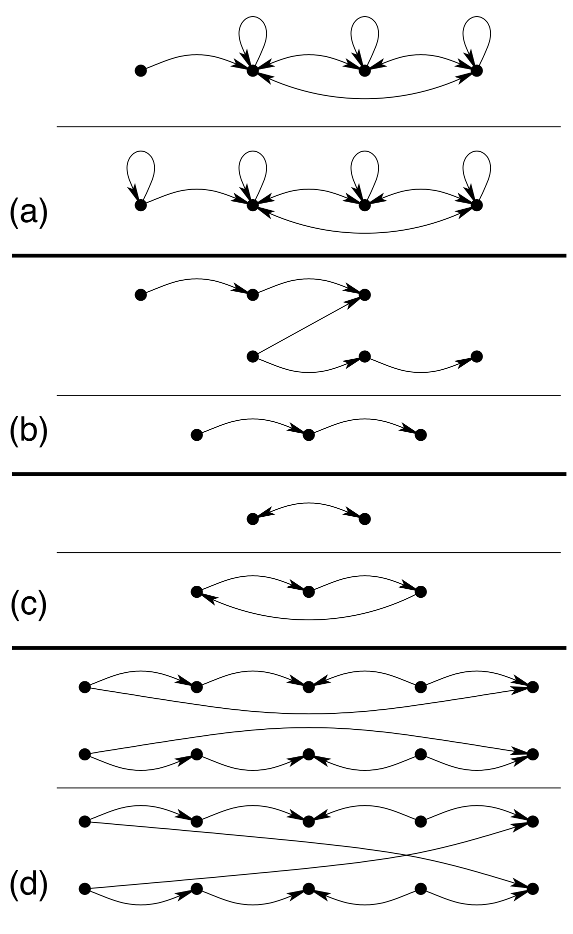


Figure 2: Graph pairs used to prove $\not\leq^{\text{bool}}_{\text{strong}}$ results in Sections 4 and 5. All edges are assumed to have the same label R.

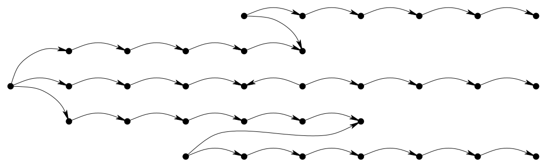


Figure 3: Query pattern used to prove Proposition 9 (3). All edges are assumed to have the same label R.

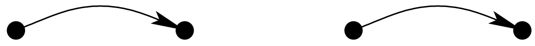


Figure 4: Graph used to prove Proposition 9 (4). Both edges are assumed to have the same label R.

for which there is no path in Figure 1(a)), that $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$. To illustrate the reasoning involved, consider the case where $F_1 = \{di, \pi\}$ and $F_2 = \{\pi\}$. By Proposition 9 (1) we obtain that $\mathcal{N}(di, \pi) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(\pi)$ and hence also that $\mathcal{N}(di, \pi) \not\leq^{\text{bool}} \mathcal{N}(\pi)$. Since $\leq^{\text{path}}$ implies $\leq^{\text{bool}}$, we hence also obtain $\mathcal{N}(di, \pi) \not\leq^{\text{path}} \mathcal{N}(\pi)$ by contraposition.

The remainder of the proof of Proposition 8 is a combinatorial analysis to verify that Proposition 9 covers all relevant cases. First, note that there are 90 ordered pairs of distinct languages $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ in $\mathcal{C}$ for which $F_1 \not\subseteq \overline{F}_1$. As illustrated above, we subsequently use the statements in Proposition 9 to establish that, for each of these pairs, $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$. Concretely, Proposition 9 (1) deals with 36 pairs, Proposition 9 (2) deals with another 24 pairs, Proposition 9 (3) deals with another 12 pairs, and, finally, Proposition 9 (4) deals with the remaining 18 pairs.

4.2 Languages with $\cap$ and without $^+$

In this subsection, we show the "only if" direction of Theorem 7, restricted to $\mathcal{C}[\cap]$, the class of languages with $\cap$ but without $^+$.

PROPOSITION 10. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be in $\mathcal{C}[\cap]$. If $F_1 \not\subseteq \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$.*

Propositions 6 and 10 combined yield the Hasse diagram of $\leq^{\text{path}}$ for $\mathcal{C}[\cap]$, shown in Figure 1(b).

Towards a proof of Proposition 10, we first establish the following.

PROPOSITION 11. *Let F_1 and F_2 be sets of nonbasic features.*

1. *If $\setminus \in \overline{F}_1$ and $\setminus \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2)$.*
2. *If $\pi \in \overline{F}_1$, and $F_2 \subseteq \{\setminus, \cap, ^+\}$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2)$.*

PROOF. For (1), consider a 3-clique, and a bow-tie consisting of two 3-cliques. It can be verified by the Brute-Force Algorithm of Section 3.2.1 that these graphs are not distinguishable in $\mathcal{N}(F_2)$. The graphs, however, are distinguishable by the boolean query expressed by $R^2 \setminus R$. For (2), it can be verified by the Brute-Force Algorithm that the graphs shown in Figure 2 (b) are not distinguishable in $\mathcal{N}(F_2)$. The graphs, however, are distinguishable by the boolean query expressed by $\pi_1(R^2) \circ R \circ \pi_2(R^2)$. □

Propositions 9 and 11 are now used to show that for every pair F_1 and F_2 of sets of nonbasic features for which $F_1 \not\subseteq \overline{F}_2$ (i.e., for which there is no path in Figure 1(b)), that $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$. To illustrate the reasoning involved, consider the case where $F_1 = \{\cap, \pi\}$ and $F_2 = \{\cap\}$. By Proposition 11 (2) we obtain $\mathcal{N}(\cap, \pi) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(\setminus, ^+)$ and hence, since $\mathcal{N}(\cap) \leq^{\text{path}} \mathcal{N}(\setminus, ^+)$, also $\mathcal{N}(\cap, \pi) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(\cap)$. Therefore, $\mathcal{N}(\cap, \pi) \not\leq^{\text{path}} \mathcal{N}(\cap)$.

