

Positive Fork Graph Calculus^{*}

Renata de Freitas¹, Sheila R. M. Veloso², Paulo Veloso³, and
Petruccio Viana¹

¹ Institute of Mathematics, UFF: Universidade Federal Fluminense; Niterói, Brazil.

² Systems and Computer Engin. Dept., Faculty of Engineering, UERJ: Universidade do Estado do Rio de Janeiro; Rio de Janeiro, Brazil.

³ Systems and Computer Engin. Program, COPPE, UFRJ; Universidade Federal do Rio de Janeiro; Rio de Janeiro, Brazil.

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, complete and decidable. Moreover, the playful nature of this calculus renders it much more intuitive than its equational counterpart.

Key words: 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.

Relation algebras [15] are appropriate to formalize some aspects of program methodology albeit their limitations on expressive and proof powers. One of the motivations for extending relation algebras is capturing important notions linked to *storing* and *retrieving* of data, which is beyond its range. One operator introduced to this end is the *fork* operator [8], which is induced by a given injective coding function \star , so that $b\star c$ 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 $\{(a, b\star c) : (a, b) \in X \text{ and } (a, c) \in Y\}$.

For instance, given relations X_1, X_2, X_3, X_4 on a set U , fork can be used to *store* the coded result of the application of X_1, X_2, X_3, X_4 to a single element $a \in U$. That is, a pair (a, b) belongs to the relation

^{*} Research partially sponsored by CNPq, FAPERJ and FAPESP.

obtained by iterated application of fork to relations X_1, X_2, X_3, X_4 on U , iff there are $b_1, b_2, b_3, b_4 \in U$ such that $b = b_1 \boxtimes (b_2 \boxtimes (b_3 \boxtimes b_4))$ and $(a, b_1) \in X_1, (a, b_2) \in X_2, (a, b_3) \in X_3, (a, b_4) \in X_4$. Fork can also be used to define projection operators to *retrieve* the data stored. For instance, one can define a new constant operator π_3^4 that outputs the third coordinate of a quadruple, i.e., a pair (a, b) belongs to the relation π_3^4 iff there are $a_1, a_2, a_3, a_4 \in U$ such that a is the encoding $a_1 \boxtimes (a_2 \boxtimes (a_3 \boxtimes a_4))$ and b is a_3 .

Abstractly, relation algebras can be defined by a set of identities specifying the behavior of the Boolean and Peircean operators as follows. The former operators behave as in Boolean algebras. The latter operators behave as in involuted monoid theory. One also adds an identity expressing a geometric aspect of the interaction of Boolean and Peircean operators [9, 14]. Fork algebras may be defined by extending relation algebras with a new binary operator ∇ (fork) and the following three identities:

- (a) $(\mathbf{I} \nabla \mathbf{E})^\top \nabla (\mathbf{E} \nabla \mathbf{I})^\top \subseteq \mathbf{I}$,
- (b) $(r \nabla s) \circ (t \nabla q)^\top = (r \circ t^\top) \sqcap (s \circ q^\top)$,
- (c) $(r \circ (\mathbf{I} \nabla \mathbf{E})) \sqcap (s \circ (\mathbf{E} \nabla \mathbf{I})) = r \nabla s$.

One of the most important characteristics of the fork algebraic apparatus is that it provides algebraic proofs of interesting program properties, 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]. Indeed, 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.

Although the validity of some identities is easily established, this is not true in general: the equational theories of relation and fork algebras are rather complex [16]. Indeed, some non-algebraic mechanisms have been developed 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 this calculus 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 language $+FL$, its formal syntax and semantics, as well as some examples of equations that are 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 positive fork language. Section 5 closes the paper with some remarks and directions for future work.

2 Positive fork language

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 ∇ on relations, called *fork* [8].

