

# Positive Fork Graph Calculus\*

Renata de Freitas   Sheila R. M. Veloso   Paulo A. S. Veloso   Petrucio Viana  
IM/UFF                      DESC/UERJ                      COPPE/UFRJ                      IM/UFF

## Abstract

We introduce and illustrate a graph calculus for proving and deciding the positive identities and inclusions of fork algebras, i.e., those without occurrences of complementation. We show that this graph calculus is sound and complete. Moreover, the playful nature of this calculus renders it much more intuitive than its equational counterpart.

*Keywords:* Positive relational calculi, fork algebras, graph calculus, completeness, decidability

## 1 Introduction

In this paper we introduce and illustrate a graph calculus for deciding the positive identities and inclusions of fork algebras, i.e., those having no occurrences of complementation.

Fork algebras were introduced with the aim of providing a wide-spectrum formalism for program specification, verification and derivation [8]. One of the characteristics that make fork algebras adequate to this end is that one can formalize important operations on data that make easy the description of program behaviors. The fork operator is induced by a given injective coding function  $\bowtie$ , so that  $b\bowtie c$  can be regarded as an encoding of the ordered pair  $(b, c)$ . Given (binary) relations  $X$  and  $Y$  on a base set  $U$ , by applying fork to  $X$  and  $Y$  we obtain the relation  $\{(a, b\bowtie c) : (a, b) \in X \text{ and } (a, c) \in Y\}$ .

Relation algebras are also appropriate to formalize some aspects of program methodology [15]. One of the motivations for enriching this formalism is capturing important notions linked to *storing* and *retrieving* of data, which is beyond the range of the relational algebraic apparatus. For instance, a pair  $(a, b)$  belongs to the relation obtained by iterated application of fork to relations  $X_1, X_2, X_3, X_4$  on  $U$ , iff there are elements  $b_1, b_2, b_3, b_4 \in U$  such that  $b = b_1\bowtie(b_2\bowtie(b_3\bowtie b_4))$  and  $(a, b_1) \in X_1, (a, b_2) \in X_2, (a, b_3) \in X_3, (a, b_4) \in X_4$ , i.e.,  $b$  may be viewed as *storing* the coded result of the parallel application of relations  $X_1, X_2, X_3, X_4$  to  $a$ . Besides storing data, fork can also be used to define projection operators to *retrieve* the data stored. For instance, one can define a new constant operator  $\pi_3^4$  that outputs the third coordinate of a quadruple, i.e., a pair  $(a, b)$  belongs to the relation  $\pi_3^4$  iff there are elements  $a_1, a_2, a_3, a_4 \in U$  such that  $a$  is the encoding  $a_1\bowtie(a_2\bowtie(a_3\bowtie a_4))$  and  $b$  is  $a_3$ .

Besides manipulation of data, fork algebras were shown to be appropriate to express other programming ideas, such as input-output specification of programs, including refinement and abstraction; program behavior, including non-determinism and parallelism; program design strategies, including case analysis and divide-and-conquer, etc. [8].

Another characteristics of the fork algebraic apparatus making it an adequate framework for formal program construction is that it provides algebraic proofs of interesting program properties. By this we mean that fork algebras are provided with an algebraic language where properties of program (schemata) can be expressed as equations and inferred from other equationally specified properties merely by replacing equals by equals.

Abstractly, relation algebras are defined by a set of identities specifying that the operators  $\sqcup$  (union),  $\sqcap$  (intersection),  $-$  (complement),  $\mathbf{E}$  (universal relation) and  $\mathbf{O}$  (empty relation) behave

---

\*Research partially sponsored by CNPq, FAPERJ and FAPESP.

as in Boolean algebras;  $\circ$  (composition),  $\top$  (reversion), and  $\mathbb{I}$  (identity relation) behave as in involuted monoid theory; together with another identity expressing a geometric aspect of the interaction of the operators  $\top$ ,  $\sqcap$ , and  $\circ$  [9, 14]. Fork algebras may be defined by extending relation algebras with a new operator  $\nabla$  (fork) and the following three identities:

- (a)  $(\mathbb{I} \nabla \mathbb{E})^\top \nabla (\mathbb{E} \nabla \mathbb{I})^\top \sqsubseteq \mathbb{I}$ ,
- (b)  $(r \nabla s) \circ (t \nabla q)^\top \approx (r \circ t^\top) \sqcap (s \circ q^\top)$ ,
- (c)  $[r \circ (\mathbb{I} \nabla \mathbb{E})] \sqcap (s \circ (\mathbb{E} \nabla \mathbb{I})) \approx r \nabla s$ .

From the above three axioms one can prove that fork is an *additive normal* operator.<sup>1</sup>

Although these identities are easily established, this is not the case in general: the equational theories of relation and fork algebras are rather complex [16]. Indeed, some non-algebraic mechanisms have been put forward to cope with the problem of establishing identities involving relations [4, 13]. This paper is a contribution in this line of development.

As a first step to overcome the difficulties mentioned above, we will introduce a formal calculus whose formulas are graphs (defining relations) and whose rules are used to derive graphs from graphs. We will prove that the calculus with graphs is sound, complete and decidable for graph inclusions. The graphical system can be directly applied to the positive fork algebraic inclusions and identities. Positive fork terms correspond to graphs and the graphical apparatus can be used to decide the equalities and identities in a playful manner.