The remainder of the proof of Proposition 10 is a combinatorial analysis to verify that Propositions 9 and 11 cover all relevant cases. First, note that there are 123 ordered pairs of distinct languages $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ in $\mathcal{C}[\cap]$ for which $F_1 \not\subseteq \overline{F}_2$. As illustrated above, we subsequently use the statements in Propositions 9 and 11 to establish that, for each of these pairs, $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$. Concretely, Proposition 9 (1) deals with 48 pairs, Proposition 9 (2) deals with another 28 pairs, Proposition 9 (4) deals with another 25 pairs, Proposition 11 (1) deals with another 18 pairs, and, finally, Proposition 11 (2) deals with the remaining 4 pairs.

4.3 Languages without $\cap$ and with $^+$

The characterization of $\leq^{\text{path}}$ for $\mathcal{C}[^+]$, the languages without $\cap$ but with $^+$, can easily be derived from the characterization of $\leq^{\text{path}}$ for $\mathcal{C}$, using the following observation.

PROPOSITION 12. *Let F_1 and F_2 be sets of nonbasic features for which $\cap \notin \overline{F}_1$, $\cap \notin \overline{F}_2$, $^+ \notin \overline{F}_1$, and $^+ \notin \overline{F}_2$. Then, $\mathcal{N}(F_1 \cup \{^+\}) \leq^{\text{path}} \mathcal{N}(F_2 \cup \{^+\})$ if and only if $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2)$.*

PROOF. The "if" is obvious. To see the "only if", assume that $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$. In Section 4.1, we showed that this can be established using one of the 4 statements of Proposition 9. Therefore, we can distinguish the following cases:

(1) Case $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2)$. By definition of $\not\leq^{\text{bool}}_{\text{strong}}$ there exists a boolean query q expressible in $\mathcal{N}(F_1)$, and two finite graphs G_1 and G_2 such that $q(G_1)$ is true, $q(G_2)$ is false, and, for any expression $e \in \mathcal{N}(F_2)$, $e(G_1)$ and $e(G_2)$ are both empty, or both nonempty. From this, we will now establish $\mathcal{N}(F_1 \cup \{^+\}) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2 \cup \{^+\})$, which implies $\mathcal{N}(F_1 \cup \{^+\}) \not\leq^{\text{path}} \mathcal{N}(F_2 \cup \{^+\})$. Towards this end, first observe that, since q is expressible in $\mathcal{N}(F_1)$, it is also expressible in $\mathcal{N}(F_1 \cup \{^+\})$. Now let e' be an expression in $\mathcal{N}(F_2 \cup \{^+\})$. We will show that again, $e'(G_1)$ and $e'(G_2)$ are both empty, or both nonempty. The crux here is that, on finite graphs, one can always compute the transitive closure by means of a finite composition. Indeed, let $n = \max(n_1, n_2)$ with n_1 the number of nodes in G_1 and n_2 the number of nodes in G_2. It is readily verified that, for any graph H with at most n nodes, and any expression $f \in \mathcal{N}(F_2)$, we have $f^+(H) = (\bigcup_{i=1}^{n} f^i)(H)$. By consistently replacing occurrences of the transitive closure operator in e' using this equality, we obtain an expression $e'' \in \mathcal{N}(F_2)$ such that $e''(G_1) = e'(G_1)$ and $e''(G_2) = e'(G_2)$. Since $e''(G_1)$ and $e''(G_2)$ are both empty or both nonempty, also $e'(G_1)$ and $e'(G_2)$ must be both empty, or both nonempty, as desired.

(2) Case $\mathcal{N}(F_1) \not\leq^{\text{path}}_{\text{strong}} \mathcal{N}(F_2)$. Similar to Case (1). (Details omitted.)

(3) Case $\pi \in \overline{F}_1$ and $F_2 \subseteq \{^{-1}, di\}$. Follows immediately from Proposition 9 (3). □

Corollary 13. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be in $\mathcal{C}[^+]$. If $F_1 \not\subseteq \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$.*

Proposition 6 and Corollary 13 combined yields that the Hasse diagram of $\leq^{\text{path}}$ for $\mathcal{C}[^+]$ can be obtained from the Hasse diagram of $\leq^{\text{path}}$ for $\mathcal{C}$ (Figure 1(a)), simply by adding $^+$ as a primitive to each language.

4.4 Languages with $\cap$ and with $^+$

As in Section 4.3, we have the following.

Proposition 14. *Let F_1 and F_2 be sets of nonbasic features for which $\cap \in \overline{F}_1$, $\cap \in \overline{F}_2$, $^+ \notin \overline{F}_1$, and $^+ \notin \overline{F}_2$. Then, $\mathcal{N}(F_1 \cup \{^+\}) \leq^{\text{path}} \mathcal{N}(F_2 \cup \{^+\})$ if and only if $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2)$.*

The proof is similar to the proof of Proposition 12. (Details omitted.)

Corollary 15. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be in $\mathcal{C}[\cap, ^+]$. If $F_1 \not\subseteq \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$.*

Proposition 6 and Corollary 15 combined yields that the Hasse diagram of $\leq^{\text{path}}$ for $\mathcal{C}[\cap, ^+]$ can be obtained from the Hasse diagram of $\leq^{\text{path}}$ for $\mathcal{C}[\cap]$ (Figure 1(b)), simply by adding $^+$ as a primitive to each language.

4.5 Cross-relationships between subdiagrams

To finish the proof of Theorem 7, we finally show the "only if" direction for the case where $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ belong to different classes.

Proposition 16. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be languages that belong to different classes among $\mathcal{C}$, $\mathcal{C}[\cap]$, $\mathcal{C}[^+]$, and $\mathcal{C}[\cap, ^+]$. If $F_1 \not\subseteq \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$.*

Towards a proof of Proposition 16, we first establish the following.

