Advances in Logic — Vol. 2

Fork Algebras in Algebra, Logic and Computer Science

Marcelo Fabián Frias

Fork Algebras in Algebra, Logic and Computer Science

Advances in Logic – Vol. 2

Fork Algebras in Algebra, Logic and Computer Science

Marcelo Fabián Frias
University of Buenos Aires, Argentina

World Scientific
New Jersey • London • Singapore • Hong Kong

Published by

World Scientific Publishing Co. Pte. Ltd.

P O Box 128, Farrer Road, Singapore 912805

USA office: Suite 1B, 1060 Main Street, River Edge, NJ 07661

UK office: 57 Shelton Street, Covent Garden, London WC2H 9HE

Library of Congress Cataloging-in-Publication Data
Frias, Marcelo Fabián, 1968–
Fork algebras in algebra ; logic and computer science / Marcelo Fabián Frias.
p. cm. -- (Advances in logic ; v. 2)
Includes bibliographical references and index.
ISBN 9810248768 (alk. paper)
1. Computer science -- Mathematics. 2. Logic, symbolic and mathematical. I. Title.
II. Series.

QA76.9.M35 F75 2002
004'.01'51--dc21 2002066193

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

Printed in Singapore by Uto-Print

to Norma, Taís and Gastón

This page is intentionally left blank

Preface

This book is the result of the research carried on by the author and some of his colleagues from 1995. Fork algebras, the subject of this book, had their origin in the early 90s as part of a formalism capable of dealing with the process of program specification and development. The contents of the book fall in with what are called Relational Methods in Computer Science.

As usually happens, applied research led to problems of a theoretical nature which were undertaken by the author and are the core of this book. Problems such as *finite axiomatizability* or *axioms independence* (Sections 4.1 and 4.2) naturally arose when investigating the relational semantics of the fork calculus.

Relational proof systems for various logics (classical, modal and multi-modal) (Sections 5.2, 6.5, 6.7) besides providing relational deductive calculi for these logics, allow us to assess the expressive power of the fork calculus and establish the foundations for a relational formalism for system specification.

Finally, in Section 7.5 we present the foundations for a relational calculus for program specification and derivation that allow us to specify and calculate program design strategies.

The author wishes to thank Armando Haeberer and Roger Maddux, who were his Ph.D. advisors and co-authors of several results in this book. Gabriel Baum has to be thanked not only as a colleague and co-author of several papers, but also as a constant source of friendship and advice. Tom Maibaum suggested that these results could be put in the form of a book. The members of the RelMiCS (*Relational Methods in Computer Science*) group are thanked for their always useful comments and criticisms. Finally, the author wishes to thank the Department of Informatics, PUC-RIO; LI-FIA, National University of La Plata; and the Department of Computer Science, School of Sciences, University of Buenos Aires, for providing an always stimulating environment.

Contents

Chapter 1
Introduction and Motivations

1.1 Software Specification, Binary Relations and Fork

Fork algebras —the subject of this book— have their origin as the foundation of a framework for software specification, verification and derivation. In our view, specification languages —as modern graphical notations like UML [G. Booch et al. (1998)]— must allow for a modular description of the different aspects that comprise a system. These aspects include structural properties, dynamic properties, temporal properties, etc. Different formalisms allow us to specify each one of these aspects, namely,

- first-order classical logic for structural properties
- propositional and first-order dynamic logic for dynamic properties,
- different modal logics for temporal properties.

Many of the previously mentioned formalisms have complete deductive systems. Nevertheless, reasoning across formalisms may be difficult if not impossible. A possible solution in order to solve this problem consists on finding an amalgamating formalism satisfying at least the following:

- the formalism must be expressive enough to interpret the specification formalisms,
- the formalism must have very simple semantics, understandable by non mathematicians,
- the formalism must have a complete and simple deductive system.

In this book we propose the formalism called *fork algebras* to this end. The formalism is presented in the form of an equational calculus, which reduces reasoning to substitution of equals by equals. The calculus is complete with respect to a very simple semantics in terms of *algebras of binary relations.*

Algebras of binary relations, such as the ones to be used in this book, have as domain a set of binary relations on some set (let us say A). Among the operations that can be defined on such domain, consider the following:

- the empty binary relation $\emptyset$,
- complement of a binary relation x with respect to a largest relation E, i.e., $\overline{x}$ —as the complement of x is denoted— is defined as $E \setminus x$,
- union of binary relations —denoted by $\cup$—, and
- intersection of binary relations —denoted by $\cap$.

Notice that the previous operations are defined on arbitrary sets, independently of whether these are binary relations or not. Actually, a set of binary relations closed under these operations is an example of *set Boolean algebra.* However, there are other operations that operate naturally on binary relations but are not defined on arbitrary sets. Among these we can mention:

- the identity binary relation on A —denoted by Id—,
- composition of binary relations —denoted by $\circ$—, and
- transposition of the pairs of a binary relation —denoted by $\smile$.

Unfortunately, a class of algebras containing these operations cannot be axiomatized by a finite number of equations [D. Monk (1964)]. In order to overcome this important drawback, we add an extra binary operation on relations called *fork.* Addition of fork has two main consequences. First, the class of algebras obtained can be axiomatized by a finite (and small) number of equations. Second, addition of fork induces a structure on the domain on top of which relations are built, i.e., rather than being the arbitrary set A, it is a set A^* closed under a binary function $*$. The definition of the operation fork (denoted by $\underline{\nabla}$) is then given by:

$$R \underline{\nabla} S = \{ \langle x, y * z \rangle : xRy \ \wedge \ xSz \} \ .$$

The definition of $\underline{\nabla}$ is depicted in Fig. 1.1. Whenever x and y are related via R, and x and z are related via S, x and $y * z$ are related via $R \underline{\nabla} S$. Notice that the definition strongly depends on the function $*$. Actually, the definition of fork evolved around the definition of the function $*$. From 1990

$$\begin{tikzcd} & R \arrow[r] & y \\ x \arrow[ur, dash] \arrow[dr, dash] & \underline{\nabla} & * \\ & S \arrow[r] & z \end{tikzcd}$$

Fig. 1.1 Fork of binary relations R and S.

(when the first class of fork algebras was introduced) until now, different alternatives were explored with the aim of finding a framework which would satisfy our needs. In the definition of the first class of fork algebras [P. Veloso et al. (1991)], function $*$ produced true set theoretical pairs, i.e., when applied to values a and b, $*(a, b)$ returned the pair $\langle a, b \rangle$. Mikulás, Sain, Simon and Németi showed in [S. Mikulás et al. (1992); I. Sain et al. (1995)] that this class of fork algebras was not finitely axiomatizable. This was done by proving that a sufficiently complex theory of natural numbers can be interpreted in the equational theory of these fork algebras, and thus leads to a non recursively enumerable equational theory. Other classes of fork algebras were defined, in which $*$ was binary tree formation or even concatenation of sequences, but these were shown to be non finitely axiomatizable too. It was in [M. Frias et al. (1995)a] where the class of fork algebras to be used in this book came up. The only requirement placed on function $*$ was that it had to be injective. This was enough to prove in [M. Frias et al. (1997)b] that the newly defined class of fork algebras was indeed finitely axiomatizable by a set of equations.