In Section 2 we describe the positive fork calculus +FC, its formal syntax and semantics, as well as some examples of equations that are identically true. In Section 3 we introduce the positive fork calculus with graphs +FG. It is built as an extension of the positive fork language hinging on an adequate notion of a graph labeled with fork terms. We also introduce inference rules to transform graphs into graphs illustrating their application. In Section 4 we establish the central metamathematical results of soundness and completeness as well as decidability. These results can be transferred directly to the the positive fork language. Section 5 closes the paper with some remarks and directions for future work.

## 2 Positive fork language: syntax and semantics

The positive fork relational language +FL is a variant of the positive relational language treated in [5, 6] by restricting semantics to structured models (cf. below) and introducing a new binary operator  $\nabla$  on relations, called *fork* [8].

The +FL *terms*, typically noted  $R, S, T$ , are generated from the set of relational variables  $\text{RVAR} = \{r_i : i \in \omega\}$  by applying the relational operators  $\mathbb{E}, \mathbb{I}, \top, \sqcap, \sqcup, \circ$ , and  $\nabla$ , according to the grammar  $R ::= r_i \mid \mathbb{E} \mid \mathbb{I} \mid R^\top \mid R \sqcap S \mid R \sqcup S \mid R \circ S \mid R \nabla S$ . The *positive fork relational inclusions* and *equalities* are the expressions of the forms  $R \sqsubseteq S$  and  $R = S$ , respectively.

A *structured universe* is a pair  $(M, \boxtimes)$ , where  $M \neq \emptyset$  and  $\boxtimes : M \times M \rightarrow M$  is an injective operation. Intuitively,  $a \boxtimes b \in M$  codifies the pair  $(a, b)$  of elements of  $M$ . The *fork* induced by  $\boxtimes$  in a structured universe  $(M, \boxtimes)$  is the binary operation  $\nabla_{\boxtimes}$  on relations on  $M$  given by  $R \nabla_{\boxtimes} S := \{(a, b \boxtimes c) \in M \times M : aRb \text{ and } aSc\}$ .

A *structured model* is a triple  $\mathfrak{M} = (M, \boxtimes, r_i^{\mathfrak{M}})_{i \in \omega}$ , where  $(M, \boxtimes)$  is a structured universe and  $r_i^{\mathfrak{M}} \subseteq M \times M$  for every  $i \in \omega$ . The *meaning*  $\llbracket R \rrbracket_{\mathfrak{M}}$  of a term  $R$  in a structured model  $\mathfrak{M}$  is defined similarly to the relational case (excluding all references to the empty relation and to complementation) together with an additional clause to treat the fork operator. Formally, given a structure model  $\mathfrak{M} = (M, \boxtimes, r_i^{\mathfrak{M}})_{i \in \omega}$ , symbols  $\mathbb{E}$  and  $\mathbb{I}$  are interpreted, respectively, as the relations  $M \times M$  and  $\{(a, b) \in M \times M : a = b\}$ ; symbols  $\sqcap, \sqcup, \top$  and  $\circ$  as  $\cap, \cup$ , conversion and composition of relations, respectively; and the meaning of a fork term  $R \nabla S$  is defined by  $\llbracket R \nabla S \rrbracket_{\mathfrak{M}} ::= \llbracket R \rrbracket_{\mathfrak{M}} \nabla_{\boxtimes} \llbracket S \rrbracket_{\mathfrak{M}}$ .

<sup>1</sup>This means that  $r \nabla \mathbb{O} \approx \mathbb{O} \approx \mathbb{O} \nabla r$ ,  $r \nabla (s \sqcup t) \approx (r \nabla s) \sqcup (r \nabla t)$  and  $(s \sqcup t) \nabla r \approx (s \nabla r) \sqcup (t \nabla r)$ .

Validity of inclusions and equalities are defined as usual. As expected, given terms  $R, S$  and model  $\mathfrak{M}$ ,  $R = S$  is valid in  $\mathfrak{M}$  iff  $R \sqsubseteq S$  and  $S \sqsubseteq R$  are both valid in  $\mathfrak{M}$ , and  $\Vdash R = S$  iff  $\Vdash R \sqsubseteq S$  and  $\Vdash S \sqsubseteq R$ . As examples of valid formulas we have the three formulas (a), (b) and (c) taken as axioms for fork algebras in Section 1. In the sequel, we will introduce the positive fork graph calculus to prove valid inclusions such as these.

### 3 Positive fork graph calculus: syntax, semantics and rules

In the positive graph relational calculus +RG [5, 6], relations are represented by (directed pseudo multi) graphs having two distinguished nodes and arcs labeled by positive relational terms. To define the positive fork graph calculus +FG, we shall label arcs with positive fork relational terms and the graphs will represent binary relations on structured universes. We would like to distinguish ordinary nodes from those in the range of a star function.

We consider a fixed set  $\text{INOD} = \{x_n : n \in \omega\}$  of *nodes*, typically denoted by  $x, y, z, u, v, w$ . Given set  $N \subseteq \text{INOD}$ , a *node equation* on  $N$  is a triple  $(u, v, w)$ , noted  $u \star v \rightarrow w$ , with  $u, v, w \in N$ ; here  $w$  is the *star* of  $u$  and  $v$ , which are, respectively, *the first* and *the second components* of  $w$ . A *table* on  $N$  is a set  $T$  of node equations on  $N$ . With respect to a table  $T$ , we call a node  $w$  *structured* iff there is some node equation  $u \star v \rightarrow w$  in  $T$ , otherwise, we call it *atomic*.

