Normal forms for binary relations

Daniel J. Dougherty[a,*], Claudio Gutiérrez[b]

[a] *Department of Computer Science, Worcester Polytechnic Institute, USA*
[b] *Department of Computer Science, Universidad de Chile, Chile*

Abstract

We consider the representable equational theory of binary relations, in a language expressing composition, converse, and lattice operations. By working directly with a presentation of relation expressions as graphs we are able to define a notion of reduction which is confluent and strongly normalizing and induces a notion of computable normal form for terms. This notion of reduction thus leads to a computational interpretation of the representable theory.

Keywords: Binary relations; Graph representation; Normal forms; Omitting minors

1. Introduction

The theory of binary relations is a fundamental conceptual and methodological tool in computer science. The formal study of relations was central to early investigations of logic and the foundations of mathematics [12,30,34–36] and has more recently found application in program specification and derivation [2,7,4,26], denotational and axiomatic semantics of programs [21,11,32,28], and hardware design and verification [8,23].

The collection of binary relations on a set has rich algebraic structure: it forms a monoid under composition, each relation has a converse, and it forms a Boolean algebra under the usual set-theoretic operations. In fact, the equational theory in this language is undecidable, since it is possible to encode set theory [36]. In this paper, we eliminate complementation as an operation, and investigate the equational theory $E_{\mathcal{R}}$ of representable relation algebras (Definition 1): this theory is the set of equations between relation expressions valid when interpreted over sets. Andréka and Bredikhin [1] have shown that $E_{\mathcal{R}}$ is decidable: our goal is a finer-grained analysis of the valid equations.

Now, the most popular framework for foundations and for implementations of theorem provers, proof-checkers, and programming languages remains the λ-calculus. It seems reasonable to say that this is due at least in part to the fact that the equational theory of λ-terms admits a computational treatment which is well-behaved: β-reduction is confluent, and terminating in typed calculi, so that the notion of *normal form* is central to the theory.

* Corresponding author. Tel.: +1 508 831 5621.
E-mail address: dd@cs.wpi.edu (D.J. Dougherty).

To our knowledge, no analogous notion of normal form for terms in the theory of relations is known. In fact, the calculus of relations has a reputation for being complex. Bertrand Russell (quoted in [31]) viewed the classical results of Peirce and Schröder on relational calculus as being "difficult and complicated to so great a degree as to doubt their utility." And in their monograph [4, p. 81] Bird and de Moor observe that "the calculus of relations has gained a good deal of notoriety for the apparently enormous number of operators and laws one has to memorize in order to do proofs effectively."

Furthermore, Andréka and Bredikhin [1] have shown that although the class of representable relation algebras has a decidable equational theory, it is not a variety. So if we are to attempt a deeper understand of $E_{\mathcal{R}}$ the usual techniques of computational equational logic, term-rewriting in particular, would seem to be unavailable: no term rewriting system can even claim to correctly present the theory, much less be a foundation for computing with relational terms.

But in this paper we suggest that a rather attractive syntactic/computational treatment of the representable theory of relations is indeed available, at least for the fragment of the theory not including complementation.

The starting point is the idea of taking certain graphs as the representation of relations. These graphs, called here "diagrams," arise very naturally and have been used since Peirce by researchers in the relation community (e.g. Tarski, Lyndon, Jónsson, Maddux, etc.). Recent formalizations appear in [1,5,8,16,18,19]. What we do here is to take graphs seriously as a notation alternative to first-order terms, i.e., to treat diagrams as first-class *syntactic* entities, and specifically as candidates for *rewriting*.

In Section 3, we explain how one can see diagram rewriting as an instance of a standard technique in automated deduction, rewriting modulo a set of equations [22]. It is well-known that certain equations inhibit classical term-rewriting techniques—the typical examples are associativity and commutativity—and that a useful response can be to pass to computing *modulo* these equations. Indeed we exhibit a set $E_{\mathcal{D}}$ of equations such that diagrams are the natural data structure for representing terms modulo $E_{\mathcal{D}}$.

We first clarify the relationship between terms and diagrams by showing that the algebra of diagrams is precisely the free algebra for a certain finite set $E_{\mathcal{D}}$ of equations between terms (Theorem 15). It is rather surprising that a finite set of equations accounts for precisely the identifications between terms induced by compiling them into diagrams.

We also characterize those graphs which arise as diagrams, as those graphs that omit certain graphs as minors (Theorem 31).

Our main result is a computational treatment of diagrams via a notion of reduction. We prove that reduction satisfies strong normalization and Church–Rosser properties, and so the theory enjoys unique (diagram-) normal forms (Theorems 34–36). These normal forms are effectively computable, giving another proof of decidability of the theory.

In light of the characterization of the set of diagrams as the free algebra for the set $E_{\mathcal{D}}$ of equations, these results can be seen—if one insists—as results about rewriting of terms modulo $E_{\mathcal{D}}$. But for us the diagram presentation is the primary one and is ultimately the closest to our intuition.

Compared with the decision procedure implicit in the method of Andréka and Bredikhin (outlined below in Theorem 5), the results here do not offer any improvement in the sense of computational complexity. But our approach enables us to systematically analyze diagrams from an algebraic perspective, complementing the traditional graph-theoretic point of view. And the existence of computable unique normal forms for diagrams provides another tool that we hope will help yield further results on relation algebras.

The role of union: It is natural to consider adding the union operation to the theory $E_{\mathcal{R}}$. But for the purposes of deciding validity, it is easily eliminated, as follows. The argument below is from [16, Section 2.21(10)].

It is readily verified that union distributes over intersection, composition, and converse. So we need only consider inclusions of the form

$$s_1 \cup \cdots \cup s_n \subseteq t_1 \cup \cdots \cup t_m,$$

where each s_i and t_j is a term over the union-free signature Σ. But such an inclusion is valid if and only if for each s_i there is a t_j such that $s_i \subseteq t_j$ is valid. For if there were an s_i such that for every t_j there is an interpretation $\mathcal{A}_j$ witnessing $s_i \not\subseteq t_j$, the Cartesian product of the $\mathcal{A}_j$ would witness the failure of

$$s_i \subseteq t_1 \cup \cdots \cup t_m$$

so that the original inclusion fails.

In light of this we restrict our attention in this paper to the union-free fragment of the full (negation-free) signature.

Related work: Bird and de Moor's [4] book is an extended presentation of the application of relational calculus to program specification and derivation, building explicitly on the theory of allegories. There, terms in relation calculus are not programs *per se*, but the authors do raise the question of how one might *execute* relation expressions [4, p. 110]. As noted there, a promising proposal is made by Lipton and Chapman in [26], where a notion of rewriting terms using the allegory axioms is presented. It should be very interesting to explore the relationship between the Lipton–Chapman model and the one presented here.

Brown and Hutton [8] apply relational methods to the problems of designing and verifying hardware circuits. They observe that people naturally reason informally about pictures of circuits and seek to provide formal basis, again based on allegories, for such reasoning; their vehicle is the relational language RUBY used to design hardware circuits. To our knowledge they do not claim decidability or normal forms for the theory they implement. An implementation of their method is distributed at [23].

The case for using a calculus of relations as a framework for concepts and methods in mathematics and computer science is compellingly made by Freyd and Scedrov in [16]. There they develop a categorical theory of *allegories*: "Allegories … are to binary relations between sets as categories are to functions between sets" [16]. In forthcoming work we apply some of the techniques of this paper to the theory of allegories.

Our diagrams are closely related to the often studied class of series–parallel graphs, arising in a variety of fields, such as electrical engineering, operations research, and the theory of concurrent processes. characterizations of families of such graphs by omitting minors have been obtained by previous authors [14,15,9] (see [29, Chapter 6], for an overview of these results). The structures we treat have a somewhat richer algebraic structure than the graphs treated in previous work, and our omitting minors theorem is correspondingly somewhat more involved.

An indication of the range of current investigations into relations and relation-calculi may be found in, for example, the books [33] or [6] or the proceedings of the roughly annual RelMiCS conferences. Two other investigations of graphical relation-calculi are the work of Kahl [24] and that of Curtis and Lowe [10].

The general topic of diagrammatic reasoning has attracted interest in several areas (see for example [3]). The present research might be viewed as a case-study in reasoning with diagrams in the general sense.

2. Preliminaries

2.1. Terms and models

The intended models of the theories we study are collections of binary relations over a set with the operations intersection $\cap$, composition ;, and converse $(\,)^\circ$, with the distinguished relation 1, the identity. The unary operator dom, the *domain* of a relation, can be defined as $\mathrm{dom}(x) = 1 \cap xx^\circ$, but for technical reasons (see Remark 16) it is useful to have an explicit symbol denoting it. Modulo the inclusion of dom the following definition is due to Andréka and Bredikhin [1].