The $+FL$ *terms*, typically denoted R, S, T , are generated from the set of relational variables $RVAR = \{r_i : i \in \omega\}$ by applying the relational operators $\mathbf{E}, \mathbf{I}, \mathbf{T}, \mathbf{\Pi}, \mathbf{\sqcup}, \mathbf{\circ}$, and $\mathbf{\nabla}$, according to the grammar $R ::= r_i \mid \mathbf{E} \mid \mathbf{I} \mid R^{\mathbf{T}} \mid R \mathbf{\Pi} S \mid R \mathbf{\sqcup} S \mid R \mathbf{\circ} S \mid R \mathbf{\nabla} S$. 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, \star) , where $M \neq \emptyset$ and $\star : M \times M \rightarrow M$ is an injective operation. Intuitively, $a \star b \in M$ codifies the pair (a, b) of elements of M . The *fork* induced by \star in a structured universe (M, \star) is the binary operation ∇_{\star} on relations on M given by $R \nabla_{\star} S := \{(a, b \star c) \in M \times M : aRb \text{ and } aSc\}$.

A *structured model* is a triple $\mathfrak{M} = (M, \star, r_i^{\mathfrak{M}})_{i \in \omega}$, where (M, \star) 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 $\mathfrak{M} = (M, \star, r_i^{\mathfrak{M}})_{i \in \omega}$, symbols \mathbf{E} and \mathbf{I} are interpreted, respectively, as the relations $M \times M$ and $\{(a, a) : a \in M\}$; symbols \sqcap , \sqcup , \top and \circ as intersection, union, conversion and composition of relations, respectively; and the meaning of a fork term $R \nabla S$ is defined by $\llbracket R \nabla S \rrbracket_{\mathfrak{M}} ::= \llbracket R \rrbracket_{\mathfrak{M}} \nabla_{\star} \llbracket S \rrbracket_{\mathfrak{M}}$.

Validity of inclusions and equalities are defined as usual. 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.

3 Positive fork graph calculus

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. In 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) , denoted $u \star v \rightarrow w$, 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.

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.

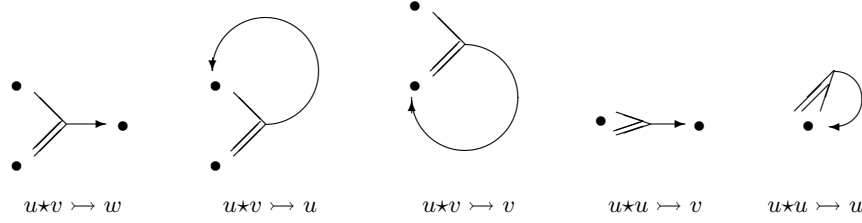


Fig. 1. Graphical representation of node equations

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* (Trm is the set of $+\text{FL}$ 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 $S = (\{x, u, v, w, y\}, \{u \star v \mapsto w\}, \{xru, x(s \sqcap l)v, w(\mathbf{E} \circ t)y\}, x, y)$, having a structured node, is represented in Figure 2.

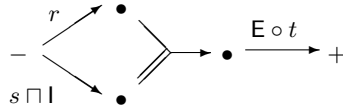


Fig. 2. Slice S with a structured node

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 $+\text{FG}$ *inclusions* and *equalities* are expressions of the forms $G \sqsubseteq H$ and $G = H$, respectively.

The semantics of slices and graphs are based on 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 graph* $G = (S_i)_{i \in I}$ in \mathfrak{M} is $\llbracket G \rrbracket_{\mathfrak{M}} = \bigcup_{i \in I} \llbracket S_i \rrbracket_{\mathfrak{M}}$.

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 (usually used to put a graph in normal form), while another rule will be used to compare graphs (usually in normal form). We will 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 $A \frac{u}{v} := \{w \frac{u}{v} R z \frac{u}{v} : w R z \in A\}$.

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.

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 $uR^T v$ 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:

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

(a) Relational rules

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

(b) Fork rule

Table 1. Elimination/Introduction rules for transforming graphs

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 wSv , 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 .

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 **GrCvr**: it allows one to infer a graph from one covered under homomorphism. The notions involved are natural extensions of the $+RG$ case [6].