Graphically, we represent nodes by dots and node equations by two-source arrows pointing to the structured node and displaying its first and second components: a single line indicates the first component whereas a double line indicates the second one. For instance, node equation  $u \star v \rightarrow w$ , where  $u, v, w$  are pairwise distinct, is represented in Figure 1.

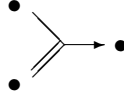


Figure 1: Graphical representation of node equation  $u \star v \rightarrow w$

An *arc* is a triple  $(u, R, v)$ , denoted  $uRv$ , where  $u, v$  are nodes, and  $R$  is a +FL term. Graphically, we represent arcs by labeled arrows linking nodes. For instance, arcs  $ur_1v, ur_2v, vr_3w, zr_4w, zr_5z$  are represented in Figure 2.

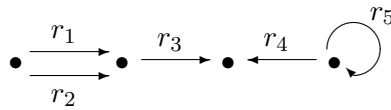


Figure 2: Graphical representation of arcs

A *slice* is a structure  $S = (N, T, A, x, y)$ , where  $N$  is a non-empty set of nodes,  $T$  is a table on  $N$ ,  $A \subseteq N \times \text{Trm} \times N$  is a set of labeled *arcs* ( $\text{Trm}$  is the set of +FL terms), and  $x, y$  are (not necessarily distinct) nodes in  $N$ , called *input* and *output*, respectively. Graphically, we represent slices by directed arc-labeled pseudo multi graphs with distinguished nodes  $x, y$  represented by  $-, +$ , respectively. For instance, the slice  $S = (\{x, u, v, w, y\}, \{u \star v \rightarrow w\}, \{xru, x(s \sqcap \text{I})v, w(\text{E} \circ t)y\}, x, y)$ , having a structured node, is represented in Figure 3.

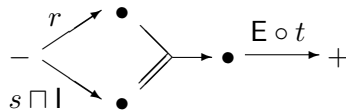


Figure 3: Slice  $S$  with a structured node

A *positive fork graph*, or simply a *graph*, typically denoted by  $G, H$ , is a finite non-empty set of slices. A graph can be represented by the juxtaposition of the representation of its slices. The  $+FG$  *inclusions* and *equalities* are expressions of the forms  $G \sqsubseteq H$  and  $G = H$ , respectively.

We now examine semantics: slices and graphs denote binary relations. First, given a slice  $S = (N, T, A, x, y)$  and a structured model  $\mathfrak{M}$  with universe  $M$ , an  $\mathfrak{M}$ -*assignment* for  $S$  is a function  $g : N \rightarrow M$  such that  $(gu, gv) \in \llbracket R \rrbracket_{\mathfrak{M}}$ , for every arc  $uRv$  in  $A$ , and  $gu \star gv = gw$ , for every node equation  $u \star v \mapsto w$  in  $T$ . Now, the *meaning* of a slice  $S$  in a model  $\mathfrak{M}$  is the subset  $\llbracket S \rrbracket_{\mathfrak{M}}$  of  $M \times M$  defined by  $(a, b) \in \llbracket S \rrbracket_{\mathfrak{M}}$  iff  $gx = a, gy = b$ , for some  $\mathfrak{M}$ -assignment  $g$  for  $S$ . The *meaning* of a graph  $G$  in a model  $\mathfrak{M}$  is the union of the meanings of its slices.

A  $+FG$  inclusion  $G \sqsubseteq H$  *holds* in a model  $\mathfrak{M}$ , denoted  $\mathfrak{M} \models G \sqsubseteq H$ , iff  $\llbracket G \rrbracket_{\mathfrak{M}} \subseteq \llbracket H \rrbracket_{\mathfrak{M}}$ . It is *valid*, denoted  $\models G \sqsubseteq H$ , iff it holds in every model. Analogously, we can define truth and validity for  $+FG$  equalities and prove results similar to those described in Section 2 for  $+FL$  inclusions and equalities. In particular, graphs  $G$  and  $H$  are called *equivalent* iff  $\models G = H$ .

The deductive apparatus of  $+FG$  is given by a set of graph transforming rules: some rules transform a graph into an equivalent one (and they will be used to put a graph in normal form), while another rule will be used to compare graphs in normal form. To state the transformation rules, we use ‘+’ for insertion of an element into a set and  $S_1 \mid S_2$  for a set of slices of a graph. We will also use the *node substitution* notation  $\frac{u}{v}$  for replacing  $u$  by  $v$ , which we extend naturally to pairs and triples as well as sets; e.g., for a set  $A$  of arcs, we put  $A \frac{u}{v} := \{w \frac{u}{v} R z \frac{u}{v} : wRz \in A\}$ .

The *transformation rules* are given in Tables 1, 2 and 3. Rules in Table 1(a) cover the relational part of the fork graphs and the rule in Table 1(b) covers similarly the fork operator. The star rules in Table 2 concern the graph tables. The rules in these three tables can be applied in both directions. In fact, each one of these rules is an abbreviation for two rules: downward and upward. The rules in Table 1 allow the *elimination* (downwards) and the *introduction* (upwards) of the operators. Table 3 presents the capital rule for comparing graphs.