Definition 1. Let Σ be the signature comprising the binary operation symbols ; and $\cap$, the unary operations $(\,)^\circ$ and dom, and the constant 1. A *subpositive set relation algebra* is a Σ-algebra

$$\mathcal{A} = \langle A, \cap, ;, (\,)^\circ, \mathrm{dom}, 1\rangle$$

in which A is a set of binary relations on some base set such that A contains the identity relation 1 and is closed under the operations intersection, composition, and converse. In $\mathcal{A}$ the symbols from Σ are interpreted in the standard way, with dom taken to be the *domain* operator as defined above. Composition is interpreted in "diagrammatic order" so that $x; y$ means "x first then y."

An arbitrary relation algebra [27] is *representable* if it is isomorphic to a subpositive set relation algebra. We denote by $\mathcal{R}$ the class of representable relation algebras over the signature Σ.

Let T_Σ denote the set of first-order *terms* over Σ and an infinite set $Vars = \{x_1, x_2, \ldots\}$ of variables. It is convenient to suppress explicit use of ; and to denote the composition $x; y$ as simply xy. An *atom* is a variable or the constant 1, and a *literal* is an atom or the converse of an atom.

Haimann [20] showed that $\mathcal{R}$ cannot be finitely axiomatized. Andréka and Bredikhin [1] showed that $\mathcal{R}$ is not a variety.

Let $E_{\mathcal{R}}$ be the set of equations valid in $\mathcal{R}$.

2.2. Graphs and diagrams

A *labelled, directed graph* is a structure $G = (V, E, s, f)$ where V is a finite set of *vertices*, $s \in V$, $f \in V$, and $E \subseteq V \times 2^{Vars} \times V$. That is, there is a distinguished *start vertex* s and a distinguished *finish vertex* f, not necessarily distinct, and edges are directed with each labelled by a set of variables from the set *Vars*. There may be several labelled edges between a given pair of vertices, and the labels need not be distinct. Multiple edges between vertices and self-loops permitted.

For brevity we may use the term *graph* to mean directed, labelled graph as above. All our graphs will be connected (in the sense that the underlying undirected graph is connected) unless stated otherwise. Let $\mathcal{G}$ denote the class of such graphs.

A *morphism* between graphs $G = (V, E, s, f)$ and $G' = (V', E', s', f')$ is a pair of related functions: the *vertex function* φ assigning to each vertex in V a vertex in V' and the *edge function* (it will cause no confusion to write this also as φ) which assigns to each edge $e \in E$ an edge $\varphi(e)$ such that

- the vertex function maps s to s' and f to f',
- if $e \in E$ is directed from a to b then $\varphi(e) \in E'$ is directed from $\varphi(a)$ to $\varphi(b)$, and
- the label of $\varphi(e)$ is the label of e.

It is easy to see that a morphism is an isomorphism (i.e., has a two-sided inverse) if and only if its edge function is a bijection. We write $G \cong H$ to indicate that G and H are isomorphic.

If $G = (V, E, s, f)$ is a graph, a *subgraph* is a graph of the form $G' = (V', E', s, f)$ with $V' \subseteq V$ and $E' \subseteq E$. Again, by convention, we assume G' connected unless stated otherwise.

A *pure graph* is a structure $G = (V, E, s, f)$ where V is a finite set of vertices, $s \in V$, $f \in V$ and E is a multiset of 2-element multisets from V. So edges are unlabelled and undirected.

Let $\mathcal{U}$ denote the class of pure graphs. If G is a (directed, labelled) graph then there is a natural pure graph $U(G)$ derived from G by forgetting the labels and directions and the edges.

Let G be a graph. If $v \in V(G)$, the graph $G - \{v\}$ denotes the subgraph obtained from G by deleting the vertex v and its incident edges. A vertex v is a *cut-vertex* if either G has a loop around v, or the pure graph $U(G - \{v\})$ is disconnected; we sometimes say that v *separates* G. A cut vertex v is a *critical vertex* if either v is s or f (but not both), or s and f lie in different components of the separated pure graph.

A *walk* in the graph G is a sequence of vertices $v_1, v_2, \ldots$ such that there is an edge from each v_i to v_{i+1} in $U(G)$. A *path* is a walk where all the vertices, except possibly the first and last, are different; a *trace* is a path from s to f. A graph is *connected* if there is a path between any two vertices.

Let $x \in \mathcal{V}(G)$. The *subgraph generated by* x is the subgraph determined by the set of vertices $\{y|$ there is a path from $x$to y using neither s nor f as internal vertices$\}$. We allow s and f as first or last vertices in such a path so as to ensure that both s and f are in subgraphs generated by vertices. Two subgraphs of G are *independent* if they have only the start and finish vertices in common.

Diagrams: Certain operations on graphs in $\mathcal{G}$ naturally correspond to the operations in Σ; we indicate those here.

When we draw our graphs we will indicate the start vertex by $\rhd$ and the finish vertex by $\lhd$; if the start and finish vertices are the same we will use $\bowtie$.

Atoms: The graph with only one vertex which is at the same time the start and finish, and no edges, will be denoted by 1.

$$1: \qquad \bowtie$$

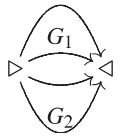
The graph with edge labelled i from the start vertex to the (distinct) finish vertex is denoted 2_i (or simply by i if no confusion arises).

$$2_i: \qquad \rhd \xrightarrow{i} \lhd.$$

To define, inductively, the remaining operations, let $G_1 = (V_1, E_1, s_1, f_1)$ and $G_2 = (V_2, E_2, s_2, f_2)$ be graphs in $\mathcal{G}$ with $V_1 \cap V_2 = \emptyset$.

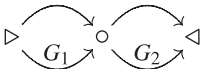
Parallel composition: $G_1 \cap G_2$ is the graph obtained by starting with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$, then identifying s_1 and s_2 (this is the new start), and identifying f_1 and f_2 (this is the new finish).

$G_1 \cap G_2$:



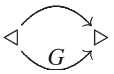
Sequential composition: $G_1; G_2$ (or just G_1G_2) is the graph obtained by starting with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$, then identifying f_1 with s_2, and defining the new start to be s_1 and the new finish to be f_2.

G_1G_2 :



Converse: G° is obtained from G by interchanging its start and finish. It is important to note that neither labels nor direction of edges changes.

G° :



Domain: The graph dom G is obtained from G by the finish node to be the same as the start node of G, which stays as the start node of dom G. It is important to note that neither the labels nor direction of edges changes.

dom G :

Definition 2. Let $\mathcal{D}$ denote the set of graphs generated by 1, the 2_i, and the operations of sequential and parallel composition, converse and domain. We call the elements of $\mathcal{D}$ *diagrams*.

Not every graph in $\mathcal{G}$ can be built using these operations: Theorem 31 characterizes the diagrams.

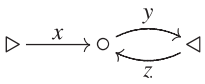
The set $\mathcal{D}$ carries a Σ-algebra structure via the operations defined above. Of course the set T_Σ of first-order terms comprises the free Σ-algebra over the set *Vars*. If we associate to the variable x_i the graph 2_i this extends naturally to the homomorphism

$$D : T_\Sigma \longrightarrow \mathcal{D},$$

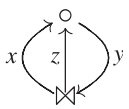
and we speak of $D(t)$ as being the diagram associated with the term t.

Examples 3. Here and below, we write lower case letters x, y, z, etc. to indicate variables, and will draw graphs labelled with these letters.

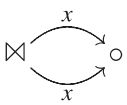
- $D(x(y \cap z^\circ))$:



- $D(1 \cap x(y \cap z^\circ))$:



- $D(1 \cap xx^\circ)$:



- $D(\mathrm{dom}\, x)$:

Note the last two examples: the terms $1 \cap xx^\circ$ and $\mathrm{dom}\, x$ have the same interpretation in all representable algebras, but their diagrams are different. This is important to the technical development below.

Definition 4. Let G and H be diagrams. Define $G \rightleftharpoons H$ if and only if there are morphisms $G \longrightarrow H$ and $H \longrightarrow G$.

The significance of the diagram representation is given in the following important theorem, from [1,16]. Recall that $\mathcal{R}$ denotes the class of algebras isomorphic to subpositive set relation algebras.

Theorem 5 (*Freyd-Scedrov, Andréka-Bredikhin*). *Let r, t be terms in the signature Σ. Then the equation $r = t$ is valid in $\mathcal{R}$ if and only if $D(r) \rightleftharpoons D(t)$.*

Proof. Whenever $\mathcal{A}$ is a subpositive set relation algebra with base set U and $R_1^U, \ldots, R_n^U$ are relations over U, we may define the graph $G_{\mathcal{A},\bar{R}}$ to have the set of vertices U and an edge (a, b) with label j for each $(a, b) \in R_j^U$. Observe that $G_{\mathcal{A},\bar{R}}$ is not necessarily in $\mathcal{D}$.