This page is intentionally left blank

Chapter 2

Algebras of Binary Relations and Relation Algebras

2.1 History and Definitions

In this section the classes of algebras of binary relations and their abstract counterpart, the class of relation algebras, will be defined. The study of algebras of binary relations began with the works of Charles Sanders Peirce [Ch. Peirce (1933)] and Augustus De Morgan [A. De Morgan (1966)] and was later continued by Ernst Schröder [E. Schröder (1895)] when looking for an algebraic counterpart of first-order reasoning, much the same as George Boole developed the so-called Boolean algebras as an algebraic counterpart to propositional reasoning.

Throughout this section and the rest of the book it will be assumed that the reader has a nodding acquaintance with elementary concepts of set-theory and first-order logic. As a reference text in both areas the reader is referred to [J. Barwise (1977)]. Given a binary relation X in a set A, and $a, b \in A$, we will denote the fact that a and b are related via the relation X by $\langle a, b\rangle \in X$ or $a X b$, indistinctly.

Definition 2.1 Let E be a binary relation on a set A, and let R be a set of binary relations satisfying:

(1) $\bigcup R \subseteq E$,
(2) Id (the identity relation on the set A), $\emptyset$ (the empty binary relation) and E belong to R,
(3) R is closed under set union ($\cup$), intersection ($\cap$) and complement relative to E ($^-$),
(4) R is closed under relational composition (denoted by $\circ$) and con-

verse (denoted by $\smile$). These two operations are defined by

$$X \circ Y = \{ \langle a,b \rangle : \exists c\,(aXc \ \wedge \ cYb) \}$$

$$\breve{X} = \{ \langle a,b \rangle : bXa \} \ .$$

Then, the structure $\langle R, \cup, \cap, ^{-}, \emptyset, E, \circ, Id, ^{\smile} \rangle$ is called an *algebra of binary relations.* The class of algebras of binary relations will be denoted by ABR.

Definition 2.2 Notice that, according to Def. 2.1, each algebra of binary relations $\mathfrak{A}$ contains a set A on which the binary relations are defined. This set will be called the *base* of $\mathfrak{A}$, and will be denoted by $B_{\mathfrak{A}}$.

Definition 2.3 An algebra of binary relations is *full* if its universe is of the form $\mathcal{P}(U \times U)$ for some set U, and is *square* if its largest relation is of the form $U \times U$.

It follows immediately from Def. 2.3 that every full algebra of binary relations is square. Also, a square algebra of binary relations, whose largest relation is $U \times U$, is a subalgebra of the full algebra of binary relations with universe $\mathcal{P}(U \times U)$.

The following theorem, besides being useful in further sections, also gives a clear understanding of the structure of algebras of binary relations. Given algebras $\mathfrak{A}$ and $\mathfrak{B}$, $\mathfrak{A} \preceq \mathfrak{B}$ means that $\mathfrak{A}$ is embeddable in $\mathfrak{B}$, i.e., $\mathfrak{A}$ is isomorphic to a subalgebra of $\mathfrak{B}$. Also, given an index set I, $\prod_{i \in I} \mathfrak{A}_i$ denotes the direct product of the algebras $(\mathfrak{A}_i)_{i \in I}$.

Theorem 2.1 *Given an algebra of binary relations $\mathfrak{A}$, there exist an index set I and full algebras of binary relations $(\mathfrak{A}_i)_{i \in I}$ such that*

$$\mathfrak{A} \preceq \prod_{i \in I} \mathfrak{A}_i \ .$$

Proof Let E be the largest relation in $\mathfrak{A}$. Since $Id \subseteq E$, $\breve{E} \subseteq E$ and $E \circ E \subseteq E$, E is an equivalence relation. Thus, there exists an index set I such that $E = \bigcup_{i \in I} E_i$, with $E_i = U_i \times U_i$. For $i \in I$, let $\mathfrak{A}_i$ be the algebra of binary relations with largest relation E_i and universe $A_i = \{ x \in E_i : x \in A \}$. Let $h : A \to \prod_{i \in I} A_i$ be defined by: $\pi_i(h(a)) = a \cap E_i$. It is an easy exercise to show that h is a one-to-one homomorphism. $\square$

In 1941 Alfred Tarski [A. Tarski (1941)] introduced the elementary theory of binary relations (ETBR) as a logical formalization of the algebras of binary relations. The elementary theory of binary relations is a formal theory where two different sorts of variables are present. The set $IndVar = \{v_1, v_2, v_3, \dots\}$ contains the so-called *individual variables*, and the set $RelVar = \{R, S, T, \dots\}$ contains the so-called *relation variables*. If we add the *relation constants* 0, 1 and 1' to the relation variables and close this set under the unary operators $^-$ and $^\smile$, and the binary operators $+, \cdot$ and $;$, we obtain the set of *relation designations*. Examples of such objects are, for instance, $\breve{R}$ (to be read 'the *converse* of R') and $R;S$ (to be read 'the *relative product* of R and S'). Atomic formulas are expressions of the form xRy (where x, y are arbitrary individual variables and R is an arbitrary relation designation) or $R = S$ (with R and S arbitrary relation designations). From the atomic formulas, we obtain compound formulas as usual, by closing the atomic formulas under the unary logical constants $\neg$, $\forall x$, $\forall y, \dots$, $\exists x$, $\exists y \dots$ (x, $y, \dots$ individual variables) and the binary logical constants $\vee$, $\wedge$, $\Rightarrow$ and $\Leftrightarrow$. We will choose a standard set of logical axioms and inference rules for our theory (see e.g. [H. Enderton (1972), Ch. 2.4]). As the axioms that explain the meaning of the relational symbols 0, 1, 1', $^-$, $^\smile$, $+$, $\cdot$ and $;$, we single out the following sentences in which x, y, z are arbitrary individual variables and R, S, T are arbitrary relation designations.

$$\forall x \forall y \,(x\,1\,y) \qquad \text{(unit definition)}$$

$$\forall x \forall y \,(\neg\, x\,0\,y) \qquad \text{(zero definition)}$$

$$\forall x \,(x\,1'\,x) \qquad \text{(reflexivity of the identity)}$$

$$\forall x \forall y \forall z \,((xRy \wedge y\,1'\,z) \Rightarrow xRz) \qquad \text{(identity is a congruence)}$$

$$\forall x \forall y \,(x\overline{R}y \Leftrightarrow \neg\, xRy) \qquad \text{(complement definition)}$$

$$\forall x \forall y \left(x\breve{R}y \Leftrightarrow yRx\right) \qquad \text{(converse definition)}$$

$$\forall x \forall y \,(xR{+}Sy \Leftrightarrow xRy \vee xSy) \qquad \text{(join definition)}$$

$$\forall x \forall y \,(xR{\cdot}Sy \Leftrightarrow xRy \wedge xSy) \qquad \text{(meet definition)}$$

$$\forall x \forall y \,(xR;Sy \Leftrightarrow \exists z\,(xRz \wedge zSy)) \qquad \text{(relative product definition)}$$

$$R = S \Leftrightarrow \forall x \forall y \,(xRy \Leftrightarrow xSy) \qquad \text{(equality definition)}$$