Proposition 17. *Let F_1 and F_2 be sets of nonbasic features.*

1. *If $\cap \in \overline{F}_1$ and $\cap \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2)$.*
2. *If $^+ \in \overline{F}_1$ and $^+ \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}} \mathcal{N}(F_2)$.*

Proof. For (1), it can be verified by the Brute-Force Algorithm of Section 3.2.1 that the graphs shown in Figure 2 (c) are not distinguishable in $\mathcal{N}(F_2)$. The graphs, however, are distinguishable by the boolean query expressed by $R^2 \cap id$. For (2), it is well known that reachability queries such as the boolean query expressed by $R \circ S^+ \circ T$ cannot be expressed in FO (see, e.g., [3]). $\square$

As detailed below, Propositions 17, 9, and 11 are now subsequently used to show that for every pair F_1 and F_2 of sets of nonbasic features for which $F_1 \not\subseteq \overline{F}_2$, that $\mathcal{N}(F_1) \not\leq^{\text{path}} \mathcal{N}(F_2)$, in the same way as in Sections 4.1 and 4.2.

The remainder of the proof of Proposition 16 is again a combinatorial analysis to verify that the above-mentioned propositions cover all relevant cases. First, note that there are 1675 ordered pairs of languages $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ that belong to two distinct classes among $\mathcal{C}$, $\mathcal{C}[\cap]$, $\mathcal{C}[^+]$, and $\mathcal{C}[\cap, ^+]$ for which $F_1 \not\subseteq \overline{F}_2$. Concretely, Proposition 17 (1) deals with 672 of these pairs, Proposition 17 (2) deals with another 508 pairs, Proposition 9 (1) deals with another 228 pairs, Proposition 9 (2) deals with another 94 pairs, Proposition 9 (3) deals with another 12 pairs, Proposition 9 (4) deals with another 130 pairs, Proposition 11 (1) deals with another 18 pairs, and, finally, Proposition 11 (2) deals with the 13 remaining pairs.

Propositions 6, 8, 10, and 16, and Corollaries 13 and 15, together prove Theorem 7.

Hence, the Hasse diagram of $\leq^{\text{path}}$ can be obtained from the subdiagrams for $\mathcal{C}$, $\mathcal{C}[\cap]$, $\mathcal{C}[^+]$, and $\mathcal{C}[\cap, ^+]$ by simply adding the canonical inclusion arrows between

1. the subdiagram for $\mathcal{C}$ and the subdiagram for $\mathcal{C}[\cap]$ (12 arrows);
2. the subdiagram for $\mathcal{C}$ and the subdiagram for $\mathcal{C}[^+]$ (12 arrows);
3. the subdiagram for $\mathcal{C}[\cap]$ and the subdiagram for $\mathcal{C}[\cap, ^+]$ (14 arrows);
4. the subdiagram for $\mathcal{C}[^+]$ and the subdiagram for $\mathcal{C}[\cap, ^+]$ (12 arrows).

So, all paths between the subdiagrams are induced by the canonical inclusion arrows above and the 5 equations from the beginning of Section 4.

5. BOOLEAN QUERIES

In this section, we characterize the order $\leq^{\text{bool}}$ of relative expressiveness for boolean queries by Theorem 20 below.

Towards the statement of this characterization, first observe that $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2)$ implies $\mathcal{N}(F_1) \leq^{\text{bool}} \mathcal{N}(F_2)$. The converse does not hold, however. Indeed, from Proposition 18 below, it follows that, e.g., $\mathcal{N}(^{-1}) \leq^{\text{bool}} \mathcal{N}(\pi)$. From Theorem 7, however, we know that $\mathcal{N}(^{-1}) \not\leq^{\text{path}} \mathcal{N}(\pi)$.

Proposition 18. *Let F be a set of nonbasic features for which $\cap \notin \overline{F}$ and $^+ \notin \overline{F}$. Then, $\mathcal{N}(F \cup \{^{-1}\}) \leq^{\text{bool}} \mathcal{N}(F \cup \{\pi\})$.*

Example 19. *To illustrate Proposition 18 (proof omitted), consider the expression $e_1 = R^3 \circ R^{-1} \circ R^3$ in $\mathcal{N}(^{-1})$. The expression $\pi_1(e_1)$ can be equivalently expressed in $\mathcal{N}(\pi)$ as $\pi_1\big(R^3 \circ \overline{\pi}_2(\pi_1(R^3) \circ R)\big)$. Now observe that, for any graph G, we have that $e_1(G)$ is nonempty if and only if $\pi_1(e_1)(G)$ is nonempty.*

Using this same observation, one can express the nonemptiness of the expression $e_2 = R \circ \overline{\pi}_2((R \circ S) \cup (R^{-1} \circ S))$ in $\mathcal{N}(^{-1}, \overline{\pi})$ by the non-emptiness of the expression $\pi_1(e_2) = \pi_1\big(R \circ \overline{\pi}_2(R \circ S) \circ \overline{\pi}_2(\pi_1(R) \circ S)\big)$ in $\mathcal{N}(\overline{\pi})$.

Proposition 18 shows that, at the level of boolean queries, $^{-1}$ does not add expressive power in the presence of π and in the absence of $\cap$ and $^+$. We thus obtain a result mirroring similar results known for XPath on trees [6, 22, 27], where downward XPath is known to be as powerful as full XPath for queries evaluated at the root. Note that Marx [19] also obtained a similar result for XPath with transitive closure, but we will show below that this no longer holds on graphs (see Proposition 28).

To accommodate the collapse of $^{-1}$ in our characterization of $\leq^{\text{bool}}$, we introduce the following notation. For a set of nonbasic features F, define $\widetilde{F}$ as follows.

$$\widetilde{F} = \begin{cases} \overline{F} \cup \{^{-1}\} & \text{if } \pi \in \overline{F},\ \cap \notin \overline{F}, \text{ and } ^+ \notin \overline{F} \\ \overline{F} & \text{otherwise} \end{cases}$$

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

We will establish the following characterization.