Now an easy induction on terms shows that for each term t in $T_\Sigma(R_1, \ldots, R_n)$ and elements $a, b \in U$ it holds $(a, b) \in t^{\mathcal{A}}[\bar{R}]$ if and only if there is a graph morphism $D(t) \longrightarrow G_{\mathcal{A},\bar{R}}$ which takes s to a and f to b.

Thus, if there is a morphism from $D(r)$ to $D(t)$ then $t \subseteq r$ is valid. This suffices to show one direction of the theorem. The converse follows from the observation that any diagram is itself a set relation algebra in an obvious way. □

Corollary 6.
- *$E_{\mathcal{R}}$ is decidable.*
- *If an equation is valid, it has the same variables on the left- and right-hand sides.*
- *If an equation is not valid, it fails in a finite model.*

The relationship between the term structure and the algebra $\mathcal{D}$ is not an isomorphism, even though we spoke of $\mathcal{D}$ as being the graphs "generated" by the Σ-operations on graphs. The reason is that certain identifications are implicit in the graph structure (for example, associativity of composition, or the equivalence of the terms $1 \cap xx^\circ$ and $\mathrm{dom}\, x$). Theorem 15 in Section 3 below clarifies this situation: $\mathcal{D}$ is the free algebra for a certain equational theory over Σ.

3. $\mathcal{D}$ as a free algebra

The algebra $\mathcal{D}$ of diagrams embodies certain equations in the sense that each graph in $\mathcal{D}$ can come from several different terms. In this section we show that these identifications can be captured by a finite set of equations.

The basic set of equations that hold are those that express associativity for composition, associativity and commutativity for $\cap$, and those governing $(\,)^\circ$. Certain other equations hold because diagrams with loops typically can be "read" in several ways. We will show that, somewhat surprisingly, with the addition of the equations in the set E_1 we will have captured all of the identifications between terms induced by the passage to diagrams. As for the equations that define dom, it turns out that dom can be completely characterized by the equations in Table 3.

We will see that these equations capture exactly the pairs of terms that are being identified when translated to diagrams.

Definition 7. The theory $E_{\mathcal{D}}$ is axiomatized by the set $E_s \cup E_1 \cup E_{\mathrm{dom}}$ of equations given in Tables 1–3.

The reader is invited to check that for each equation in $E_{\mathcal{D}}$ the left- and right-hand sides compile to the same diagram. That is, each of these equations is valid in $\mathcal{D}$. For example,

Table 1
The equations E_s

$$(xy)z = x(yz) \tag{1}$$

$$(x \cap y) \cap z = x \cap (y \cap z) \tag{2}$$

$$x \cap y = y \cap x \tag{3}$$

$$x^{\circ\circ} = x \tag{4}$$

$$(x \cap y)^\circ = x^\circ \cap y^\circ \tag{5}$$

$$(xy)^\circ = y^\circ x^\circ \tag{6}$$

Table 2
The equations E_1

$$x1 = x \tag{7}$$

$$1 \cap 1 = 1 \tag{8}$$

$$(1 \cap x)(1 \cap y) = 1 \cap (x \cap y) \tag{9}$$

$$x \cap y(1 \cap z) = (x \cap y)(1 \cap z) \tag{10}$$

$$1 \cap (x \cap y^\circ)z = 1 \cap x(y \cap z) \tag{11}$$

Table 3
The equations for dom

$$\mathrm{dom}\, 1 = 1 \tag{12}$$

$$(\mathrm{dom}\, x)^\circ = \mathrm{dom}\, x \tag{13}$$

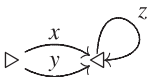
$$\mathrm{dom}((\mathrm{dom}\, x)y) = (\mathrm{dom}\, x) \cap (\mathrm{dom}\, y) \tag{14}$$

$$\mathrm{dom}(x(\mathrm{dom}\, y)) = \mathrm{dom}(xy) \tag{15}$$

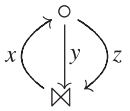
$$\mathrm{dom}((x \cap y)z) = 1 \cap x((\mathrm{dom}\, z)y^\circ) \tag{16}$$

$$x \cap y\, \mathrm{dom}\, z = (x \cap y)\mathrm{dom}\, z \tag{17}$$

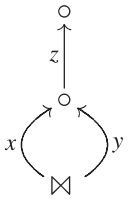
is the diagram for $(1 \cap x)(1 \cap y)$ as well the diagram for $(1 \cap x \cap y)$. See Eq. (9). Similarly,



represents both $x \cap y(1 \cap z)$ and $(x \cap y)(1 \cap z)$ (Eq. (10)). Finally,



captures Eq. (11), and the next captures Eq. (16):



These simple checks give the proof of the following lemma:

Lemma 8. *$(E_s \cup E_1 \cup E_{\mathrm{dom}})$ is a sub-theory of the theory of subpositive relational algebras.*

The main work of this section is to show that these equations capture all the identifications between terms induced by $\mathcal{D}$.

It might be thought that the equation $x \cap x = x$ should be part of the theory above. Certainly this is a valid equation, indeed a fundamental part of our intuition about relations (as sets). But recall that we are here characterizing the equations which yield *isomorphisms* between diagrams, not merely the validities.

The first step is to show that these equations allow us to impose a certain taxonomy on terms. The taxonomy is rather complex, but we will see that it captures all terms, up to $E_{\mathcal{D}}$-equivalence, and will be useful in future reasoning.

Definition 9. The set of terms $\hat{T}$ is the union of the following sets defined by mutual recursion:

- The set $\{1\}$.
- The set L of *literals*: $L = \{1\} \cup \mathit{Vars} \cup \{x^\circ : x \in \mathit{Vars}\}$.
- T_p, the set of *parallel terms*: a term $t \in T_p$ iff

  $$t = t_1 \cap \cdots \cap t_n$$

  with $n \geqslant 2$, and $t_i \in L \cup T_s$.
- T_{dom}, the set of *domain* terms: a term $t \in T_{\mathrm{dom}}$ iff

  $$t = \mathrm{dom}(ct')$$

  where $c \in L$ and $t' \in \hat{T}$.
- T_d, the set of *diagonal* terms: a term $t \in T_d$ iff

  $$t = 1 \cap t_1 \cap \cdots \cap t_n$$

  for $t_1 \in L \cup T_s$ and $t_i \in L \cup T_s \cup T_{\mathrm{dom}}$.
- T_s, the set of *sequential terms*: a term $t \in T_s$ iff

  $$t = t_1 \ldots t_n$$

  with $n \geqslant 2$, and $t_1, t_n \in L \cup T_p$ and the terms t_i $(1 < i < n)$ in $L \cup T_p \cup T_d \cup T_{\mathrm{dom}}$.

- T_{dt}, the set of terms of the form

  $$t_d t$$

  with $t_d \in T_d \cup T_{\mathrm{dom}}$ and $t \in L \cup T_s \cup T_p$.
- T_{td}, the set of terms of the form

  $$t t_d$$

  with $t_d \in T_d \cup T_{\mathrm{dom}}$ and $t \in L \cup T_s \cup T_p$.
- T_{dtd}, the set of terms of the form

  $$t_1 t t_2$$

  with $t_1, t_2 \in T_d \cup T_{\mathrm{dom}}$ and $t \in L \cup T_s \cup T_p$.

We wish to show that any term is equivalent modulo $E_{\mathcal{D}}$ to a term in $\hat{T}$. To do this it is useful to introduce a notion of *rewriting* on terms.

3.1. Rewriting for $E_{\mathcal{D}}$

Roughly speaking, we will orient the equations of $E_{\mathcal{D}}$ from left to right and rewrite terms under this reduction. In the proof of the taxonomy result Proposition 14 it will be convenient to be able to focus on the terms which are irreducible under this notion of reduction.

We must do a little more than simply orient the equations in $E_{\mathcal{D}}$, because of the symmetries inherent in the system. We need to account for the commutativity of $\cap$, certainly, and also for the fact that certain derivable equalities involving composition must be made explicit in the rewriting context.

Lemma 10. *The following equations are derivable in $E_{\mathcal{D}}$*

$$\begin{aligned} 1x &= x, \\ 1 &= 1^\circ, \\ 1 \cap x^\circ &= 1 \cap x, \\ 1 \cap z(x \cap y^\circ) &= 1 \cap (y \cap z)x, \\ x \cap (1 \cap z)y &= (1 \cap z)(x \cap y), \\ x \cap (\mathrm{dom}\, z)y &= (\mathrm{dom}\, z)(x \cap y). \end{aligned}$$

Proof. The fact that $1x = x$ holds is as follows: $(1x) = (1x)^{\circ\circ} = (x^\circ 1^\circ)^\circ = (x^\circ)^\circ = x$.