From the elementary theory of binary relations, Tarski [A. Tarski (1941)] introduced the *calculus of relations* (CR). The calculus of relations is defined as a restriction of the elementary theory of binary relations. Formulas of the calculus of relations are those formulas of the elementary theory of binary relations where no variables over individuals occur. As axioms of the calculus of relations, Tarski chose a subset of formulas without variables over individuals valid in the elementary theory of binary relations. The formulas Tarski chose as axioms are, besides a set of axioms for the logical connectives, the following:

(1) $(R = S \wedge R = T) \Rightarrow S = T$
(2) $R = S \Rightarrow (R+T = S+T \wedge R{\cdot}T = S{\cdot}T)$
(3) $R+S = S+R \wedge R{\cdot}S = S{\cdot}R$
(4) $(R+S)\cdot T = (R{\cdot}T) + (S{\cdot}T) \wedge (R{\cdot}S) + T = (R+T)\cdot(S+T)$
(5) $R+0 = R \wedge R{\cdot}1 = R$
(6) $R+\overline{R} = 1 \wedge R{\cdot}\overline{R} = 0$
(7) $\overline{1} = 0$
(8) $\breve{\breve{R}} = R$
(9) $(R;S)^{\smile} = \breve{S};\breve{R}$
(10) $(R;S);T = R;(S;T)$
(11) $R;1' = R$
(12) $(R;S)\cdot T = 0 \Rightarrow (S;T)\cdot\breve{R} = 0$
(13) $R;1 = 1 \vee 1;\overline{R} = 1$

Axioms (1)–(7) are an axiomatization for Boolean algebras, axioms (8)–(12) axiomatize the relative operators.

As is customary in universal algebra, an algebra $\mathfrak{A}$ with universe A is *simple* if:

- $|A| \geq 2$,
- $\mathfrak{A}$ has exactly two homomorphic images.

It follows from [A. Tarski (1941), p. 85] that formula (13) is equivalent to the formula

$$\forall R (R \neq 0 \Rightarrow 1;R;1 = 1) . \tag{2.1}$$

It is proved in [B. Jónsson et al. (1952), Thm. 4.10(iii)] that (2.1) forces models to be simple, and therefore so does formula (13). The models of the calculus of relations motivate the following definition.

Definition 2.4 A *relation algebra* is an algebra $\langle A, +, \cdot, ^{-}, 0, 1, ;, 1', ^{\smile} \rangle$ where $+$, $\cdot$ and ; are binary operations, $^{-}$ and $^{\smile}$ are unary, and 0, 1 and 1' are distinguished elements. Furthermore, the reduct $\langle A, +, \cdot, ^{-}, 0, 1 \rangle$ is a Boolean algebra, and the following identities are satisfied for all $x, y, z \in A$:

$$x;(y;z) = (x;y);z, \tag{Ax. 1}$$

$$(x+y);z = x;z + y;z, \tag{Ax. 2}$$

$$(x+y)^{\smile} = \breve{x}+\breve{y}, \tag{Ax. 3}$$

$$\breve{\breve{x}} = x, \tag{Ax. 4}$$

$$x;1' = 1';x = x, \tag{Ax. 5}$$

$$(x;y)^{\smile} = \breve{y};\breve{x}, \tag{Ax. 6}$$

$$x;y \cdot z = 0 \text{ iff } z;\breve{y} \cdot x = 0 \text{ iff } \breve{x};z \cdot y = 0. \tag{Ax. 7}$$

As an immediate consequence of Defs. 2.1 and 2.4 we obtain the following theorem.

Theorem 2.2 *Every algebra of binary relations is a relation algebra.*

Proof The proof consists of showing that axioms Ax. 1–Ax. 7 hold in any algebra of binary relations, and is left as an insightful exercise for the reader. □

We will denote the class of all relation algebras by RA and by $\leq$ the ordering induced by the Boolean reduct. Thus, we will use the notation $x \leq y$ as a shorthand for the equation $x+y = y$. Elements from either algebras of binary relations or relation algebras will be generally called *relations.* In case it is necessary to mention elements from an algebra of binary relations, we will call them *binary relations* or *concrete relations.* We denote by 0' the *diversity* relation $\overline{1'}$.

Alternative (and equivalent) axiomatizations for the calculus of relations can be obtained by replacing Ax. 7 in Def. 2.4 by any of the following two formulas:

$$(x;y) \cdot z \;\leq\; (x \cdot z;\breve{y})\,;\,(y \cdot \breve{x};z)\,, \tag{2.2}$$

$$x;y \;\leq\; z \iff \breve{x};\overline{z} \leq \overline{y} \iff \overline{z};\breve{y} \;\leq\; \overline{x}. \tag{2.3}$$

Notice that (2.2) is an equation, and therefore the class of relation algebras is a finitely based variety, i.e., it is axiomatizable by a finite set of equations. If we add formula (13) to the axiomatization of relation algebras, we obtain the class of simple relation algebras. It is proved in [L. Chin et al. (1951)] that axioms (1)–(12) can be proved from Ax. 1–Ax. 7 and viceversa.

At the end of his paper [A. Tarski (1941)], Tarski asked the following questions:

(1) Is every model of the calculus of relations isomorphic to an algebra of binary relations?

(2) Is it true that every formula of the calculus of relations that is valid in all algebras of binary relations is provable in the calculus of relations?

(3) Is it true that every formula of the elementary theory of binary relations can be transformed into an equivalent formula of the calculus of relations?

The answer to these questions is negative in all cases. The first question was answered negatively by Roger Lyndon [R. Lyndon (1950); R. Lyndon (1956)] by exhibiting a non-representable relation algebra, i.e., a relation algebra that is not isomorphic to any algebra of binary relations. The immediate consequence of this result is that there exist properties valid in all algebras of binary relations which can be false in some relation algebras. The second question was answered by Lyndon, who showed that the equation

$$a;b \cdot c;d \cdot e;f \;\leq\; a;\left[\breve{a};c \cdot b;\breve{d} \cdot \left(\breve{a};e \cdot b;\breve{f}\right);\left(\breve{e};c \cdot f;\breve{d}\right)\right];d$$