Given slices $S = (N, T, A, x, y)$ and $S' = (N', T', A', x', y')$, a *homomorphism* from S' to S (denoted $\theta : S' \rightarrow S$) is a function $\theta : N' \rightarrow N$ that preserves the slice structure: $\theta w \star \theta u \mapsto \theta v \in T$, for every node equation $w \star u \mapsto v$ in T' ; $\theta u R \theta v \in A$, for every arc $u R v$

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

Table 2. Star rules for transforming graphs

$$\text{GrCvr} \frac{G}{H} \quad \text{if } G \leftarrow H$$

Table 3. Homomorphism rule GrCvr for transforming graphs.

in A' ; $\theta x' = x$ and $\theta y' = y$. Given graphs G and H , we say that H *covers* G (denoted $G \leftarrow H$) iff, for each slice S of G , there is a slice S' of H and a homomorphism $\theta : S' \rightarrow S$.

Rule GrCvr can be applied only downwards, but a special case of GrCvr 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.

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

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 **GrCvr** rule, followed by applications of introduction, star and **ErUn** rules.

To illustrate the system in action, 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 $\vdash G_R \sqsubseteq G_S$. A graph proof of the **+FG** axiom (a) is indicated in Figure 3.

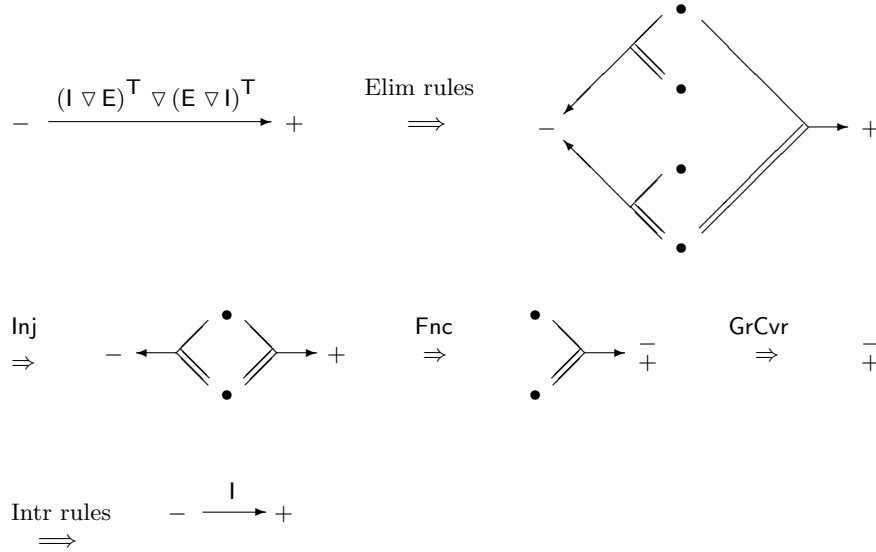


Fig. 3. Graph proof of fork axiom (a)

+FG also can model a kind of parallel product through the introduction of the operator *cross*. Given a pair of relations X and Y on a set U , by applying *cross* to X and Y we obtain the relation $\{(\star ab, \star cd) : (a, c) \in X \text{ and } (b, d) \in Y\}$. A fork algebraic definition of *cross* is given by [8]: $r \otimes s := ((I \nabla E)^T \circ r) \nabla ((E \nabla I)^T \circ s)$. In [8], \otimes is extensively used to prove identities as

$$(d) (r \nabla s) \circ (t \otimes u) = (r \circ t) \nabla (s \circ u),$$

describing the iterated behaviour of the algebraic operators. Figure 4 contains an equational proof of (d).