The second equation follows from a similar trick: write $x \cap (1 \cap z)y$ as $(x \cap ((1 \cap z)y))^{\circ\circ}$ and use Eq. (10).

For the third,

$$1 = (1^\circ)^\circ = (11^\circ)^\circ = 1^{\circ\circ}1^\circ = 11^\circ = 1^\circ.$$

For the fourth,

$$1 \cap x = 1 \cap (x \cap 1) = 1 \cap 1(x \cap 1) = 1 \cap (1 \cap x^\circ)1 = 1 \cap (1 \cap x^\circ) = 1 \cap x^\circ.$$

The fifth is proved as follows:

$$1 \cap z(x \cap y^\circ) = 1 \cap (z(x \cap y^\circ))^\circ = 1 \cap (x^\circ \cap y)z^\circ = 1 \cap x^\circ(y^\circ \cap z^\circ) = 1 \cap (z \cap y)x.$$

Finally for the last, use the second equation plus the fact that $\mathrm{dom}\, x = 1 \cap \mathrm{dom}\, x$ which follows from Eq. (16). □

Definition 11. Consider the rewrite system obtained from $E_{\mathcal{D}}$ as follows:
(1) Orient each equation in $E_{\mathcal{D}}$ from left to right, with the exception of (3) and (11);

(2) then add the rules
- $1x \longrightarrow x$,
- $1 \cap \mathrm{dom}\, x \longrightarrow \mathrm{dom}\, x$,
- $x \cap (1 \cap z)y \longrightarrow (1 \cap z)(x \cap y)$, and
- $x \cap (\mathrm{dom}\, z)y \longrightarrow (\mathrm{dom}\, z)(x \cap y)$;

(3) finally, for each rule involving $\cap$, add the obvious corresponding rules which arise due to the commutativity of $\cap$. Say that a term is *$E_{\mathcal{D}}$-irreducible* if it is irreducible in this system.

As an example of the rules added by the third construction above, the rule $(1 \cap x^\circ) \to (1 \cap x)$ gives rise to the additional rule $(x^\circ \cap 1) \to (x \cap 1)$. Similarly the rule $x \cap (\mathrm{dom}\, z)y \longrightarrow (\mathrm{dom}\, z)(x \cap y)$ induces $(\mathrm{dom}\, z)y \cap x \longrightarrow (\mathrm{dom}\, z)(y \cap x)$.

It is clear why we omit Eq. (3) in forming reduction rules since it would obviously induce a non-terminating reduction. Eq. (11) is more subtle: its corresponding reduction is omitted here simply because it would cause a problem with the proof technique and because we do not need it for Proposition 14.

Lemma 12. *The reduction system is terminating, that is, there are no infinite reduction sequences out of any terms.*

Proof. It suffices to exhibit a lexicographic path ordering [13,25] under which each rewrite rule is decreasing. We use the following order on the symbols of Σ:

$$\circ \succ \mathrm{dom} \succ \cap \succ ;$$

and use left-to-right priority in comparing subterms lexicographically. It is a straightforward verification that for each rule $l \to r$ from Definition 11 the term l is greater than the term r.

Corollary 13. *Any term is $E_{\mathcal{D}}$-equivalent to an $E_{\mathcal{D}}$-irreducible one.*

Proof. By Lemmas 10 and 12. □

Proposition 14 (*Standard forms for terms*). *If t is $E_{\mathcal{D}}$-irreducible then t is in $\hat{T}$.*

Proof. We will prove by induction on the length of t that the only irreducible terms are those in $\hat{T}$. Let t be given, $t \neq 1$, $t \notin L$. Modulo the associativity of intersection and composition, we may consider t to be in one the forms:

$$t = t_1 \cap \cdots \cap t_n \quad (n \geqslant 2) \ \text{ no } t_i \text{ is an intersection}$$

or

$$t = t_1 \cdots t_n \quad (n \geqslant 2) \ \text{ no } t_i \text{ is a composition}$$

or

$$t = \mathrm{dom}(t').$$

Consider the first case. Suppose first that no t_i is 1; we claim that in fact $t \in T_p$. For this we must show that each t_i is either in L or T_s. By the induction hypothesis each t_i is in $\hat{T}$; we have assumed that none is 1, or is an intersection, so it remains only to show that no t_i is in T_{dt}, T_{td}, or T_{dtd}. If, say, some t_i were in T_{dt} then t would be of the form $u \cap (1 \cap r)v$. But this term is not irreducible, a contradiction. The cases of some t_i in T_{td} or T_{dtd} are similar.

Next suppose that some $t = t_1 \cap \cdots \cap t_n$ and some t_i is 1. Then since t is irreducible exactly one of the t_i is 1. Without loss of generality, then, t is $1 \cap t_2 \cap \cdots \cap t_n$. We may use the same reasoning as in the previous paragraph to conclude that each t_i, $i \geqslant 2$ is in $L \cup T_s \cup T_{\mathrm{dom}}$, so that t itself is in T_d.

Now consider the case $t = t_1, \ldots, t_n$. In particular, no t_i is in $T_s \cup T_{dt} \cup T_{td} \cup T_{dtd}$, and clearly no t_i is 1. If neither t_1 nor t_n is in T_d then we are done: t is in T_s. So suppose for example that $t_1 \in T_d$, but $t_n \notin T_d$; we will show that t is in T_{dt} (the other two possibilities are argued similarly). We have $t = (1 \cap u)t_2 \cdots t_n$, $(n \geqslant 2)$ and we wish to show

that $w = t_2 \cdots t_n$ is in $L \cap T_s \cup T_p$. By induction, w is in $\hat{T}$; we need to exclude the possibilities that $w = 1$, $w \in T_d$, $w \in T_{dt}$, $w \in T_{td}$, or $w \in T_{dtd}$. The first is impossible by irreducibility, the last two are impossible since we assumed $t_n \notin T_d$. And if w were in T_d or T_{dt} then t_1 and t_2 would have been of the form $(1 \cap x_1)$ and $(1 \cap x_2)$, respectively. But $(1 \cap x_1)(1 \cap x_2)$ is reducible, a contradiction.

Finally, consider the case $t = \mathrm{dom}\, t'$. Because t is irreducible, clearly $t' \neq 1$. Now assume $t' = t_1 \cdots t_n$. If $t_1 \notin L$, then t_1 is parallel or a domain. In both cases t can be reduced using the rules coming from Eqs. (16) or (14), contradiction. Now assume $t' = t_1 \cap \cdots \cap t_n$. Using Eq. (14) with $z = 1$ it can be shown that t could be reduced, contradiction. If $t' = \mathrm{dom}(t'')$, using Eq. (16) it can be shown than t was not irreducible, contradiction. □

Theorem 15. *$\mathcal{D}$ is the free algebra over the set of labels for the set of equations $E_{\mathcal{D}}$. Indeed, for any terms r and t, $D(r)$ and $D(t)$ are isomorphic if and only if $E_{\mathcal{D}} \vdash r = t$.*

Proof. It is straightforward to verify that each of the equations in $E_{\mathcal{D}}$ is valid in $\mathcal{D}$ in the sense that for each equation the two sides compile to the same graph.

For the other direction we consider terms r and t such that $D(r)$ and $D(t)$ are isomorphic and we show that $E_{\mathcal{D}} \vdash r = t$. Since any term is $E_{\mathcal{D}}$-equivalent to any of its irreducible forms, we may suppose without loss of generality that r and t are both irreducible. The proof is by induction on the structure of t.

If $t = 1$ or $t \in L$ then it is clear that no other irreducible term compiles to the same graph; in other words r and t are the same term.

If t is a domain, then $t = \mathrm{dom}(ct')$ (recall that t is irreducible). Clearly r must be of the form $r = \mathrm{dom}(cr')$. Use induction hypothesis over t' and r'.

Suppose $t \in T_s$, say $t = t_1, \ldots, t_n$, $(n \geqslant 2)$. It is clear that r is of the form $r_1, \ldots, r_n$, with $D(t_1) \cong D(r_i)$ for each i. By induction, for each i we have $E_{\mathcal{D}} \vdash r_i = t_i$, so that $E_{\mathcal{D}} \vdash r = t$.

Very similar arguments apply to the cases when $t \in T_{dt}$, $t \in T_{td}$, and $t \in T_{dtd}$.