THEOREM 20. *$\mathcal{N}(F_1) \leq^{\text{bool}} \mathcal{N}(F_2)$ if and only if $F_1 \subseteq \widetilde{F}_2$.*

The "if" direction of Theorem 20 is shown in Proposition 21.

PROPOSITION 21. *If $F_1 \subseteq \widetilde{F}_2$, then $\mathcal{N}(F_1) \leq^{\text{bool}} \mathcal{N}(F_2)$.*

PROOF. We distinguish two cases. If $F_1 \subseteq \overline{F}_2$, then $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2)$, by Proposition 6, whence $\mathcal{N}(F_1) \leq^{\text{bool}} \mathcal{N}(F_2)$. In the other case, $\pi \in \overline{F}_2$, $\cap \notin \overline{F}_2$, and $^+ \notin \overline{F}_2$, and $F_1 \subseteq \overline{F}_2 \cup \{^{-1}\}$. By definition, $\overline{F}_2 \cup \{^{-1}\} = \overline{F_2 \cup \{^{-1}\}}$, since $\pi \in \overline{F}_2$. Hence, $F_1 \subseteq \overline{F_2 \cup \{^{-1}\}}$. Then, by Theorem 7, $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2 \cup \{^{-1}\})$. Since $F_2 \cup \{^{-1}\} \subseteq \overline{F}_2 \cup \{^{-1}\}$, it also follows that $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(\overline{F}_2 \cup \{^{-1}\})$. Since $\cap \notin \overline{F}_2$ and $^+ \notin \overline{F}_2$, we have that $\mathcal{N}(\overline{F}_2 \cup \{^{-1}\}) \leq^{\text{bool}} \mathcal{N}(\overline{F}_2 \cup \{\pi\}) = \mathcal{N}(\overline{F}_2)$, by Proposition 18. By combining these, we finally find that $\mathcal{N}(F_1) \leq^{\text{bool}} \mathcal{N}(\overline{F}_2)$. Proposition 21 now follows from the fact that $\mathcal{N}(\overline{F}_2)$ and $\mathcal{N}(F_2)$ are equivalent at the level of path queries and hence also at the level of boolean queries. $\square$

The "only if" direction of Theorem 20, requires a detailed analysis, which proceeds along the same lines as the analysis in Section 4. We first establish the "only if" direction for the cases where $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ belong to the same class among $\mathcal{C}$, $\mathcal{C}[\cap]$, $\mathcal{C}[^+]$, $\mathcal{C}[\cap, ^+]$, and then consider the case where $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ belong to distinct classes.

5.1 Languages without $\cap$ and without $^+$

In this subsection, we show the "only if" direction of Theorem 20, restricted to $\mathcal{C}$, the class of languages without $\cap$ and without $^+$.

PROPOSITION 22. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be in $\mathcal{C}$. If $F_1 \not\subseteq \widetilde{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}} \mathcal{N}(F_2)$.*

Propositions 21 and 22 combined yield the Hasse diagram of $\leq^{\text{bool}}$ for $\mathcal{C}$, shown in Figure 5. It is indeed readily verified that for any two languages $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ in $\mathcal{C}$, there is a path from $\mathcal{N}(F_1)$ to $\mathcal{N}(F_2)$ in Figure 5 if and only if $F_1 \subseteq \widetilde{F}_2$.

Towards a proof of Proposition 22, we first establish the following.

PROPOSITION 23. *Let F be a set of nonbasic features. If $^{-1} \in \overline{F}$, then $\mathcal{N}(F) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(di, ^+)$.*

PROOF. It can be verified by the Brute-Force Algorithm of Section 3.2.1 that the graphs shown in Figure 2 (c) are not distinguishable in $\mathcal{N}(di, ^+)$. The graphs, however, are distinguishable by the boolean query expressed by $R^2 \circ R^{-1} \circ R^2$. $\square$

As detailed below, Propositions 9 and 23 are now subsequently used to show that for every pair F_1 and F_2 of sets of nonbasic features for which $F_1 \not\subseteq \widetilde{F}_2$, that $\mathcal{N}(F_1) \not\leq^{\text{bool}} \mathcal{N}(F_2)$, in the same way as in Sections 4.1 and 4.2.

The remainder of the proof of Proposition 22 is again a combinatorial analysis to verify that the above-mentioned propositions cover all relevant cases. First, note that there are 34 ordered pairs of distinct languages $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ in $\mathcal{C}$ for which $F_1 \not\subseteq \widetilde{F}_2$. (These are exactly the pairs for

$$\begin{tikzcd}
\mathcal{N}(\boxed{^{-1}, di, \overline{\pi}}, \pi) = \mathcal{N}(\boxed{di, \overline{\pi}}, \pi) & \\
 & \mathcal{N}(\boxed{^{-1}, \overline{\pi}}, \pi) = \mathcal{N}(\boxed{\overline{\pi}}, \pi) \arrow[ul] \\
\mathcal{N}(\boxed{^{-1}, di, \pi}) = \mathcal{N}(\boxed{di, \pi}) \arrow[uu] & \\
 & \mathcal{N}(\boxed{^{-1}, \pi}) = \mathcal{N}(\boxed{\pi}) \arrow[ul] \arrow[uu] \\
\mathcal{N}(\boxed{^{-1}, di}) \arrow[uu] & \\
 & \mathcal{N}(\boxed{^{-1}}) \arrow[ul] \arrow[uu] \\
\mathcal{N}(\boxed{di}) \arrow[uu] & \\
 & \mathcal{N} \arrow[ul] \arrow[uu]
\end{tikzcd}$$

Figure 5: The Hasse diagram of $\leq^{\text{bool}}$ for $\mathcal{C}$. For each language, the boxed features are a minimal set of nonbasic features defining the language, while the other features can be derived from them in the sense of Theorem 7 (using the appropriate interdependencies).

which there is no path from $\mathcal{N}(F_1)$ to $\mathcal{N}(F_2)$ in Figure 5.) Concretely, Proposition 9 (1) deals with 16 of these pairs, Proposition 9 (2) deals with another 9 pairs, Proposition 9 (3) deals with another 6 pairs, and, finally, Proposition 23 deals with the remaining 3 pairs.