---


$$\begin{array}{l}
\text{Unv} \quad \frac{N, T, A + uEv, x, y}{N, T, A, x, y} \qquad \text{Idn} \quad \frac{N, T, A + ulv, x, y}{N \frac{u}{v}, T \frac{u}{v}, A \frac{u}{v}, x \frac{u}{v}, y \frac{u}{v}} \\
\text{Cnv} \quad \frac{N, T, A + uR^T v, x, y}{N, T, A + vRu, x, y} \qquad \text{Int} \quad \frac{N, T, A + uR \sqcap Sv, x, y}{N, T, A + uRv + uSv, x, y} \\
\text{Uni} \quad \frac{N, T, A + uR \sqcup Sv, x, y}{(N, T, A + uRv, x, y) \mid (N, T, A + uSv, x, y)} \\
\text{Cmp} \quad \frac{N, T, A + uR \circ Sv, x, y}{N + w, T, A + uRw + wSv, x, y} \quad \text{if } w \notin N
\end{array}$$

(a) Relational rules

---


$$\text{Frk} \quad \frac{N, T, A + uR \nabla Sv, x, y}{N + v_1 + v_2, T + v_1 \star v_2 \mapsto v, A + uRv_1 + uSv_2, x, y} \quad \text{if } v_1, v_2 \notin N$$

(b) Fork rule

---

Table 1: Elimination/Introduction rules for transforming graphs

We will explain each bidirectional rule in the downward direction. Each rule in Tables 1 and 2 states that the meaning of graph does not change when applying the local transformation specified in the rule, leaving the rest of graph untouched. Soundness will follow from the explanations.

Rules in Table 1(a) concern the relational part of the fork graphs, being similar to those of +RG [6]. Rule **Unv** allows erasing an arc labeled by **E** from a slice. Rule **ldn** allows one to erase an arc  $ulv$  and a node  $u$ , renaming nodes and redirecting arcs accordingly. Rule **Cnv** allows replacing an arc  $uR^\top v$  by  $vRu$ . Rule **Int** allows one to replace an arc  $uR \sqcap Sv$  by two others  $uRv$  and  $uSv$ .

Rule **Uni** allows one to replace a slice  $C$  having an arc  $uR \sqcup Sv$ , by two other slices  $C_R$  and  $C_S$ , obtained from  $C$  by replacing the arc  $uR \sqcup Sv$  by a new arc:  $uRv$  for  $C_R$  and  $uSv$  for  $C_S$ .

Rule **Cmp** allows one to replace an arc  $uR \circ Sv$  by two others,  $uRw$  and  $uSv$ , with a new node  $w$ .

Table 1(b) presents the **Frk** rule concerning the fork operator: it allows one to replace an arc  $uR \nabla Sv$  by two other arcs  $uRv_1$  and  $uSv_2$ , with two new nodes  $v_1$  and  $v_2$ .

---

Tot	$\frac{N, T, A, x, y}{N + w, T + u \star v \mapsto w, A, x, y} \quad \text{if } w \notin N$
Fnc	$\frac{N, T + u \star v \mapsto w_1 + u \star v \mapsto w_2, A, x, y}{N \frac{w_1}{w_2}, (T + u \star v \mapsto w_1) \frac{w_1}{w_2}, A \frac{w_1}{w_2}, x \frac{w_1}{w_2}, y \frac{w_1}{w_2}}$
Inj	$\frac{N, T + u_1 \star v_1 \mapsto w + u_2 \star v_2 \mapsto w, A, x, y}{(N \frac{u_1}{u_2}) \frac{v_1}{v_2}, [(T + u_1 \star v_1 \mapsto w) \frac{u_1}{u_2}] \frac{v_1}{v_2}, (A \frac{u_1}{u_2}) \frac{v_1}{v_2}, (x \frac{u_1}{u_2}) \frac{v_1}{v_2}, (y \frac{u_1}{u_2}) \frac{v_1}{v_2}}$

---

Table 2: Star rules for transforming graphs

Table 2 presents the rules concerning graph tables. Rule **Tot** allows one to add a new node as the star of given nodes. Rule **Fnc** allows one to identify nodes that are the star of the same pair of nodes. Rule **Inj** allows one to identify nodes that are the first and the second components, respectively, of a node. Each one of these rules is sound since  $\star$  is a total, injective function.

Table 3 presents the capital rule **Hom**: it allows one to infer a graph from one covered under homomorphism. The notions involved are natural extensions of the +RG case [6].

---

Hom	$\frac{G}{H} \quad \text{if } G \leftarrow H$
-----	---

---

Table 3: Homomorphism rule **Hom** for transforming graphs.

Given slices  $S = (N, T, A, x, y)$  and  $S' = (N', T', A', x', y')$ , a *homomorphism* from  $S'$  to  $S$  (noted  $\theta : S' \rightarrow S$ ) is a function  $\theta : N' \rightarrow N$  that preserves the slice structure in the sense:  $\theta w \star \theta u \mapsto \theta v \in T$ , for every node equation  $w \star u \mapsto v$  in  $T'$ ;  $\theta u R \theta v \in A$ , for every arc  $u R v$  in  $A'$ ;  $\theta x' = x$  and  $\theta y' = y$ . Given graphs  $G$  and  $H$ , we say that  $H$  *covers*  $G$  (noted  $G \leftarrow H$ ) iff, for each slice  $S$  of  $G$ , there is a slice  $S'$  of  $H$  and a homomorphism  $\theta : S' \rightarrow S$ .