Suppose $t \in T_p$ say $t = t_1 \cap \cdots \cap t_n$, $(n \geqslant 2)$. Then r is of the form $r_1 \cap \cdots \cap r_{n'}$, $(n' \geqslant 1)$. If any of the t_i or r_j are literals then by irreducibility each occurs exactly once in t and in r. Isolating these literals, write $t = t_1 \cap \cdots \cap t_k \cap l_1 \cap \cdots \cap l_p$ and $r = r_1 \cap \cdots \cap r_{k'} \cap l_1 \cap \cdots \cap l_p \quad (k, k' \geqslant 0)$. Letting $t^* = t_1 \cap \cdots \cap t_k$ and $r^* = r_1 \cap \cdots \cap r_{k'}$, it suffices to show that $E_{\mathcal{D}} \vdash t^* = r^*$. We know that $D(t^*) \cong D(r^*)$. Any isomorphism must map traces to traces and so in fact $k = k'$ and there is a permutation π of indices such that $t_i \cong r_{\pi(i)}$ $(1 \leqslant i \leqslant k)$. By induction $E_{\mathcal{D}} \vdash t_i = r_{\pi(i)}$ $(1 \leqslant i \leqslant k)$ and the result follows.

Finally suppose $t \in T_d$. Certainly r must be in T_d as well. Furthermore, by removing the simple loops similarly as in the previous case we may restrict attention to the case $t = 1 \cap t_1 \cap \cdots \cap t_n$, $r = 1 \cap r_1 \cap \cdots \cap r'_n$, $\quad (n, n' \geqslant 1)$ with each t_i and r_j in $T_s \cup T_{\mathrm{dom}}$. The isomorphism between $D(t)$ and $D(r)$ enables us to conclude $n = n'$ and that there is a permutation π of indices with $D(1 \cap t_i) \cong D(1 \cap r_{\pi(i)})$, $(1 \leqslant i \leqslant n)$. By Eq. (9) it suffices to conclude from this that, for each i, $E_{\mathcal{D}} \vdash 1 \cap t_i = 1 \cap r_{\pi(i)}$.

If $t_i \in T_{\mathrm{dom}}$, then $D(1 \cap t_i) = D(t_i)$, and similarly for r_i, so apply induction hypothesis.

So assume that $t_i \in T_s$, i.e. $t_i = t'_1, \ldots, t'_{n'}$ with $n' \geqslant 2$, and $t'_1, t'_{n'} \in L \cup T_p$. If $t'_1 = a$ and $t'_{n'} = b$ are in L; it must be the case that $r_{\pi(i)} = ar'b$ or $r_{\pi(i)} = b^\circ r'' a^\circ$ for some r', r''. Then use Eq. (3.1). It remains to analyze the cases when t'_1 or t'_n are in T_p. There are two cases:

- $t_i = (u_1 \cap u_2)z$,
- $t_i = w(v_1 \cap v_2)$.

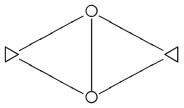
Eq. (11) and its symmetric counterpart deal precisely with these symmetries. This concludes the proof. □

4. Diagrams are characterized by omitting minors

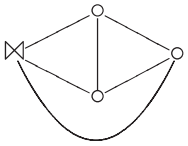
In this section we characterize the graphs which are diagrams, that is, those graphs which arise as $D(t)$ for some term t. A moment's thought makes it clear that changing the directions of the edges of a directed graph, or the labels, cannot affect whether it is a diagram. So in this section we are led to focus on the underlying pure graph structure of a directed graph.

The (undirected) pure graphs *Diamond* and *Diamond ring* are two important examples of pure graphs. The former graph is well-known in circuit theory.

Diamond



Diamond ring



An *elementary contraction* of the pure graph P is a pure graph obtained from P by
(1) removing two adjacent vertices u, v and all edges between them and adding a new vertex w, and
(2) for each edge between u and v with label e, adding one edge between w and w with label e; for each edge between z and u ($z \neq v$) with label e, add an edge between z and w with label e; for each edge between z and v ($z \neq u$) with label e, add an edge between z and w with label e;
(3) if either u or v is the start (resp., the finish) vertex of P then w becomes the start (resp., finish) of the new graph.
The graph H is a *contraction* of P if it is obtained from P by a sequence of elementary contractions. The graph H is a *minor* of P if H is a subgraph of a contraction of P.

A graph P *omits* graph H if H is not a minor of P.

The main result of this section, Theorem 31, is that a graph G is a diagram if and only if $U(G)$ omits both diamond and diamond ring. Observe that it is not obvious that the class of diagrams is even closed under subgraph.

Remark 16. Here we can see why the operator dom was added to our signature, even though it is definable (up to the relation $\rightleftharpoons$). Consider the graph

which is the diagram for $\mathrm{dom}\, x$. Comparing this graph with the graph for $(1 \cap xx^\circ)$ we see that they are related by $\rightleftharpoons$, reflecting the fact that semantically $\mathrm{dom}\, x$ is equal to $(1 \cap xx^\circ)$. But although the above graph certainly omits both diamond and diamond ring it is not the diagram of any term over the "standard" relational algebra signature.

It is remarkable that such a slight alteration of the relational algebra signature suffices to yield the omitting minors theorem. One may view this as an argument for the naturalness of the domain of a relation as a primitive concept.

Lemma 17. *Let H be a minor of P. If there exists a diagram D with $P = U(D)$ then there exists a diagram D' with $H = U(D')$.*

Proof. It suffices to prove this when H obtained from P by an elementary contraction. The argument is an easy induction over the structure of the diagram $D(t)$ whose underlying graph is P. □

Proposition 18. *Let $P = U(D)$ for some diagram D. Then P has neither diamond nor diamond ring as minor.*

Proof. As a consequence of Lemma 17, in order to show that no such graph contains diamond or diamond ring as a minor, it suffices to show that no such graph contains diamond or diamond ring as a *subgraph*.

By an induction over terms t one can show simultaneously that $U(D(t))$ does not contain diamond or diamond ring as subgraph. □

The converse is of course harder to prove. The essence of the proof is to show that when P omits diamond and diamond ring then P admits a decomposition into parallel and sequential components.

4.1. Decomposing and unfolding graphs

Definition 19. Let P be a pure graph and let u and v be vertices in P. Define $u \sim v$ if there exists a path between u and v not involving s or f.

It is clear that $\sim$ is an equivalence relation. An equivalence class of the $\sim$-relation over P will not contain s nor f, but these classes determine subgraphs of P once s and f are included, as follows.

Definition 20. Let P be a pure graph, let X be a set of vertices of P, and let x_0, x_1 be elements of X. This data generates a pure graph with vertices X, whose edges are the edges in P between vertices from X, with start vertex x_0 and finish vertex x_1. We denote this graph by $\langle X, x_0, x_1\rangle_P$.

Definition 21 (*parallel branches*). Let P be a graph with $s \neq f$. A *parallel branch* of P is either
- a subgraph 2_a of P, or
- a graph $\langle X \cup \{s, f\}, s, f\rangle_P$ when X is a $\sim$-equivalence class of P.

Lemma 22. *Suppose that neither s nor f separates P. Let* $P_1, \ldots, P_k$ *be all the parallel branches of P. If* $k \geqslant 2$ *then* $P = P_1 \cap \cdots \cap P_k$.

Proof. From the hypothesis, it follows that every edge of P either connects s and f or is incident upon some vertex not equal to s or f. Therefore, each edge lies in some parallel branch. It follows that the collection of graphs P_i comprises the vertices and edges of P, and that they have only s and/or f in common. The result will follow once we show that each of the components is connected.

Consider some $P_i = \langle X \cup \{s, f\}, s, f\rangle_P$, in the non-trivial case, when $X \neq \emptyset$. It suffices to show that each of s and f are adjacent to a vertex in X. Suppose s is not. Then since X is a connected component of $P - \{s, f\}$, it is in fact a connected component of $P - \{f\}$, which means that f disconnects the original graph P, a contradiction. By a similar argument, f must be adjacent to a vertex in X. □

The next notion is a "sequential" analogue of parallel branches.

Definition 23. Let w be a critical vertex of P. Then each of the connected components of $P - \{w\}$ gives rise to a graph as follows:
- Let X_s be the component of $P - \{w\}$ containing s. Then $\langle X_s \cup \{w\}, s, w\rangle_P$ is denoted P_s^w.
- Let X_f be the component of $P - \{w\}$ containing f. Then $\langle X_f \cup \{w\}, w, f\rangle_P$ is denoted P_f^w.
- Let $X_1, \ldots, X_k$ be the components of $P - \{w\}$ containing neither s nor f; and if there are loops on vertex w then let X_0 be $\emptyset$. Then for each such X_i, the graph $\langle X_i \cup \{w\}, w, w\rangle_P$ is denoted P_i^w.