5.2 Languages with $\cap$ and without $^+$

In this subsection, we show the "only if" direction of Theorem 20, restricted to $\mathcal{C}[\cap]$, the class of languages with $\cap$ but without $^+$.

PROPOSITION 24. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be in $\mathcal{C}[\cap]$. If $F_1 \not\subseteq \widetilde{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2)$.*

Notice that since $\cap \in \overline{F}_2$, $\widetilde{F}_2 = \overline{F}_2$. Hence, Proposition 7 and Proposition 24 combined show that $\leq^{\text{bool}}$ coincides with $\leq^{\text{path}}$ on $\mathcal{C}[\cap]$. As a result, the Hasse diagram of $\leq^{\text{bool}}$ for $\mathcal{C}[\cap]$ is the same as the Hasse diagram of $\leq^{\text{path}}$ for $\mathcal{C}[\cap]$ shown in Figure 1(b). Note that, in addition, all separations are strong.

Towards a proof of Proposition 24, we first establish the following.

PROPOSITION 25. *Let F_1 and F_2 be sets of nonbasic features. If $^{-1} \in F_1$, $\cap \in F_1$, and $^{-1} \notin F_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{strong}} \mathcal{N}(F_2)$.*

PROOF. The graphs G_1 and G_2 shown in Figure 2 (d), top and bottom, are distinguished by the boolean query q

expressed by $(R^2 \circ R^{-1} \circ R) \cap R$. On these graphs, the Brute-Force Algorithm of Section 3.2.1 does not terminate in a reasonable time. Using a variant of the well-known algorithm by Paige and Tarjan [24], it can be verified in polynomial time, however, that for each pair $(a_1, b_1) \in \mathrm{adom}(G_1)^2$, there exists $(a_2, b_2) \in \mathrm{adom}(G_2)^2$ such that $(G_1, a_1, b_1) \simeq_k (G_2, a_2, b_2)$ for any depth k. From Proposition 5, it follows that q is not expressible in $\mathcal{N}(F_2)$. $\square$

The remainder of the proof of Proposition 24 proceeds as the proof of Proposition 10, except that Proposition 25 is used instead of Proposition 9 (4).

5.3 Languages without $\cap$ and with $^+$

In this subsection, we show the "only if" direction of Theorem 20, restricted to $\mathcal{C}[^+]$, the class of languages without $\cap$ but with $^+$.

Proposition 26. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be in $\mathcal{C}[^+]$. If $F_1 \not\subseteq \widetilde{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}} \mathcal{N}(F_2)$.*

Notice that since $^+ \in F_2$, $\widetilde{F}_2 = \overline{F}_2$. Hence, Proposition 7 and Proposition 26 combined show that $\leq^{\mathrm{bool}}$ coincides with $\leq^{\mathrm{path}}$ on $\mathcal{C}[^+]$. As a result, the Hasse diagram of $\leq^{\mathrm{bool}}$ for $\mathcal{C}[^+]$ is the same as the Hasse diagram of $\leq^{\mathrm{path}}$ for $\mathcal{C}[^+]$. (Recall that the latter is obtained from Figure 1(a) by adding $^+$ as a primitive to each language.) Observe in particular that the collapses for $\leq^{\mathrm{bool}}$ compared to $\leq^{\mathrm{path}}$ in the case of languages without $\cap$ and without $^+$ disappear again when $^+$ is added.

Towards a proof of Proposition 26, we first remark the following.

Remark 27. Let F_1, F_2, and G be arbitrary sets of nonbasic features. (In particular, they may or may not contain $\cap$, $\setminus$, or $^+$.) While it is straightforward that $\mathcal{N}(F_1) \leq^{\mathrm{path}} \mathcal{N}(F_2)$ implies $\mathcal{N}(F_1 \cup G) \leq^{\mathrm{path}} \mathcal{N}(F_2 \cup G)$, the corresponding statement for $\leq^{\mathrm{bool}}$ does *not* hold. Indeed, by Proposition 18, $\mathcal{N}(^{-1}, \pi) \leq^{\mathrm{bool}} \mathcal{N}(\pi)$. However, $\mathcal{N}(^{-1}, \pi, ^+) \not\leq^{\mathrm{bool}} \mathcal{N}(\pi, ^+)$, as will follow from Proposition 28 below (proof omitted).

Proposition 28. *Let F_1 and F_2 be sets of nonbasic features. If $^{-1} \in \overline{F}_1$, $^+ \in \overline{F}_1$, and $^{-1} \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}} \mathcal{N}(F_2)$.*

The remainder of the proof of Proposition 26 proceeds as the proof of Proposition 8, except that Proposition 28 is used instead of Proposition 9 (4).

5.4 Languages with $\cap$ and with $^+$

As in Section 4.4, we have the following.