is valid in all algebras of binary relations, but fails in the relation algebra presented in [R. Lyndon (1950)]. McKenzie [R. McKenzie (1970)] also answered the first question by presenting a small non representable relation algebra (Lyndon's algebra has fifty-six atoms, while McKenzie's has only four). Notice that since Tarski proved in [A. Tarski (1955)] that the class of representable relation algebras is a variety, i.e., it is axiomatizable with a set of equations, there must exist an equation that fails in McKenzie's algebra, providing a negative answer to the second question. An example of such an equation is given in [A. Tarski et al. (1987), p. 55]. With regard to the third question, a result due to Korselt and whose proof is included in [L. Löwenheim (1915)] shows that the expressive power of the calculus of relations is that of a proper restriction of first-order logic. The logical counterpart of the calculus of relations — denoted by $\mathcal{L}^{\times}$ in [A. Tarski et al. (1987)] — is equivalent (*equipollent* is the technical term) with a three variables fragment of first-order predicate logic (see [A. Tarski et al. (1987), Ch. 3.9] for a detailed proof of this). If we recall our mission of devising a framework suitable for system specification, such lack of expressiveness has a negative impact since first-order specifications of systems are not likely to have meaningful translations into the calculus of relations. In [A. Tarski et al. (1987), §3.4(iv)] Tarski and Givant present the following formula, which is not equivalent to any sentence of the calculus of relations:

$$\forall x \forall y \forall z \exists u \, (u 0\text{'} x \ \wedge \ u 0\text{'} y \ \wedge \ u 0\text{'} z) \ . \tag{2.4}$$

One way to convince oneself that this is indeed the case is by attempting to reduce this formula to a relational expression using the definitions of the relational operations. In Ch. 2.2 we will come back to this formula.

For a more detailed study in the origin of relation algebras and the calculus of relations the reader is referred to [R. Maddux (1998); R. Maddux (1991)] and [C. Brink et al. (1997), Ch. 2].

In Def. 2.5 below we introduce some terminology to be used in further sections.

Definition 2.5 A relation F is called *functional* if $\breve{F};F \leq 1\text{'}$. A relation I is called *injective* if $I;\breve{I} \leq 1\text{'}$. A relation S is called *symmetric* if $\breve{S} = S$. A relation T is called *transitive* if $T;T \leq T$. A relation D is called *left-ideal* if $D = 1;D$, and *right-ideal* if $D = D;1$. A relation C is called *constant* if it is functional, left-ideal and $C;1 = 1$. Intuitively, constant relations are alike constant functions (i.e., they map all inputs to a single value). We will generally denote the constant relation whose output is the value v by

C_v. By *Dom* (R) we denote the term $(R;\breve{R})\cdot 1'$ (the *domain* of the relation R), and by *Ran* (R) we denote the term $(\breve{R};R)\cdot 1'$ (the *range* of the relation R). Given a binary relation R, by $\mathsf{dom}\,(R)$ and $\mathsf{ran}\,(R)$ we denote the sets $\{\,x : \exists y(\langle x,y\rangle \in R)\,\}$ and $\{\,y : \exists x(\langle x,y\rangle \in R)\,\}$, respectively. In general, we will denote algebras and structures by capital german letters $(\mathfrak{A}, \mathfrak{B}, \ldots)$, and their universes by the associated roman letter $(A, B, \ldots)$. Given algebras $\mathfrak{A}$ and $\mathfrak{B}$, by $\mathfrak{A} \preceq \mathfrak{B}$ we denote the fact that $\mathfrak{A}$ is embeddable in $\mathfrak{B}$ (i.e., there exists a one–to–one homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$). Given a set S, by $\mathcal{P}\,(S)$ we denote the *power set of* S.

The reader is invited to verify that when interpreted in algebras of binary relations, the conditions in Def. 2.5 characterize familiar notions. For example, a binary relation F satisfying the condition $\breve{F};F \leq 1'$ will in effect be functional.

2.2 Arithmetical Properties

In this section a list of properties that are true in all relation algebras is presented. These properties will be used in further sections. Within the proof of Thm. 2.3, a reference to the nth property stated within the same theorem will have the shape 'by n'.

Theorem 2.3 *The following properties are valid in all relation algebras for all relations R, S, T, F, G and I:*

(1) $R;0 = 0;R = 0$.
(2) $\breve{1} = 1$.
(3) $1;1 = 1$.
(4) $(R+S)^{\smile} = \breve{R}+\breve{S}$.
(5) $(R\cdot S)^{\smile} = \breve{R}\cdot\breve{S}$.
(6) *If* $R \leq 1'$ *then* $\breve{R} = R$.
(7) *If* $R, S \leq 1'$ *then* $R;S = R\cdot S$.
(8) *If* $R \leq 1'$ *then* $(R;1)\cdot S = R;S$ *and* $(1;R)\cdot S = S;R$.
(9) *If* $F+G = 1'$ *and* $F\cdot G = 0$, *then* $\overline{F;1} = G;1$.
(10) *Dom* $(R) = (R;1)\cdot 1'$ *and Ran* $(R) = (1;R)\cdot 1'$.
(11) *Dom* $(R)\,;R = R$ *and* $R;$*Ran* $(R) = R$.
(12) *Dom* $(R+S) =$ *Dom* (R) $+$*Dom* (S), *i.e., Dom is additive. Similarly, Ran* $(R+S) =$ *Ran* (R) $+$*Ran* (S).

(13) $Dom\left(\breve{R}\right) = Ran\,(R)$ *and* $Ran\left(\breve{R}\right) = Dom\,(R)$.
(14) $R;1 = Dom\,(R)\,;1$ *and* $1;R = 1;Ran\,(R)$.
(15) $\breve{\overline{R}} = \overline{\breve{R}}$.
(16) $(R{\cdot}S)\,;T \leq (R;T) \cdot (S;T)$ *and* $R;\,(S{\cdot}T) \leq (R;S) \cdot (R;T)$.
(17) *If* F *is a functional relation then* $F;\,(R{\cdot}S) = (F;R) \cdot (F;S)$.
(18) *If* F *is a functional relation,* $G \leq F$, *and* $Dom\,(G) = Dom\,(F)$ *then* $G = F$.
(19) *If* F *is a functional relation then* $Dom\,(F)\,;\overline{F;R} = F;\overline{R}$.
(20) *If* I *is an injective relation then* $(R{\cdot}S)\,;I = (R;I) \cdot (S;I)$.
(21) *If* I *is an injective relation then* $\overline{R;I};Ran\,(I) = \overline{R};I$.
(22) *If* $F \leq 1'$ *then* $F;R \cdot S = F;\,(R{\cdot}S)$ *and* $R;F \cdot S = (R{\cdot}S)\,;F$.

Proof

1. See [L. Chin et al. (1951)] Cor. 2.4.
2. See [L. Chin et al. (1951)] Thm. 1.7.
3. See [L. Chin et al. (1951)] Thm. 2.6.
4. See [L. Chin et al. (1951)] Thm. 1.11.
5. See [L. Chin et al. (1951)] Thm. 1.9.
6. See [L. Chin et al. (1951)] Thms. 3.2 and 3.5.
7. See [L. Chin et al. (1951)], Cor. 3.12.
8. By monotonicity and Ax. 5

$$R;S \leq R;1 \quad \text{and} \quad R;S \leq 1';S = S\ .$$

Then,

$$R;S \leq (R;1)\cdot S\ .$$

We also have

$$\begin{aligned}
R;S &= (R{\cdot}R)\,;S && \text{(BA)}\\
&= (R;R)\,;S && \text{(by 7)}\\
&= \left(R;\breve{R}\right);S && \text{(by 6)}\\
&= R;\left(\breve{R};S\right) && \text{(by Ax. 1)}\\
&\geq (R \cdot S;\breve{1});(1 \cdot \breve{R};S) && \text{(by monotonicity)}\\
&\geq (R;1)\cdot S\ . && \text{(by (2.2))}
\end{aligned}$$

The other case is proved similarly.