Lemma 24. *Let w be a critical vertex of g. Then* $P = (P_s^w)(P_1^w)\cdots(P_k^w)(P_f^w)$.

Proof. As in Lemma 22 the work to be done is to show that each of the indicated graphs is connected. It is clear that each of the P_i^w is connected since their underlying vertex sets are connected components of a graph. We need to check that P_s^w and P_f^w are connected. To see that P_s^w is connected, for example, it suffices to see that w is adjacent to some vertex in X_s, which is the component of $P - \{w\}$ containing s. But since w is in the field of $\prec$ there is a path from s to w. The argument for P_f^w is similar. □

In many ways the graphs with $s \neq f$ are better-behaved than those with $s = f$. For graphs P of the latter type it will be convenient to identify a certain graph h such that P can be obtained as $1 \cap h$. In order to do this in a well-defined way it will be useful to assume that the set of vertices over which our graphs are defined is well-ordered.

Definition 25. Assume a well-ordering of the set of all graph-vertices. Let P be a non-trivial graph satisfying $s = f$. We define the graph *unfold*(P) as follows:
(1) If P is a domain graph, then $P = \mathrm{dom}(a\,P')$. Define *unfold*$((\,)P)$ as $a\mathrm{dom}(P')a^\circ$.

(2) Assume P as $1 \cap P_1 \cap \cdots \cap P_m$, where no P_i can be so decomposed; the definition is by induction on m.
- If $m \geqslant 2$ then $\mathit{unfold}(P) = \mathit{unfold}(P_1) \cap \cdots \cap \mathit{unfold}(P_m)$.
- If $m = 1$ then let $v_1, \ldots, v_n$ be the set of vertices not equal to s but adjacent to s in P, taken in order. If $n = 0$ then P must consist of a simple loop, and we define $\mathit{unfold}(P)$ to be the pure graph with edges $e_1, \ldots, e_k$ between the two vertices s and f.
  Otherwise $\mathit{unfold}(P)$ is obtained as follows. Add a new vertex, make it the new f (leave the old s) and redirect the edge between s and v_n in P to go between the new f and v_n.

It is not difficult to see that $P \cong 1 \cap \mathit{unfold}(P)$.

Proposition 26. *Let P satisfy $s = f$. If P omits diamond and diamond ring then $\mathit{unfold}(P)$ omits diamond and diamond ring.*

Proof. If $P = \mathrm{dom}(a P')$, it omits diamond ring if and only if P' does. Apply induction on the size of the graph.

Now, assume P is diagonal. Let $H = \mathit{unfold}(P)$. Letting x be the vertex v_n in the description of the unfolding procedure, we note that in H the (new) finish vertex is adjacent only to x.

Certainly, if diamond ring is a minor of H then it is a minor of P. To see that this holds of diamond as well, suppose for sake of contradiction that diamond is a minor of H. This means that in fact diamond must be a minor of the diagram obtained from H by the elementary contraction which collapses the finish vertex to x. In particular, this means that diamond is a minor of the original diagram P—more precisely, a graph with the same shape as diamond but with x in place of f is a minor of P. But now, since there are edges between x and s in P, if we perform the elementary contraction collapsing x and $s (= f)$ in P we can see that diamond ring is a minor in P, a contradiction. □

Caution. It is not the case that "$P = 1 \cap H$ and P omits diamond ring implies that H omits diamond." Just take diamond itself as H—then $1 \cap H$ will omit diamond ring. This is one reason for the care in defining the unfolding relation.

4.2. Coherence

Here we establish some fundamental facts about the order $\prec$ (defined in Section 2). In this section all graphs P have $s \neq f$.

When u and v are distinct vertices of P define $u \prec v$ if there is a trace in P containing u and v with u preceding v.

Not all vertices belong to traces in general, so we define the *field* of $\prec$ to be $\{u | u \text{ belongs to a trace}\}$. Then define $u \preccurlyeq v$ if either $u \prec v$ or $u = v$ and u is in the field of $\prec$.

The relation $\prec$ is not in general, a partial order: besides the fact that $u \preccurlyeq u$ can fail, there are graphs with distinct vertices u and v such that $u \prec v \prec u$ (the graph diamond is an example).

Say that a graph P is *coherent* if it is connected and if there do not exist u and v with $u \prec v$ and $v \prec u$.

Lemma 27. *A connected graph P omits diamond if and only if P is coherent.*

Proof. It is easy to see that if diamond is a minor of P, then P is not coherent. For the converse, suppose P is not coherent. Then for u, v we have traces $s \ldots u \ldots v \ldots f$ and $s \ldots v \ldots u \ldots f$. Note that $u \neq v$ and either u or v can be s or f. Using the four vertices s, u, v, f and the paths among them given by the traces, it is easy to see that diamond is in fact a subgraph of a contraction of P. □

Lemma 28. *Suppose P is coherent. Then $\preccurlyeq$ is a lattice order on its field.*

Proof. First let us establish that $\preccurlyeq$ defines a partial order in $V(P)$. For every vertex $v \preccurlyeq v$ holds by hypothesis—every vertex is incident to an edge because P is connected. For transitivity suppose that $v \preccurlyeq u$ and $u \preccurlyeq w$. Then there are traces $s \ldots v \ldots u \ldots f$ and $s \ldots u \ldots w \ldots f$ (where $\ldots$ indicates a path). Consider the walk $s \ldots v \ldots u \ldots w \ldots f$ obtained by concatenating the corresponding parts of the paths in the traces shown. We claim that this walk is a path, i.e., it has no repeated vertices. Suppose not; then some vertex z must occur twice in it, but because $s \ldots v \ldots u$ and $u \ldots w \ldots f$ are paths, it must occur once in $s \ldots v \ldots u$ and once in $u \ldots w \ldots f$, which is not possible because then

$u \prec z$ and $z \prec u$. Hence $v \preccurlyeq w$. As for antisymmetry, it follows immediately from the hypothesis: the only possibility for $v \preccurlyeq u$ and $u \preccurlyeq v$ is that $v = u$.

To see that $\preccurlyeq$ is indeed a lattice order, first note the presence of s and f ensure that lower bounds and upper bounds exist. It remains to show the existence of greatest lower bounds and least upper bounds. For the former, suppose we have two vertices u and v and w_1, w_2 with each $w_i \preccurlyeq u$, v and each w_i maximal with respect to this property. Suppose $w_1 \neq w_2$. Then consider the four traces $s \ldots w_1 \ldots u \ldots f, s \ldots w_1 \ldots v \ldots f, s \ldots w_2 \ldots u \ldots f$ and $s \ldots w_2 \ldots v \ldots f$ and check that from them you can deduce $u \prec v$ and $v \prec u$, which is a contradiction. Hence $w_1 = w_2$ and it is the greatest lower bound. The argument for least upper bounds is analogous. □

We let $u_\wedge v$ and $u_\vee v$ denote, respectively, the greatest lower bound and least upper bound of u and v. When we use this notation we are implicitly assuming that u and v are in the field of $\prec$.

Lemma 29. *Suppose P is coherent. If u and v satisfy $u_\wedge v = s$ and $u_\vee v = f$ then u and v are in different parallel branches.*

Proof. Suppose for sake of contradiction that there exists a path $\pi : u \twoheadrightarrow v$ containing neither s nor f. We claim that in this case we have $u \prec v$ and $v \prec u$, contradicting coherence. By symmetry it suffices to argue $u \prec v$. Consider the paths

$$\alpha_1 : s \twoheadrightarrow u; \quad \alpha_2 : u \twoheadrightarrow f; \quad \beta_1 : s \twoheadrightarrow v; \quad \beta_2 : v \twoheadrightarrow f,$$

where $\alpha_1\alpha_2$ and $\beta_1\beta_2$ each are traces.

For this it suffices to show that $\alpha_1\pi\beta_2$ is a trace. Suppose there were a vertex x in both α_1 and π. Then if we choose x to be $\prec$-minimal, then $x \prec v$ by virtue of the trace formed by portion of α_1 up to x, followed by the rest of π, followed by β_2 (this is a trace by the minimality of x). Certainly $x \prec u$, so we conclude that x is a lower bound for u and v. But x cannot be s since it is from π. This contradicts $u_\vee v = s$. A completely symmetric argument shows that $\pi \cap \beta_2 = \emptyset$. To see that $\alpha_1 \cap \beta_2 = \emptyset$, suppose for sake of contradiction that x is a vertex on each of these paths. This would imply $v \prec x$ and $x \prec u$ and thus $v \prec u$ contradicting coherence. □

Lemma 30. *Suppose P is coherent. Let u, v, and w satisfy $u_\wedge v = s$ and $u_\vee v = w \prec f$. Then any trace through u or v includes w.*

Proof. For sake of contradiction let τ be a trace through v which misses w. A contradiction to coherence will be established if we verify that $w \prec v$, since $v \prec w$ is given. There are paths

$$\alpha_1 : s \twoheadrightarrow u; \quad \alpha_2 : u \twoheadrightarrow w; \quad \beta_1 : s \twoheadrightarrow v; \quad \beta_2 : v \twoheadrightarrow w,$$

where $\alpha_1\alpha_2$ and $\beta_1\beta_2$ each are paths.