$$\begin{aligned}
(r \nabla s) \circ (t \otimes u) &= (r \nabla s) \circ (t \otimes u)^{\top\top} \\
&= (r \nabla s) \circ [((I \nabla E)^{\top} \circ t) \nabla ((E \nabla I)^{\top} \circ u)]^{\top\top} \\
&= (r \nabla s) \circ [((I \nabla E)^{\top} \circ t \circ (I \nabla E)) \sqcap ((E \nabla I)^{\top} \circ u \circ (E \nabla I))]^{\top\top} \\
&= (r \nabla s) \circ [((I \nabla E)^{\top} \circ t \circ (I \nabla E))^{\top} \sqcap ((E \nabla I)^{\top} \circ u \circ (E \nabla I))^{\top}]^{\top\top} \\
&= (r \nabla s) \circ [((I \nabla E)^{\top} \circ t^{\top} \circ (I \nabla E)^{\top\top}) \sqcap ((E \nabla I)^{\top} \circ u^{\top} \circ (E \nabla I)^{\top\top})]^{\top\top} \\
&= (r \nabla s) \circ [((I \nabla E)^{\top} \circ t^{\top} \circ (I \nabla E)) \sqcap ((E \nabla I)^{\top} \circ u^{\top} \circ (E \nabla I))]^{\top} \\
&= (r \nabla s) \circ [((I \nabla E)^{\top} \circ t^{\top}) \sqcap ((E \nabla I)^{\top} \circ u^{\top})]^{\top} \\
&= [r \circ ((I \nabla E)^{\top} \circ t^{\top})]^{\top} \sqcap [s \circ ((E \nabla I)^{\top} \circ u^{\top})]^{\top} \\
&= [r \circ [t^{\top\top} \circ (I \nabla E)^{\top\top}]] \sqcap [s \circ [u^{\top\top} \circ (E \nabla I)^{\top\top}]] \\
&= [r \circ [t \circ (I \nabla E)]] \sqcap [s \circ [u \circ (E \nabla I)]] \\
&= (r \circ t) \nabla (s \circ u).
\end{aligned}$$

Fig. 4. Equational proof of (d)

Figure 5 presents an alternative graph proof, based on the derived rule **Paral** in Table 5, whose graph derivation is in Figure 6.

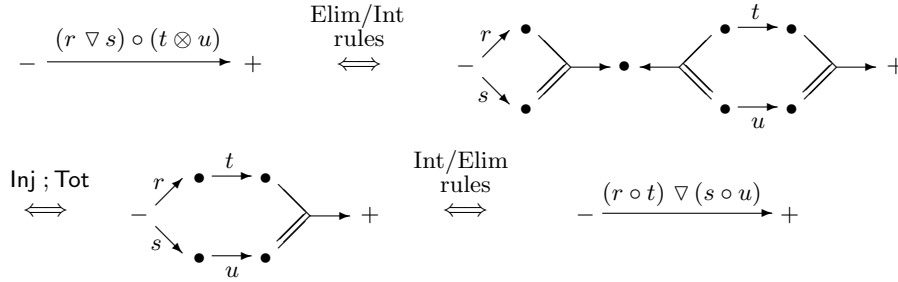


Fig. 5. Graph proof of (d)

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 1.

Paral	$N, T, A \cup \{uR \otimes Sv\}, x, y$
	$N \cup \{w_1, w_2, w_3, w_4\}, T \cup \{w_1 \star w_3 \mapsto u, w_2 \star w_4 \mapsto v\}, A \cup \{w_1 R w_2, w_3 S w_4\}, x, y$
	if $w_1, w_2, w_3, w_4 \notin N$

Table 5. Derived introduction/elimination rule for \otimes .

Lemma 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 GrCvr 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.

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 $i, 1 \leq \dots i \dots \leq n - 1$, $u_i \star v_i \mapsto u_{i+1} \in T$ 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 $i, 1 \leq \dots i \dots \leq n - 1$, $u_i \star v_i \mapsto u_{i+1} \in T$ 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 $v = v_1$.

We call a graph G *basic* iff every arc in G is labeled by a relational variable, *functionally 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 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 $\mathfrak{M}_S = (N^*, \star, r_i^{\mathfrak{M}_S})_{i \in \omega}$ as follows:

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

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 $g : N' \rightarrow N^*$ 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 $u_n \star v_n \rightarrow w \in T'$, we have $gu_n \star gv_n = gw$. Hence, $gu_n \star gv_n \rightarrow gw \in T$. 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.⁴

We thus immediately have our central equivalences.