9. In order to prove this property, we will show that $G;1$ is the complement of $F;1$.

$$\begin{aligned}(F;1)\cdot(G;1)=0 &\iff \breve{F};G;1 \cdot 1=0 && \text{(Ax. 7)}\\ &\iff F;G;1=0 && \text{(BA)}\\ &\iff (F\cdot G);1=0 && \text{(by 7)}\\ &\iff 0=0\ . && \text{(Hyp.)}\end{aligned}$$

We also have

$$\begin{aligned}F;1 + G;1 &= (F+G);1 && \text{(Ax. 2)}\\ &= 1';1 && \text{(Hyp.)}\\ &= 1\ . && \text{(Ax. 5)}\end{aligned}$$

From $(F;1)\cdot(G;1)=0$ and $(F;1)+(G;1)=1$, we deduce $\overline{F;1}=G;1$.

10. We will prove that $Dom(R) \le (R;1)\cdot 1'$ and $Dom(R) \ge (R;1)\cdot 1'$

$$\begin{aligned}Dom(R) &= \left(R;\breve{R}\right)\cdot 1' && \text{(by Def. } Dom)\\ &\le (R;1)\cdot 1'\ . && \text{(by monotonicity)}\end{aligned}$$

$$\begin{aligned}Dom(R) &= \left(R;\breve{R}\right)\cdot 1' && \text{(by Def. } Dom)\\ &= \left((R\cdot 1';1);(1\cdot \breve{R};1')\right)\cdot 1' && \text{(by Ax. 5 and BA)}\\ &\ge (R;1)\cdot 1'\cdot 1' && \text{(by (2.2) and 2)}\\ &= (R;1)\cdot 1'\ . && \text{(BA)}\end{aligned}$$

The proof for *Ran* follows in a similar way.

11. The proof follows by 10 and [G. Schmidt et al. (1993)] Prop. 2.4.2.

12.

$$\begin{aligned}Dom(R+S) &= ((R+S);1)\cdot 1' && \text{(by 10)}\\ &= (R;1 + S;1)\cdot 1' && \text{(by Ax. 2)}\\ &= ((R;1)\cdot 1') + ((S;1)\cdot 1') && \text{(BA)}\\ &= Dom(R) + Dom(S)\ . && \text{(by 10)}\end{aligned}$$

The proof for *Ran* follows in a similar way.

13.

$$\begin{aligned} Dom\left(\breve{R}\right) &= \left(\breve{R};\breve{\breve{R}}\right)\cdot 1' && \text{(by Def. } Dom) \\ &= \left(\breve{R};R\right)\cdot 1' && \text{(by Ax. 4)} \\ &= Ran\,(R)\ . && \text{(by Def. } Ran) \end{aligned}$$

The proof for $Ran\left(\breve{R}\right)$ follows in a similar way.

14. We will show that $R;1 = Dom\,(R)\,;1$. The case with *Ran* is proved analogously.

$$\begin{aligned} Dom\,(R)\,;1 &\geq\ Dom\,(R)\,;R;1 && \text{(by monotonicity)} \\ &= R;1\ . && \text{(by 11)} \end{aligned}$$

Also,

$$\begin{aligned} Dom\,(R)\,;1 &= ((R;1)\cdot 1')\,;1 && \text{(by 10)} \\ &\leq\ \left(R\cdot 1';1\right);\left(1\cdot \breve{R};1'\right);1 && \text{(by (2.2) and 2)} \\ &\leq\ R;1;1 && \text{(by monotonicity)} \\ &= R;1\ . && \text{(by 3)} \end{aligned}$$

15. See [L. Chin et al. (1951)] Thm. 1.10.

16. By monotonicity we have

$$(R\cdot T)\,;S\ \leq\ R;S \quad\text{and}\quad (R\cdot T)\,;S\ \leq\ T;S\ .$$

Thus,

$$(R\cdot T)\,;S\ \leq\ (R;S)\cdot(T;S)\ .$$

17. See [L. Chin et al. (1951)] Thm. 4.2.

18. See [G. Schmidt et al. (1993)] Prop. 4.2.2 (iv).

19. In order to prove this result we will use the following property of Boolean algebras. Let R, S and T be arbitrary, then

$$R\cdot S = 0 \text{ and } R+S = T \ \text{ implies }\ R = T\cdot\overline{S} \text{ and } S = T\cdot\overline{R}\ . \quad (2.5)$$

We will begin by proving that the hypothesis of (2.5) are satisfied for suitable instantiations of R, S and T.

$$\begin{aligned} F;\overline{R} \cdot F;R &= F;\left(\overline{R}\cdot R\right) && \text{(by 17)}\\ &= F;0 && \text{(by BA)}\\ &= 0\,. && \text{(by 1)} \end{aligned}$$

$$\begin{aligned} F;\overline{R} + F;R &= F;\left(\overline{R}+R\right) && \text{(by Ax. 2)}\\ &= F;1 && \text{(by BA)}\\ &= Dom(F)\,;1\,. && \text{(by 14)} \end{aligned}$$

Then, once the hypothesis that allow the application of (2.5) has been established, we proceed as follows:

$$\begin{aligned} Dom(F)\,;\overline{F;R} &= Dom(F)\,;1 \cdot \overline{F;R} && \text{(by 8)}\\ &= F;\overline{R}\,. && \text{(by (2.5))} \end{aligned}$$

20. The rationale of most proofs involving injective relations consists of transforming the original property to a related property of functional relations. This property is usually obtained by applying the converse operator twice to some expression, which yields an expression equivalent to the original one. Once this new property has been stated, known properties of functional relations are used.
For this specific case, since I is an injective relation, $\breve{I}$ is functional. Then,

$$\begin{aligned} (R\cdot S)\,;I &= (((R\cdot S)\,;I)^{\smile})^{\smile} && \text{(by Ax. 4)}\\ &= \left(\breve{I};(R\cdot S)^{\smile}\right)^{\smile} && \text{(by Ax. 6)}\\ &= \left(\breve{I};\left(\breve{R}\cdot\breve{S}\right)\right)^{\smile} && \text{(by 5)}\\ &= \left(\left(\breve{I};\breve{R}\right)\cdot\left(\breve{I};\breve{S}\right)\right)^{\smile} && \text{(by 17)}\\ &= \left(\breve{I};\breve{R}\right)^{\smile}\cdot\left(\breve{I};\breve{S}\right)^{\smile} && \text{(by 5)}\\ &= \left(\breve{\breve{R}};\breve{\breve{I}}\right)\cdot\left(\breve{\breve{S}};\breve{\breve{I}}\right) && \text{(by Ax. 6)}\\ &= (R;I)\cdot(S;I)\,. && \text{(by Ax. 4)} \end{aligned}$$