Let y be $\prec$-maximal among the vertices common to β_2 and τ, define β_2' to be that part of (the converse of) β_2 from w to y and define τ_2 to be the final segment of τ originating with y. It suffices to see that $\alpha_1\alpha_2\beta_2'\tau_2$ is a trace.

Certainly $\alpha_1 \cap \alpha_2 = \emptyset$. Also, $\alpha_1 \cap \beta_2'\tau_2 = \emptyset$ since if there were an x in common to these, we would have $v \prec x$ and $x \prec u$. We know that $\alpha_2 \cap \beta_2\tau_2 = \emptyset$ since if x were in common to these, it would be a smaller upper bound for u and v. Finally, $\tau_1 \cap \beta_2' = \emptyset$ by construction. □

4.3. The omitting-minors theorem

Theorem 31. *Let $G \in \mathcal{G}$ and let $P = U(G)$. Then G is a diagram if and only if P omits both diamond and diamond ring.*

Proof. The "only if" half of this is precisely Proposition 18, repeated here for emphasis.

The proof of the other direction is by induction on the number of vertices other than s and f in G. If this number is 0 then G is either a single point, perhaps with some loops, or consists of two points with some edges between them; in either case the result is clear.

For the induction step we first identify two cases: $s \neq f$ and $s = f$. We analyze the first case in detail and reduce the second case to the first.

When $s \neq f$: Consider the situation when vertex s separates the graph. Let C_f be the connected component of $P - \{s\}$ containing f and let S_0 be the union of the other connected components of $P - \{s\}$. Then $P = (\langle S_0, s, s\rangle_P)(\langle C_f, , s\rangle_f P)$, the induction hypothesis applies to each of the indicated components, and we are done.

If s does not separate the graph but f does then the analysis is similar.

So suppose neither s nor f separates P, and let P have parallel branches $P_1, \ldots P_k$; first suppose that $k \geqslant 2$: $P = P_1 \cap \cdots \cap P_k$ by Lemma 22. For each P_i which is not a simple loop on s or on f the induction hypothesis applies, and such a subgraph is a diagram. The remaining simple loops are of course diagrams, so P is.

We are left with the situation where there is a single parallel branch. Let $v_1, \ldots v_n$ be the vertices other than f which are adjacent to s ($k \geqslant 1$). Claim: *each* v_i *is in the field of* $\prec$. Otherwise, if some v_i were not, then all paths between f and v_i would involve s which means that s separates v_i from f, contrary to our assumption. By Lemma 29 the least upper bound w of the v_i is not f, and by Lemma 30 the vertex w is a critical vertex.

We now apply Lemma 24 and the induction hypothesis. This completes the argument in the case that $s \neq f$.

When $s = f$: Consider the graph *unfold*(P). Since P omits diamond ring, *unfold*(P) omits diamond. This graph has the same nodes other than s and f, and so the argument above applies, and any labelling of *unfold*(P) is a diagram. Since $P = 1 \cap$ *unfold*(P), any labelling of P is a diagram. □

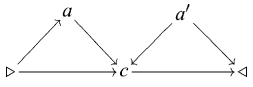
The following corollary is crucial to the soundness of the notion of reduction defined in Section 5.

Corollary 32. *If G is in $\mathcal{D}$ and G' is a connected subgraph of G containing s and f then G' is in $\mathcal{D}$. We may refer to G' as a* subdiagram *of G.*

If $\varphi : G \to H$ is a morphism of diagrams then the restriction of φ to a subdiagram of G is a morphism of diagrams, the image $\varphi[G]$ of φ is a diagram, and the inclusion of $\varphi[G]$ into H is a diagram morphism.

Proof. The fact that G' is a diagram follows from the fact that $U(G')$ omits diamond and diamond ring, since $U(G)$ does. In the same way, $\varphi[G]$ is a diagram; the other assertions concerning diagram-morphisms are immediate. □

It is worth observing that the class of diagrams is not closed under homomorphic images. Consider the following diagram (the labels do not matter).



The graph homomorphism which collapses a and a' results in a graph G whose underlying pure graph $U(G)$ is diamond.

5. Normalization for diagrams

In this section we define a notion of reduction on diagrams and prove that for each diagram we can compute a unique *normal form*. The techniques were introduced in [17] in a more general setting; to keep this paper self-contained we present the development specialized to our setting.

In this section, it will be convenient to refer to a diagram-morphism as being "injective" when its edge-function is injective, and similar for "surjective".

Recall the notation: $G \rightleftharpoons H$ if and only if there are morphisms $G \longrightarrow H$ and $H \longrightarrow G$. Theorem 5 highlighted the significance of this notion.

Clearly $\rightleftharpoons$ is an equivalence relation. Our goal is to prove that each diagram has a computable canonical $\rightleftharpoons$-representative.

Definition 33. Given diagrams G and H, define the relation $\Longrightarrow$ by: $G \Longrightarrow H$ if and only if there is a surjective ρ which is not an isomorphism and an injective μ such that $G \xrightarrow{\rho} H \xrightarrow{\mu} G$.

We denote the reflexive-transitive closure of $\Longrightarrow$ by $\overset{*}{\Longrightarrow}$, where reflexivity is defined up to isomorphism. Thus, $G \overset{*}{\Longrightarrow} H$ means either $G \cong H$ or there is a finite sequence $G \Longrightarrow c_1 \Longrightarrow \cdots \Longrightarrow c_n \Longrightarrow H$.

Corollary 32 plays a role here ensuring that subdiagrams of diagrams are diagrams themselves. So when G is a diagram any image of G under $\Longrightarrow$ is guaranteed to be a diagram.

It is clear that $G \overset{*}{\Longleftrightarrow} H$ implies that $G \rightleftharpoons H$. The main result of this section is a strong converse.

Note that there are no infinite $\Longrightarrow$-reductions out of any a.

Theorem 34 (*confluence for* $\rightleftharpoons$). *If* $G \rightleftharpoons H$, *then there exists* K *such that* $G \overset{*}{\Longrightarrow} K \overset{*}{\Longleftarrow} H$.

Proof. The proof is by noetherian induction over the multiset extension of $\Longrightarrow$. We are given $G \xrightarrow{\varphi} H \xrightarrow{\gamma} G$. If both φ and γ are injective then G and H are isomorphic and we may take K to be G. Otherwise, by symmetry we may suppose φ is not injective without loss of generality. Factor the morphism $\gamma \circ \varphi$ as $G \xrightarrow{\rho} X \overset{\mu}{\hookrightarrow} G$ with ρ surjective and μ injective. Now, ρ is not injective, otherwise $\gamma \circ \varphi$ would be, contradicting the assumption that φ is not injective. In particular ρ is not an isomorphism, and so $G \Longrightarrow X$. Since $X \rightleftharpoons H$ we may apply the induction hypothesis to $\{X, H\}$, obtaining K with $G \Longrightarrow X \overset{*}{\Longrightarrow} K \overset{*}{\Longleftarrow} H$ as desired. □

Thus, the relation $\Longrightarrow$ is a terminating and confluent abstract reduction system capturing the equivalence relation $\rightleftharpoons$:

Theorem 35 (*normal forms for diagrams*). *If* $G \rightleftharpoons H$ *in* $\mathcal{D}$ *then there is a* K, *unique up to isomorphism*, *such that* $G \Longrightarrow K$ *and* $H \Longrightarrow K$ *and* K *is* $\Longrightarrow$*-irreducible.*

For each graph G, *there is a graph* $\mathrm{nf}(G)$ *such that* $\mathrm{nf}(G)$ *is* $\Longrightarrow$*-irreducible and such that for any* H, $G \rightleftharpoons H$ *in* $\mathcal{D}$ *if and only if* $\mathrm{nf}(G) \cong \mathrm{nf}(H)$. *The graph* $\mathrm{nf}(G)$ *is unique up to isomorphism. The graph* $\mathrm{nf}(G)$ *is effectively computable from* G.

Proof.

(1) Since there are no infinite $\Longrightarrow$-reductions, we may let G' and H' be any $\Longrightarrow$-irreducible objects such that $G \overset{*}{\Longrightarrow} G'$ and $H \overset{*}{\Longrightarrow} H'$, respectively. Then $G' \rightleftharpoons H'$. But Proposition 34 and the irreducibility of G' and H' imply that G' and H' are isomorphic. We may take K to be G'.