Rule **Hom** can be applied only downwards, but a special case of **Hom** can be applied in both directions, namely, the derived rule **ErUn** for erasing useless nodes presented in Table 4. A node is *useless* in a slice iff it is not distinguished and does not occur in its arcs nor in its table.

$$\text{ErUn} \quad \frac{N + w, T, A, x, y}{N, T, A, x, y} \quad \text{if } w \text{ is useless}$$

Table 4: Derived rule **ErUn** for erasing useless nodes.

An inclusion  $G \sqsubseteq H$  is a **+FG theorem**, denoted  $\vdash G \sqsubseteq H$ , iff  $H$  can be obtained from  $G$  by applications of the inference rules. We will call graphs  $G$  and  $H$  *provably equivalent* iff  $\vdash G \sqsubseteq H$  and  $\vdash H \sqsubseteq G$ . We call a proof *normal* iff it consists of applications of elimination, star and **ErUn** rules, followed by a single application of the **Hom** rule, followed by applications of introduction, star and **ErUn** rules. To illustrate the system in action, we show how to prove the fork algebra axioms mentioned in Sections 1 and 2. We associate to a fork relational term  $R$  its graph  $G_R := \{(\{x, y\}, \emptyset, \{xRy\}, x, y)\}$ . Then, it is clear that  $\llbracket G_R \rrbracket_{\mathfrak{M}} = \llbracket R \rrbracket_{\mathfrak{M}}$ , for every model  $\mathfrak{M}$ . So, we can reduce inclusions between fork relational terms to inclusions between their associated graphs:  $\models R \sqsubseteq S$  iff  $\vdash G_R \sqsubseteq G_S$ . Graph proofs of the **+FG** formulas associated to (a), (b) and (c) are indicated in Figures 4, 5 and 6, respectively.

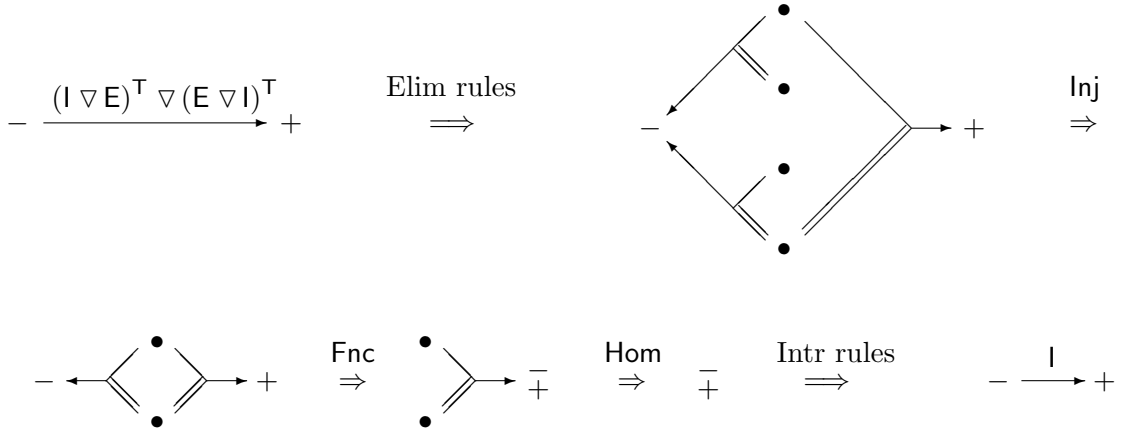


Figure 4: Graph proof of fork axiom (a)

## 4 Metamathematics of the fork graph calculus

In this section, we prove soundness, completeness and decidability of the positive fork graph calculus with respect to the **+FG** valid inclusions. The approach presented here is a substantial extension of the one given in [5], for **+RG**.

Soundness of **+FG** is an immediate consequence of the Lemma 4.1.

**Lemma 4.1** *Consider graphs  $G$  and  $H$ . If  $H$  is obtained from  $G$  by applications of the rules in Tables 1, 2 and 4, then  $\models G = H$ . If  $H$  is obtained from  $G$  by the rule **Hom** then  $\models G \sqsubseteq H$ .*

For the remaining results, we will define a normal form for graphs and show that inclusion of graphs can be reduced to inclusion of their normal forms.

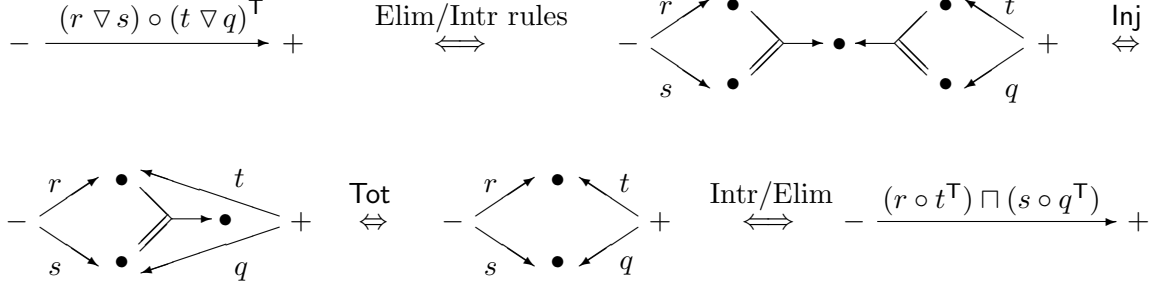


Figure 5: Graph proof of fork axiom (b)