21.

$$\begin{aligned}
\overline{R;I};Ran\,(I) &= ((\overline{R;I};Ran\,(I))^{\smile})^{\smile} && \text{(by Ax. 4)}\\
&= ((Ran\,(I))^{\smile};\overline{R;I}^{\smile})^{\smile} && \text{(by Ax. 6)}\\
&= \Big((Ran\,(I))^{\smile};\overline{(R;I)^{\smile}}\Big)^{\smile} && \text{(by 15)}\\
&= \Big((Ran\,(I))^{\smile};\overline{\breve{I};\breve{R}}\Big)^{\smile} && \text{(by Ax. 6)}\\
&= \Big((Ran\,(I))^{\smile};\breve{I};\overline{\breve{R}}\Big)^{\smile} && \text{(by 19)}\\
&= \Big((Ran\,(I))^{\smile};\breve{I};\breve{\overline{R}}\Big)^{\smile} && \text{(by 15)}\\
&= \breve{\breve{\overline{R}}};\breve{\breve{I}};((Ran\,(I))^{\smile})^{\smile} && \text{(by Ax. 6)}\\
&= \overline{R};I;Ran\,(I) && \text{(by Ax. 4)}\\
&= \overline{R};I\ . && \text{(by 11)}
\end{aligned}$$

22. Since $F \leq 1'$, F is a functional relation. Then, $F;(R{\cdot}S) = F;R \cdot F;S$. By monotonicity,

$$F;R \cdot S \leq F;R\ . \tag{2.6}$$

Also,

$$\begin{aligned}
F;R \cdot S &\leq F;1 \cdot S && \text{(by monotonicity)}\\
&= F;S\ . && \text{(by 8)}
\end{aligned}$$

Then,

$$F;R \cdot S \leq F;S\ . \tag{2.7}$$

Thus, by (2.6), (2.7) and BA,

$$F;R \cdot S \leq F;R \cdot F;S\ . \tag{2.8}$$

On the other hand,

$$\begin{aligned}
F;R \cdot F;S &\leq F;R \cdot 1';S && \text{(by monotonicity)}\\
&= F;R \cdot S\ . && \text{(by Ax. 5)}
\end{aligned}$$

Then,

$$F;R \cdot F;S \leq F;R \cdot S\ . \tag{2.9}$$

Joining (2.8) and (2.9),

$$F;R \cdot F;S = F;R \cdot S \ .$$

The case when F appears on the right hand side is proved analogously.

$\square$

As a source for additional arithmetical properties of relation algebras, we direct the reader to any of [L. Chin et al. (1951); G. Schmidt et al. (1993); A. Tarski (1941)].

Chapter 3
Proper and Abstract Fork Algebras

3.1 On the Origin of Fork Algebras

Let us recall formula (2.4):

$$\forall x \forall y \forall z \exists u \, (u 0\text{'} x \;\wedge\; u 0\text{'} y \;\wedge\; u 0\text{'} z)\,.$$

We have already mentioned in a previous chapter that it is not equivalent to any sentence of the calculus of relations. In order to overcome this limitation, it seems enough to have some operator ∇ and a binary function $\star$ satisfying the following equivalence:

$$uRx \;\wedge\; uRy \qquad \Longleftrightarrow \qquad uR\nabla R \star (x, y)\ . \tag{3.1}$$

Under these conditions we can proceed as follows:

$$\begin{array}{ll}
 & \forall x \forall y \forall z \exists u \, (u 0\text{'} x \wedge u 0\text{'} y \wedge u 0\text{'} z) \\
\Longleftrightarrow & \{\, \text{by } (3.1) \,\} \\
 & \forall x \forall y \forall z \exists u \, (u 0\text{'} x \;\wedge\; u 0\text{'} \nabla 0\text{'} \star (y, z)) \\
\Longleftrightarrow & \{\, \text{by def. of } \breve{} \,\} \\
 & \forall x \forall y \forall z \exists u \left(x \breve{0\text{'}} u \;\wedge\; u 0\text{'} \nabla 0\text{'} \star (y, z) \right) \\
\Longleftrightarrow & \{\, \text{by def. of } ; \,\} \\
 & \forall x \forall y \forall z \left(x \breve{0\text{'}} ; (0\text{'} \nabla 0\text{'}) \star (y, z) \right) \\
\Longleftrightarrow & \{\, \text{by elementary logic} \,\} \\
 & \forall x \forall y \forall z \left(x \breve{0\text{'}} ; (0\text{'} \nabla 0\text{'}) \star (y, z) \iff \mathit{true} \right)
\end{array}$$

$$\begin{aligned}
\iff\quad & \{\,\text{by def. of } 1\,\}\\
& \forall x \forall y \forall z \left(x \breve{0'} \,;\, (0' \nabla 0') \star (y,z) \iff x 1 y \;\wedge\; x 1 z\right)\\
\iff\quad & \{\,\text{by } (3.1)\,\}\\
& \forall x \forall y \forall z \left(x \breve{0'} \,;\, (0' \nabla 0') \star (y,z) \iff x 1 \nabla 1 \star (y,z)\right)\\
\iff\quad & \{\boxed{\text{by def. of} =}\}\\
& \breve{0'} \,;\, (0' \nabla 0') = 1 \nabla 1\ .
\end{aligned}$$

Even though intuitively clear, the framed justification above needs an extra assumption, namely, that all elements that can appear in the range of $R \nabla S$ are indeed of the form $\star(a,b)$, for suitable a and b. Thus, we define the operator ∇ (called *fork*) by the following formula from the elementary theory of binary relations:

$$\forall x \forall y \left(x R \nabla S y \iff \exists u \exists v \left(y = \star(u,v) \;\wedge\; x R u \;\wedge\; x S v\right)\right). \qquad (3.2)$$

The development of the classes of proper and abstract fork algebras (to be introduced in Section 3.2) evolved around the meaning of the notation $\star(x,y)$. The study on fork algebras begun in [A. Haeberer et al. (1991)] when looking for a framework adequate as the foundation of a calculus for system specification, construction and verification. There, fork algebras are built using *finite trees*, i.e., the notation $\star(x,y)$ meant the tree with subtrees x and y. In [P. Veloso et al. (1991)], the definition is changed and *finite strings* are used instead of trees (the notation $\star(x,y)$ then meant the concatenation of strings x and y). Thus, the base of a fork algebra changed from a free groupoid to a free monoid. In [P. Veloso et al. (1992)], the base set is once again made out of finite trees. In all the previously mentioned articles, no axiomatization of the class of abstract fork algebras is given, but rather some valid properties are stated (an incomplete set of properties in every case). It was proved by Mikulás, Sain and Simon [S. Mikulás et al. (1992)] that the class of ABR extended with an operator ∇ defined as in (3.2) with $\star(x,y)$ being either concatenation or binary tree formation is not finitely axiomatizable. It is in [A. Haeberer et al. (1993)a] and its published version [A. Haeberer et al. (1993)b] where the current axiomatization for abstract fork algebras is first used, since the version presented in [P. Veloso et al. (1993)] had an extra non-equational axiom necessary in order to achieve representability. On the other hand, [P. Veloso et al. (1993)] use the current definition of proper fork algebras (with $\star(x,y)$ denoting the application of a binary injective mapping to x and y), while [A. Haeberer et al. (1993)a] and [A. Haeberer et al. (1993)b] still resort to finite trees.