(2) By the previous part, taking H to be G. The fact that $\mathrm{nf}(G)$ is effectively computable is a consequence of the fact that the reduction $\Longrightarrow$ itself is computable and that each application of $\Longrightarrow$ reduces the number of vertices in the graph. □

5.1. Application to $E_{\mathcal{R}}$

The reduction $\Longrightarrow$ induces a rewriting relation on terms, as follows.

$$t \xrightarrow{\mathcal{D}} u \quad \text{if } D(t) \Longrightarrow D(u).$$

Theorem 36 (*normal forms for terms*). *Given term* t *we can effectively compute a term* $\mathrm{nf}(t)$ *such that* $t \overset{\mathcal{D}}{\twoheadrightarrow} \mathrm{nf}(t)$ *and* $\mathrm{nf}(t)$ *is irreducible under* $\xrightarrow{\mathcal{D}}$. *Furthermore*, *for any term* u, *it holds* $t = u$ *in* $\mathcal{R}$ *if and only if* $D(\mathrm{nf}(t))$ *is isomorphic to* $D(\mathrm{nf}(u))$

Proof. In light of Theorem 35, we may take t_0 to be any term such that $D(t_0)$ is the normal form of $D(t)$ under $\Longrightarrow$. □

In the language of term-rewriting this result can be viewed as establishing the existence of unique normal forms for the theory $E_{\mathcal{R}}$ modulo the theory $E_{\mathcal{D}}$. The novelty of course is that the normal forms are graphs rather than terms, and so the notion of "modulo $E_{\mathcal{D}}$" is not based on equational provability but rather on the identity of diagram-forms for terms.

The normal form result gives another perspective on the known decidability result for the theory $E_{\mathcal{R}}$. Furthermore, the relation $\Longrightarrow$ can be said to constitute a computational interpretation of $E_{\mathcal{R}}$.

6. Conclusion

We have explored in some depth the technique of working with diagrams, graphical representations of relation-algebra terms. A given diagram can represent many terms, inducing an equivalence relation on terms; we have given a set of equations which characterizes this relation, so that the set of diagrams can be seen as a free algebra. We have characterized the set of graphs which arise as diagrams of terms by an omitting-minors theorem.

We examined the equational theory $E_{\mathcal{R}}$ of representable binary relations; we defined a notion of reduction of a diagram which yields a computational interpretation of this theory. The notion of reduction of diagrams is terminating and confluent, so we may compute unique normal forms for a term modulo the theory $E_{\mathcal{D}}$.

Acknowledgments

The authors are grateful to Jim Lipton for several illuminating discussions and to Roger Maddux for technical advice and encouragement. The diagrams were built using the XY-pic package from Kris Rose and Ross Moore.

References

[1] H. Andréka, D. Bredikhin, The equational theory of union-free algebras of relations, Algebra Universalis 33 (1995) 516–532.
[2] R.C. Backhouse, P.F. Hoogendijk, Elements of a relational theory of datatypes, in: B. Möller, H. Partsch, S. Schuman (Eds.), Formal Program Development, Lecture Notes in Computer Science, Vol. 755, Springer, Berlin, 1993, pp. 7–42.
[3] J. Barwise, G. Allwein (Eds.), Logical Reasoning with Diagrams, Oxford University Press, Oxford, 1996.
[4] R. Bird, O. de Moor, Algebra of Programming, Prentice-Hall, Englewood Cliffs, NJ, 1997.
[5] D.A. Bredikhin, The equational theory of relation algebras with positive operations, Izv. Vyssh. Uchebn. Zaved. Mat. 3 (1993) 23–30 (in Russian).
[6] C. Brink, W. Kahl, G. Schmidt (Eds.), Relational Methods in Computer Science, Advances in Computing, Springer, Wien, New York, 1997, ISBN 3-211-82971-7.
[7] P. Broome, J. Lipton, Combinatory logic programming: computing in relation calculi, in: M. Bruynooghe (Ed.), Logic Programming, MIT Press, Cambridge, MA, 1994.
[8] C. Brown, G. Hutton, Categories, allegories and circuit design, in: Logic in Computer Science, IEEE Computer Society Press, Silver Spring, MD, 1994, pp. 372–381.
[9] T. Brylawski, Series–parallel networks, Trans. Amer. Math. Soc. 154 (1971) 1–22.
[10] S. Curtis, G. Lowe, Proofs with graphs, Sci. Comput. Programming 26 (1996) 197–216.
[11] J.W. De Bakker, W.P. De Roever, A calculus for recursive program schemes, in: M. Nivat (Ed.), Automata, Languages and Programming, North-Holland, Amsterdam, 1973, pp. 167–196.
[12] A. De Morgan, On the syllogism no. IV, and on the logic of relations, Trans. Cambridge Philos. Soc. 10 (1860) 331–358.
[13] N. Dershowitz, Orderings for term-rewriting systems, Theoret. Comput. Sci. 17 (3) (1982) 279–301.
[14] G.A. Dirac, A property of 4-chromatic graphs and some remarks on critical graphs, J. London Math. Soc. 27 (1952) 85–92.
[15] R.J. Duffin, Topology of series–parallel networks, J. Math. Anal. Appl. 10 (1965) 303–318.
[16] P. Freyd, A. Scedrov, Categories and Allegories, North-Holland Mathematical Library, Vol. 39, North-Holland, Amsterdam, 1990.
[17] C. Gutiérrez, Normal forms for connectedness in categories, Ann. Pure Appl. Logic 108 (1–3) (2001) 237–247 (Special issue devoted to the XI Simposio Latinoamericano de Logica Matematica, Venezuela, July 1998).
[18] M. Haiman, Linear lattice proof theory: an overview, Universal Algebra and Lattice Theory, Lecture Notes in Mathematics, Vol. 1149, Springer, Berlin, 1985, pp. 129–141.
[19] M. Haiman, Proof theory for linear lattices, Adv. Math. 58 (3) (1985) 209–242.
[20] M. Haiman, Arguesian lattices which are not linear, Bull. Amer. Math. Soc. 16 (1987) 121–123.
[21] C.A.R. Hoare, An axiomatic basis for computer programming, CACM 12 (10) (1969) 576–583.
[22] G. Huet, Confluent reductions: abstract properties and applications to term rewriting systems, J. Assoc. Comput. Mach. 27 (4) (1980) 797–821.
[23] G. Hutton, E. Meijer, E. Voermans, A tool for relational programmers, Distributed on the mathematics of programming electronic mailing list, January 1994, ⟨http://www.cs.nott.ac.uk/∼gmh/allegories.gs⟩.
[24] W. Kahl, Relational matching for graphical calculi of relations, J. Inform. Sci. 119 (3–4) (1999) 253–273.
[25] S. Kamin, J.-J. Lévy, Two generalizations of the recursive path ordering, Unpublished note, Department of Computer Science, University of Illinois, Urbana, IL, February 1980.
[26] J. Lipton, E. Chapman, Some notes on logic programming with a relational machine, in: A. Jaoua, P. Kempf, G. Schmidt (Eds.), Relational Methods in Computer Science, Technical Report Nr. 1998-03, Fakultät für Informatik, Universität der Bundeswehr München, July 1998, pp. 1–34.

[27] R. Maddux, The origins of relation algebras in the development and axiomatization of the calculus of relations, Studia Logica (50) (1991) 421–455.
[28] R. Maddux, Relation-algebraic semantics, Theoret. Comput. Sci. 160 (1996) 1–85.
[29] J. Oxley, Graphs and series–parallel networks, in: N. White (Ed.), Theory of Matroids, Cambridge, 1986 (Chapter 6).
[30] C.S. Peirce, Collected Papers, Harvard University Press, Cambridge, MA, 1933.
[31] V. Pratt, Origins of the Calculus of Binary Relations, IEEE Computer Society Press, Silver Spring, MD, 1992 pp. 248–254.
[32] J.G. Sanderson, A Relational Theory of Computing, Lecture Notes in Computer Science, Vol. 82, Springer, Berlin, 1980.
[33] G.W. Schmidt, T. Ströhlein, Relations and Graphs: Discrete Mathematics for Computer Scientists, in: EATCS Monographs on Theoretical Computer Science, Springer, Berlin, 1993.
[34] E. Schröder, Vorlesungen über der Algebra der Logik, Vol. 3, "Algebra und Logik der Relative", Teubner, Stuttgart, 1895.
[35] A. Tarski, On the calculus of relations, J. Symbolic Logic 6 (3) (1941) 73–89.
[36] A. Tarski, S. Givant, A Formalization of Set Theory Without Variable, AMS Colloquium Publications, Vol. 41, American Mathematical Society, Providence, RI, 1988.