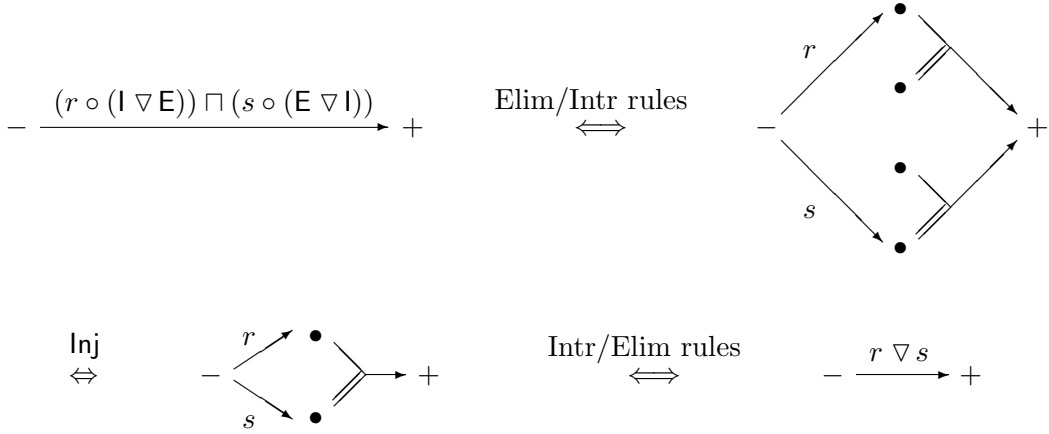


Figure 6: Graph proof of fork axiom (c)

A table  $T$  induces a relation  $\star_T$  on  $(N \times N) \times N$  such that  $(u, v) \star_T w$  iff  $u \star v \mapsto w \in T$ . We call a table *functional* or *injective* iff the induced relation is functional or injective, respectively.

A *star-path* in a slice is a sequence  $(u_1, v_1, \dots, u_n, v_n, w)$  of nodes such that  $u_n \star v_n \mapsto w \in T$  and for each  $i, i \leq 1 \dots \leq n-1$ ,  $u_i \star v_i \mapsto u_{i+1} \in T$  or  $u_i \star v_i \mapsto v_{i+1} \in T$ . A *star-cycle* in a slice is a sequence  $(u_1, v_1, \dots, u_n, v_n)$  of nodes such that  $u_n \star v_n \mapsto u_1 \in T$  or  $u_n \star v_n \mapsto v_1 \in T$ , and for each  $i, i \leq 1 \dots \leq n-1$ ,  $u_i \star v_i \mapsto u_{i+1} \in T$  or  $u_i \star v_i \mapsto v_{i+1} \in T$ .

Now, we introduce the notion of essential node, which will play a central role in the definition of normal form for graphs. A node  $v$  is *essential* in a slice  $S$  if it is  $n$ -essential in  $S$ , for some  $n \in \mathbb{N}$ . A node  $v$  is *0-essential* in a slice  $S$  if  $v$  is distinguished in  $S$ , or  $v$  is an extreme of an arc in  $S$ , or  $v$  is an element of a star-cycle in  $S$ . A node  $v$  is  $n$ -essential in a slice  $S$ , for  $n > 0$ , if there is a star-path  $(u_1, v_1, \dots, u_n, v_n, w)$  in  $S$  such that  $v$  is not  $m$ -essential in  $S$ , for  $m < n$ ,  $w$  is 0-essential in  $S$ , and  $v = u_1$  or  $v = v_1$ .

For instance, consider the slice  $S = (N, T, A, x, y)$ , with  $N = \{a, b, c, d, e, f, g, h, j, i, k, l, x, y\}$ ,  $T = \{a \star b \mapsto e, b \star c \mapsto y, d \star e \mapsto g, y \star f \mapsto h, k \star l \mapsto k\}$ ,  $A = \{gri\}$  (cf. Figure 7). Here, nodes  $x, y$  are 0-essential, as they are distinguished; nodes  $k, l$  are 0-essential, as  $(k, l)$  is a star-cycle; nodes  $g, i$  are 0-essential, as they are the extremes of arc  $gri$ ; nodes  $d, e$  are 1-essential, as  $(d, e, g)$  is a star-path,  $g$  is 0-essential, and  $d, e$  are not 0-essential; nodes  $b, c$  are 1-essential, as  $(b, c, y)$  is a star-path,  $y$  is 0-essential, and  $b, c$  are not 0-essential; node  $a$  is 2-essential, as  $(a, b, d, e, g)$  is a star-path,  $g$  is 0-essential, and  $a$  is not 1-essential nor 0-essential; nodes  $f, h, j$  are non-essential.

We call a graph  $G$  *basic* iff every arc in  $G$  is labeled by a relational variable, *functionally*

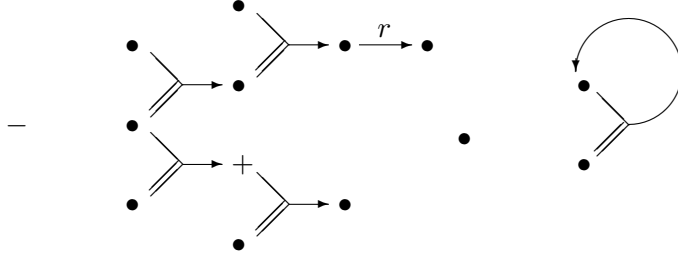


Figure 7: Example of essential and non-essential nodes.