In order to complete the definition of fork given in (3.2), we will request $\star$ to be an injective mapping, i.e., it must satisfy the sentence:

$$\forall x \forall y \forall u \forall v \, (\star(x,y) = \star(u,v) \quad \Rightarrow \quad x = u \, \wedge \, y = v) \ . \tag{3.3}$$

In the following sections we get into the technical details in order to define the classes of proper (also called standard) fork algebras and abstract fork algebras.

3.2 Definition of the Classes

The class of proper fork algebras (PFA for short) is the extension of the class of algebras of binary relations [B. Jónsson et al. (1952); A. Tarski (1941)] with fork. The operator fork induces a structure on the base of proper fork algebras. The objects, instead of being binary relations on a plain set, are binary relations on a structured domain $\langle A, * \rangle$, where $*$ is, by (3.3), an injective binary function on A.

Definition 3.1 A *star proper fork algebra* is a two-sorted algebraic structure $\langle R, U, \cup, \cap, ^-, \emptyset, E, \circ, Id, ^\smile, \underline{\nabla}, * \rangle$ with domains R and U, such that:

(1) $\langle R, \cup, \cap, ^-, \emptyset, E, \circ, Id, ^\smile \rangle$ is an algebra of binary relations on the set U,

(2) $* : U \times U \longrightarrow U$ is a binary function that is injective on the restriction of its domain to E,

(3) R is closed under fork of binary relations, defined by:

$$S \underline{\nabla} T = \{ \langle x, *(y,z) \rangle : xSy \ \wedge \ xTz \} \ .$$

A graphical interpretation of $\underline{\nabla}$ is given in Fig. 3.1. Notice that in Fig. 3.1, in order to picture the fork of relations R and S, we use a two-dimensional notation. Such notation will be used intensively throughout the book in order to obtain shorter and (hopefully) more readable expressions.

In Def. 3.1(2), notice that E is a binary relation on U, and therefore the restriction of $U \times U$ to E is adequate. Proper fork algebras are obtained as reducts (some operations and domains are forgotten) of star proper fork algebras.

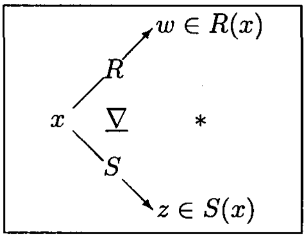

Fig. 3.1 The operator fork.

Definition 3.2 We define the class of proper fork algebras (denoted by PFA) as **Rd**$\star$PFA, where the operation **Rd** takes reducts to the similarity type $\langle \cup, \cap, ^-, \emptyset, U \times U, \circ, Id, ^\smile, \underline{\nabla} \rangle$.

Notice that proper fork algebras are obtained from star proper fork algebras by forgetting the domain U and the function $*$.

Definition 3.3 A proper fork algebra is *full* if its universe is of the form $\mathcal{P}(U \times U)$ for some set U, and is *square* if its largest relation is of the form $U \times U$.

It follows immediately from Def. 3.3 that every full proper fork algebra is square, and a square proper fork algebra whose largest relation is $U \times U$ is a subalgebra of the full proper fork algebra with universe $\mathcal{P}(U \times U)$. We will denote the class of full proper fork algebras by FullPFA and the class of square proper fork algebras by SPFA.

Theorem 3.1 *Given a proper fork algebra $\mathfrak{A}$, there exist an index set I and full proper fork algebras $(\mathfrak{A}_i)_{i \in I}$ such that*

$$\mathfrak{A} \preceq \prod_{i \in I} \mathfrak{A}_i \, .$$

Proof Follow the lines of the proof of Thm. 2.1 and prove that h is a one-to-one fork algebra homomorphism. □

In Def. 3.1 the function $*$ performs the role of pairing, encoding pairs of objects into single objects. It is important to notice that there are $*$ functions which are distinct from set-theoretical pair formation, i.e., $*(x, y)$ differs from $\{ x, \{ x, y \} \}$.

Notice that in order to define a FullPFA $\mathfrak{A}$, it suffices to provide the set $B_{\mathfrak{A}}$ and an injective mapping $* : B_{\mathfrak{A}} \times B_{\mathfrak{A}} \to B_{\mathfrak{A}}$.

Given a PFA $\mathfrak{A}$ with base $B_{\mathfrak{A}}$, it is possible to single out those elements that do not represent pairs (if there are any). Notice that the term $\overline{1\nabla 1}$ stands for the binary relation

$$\{\langle x,y\rangle : x\in B_{\mathfrak{A}} \ \wedge\ \forall u,v\in B_{\mathfrak{A}}\,(y\neq *(u,v))\,\}\ .$$

Thus, the term $Ran\left(\overline{1\nabla 1}\right)$ distinguishes those elements from the base that are not pairs. In what follows we will denote by $1'_{\mathsf{U}}$ the term $Ran\left(\overline{1\nabla 1}\right)$, by 1_{U} the term $1;1'_{\mathsf{U}}$ and by ${}_{\mathsf{U}}1$ the term $1'_{\mathsf{U}};1$. We will call the elements from the base in the domain of $1'_{\mathsf{U}}$ *urelements*, and will denote the set of urelements of a fork algebra $\mathfrak{A}$ by $Urel_{\mathfrak{A}}$.

Under the previous definitions, the equation

$$1;1'_{\mathsf{U}};1=1 \tag{3.4}$$

is valid in a proper fork algebra $\mathfrak{A}$ only in case $Urel_{\mathfrak{A}}$ is nonempty. We will denote by PFAU the subclass of proper fork algebras with a nonempty set of urelements (i.e., satisfying (3.4)). Also, we denote by SPFAU the class of square proper fork algebras with urelements.

In the proof of several theorems to come, it will be necessary to explicitly construct proper fork algebras. By the 'full fork algebra with set of urelements U', we mean the following construction:

(1) Construct the absolutely free groupoid $\langle U^*, *\rangle$ with set of generators U
(2) Construct the full fork algebra on the set U^*, the operation $\underline{\nabla}$ being defined by the condition $R\underline{\nabla}S=\{\langle x,*(y,x)\rangle : xRy \text{ and } xSz\}$

As a particular instance of the application of the operator fork, we have the relation $Id\underline{\nabla}Id$. When this relation is interpreted in a proper fork algebra, it produces two copies of a given input element. Fig. 3.2 illustrates its definition. This relation will be denoted by $\mathit{2}$.