Proposition 29. *Let F_1 and F_2 be sets of nonbasic features for which $\cap \in \overline{F}_1$, $\cap \in \overline{F}_2$, $^+ \notin \overline{F}_1$, and $^+ \notin \overline{F}_2$. Then, $\mathcal{N}(F_1 \cup \{^+\}) \leq^{\mathrm{bool}} \mathcal{N}(F_2 \cup \{^+\})$ if and only if $\mathcal{N}(F_1) \leq^{\mathrm{bool}} \mathcal{N}(F_2)$.*

Proof. By Remark 27, we may not straightforwardly infer from $\mathcal{N}(F_1) \leq^{\mathrm{bool}} \mathcal{N}(F_2)$ that $\mathcal{N}(F_1 \cup \{^+\}) \leq^{\mathrm{bool}} \mathcal{N}(F_2 \cup \{^+\})$. However, we established in Section 5.2 that $\mathcal{N}(F_1) \leq^{\mathrm{bool}} \mathcal{N}(F_2)$ implies $\mathcal{N}(F_1) \leq^{\mathrm{path}} \mathcal{N}(F_2)$, whence also that $\mathcal{N}(F_1 \cup \{^+\}) \leq^{\mathrm{path}} \mathcal{N}(F_2 \cup \{^+\})$ and $\mathcal{N}(F_1 \cup \{^+\}) \leq^{\mathrm{bool}} \mathcal{N}(F_2 \cup \{^+\})$. This settles the "if". To see the "only if", assume that $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}} \mathcal{N}(F_2)$. By Propositions 21 and 24, it follows that $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}}_{\mathrm{strong}} \mathcal{N}(F_2)$, whence $\mathcal{N}(F_1 \cup \{^+\}) \not\leq^{\mathrm{bool}}_{\mathrm{strong}} \mathcal{N}(F_2 \cup \{^+\})$. $\square$

Corollary 30. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be in $\mathcal{C}[\cap, ^+]$. If $F_1 \not\subseteq \widetilde{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}}_{\mathrm{strong}} \mathcal{N}(F_2)$.*

Notice that since $^+ \in F_2$, $\widetilde{F}_2 = \overline{F}_2$. Hence, Proposition 7 and Corollary 30 combined show that $\leq^{\mathrm{bool}}$ coincides with $\leq^{\mathrm{path}}$ on $\mathcal{C}[\cap, ^+]$. As a result, the Hasse diagram of $\leq^{\mathrm{bool}}$ for $\mathcal{C}[\cap, ^+]$ is the same as the Hasse diagram of $\leq^{\mathrm{path}}$ for $\mathcal{C}[\cap, ^+]$. (Recall that the latter is obtained from Figure 1(b) by adding $^+$ as a primitive to each language.) Note that, in addition, all separations are strong.

5.5 Cross-relationships between subdiagrams

To finish the proof of Theorem 20, we finally show the "only if" direction for the case where $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ belong to different classes.

Proposition 31. *Let $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ be languages that belong to different classes among $\mathcal{C}$, $\mathcal{C}[\cap]$, $\mathcal{C}[^+]$, and $\mathcal{C}[\cap, ^+]$. If $F_1 \not\subseteq \widetilde{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}} \mathcal{N}(F_2)$.*

Towards a proof of Proposition 31, we first establish the following.

Proposition 32. *Let F_1 be a set of nonbasic features. If $^{-1} \in \overline{F}_1$, and $F_2 \subseteq \{\setminus, \cap, ^+\}$, then $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}}_{\mathrm{strong}} \mathcal{N}(F_2)$.*

Proof. It can be verified by the Brute-Force Algorithm of Section 3.2.1 that the graphs shown in Figure 2 (b) are not distinguishable in $\mathcal{N}(F_2)$. The graphs, however, are distinguishable by the boolean query expressed by $R^2 \circ R^{-1} \circ R^2$. $\square$

As detailed below, Propositions 17, 9, 11, 23, 25, 28, and 32 are now subsequently used to show that for every pair F_1 and F_2 of sets of nonbasic features for which $F_1 \not\subseteq \widetilde{F}_2$, that $\mathcal{N}(F_1) \not\leq^{\mathrm{bool}} \mathcal{N}(F_2)$, in the same way as in Sections 5.1–5.4.

The remainder of the proof of Proposition 31 is again a combinatorial analysis to verify that the above-mentioned propositions cover all relevant cases. First, note that there are 1370 ordered pairs of languages $\mathcal{N}(F_1)$ and $\mathcal{N}(F_2)$ that belong to two distinct classes among $\mathcal{C}$, $\mathcal{C}[\cap]$, $\mathcal{C}[^+]$, and $\mathcal{C}[\cap, ^+]$ for which $F_1 \not\subseteq \widetilde{F}_2$. Concretely, Proposition 17 (1) deals with 560 of these pairs, Proposition 17 (2) deals with another 460 pairs, Proposition 9 (1) deals with another 184 pairs, Proposition 9 (2) deals with another 68 pairs, Proposition 9 (3) deals with another 6 pairs, Proposition 11 (1) deals with another 21 pairs, Proposition 11 (2) deals with another 20 pairs, Proposition 23 deals with another 3 pairs, Proposition 25 deals with another 18 pairs, Proposition 28 deals with another 26 pairs, and, finally, Proposition 32 deals with the 4 remaining pairs.

Propositions 21, 22, 24, 26, and 31, and Corollary 30, together prove Theorem 20.

Hence, the Hasse diagram of $\leq^{\mathrm{bool}}$ can be obtained from the subdiagrams for $\mathcal{C}$, $\mathcal{C}[\cap]$, $\mathcal{C}[^+]$, and $\mathcal{C}[\cap, ^+]$ by simply adding the canonical inclusion arrows between

1. the subdiagram for $\mathcal{C}$ and the subdiagram for $\mathcal{C}[\cap]$ taking into account only the representations with the feature sets that are minimal with respect to inclusion (8 arrows);
2. the subdiagram for $\mathcal{C}$ and the subdiagram for $\mathcal{C}[^+]$ taking into account only the representations with the feature sets that are minimal with respect to inclusion (8 arrows);

3. the subdiagram for $\mathcal{C}[\cap]$ and the subdiagram for $\mathcal{C}[\cap, {}^+]$ (14 arrows);
4. the subdiagram for $\mathcal{C}[{}^+]$ and the subdiagram for $\mathcal{C}[\cap, {}^+]$ (12 arrows).

So, all paths between the subdiagrams are induced by the canonical inclusion arrows above, the 5 equations from the beginning of Section 4, and Proposition 18.

6. QUERIES ON UNLABELED GRAPHS

In this section, we consider the case in which the set Λ of edge labels is a singleton. In other words, a graph G is then a relational structure consisting of a set of nodes V and a relation $E \subseteq V \times V$, the set of edges of G. We use the notation $\leq^{\text{path}}_{\text{unl}}$ and $\leq^{\text{bool}}_{\text{unl}}$ to compare the expressiveness of languages on such unlabeled graphs.