*injective* iff every table in  $G$  is functional and injective, and *lean* iff every node in  $G$  is essential. We say that  $G$  is in *normal form* iff  $G$  is basic, functionally injective and lean.

**Lemma 4.2 (Conversion to normal form)** *Every graph  $G$  can be effectively converted (by elimination, star and ErUn rules) into a provably equivalent NFG in normal form.*

PROOF. By applying elimination rules we reduce  $G$  to a basic  $G_1$  (by induction on the number of operators in the labels of arcs in the graph). By applying rules Fnc and Inj we reduce  $G_1$  to a functionally injective  $G_2$  (by induction on the number of nodes in the slices of the graph). We can reduce  $G_2$  to a lean NFG, noting that, as the table of  $G_2$  is injective, a non-essential node occurs in a star-path or is useless. If it occurs in star-path, this star-path has a sink which is not a 0-essential node and occurs in exactly one node-equation, whence this sink can be eliminated by rule Tot. In the latter case, we use rule ErUn. We thus effectively convert  $G$  into NFG in normal form, they are provably equivalent since the rules used are reversible. ■

Thus, by soundness, we can reduce inclusions to inclusion of normal forms.

**Corollary 4.1 (Reduction)** *Given fork graphs  $G$  and  $H$ ,  $\models G \sqsubseteq H$  iff  $\models \text{NFG} \sqsubseteq \text{NFH}$ .*

The next lemma is our main tool for establishing the remaining results. Its proof is based on a construction of a model induced by a slice in normal form.

**Lemma 4.3** *Given fork graphs  $G$  and  $H$  in normal form, if  $\models G \sqsubseteq H$ , then  $H$  covers  $G$ .*

PROOF. Assume  $\models G \sqsubseteq H$ . Let  $S = (N, T, A, x, y)$  be a slice of  $G$ . Construct structured model  $\mathfrak{M}_S = (N^\star, \star, r_i^{\mathfrak{M}_S})_{i \in \omega}$  as follows:

- $N^\star := \bigcup_{k \in \mathbb{N}} N_k$ , with  $N_0 := N$  and  $N_{k+1} := N_k \cup \{(u, v) \in N_k \times N_k : \forall w \in N_k (u \star v \mapsto w \notin T)\}$ ,
- $r_i^{\mathfrak{M}_S} := \{(u, v) \in N \times N : ur_i v \in A\}$ ,
- $u \star v := \begin{cases} w & \text{if } u \star v \mapsto w \in T \\ (u, v) & \text{otherwise.} \end{cases}$

It is easy to see that  $(x, y) \in \llbracket G \rrbracket_{\mathfrak{M}_S}$ .

Also,  $\mathfrak{M}_S$  is a model, since by construction,  $\star$  is total as well as injective and is functional. Since  $\models G \sqsubseteq H$ , we have  $(x, y) \in \llbracket H \rrbracket_{\mathfrak{M}_S}$ . Thus, consider a slice  $S' = (N', A', T', x', y')$  of  $H$  and an  $\mathfrak{M}_S$ -assignment  $g : N' \rightarrow N^\star$  with  $gx' = x$ ,  $gy' = y$ .

We claim that the range of  $g$  is a subset of  $N$ . In fact, given  $v \in N'$ , we have that  $v$  is essential, since  $H$  is in normal form. If  $v$  is distinguished in  $S'$ , or an extreme of an arc in  $S'$ , then  $gv \in N$ .

If  $v$  is in a star-cycle in  $S'$ , then  $gv \in N$ , since the nodes introduced in the construction of  $\mathfrak{M}_S$  do not belong to cycles. If  $v$  is in a star-path  $(u_1, v_1, \dots, u_n, v_n, w)$  in  $S'$  such that  $v = u_1$  or  $v = v_1$  and  $w$  is a 0-essential node in  $S'$ , then, by previous cases,  $gw \in N$ . Since  $u_n \star v_n \mapsto w \in T'$ , we have  $gu_n \star gv_n = gw$ . Hence,  $gu_n \star gv_n \mapsto gw \in T$ . So,  $gu_n, gv_n \in N$ . Applying the same reasoning backwards, we obtain  $gv \in N$ .

To show that  $g$  is a homomorphism from  $S'$  to  $S$ , it remains to see that  $g$  preserves arcs and tables, but this is clear, because  $g$  is an  $\mathfrak{M}_S$ -assignment.<sup>2</sup> ■

We thus immediately have our central equivalences.

**Proposition 4.1** *Given fork graphs  $G$  and  $H$ , the following assertions are equivalent.*

1. *The inclusion  $G \sqsubseteq H$  is valid:  $\models G \sqsubseteq H$ .*
2. *The inclusion  $\text{NFG} \sqsubseteq \text{NFH}$  is valid:  $\models \text{NFG} \sqsubseteq \text{NFH}$ .*
3.  *$\text{NFH}$  covers  $\text{NFG}$ :  $\text{NFG} \leftarrow \text{NFH}$ .*
4. *The inclusion  $G \sqsubseteq H$  is a theorem:  $\vdash G \sqsubseteq H$ .*

We thus have our main results: completeness (of normal) proofs and decidability.