Proposition 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 $\nu G \sqsubseteq \nu H$ is valid: $\models \nu G \sqsubseteq \nu H$.*

⁴ For a node equation $u \star v \rightarrow w \in T'$, we have $gu \star gv = gw$, and as $gw \in N$, we have $gu \star gv \rightarrow gw \in T$. 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 $gur_i gv \in A$.

3. νH covers νG : $\nu G \leftarrow \nu H$.
4. The inclusion $G \sqsubseteq H$ is a theorem: $\vdash G \sqsubseteq H$.

We thus have completeness (of normal) proofs and decidability.

Theorem 1. *Given a fork graph G and H , consider the inclusion $G \sqsubseteq H$.*

- (a) *If $G \sqsubseteq H$ is valid, then there is a normal proof of H from G .*
- (b) *The inclusion $G \sqsubseteq H$ is valid iff νH covers νG .*

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 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* [1–4, 10], the approach based on the *rewriting systems* [11–13], and the *logic systematic* approach [3, 5, 6]. 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. Brown, C., Hutton, G.: Categories, allegories and circuit design. In: 9th IEEE Symp. Logic in Computer Science pp. 372–381, IEEE Computer Society Press (1994)
2. Brown, C., Jeffrey, A.: Allegories of circuits. In: Proc. Logical Foundations of Computer Science pp. 56–68, St. Petersburg (1994)
3. Curtis, S., Lowe, G.: A graphical calculus. In: Mller, B. (ed) Mathematics of Program Construction. LNCS, vol. 947, pp. 214–231 Springer, Berlin (1995)
4. Curtis, S., Lowe, G. Proofs with graphs. In: Sci. Comput. Program., vol. 26 pp. 197–216 (1996)
5. de Freitas, R.P., Veloso, P.A.S., Veloso, S.R.M. , Viana, P.: Reasoning with graphs. ENTCS, vol.165, pp. 201–212 (2006)
6. de Freitas, R.P., Veloso, P.A.S., Veloso, S.R.M. , Viana, P.: On positive relational calculi. Logic J. IGPL, vol.15, pp.577–601, Oxford University Press (2006)
7. Freyd, P.J, Scedrov, A.: Categories, Allegories. North-Holland, Amsterdam (1990)
8. Frias, M.: Fork Algebras in Algebra, Logic and Computer Science. World Scientific (2002)
9. Hirsch, R., Hodkinson, I.: Relation Algebras by Games. Elsevier, Amsterdam (2002)
10. Hutton, G.: A relational derivation of a functional program. In: Lecture Notes of the STOP Summer School on Constructive Algorithms, Ameland (1992)
11. Kahl, W.: Algebraic graph derivations for graphical calculi. In: d’Amore, F., Franciosa, P.G., Marchetti-Spaccamela, A. (eds.) Graph Theoretic Concepts in Computer Science WG’96 LNCS vol. 1197, pp. 224–238, Springer, Berlin (1997)
12. Kahl, W.: Relational treatment of term graphs with bound variables. Logic J. IGPL, vol.6, pp. 259–303 Oxford University Press (1998)
13. Kahl, W.: Relational matching for graphical calculi of relations. Inform. Sciences, vol.119, n. 3–4, pp. 253–273 Elsevier, New York (1999)
14. Maddux, R.D.: Relation Algebras. Elsevier, Amsterdam (2006)
15. Maddux, R.D.: Relation-algebraic semantics. Theor. Comput. Sci., vol. 160, pp. 1–85, Elsevier, Amsterdam (1996)
16. Mikulás, Sz., Sain, I., Simon, A.: Complexity of the equational theory of relation algebras with projection elements. Bull. Sect. Logic Univ. Lódź, vol. 21, pp. 103–111 (1992)