Of course, $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2)$ implies $\mathcal{N}(F_1) \leq^{\text{path}}_{\text{unl}} \mathcal{N}(F_2)$, and $\mathcal{N}(F_1) \leq^{\text{bool}} \mathcal{N}(F_2)$ implies $\mathcal{N}(F_1) \leq^{\text{bool}}_{\text{unl}} \mathcal{N}(F_2)$.

We must emphasize that, with the exception of Proposition 17 (2), none of the separation results from Sections 4 and 5 depend on the presence of multiple edge labels, and therefore already hold in the case of unlabeled graphs. For Proposition 17 (2), however, at the level of general path queries, we can establish the following counterpart using the well-known fact that transitive closure of a binary relation is not expressible in FO, and hence also not in any $\mathcal{N}(F)$ without transitive closure [26]:

PROPOSITION 33. *Let F_1 and F_2 be sets of nonbasic features. If ${}^+ \in \overline{F}_1$, and ${}^+ \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{path}}_{\text{unl}} \mathcal{N}(F_2)$.*

We therefore have

PROPOSITION 34. *Let F_1 and F_2 be sets of nonbasic features. Then, $\mathcal{N}(F_1) \leq^{\text{path}}_{\text{unl}} \mathcal{N}(F_2)$ if and only if $\mathcal{N}(F_1) \leq^{\text{path}} \mathcal{N}(F_2)$.*

We may thus conclude that $\leq^{\text{path}}_{\text{unl}}$ coincides with $\leq^{\text{path}}$.

At the level of boolean queries, as we show next, a counterpart of Proposition 17 (2), similar to Proposition 33, does not exist. It will actually turn out that, on unlabeled graphs, adding transitive closure does not always add expressive power.

However, we have the following weaker result.

PROPOSITION 35. *Let F_1 and F_2 be sets of nonbasic features.*

1. *If ${}^+ \in \overline{F}_1$, $\cap \in \overline{F}_1$, and ${}^+ \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{unl}} \mathcal{N}(F_2)$.*
2. *If ${}^+ \in \overline{F}_1$, ${}^{-1} \in \overline{F}_1$, and ${}^+ \notin \overline{F}_2$, then $\mathcal{N}(F_1) \not\leq^{\text{bool}}_{\text{unl}} \mathcal{N}(F_2)$.*

PROOF. For (1), it is well known that the query that checks whether a graph contains a cycle cannot be expressed in FO (see, e.g., [3]). The query, however, is expressed by $R^+ \cap id$. For (2), a classical Ehrenfeucht-Fraïssé argument shows that the boolean query expressed by $R^2 \circ (R \circ R^{-1})^+ \circ R^2$ cannot be expressed in FO (see, e.g., [3]). $\square$

The languages not covered by the statements in Proposition 35 are $\mathcal{N}({}^+)$, $\mathcal{N}(\pi, {}^+)$, $\mathcal{N}(di, {}^+)$, $\mathcal{N}(\overline{\pi}, {}^+)$, $\mathcal{N}(di, \pi, {}^+)$, and $\mathcal{N}(di, \overline{\pi}, {}^+)$. We have the following collapses for $\leq^{\text{bool}}_{\text{unl}}$ compared to $\leq^{\text{bool}}$.

PROPOSITION 36. *The following collapses occur at the level of boolean queries on unlabeled graphs, but not on labeled graphs:*

$$\mathcal{N}({}^+) \leq^{\text{bool}}_{\text{unl}} \mathcal{N}, \tag{1}$$

$$\mathcal{N}(\pi, {}^+) \leq^{\text{bool}}_{\text{unl}} \mathcal{N}(\pi), \text{ and} \tag{2}$$

$$\mathcal{N}(di, {}^+) \leq^{\text{bool}}_{\text{unl}} \mathcal{N}(di). \tag{3}$$

PROOF. We first sketch the proof of Statement (2). Let e be an expression in $\mathcal{N}(\pi, {}^+)$. We show that a sufficient condition for $e(G)$ to be nonempty is that $R^m(G)$ is nonempty, where m is a natural number computable from e. We then show that on a graph G with $R^m(G)$ empty, e can equivalently be expressed by an expression e' in $\mathcal{N}(\pi)$. From the above, it follows immediately that $e(G)$ is nonempty if and only if $R^m \cup e'(G)$ is nonempty. This proves Statement (2). Additionally, we show that, if e is in $\mathcal{N}({}^+)$, then e' is in $\mathcal{N}$, which proves Statement (1). The proof of Statement (3) is omitted. $\square$

It is still open whether the languages $\mathcal{N}(\overline{\pi}, {}^+)$, $\mathcal{N}(di, \pi, {}^+)$, and/or $\mathcal{N}(di, \overline{\pi}, {}^+)$ collapse to any language without ${}^+$. It is conjectured that $\mathcal{N}(\overline{\pi}, {}^+) \leq^{\text{bool}}_{\text{unl}} \mathcal{N}(\overline{\pi})$; $\mathcal{N}(di, \pi, {}^+) \leq^{\text{bool}}_{\text{unl}} \mathcal{N}(di, \pi)$; and $\mathcal{N}(di, \overline{\pi}, {}^+) \leq^{\text{bool}}_{\text{unl}} \mathcal{N}(di, \overline{\pi})$.

7. CONCLUSIONS

In this paper, we considered all languages defined by the basic features union, composition, and the identity relation, and none, some, or all the nonbasic features intersection, set difference, projection, coprojection, converse, transitive closure, and the diversity relation. While some of these languages have been studied in the context of XPath, they are there evaluated over trees. In our work, we have studied these languages as simple navigational query languages for graphs. The main contribution of this paper is that we have been able to perform a complete comparison of all the languages under consideration, and this both at the level of general path queries and the level of boolean queries.