**Theorem 4.1** *Given a fork graph  $G$  and  $H$ , consider the inclusion  $G \sqsubseteq H$ .*

- (a) *If the inclusion  $G \sqsubseteq H$  is valid, then there is a normal proof of  $H$  from  $G$ .*
- (b) *The inclusion  $G \sqsubseteq H$  is valid iff  $\text{NFH}$  covers  $\text{NFG}$ .*

## 5 Perspectives

We have presented a graph calculus for proving and deciding the positive identities and inclusions of fork algebras. In this calculus, formulas are graphs and the rules derive graphs from graphs. This calculus is sound, complete and decidable for graph inclusions. We have illustrated how this graphical apparatus can be directly applied to the positive fork algebraic inclusions and identities. Our fork calculus considered structured universes with a total injective function  $\star$ . A natural extension would be providing sound and complete calculi for structured universes with weaker restrictions imposed on  $\star$ .

Proofs of inclusions and identities from hypotheses are also interesting. Extending our system to cope with this non-decidable case will involve more elaborated work.

Pictures have been proposed by some authors as a tool to help investigating and applying relational formalisms. Here, we mention three main lines of research.

The approach based on the *theory of allegories* [1, 2, 3, 4, 10] views pictures as arrows in a (unitary pretabular) allegory [7] and uses laws directly associated to the valid allegorical identities for transforming pictures. Results of the theory of allegories are used to show that two pictures can be proved equal by using the laws on pictures iff they represent the same relation.

The approach based on the *rewriting systems* [11, 12, 13] endows pictures with a relational semantics, which allows them to be interpreted as terms of an algebraic language. A rewriting mechanism for pictures is built as a variant of the algebraic approach to graph rewriting. Rewriting sequences can be used as proofs in a way that leads to a very general and flexible tool for the proof of relational algebraic identities.

The *logic systematic* approach [3, 5, 6] considers pictures as ordinary formulas of a (non-orthodox) logical system and a set of inference rules is provided for deriving pictures from pictures. This approach emphasizes notions of normal form and homomorphism for pictures, which are used to decide the positive inclusions and equalities of relation algebra.

---

<sup>2</sup>For a node equation  $u \star v \mapsto w \in T'$ , we have  $gu \star gv = gw$ , and as  $gw \in N$ , we have  $gu \star gv \mapsto gw \in T$ . For an arc  $ur_i v \in A'$ , we have  $(gu, gv) \in \llbracket r_i \rrbracket_{\mathfrak{M}_S}$ , and by the definition of  $\mathfrak{M}_S$ , one has  $gur_i gv \in A$ .

Each one of these approaches has its own flavor, techniques of investigations and line of results. Nevertheless, they are not completely disjoint sharing many characteristics whose interactions deserve further investigation. The work reported here may also be viewed as a first contribution in this direction in that we extend the logic systematic approach to the positive fork language. We thus provide a basis for a new formalism, which is not only more widely applicable but also provides a common denominator of the above three lines of investigation.

## References

- [1] C. BROWN & G. HUTTON. Categories, allegories and circuit design. In *9th IEEE Symp. Logic in Computer Science*: 372–381, 1994.
- [2] C. BROWN & A. JEFFREY. Allegories of circuits. In *Proc. Logical Foundations of Computer Science*: 56–68; St. Petersburg, 1994.
- [3] S. CURTIS & G. LOWE. A graphical calculus. In *Mathematics of Program Construction*, LNCS **947**: 214–231; Berlin: Springer, 1995.
- [4] S. CURTIS & G. LOWE. Proofs with graphs. *Sci. Comput. Program.* **26**: 197–216, 1996.
- [5] R.P. DE FREITAS, P.A.S. VELOSO, S.R.M. VELOSO & P. VIANA. Reasoning with graphs. *ENTCS* **165**: 201–212, 2006.
- [6] R. DE FREITAS, P.A.S. VELOSO, S.R.M. VELOSO & P. VIANA. On positive relational calculi. *Logic J. IGPL* **15**: 577–601, 2006.
- [7] P.J. FREYD & A. SCEDROV. *Categories, Allegories*. Amsterdam: North-Holland, 1990.
- [8] M. FRIAS. *Fork Algebras in Algebra, Logic and Computer Science*. World Scientific, 2002.
- [9] R. HIRSCH & I. HODKINSON. *Relation Algebras by Games*. Amsterdam: Elsevier, 2002.
- [10] G. HUTTON. A relational derivation of a functional program. In *Lecture Notes of the STOP Summer School on Constructive Algorithms*; Ameland, 1992.
- [11] W. KAHL. Algebraic graph derivations for graphical calculi. In F. d’Amore, P.G. Franciosa & A. Marchetti-Spaccamela (eds.) *Graph Theoretic Concepts in Computer Science, 22nd International Workshop, WG’96* LNCS **1197**: 224–238, Berlin: Springer, 1992.
- [12] W. KAHL. Relational treatment of term graphs with bound variables. *Logic J. IGPL* **6**: 259–303, 1998.
- [13] W. KAHL. Relational matching for graphical calculi of relations. *Inform. Sciences* **119**(3–4):253–273, 1999.
- [14] R.D. MADDUX. *Relation Algebras*. Amsterdam: Elsevier, 2006.
- [15] R.D. MADDUX. Relation-algebraic semantics. *Theor. Comput. Sci.* **160**: 1–85, 1996.
- [16] SZ. MIKULÁS, I. SAIN & A. SIMON. Complexity of the equational theory of relation algebras with projection elements. *Bull. Sect. Logic Univ. Łódź* **21**: 103–111, 1992.