Especially in the case of boolean queries, we encountered some nontrivial separations and a few surprising collapses. With regard to the latter, we refer in particular to the result stating that converse can be eliminated in the presence of projection and in the absence of intersection and transitive closure. However, these collapses disappear again if intersection or transitive closure is added.

Finally, we discussed the restrictions of the above languages to unlabeled graphs. We first established that the relative expressiveness of these languages at the level of general path queries remains unchanged by this restriction. At the level of boolean queries, however, some additional collapses occur. Using intricate arguments, we were able to establish that the three languages defined by the basic features augmented by transitive closure, respectively projection and transitive closure, respectively the diversity relation and transitive closure, collapse to their counterparts without transitive closure. For three remaining languages with transitive closure, it remains open whether such collapse occurs, though the present authors conjecture that this is the case.

8. FURTHER RESEARCH

There are alternative modalities for expressing boolean queries apart from interpreting the nonemptiness of an expression as "true" and emptiness as "false". For example, we can switch these interpretations and interpret nonemptiness as "false" and emptiness as "true". Yet another possibility is to consider a boolean query q expressible if there are two expressions e_1 and e_2 such that $e_1(G) = e_2(G)$ if, and only if, $q(G)$ is "true", for all G. For some of our languages, these alternative modalities would not make a difference, but it would for others. Looking into these alternative modalities is an interesting topic for further research.

In the present paper, we have been focusing on expressive power, but, of course, it is also interesting to investigate the decidability of satisfiability or containment of expressions. Much is already known. From the undecidability of FO^3, it follows that the most powerful language without transitive closure is undecidable, and the same holds even without converse. From the decidability of ICPDL [14], all languages without set difference are decidable, although this is not yet known if these languages are restricted to finite relations. An interesting topic for further research is the decidability of satisfiability or validity of the languages with set difference, but without the diversity relation.

Another natural question is whether the invariance under arrow logic bisimulation, that we use as a tool to prove some nonexpressibility results, actually provides characterizations of indistinguishability in the various languages (say, up to some quantifier rank), as is the case for modal logic [15]. We have in fact proved this in a number of cases [11]. A further question then is whether van Benthem-style expressive completeness results [23] can be established.

Finally, there are still other interesting operators on binary relations that can be considered. A good example is residuation [25], a derived operator of the calculus of relations, and interesting to consider separately, as we have done for projection and coprojection. Residuation is interesting from a database perspective because it corresponds to the set containment join (e.g., [18]).

9. REFERENCES

[1] RDF primer, 2004. `http://www.w3.org/TR/rdf-primer/`.

[2] S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufmann, 1999.

[3] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of databases*. Addison Wesley, Reading, MA, 1995.

[4] R. Angles and C. Gutiérrez. Survey of graph database models. *ACM Comput. Surv.*, 40(1):1–39, 2008.

[5] F. Baader, D. Calvanese, D. McGuiness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.

[6] M. Benedikt, W. Fan, and G. M. Kuper. Structural properties of XPath fragments. In *ICDT*, pages 79–95, 2003.

[7] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.

[8] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 2001.

[9] D. Calvanese, G. D. Giacomo, M. Lenzerini, and M. Y. Vardi. Containment of conjunctive regular path queries with inverse. In *KR*, pages 176–185, 2000.

[10] A. Deutsch and V. Tannen. Optimization properties for classes of conjunctive regular path queries. In *DBPL*, pages 21–39, 2001.

[11] G. Fletcher, M. Gyssens, D. Leinders, J. Van den Bussche, D. Van Gucht, S. Vansummeren, and Y. Wu. Unpublished results, 2010.

[12] D. Florescu, A. Levy, and A. Mendelzon. Database techniques for the World-Wide Web: A survey. *SIGMOD Record*, 27(3):59–74, 1998.

[13] M. J. Franklin, A. Y. Halevy, and D. Maier. From databases to dataspaces: a new abstraction for information management. *SIGMOD Record*, 34(4):27–33, 2005.

[14] S. Göller, M. Lohrey, and C. Lutz. PDL with intersection and converse: satisfiability and infinite-state model checking. *Journal of Symbolic Logic*, 74(1):279–314, 2009.

[15] V. Goranko and M. Otto. Model theory of modal logics. In P. Blackburn, J. van Benthem, and F. Wolter, editors, *Handbook of Modal Logic*, pages 249–329. Elsevier, 2007.

[16] D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. MIT Press, 2000.

[17] R. D. Maddux. *Relation algebras*. Elsevier, Amsterdam, 2006.

[18] N. Mamoulis. Efficient processing of joins on set-valued attributes. In *ACM SIGMOD*, pages 157–168, 2003.

[19] M. Marx. Conditional XPath. *ACM Trans. Database Syst.*, 30(4):929–959, 2005.

[20] M. Marx and M. de Rijke. Semantic characterizations of navigational XPath. *SIGMOD Record*, 34(2):41–46, 2005.

[21] M. Marx and Y. Venema. *Multi-Dimensional Modal Logic*. Springer, 1997.

[22] D. Olteanu. Forward node-selecting queries over trees. *ACM Trans. Database Syst.*, 32(1):3, 2007.

[23] M. Otto. Model theoretic methods for fragments of FO and special classes of (finite) structures. Survey at 2006 Durham workshop on Finite and Algorithmic Model Theory, 2008.

[24] R. Paige and R. Tarjan. Three partition refinement algorithms. *SIAM J. Comput.*, 16(6):973–989, 1987.

[25] V. R. Pratt. Origins of the calculus of binary relations. In *LICS*, pages 248–254, 1992.

[26] A. Tarski and S. Givant. *A formalization of set theory without variables*. American Mathematical Society, 1987.

[27] Y. Wu, D. Van Gucht, M. Gyssens, and J. Paredaens. A study of a positive fragment of path queries: expressiveness, normal form, and minimization. *The Computer Journal*, 2010. Published online, 11 July.