Given a pair of binary relations, the operation called *cross* (and denoted by $\otimes$) performs a kind of parallel product. A graphic representation of cross is given in Fig. 3.3. Its set theoretical definition is given by

$$R\otimes S=\{\langle *(x,y),*(w,z)\rangle : xRw\wedge ySz\}\,.$$

It is not difficult to check that cross is definable from the other relational operators with the use of fork. It is a simple exercise to show that if V is

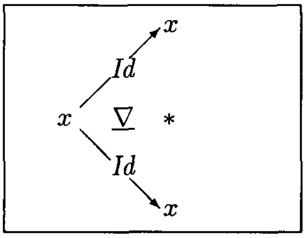

Fig. 3.2 The relation $\mathcal{2}$.

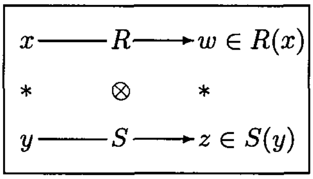

Fig. 3.3 The operator cross.

the greatest relation in a proper fork algebra, then

$$R \otimes S = ((Id \underline{\nabla} V)^{\smile} \circ R) \underline{\nabla} ((V \underline{\nabla} Id)^{\smile} \circ S).$$

Much the same as relation algebras are an abstract version of algebras of binary relations, proper fork algebras also have their abstract counterpart, the class of abstract fork algebras (AFA). As we will see in the next definition, the class of abstract fork algebras can be axiomatized with a finite set of equations, and therefore is a finitely based variety.

Definition 3.4 An abstract fork algebra is an algebraic structure

$$\langle R, +, \cdot, ^{-}, 0, 1, ;, 1', ^{\smile}, \nabla \rangle,$$

where $\langle R, +, \cdot, ^{-}, 0, 1, ;, 1', ^{\smile} \rangle$ is a relation algebra and for all $r, s, t, q \in R$,

$$r \nabla s = (r ; (1' \nabla 1)) \cdot (s ; (1 \nabla 1')), \tag{Ax. 8}$$

$$(r \nabla s) ; (t \nabla q)^{\smile} = (r ; \breve{t}) \cdot (s ; \breve{q}), \tag{Ax. 9}$$

$$(1'\nabla 1)^{\smile}\nabla(1\nabla 1')^{\smile} \leq 1'. \qquad \text{(Ax. 10)}$$

The class of simple abstract fork algebras (SAFA) contains those abstract fork algebras satisfying formula (2.1). Those abstract fork algebras that satisfy (3.4) are said to have urelements, and their class is denoted by AFAU. The class of simple abstract fork algebras with urelements is denoted by SAFAU. *At*FAU denotes the class of *atomic* fork algebras with urelements (i.e., fork algebras with urelements whose Boolean reduct is an atomic Boolean algebra), and *At*SFAU denotes the class of atomic and simple fork algebras with urelements.

From the abstract definition of fork induced by the axioms in Def. 3.4, it is possible to define cross by the equation

$$R\otimes S = ((1'\nabla 1)^{\smile};R)\ \nabla\ ((1\nabla 1')^{\smile};S)\ . \qquad (3.5)$$

When interpreted in a proper fork algebra, the relations $(1'\nabla 1)^{\smile}$ and $(1\nabla 1')^{\smile}$ behave as projections (they are actually called quasi-projections), projecting components from pairs constructed with an injective function $*$. We call them π and ρ respectively. They will allow us to cope in further sections with the lack of variables over individuals in the language of abstract fork algebras. Figure 3.4 illustrates the meaning of these relations.

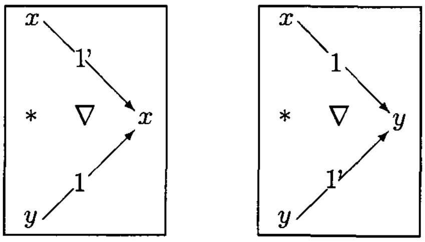

Fig. 3.4 The projections π and ρ.

Notice that under the previous definitions of π and ρ, (3.5) can be spelt in a simpler form as follows:

$$R\otimes S = (\pi;R)\ \nabla\ (\rho;S)\ .$$

3.3 Arithmetical Properties

Theorem 3.2 *The following properties hold in all fork algebras for all relations F, I, R, S, T and U.*

(1) $(R \nabla S)\,;\breve{2} = R \cdot S$.
(2) $(R \nabla S)\,;\pi = \mathit{Dom}\,(S)\,;R$ *and* $(R \nabla S)\,;\rho = \mathit{Dom}\,(R)\,;S$.
(3) $R;(S \nabla T) \leq (R;S) \nabla (R;T)$.
(4) *Let F be functional, then* $F;(R \nabla S) = (F;R) \nabla (F;S)$.
(5) *If* $F \leq 1'$ *then* $(F;R) \nabla S = F;(R \nabla S)$.
(6) $(R \nabla S) \cdot (T \nabla U) = (R \cdot T) \nabla (S \cdot U)$.
(7) $(R \otimes S)^{\smile} = \breve{R} \otimes \breve{S}$.
(8) $(R \otimes S) \cdot (T \otimes U) = (R \cdot T) \otimes (S \cdot U)$.
(9) $(R \nabla S)\,;(T \otimes U) = (R;T) \nabla (S;U)$.
(10) $(R \otimes S)\,;(T \otimes U) = (R;T) \otimes (S;U)$.
(11) $(R+S) \otimes T = (R \otimes T) + (S \otimes T)$, *i.e.,* $\otimes$ *is additive. Similarly,* $R \otimes (S+T) = (R \otimes S) + (R \otimes T)$.
(12) $(R \otimes 1')\,;\pi = \pi;R$ *and* $(1' \otimes R)\,;\rho = \rho;R$.
(13) *The relations π and ρ are functional.*
(14) $\breve{\pi};\rho = 1$.
(15) $\mathit{Dom}\,(\pi) = \mathit{Dom}\,(\rho) = 1' \otimes 1'$.
(16) $\mathit{Dom}\,(\pi;R) = \mathit{Dom}\,(R) \otimes 1'$ *and* $\mathit{Dom}\,(\rho;R) = 1' \otimes \mathit{Dom}\,(R)$.
(17) $(\breve{R} \otimes 1');\breve{2} = \mathit{Dom}\,((1' \otimes R)\,;\breve{2})\,;\rho$.
(18) *Let F be functional, then* $\breve{\pi} \cdot 1;(1' \nabla F) = 1' \nabla F$.
(19) *If I is injective, then* $(1' \otimes R;I)\,;\breve{2} = \mathit{Dom}\left((\breve{I} \otimes R);\breve{2}\right);\pi$ *and* $(R;I \otimes 1')\,;\breve{2} = \mathit{Dom}\left((R \otimes \breve{I});\breve{2}\right);\rho$.
(20) $(1' \otimes 1')\,;\overline{R \otimes S};(1' \otimes 1') = \left(\overline{R} \otimes 1\right) + \left(1 \otimes \overline{S}\right)$.

Proof

1. If we recall that 2 equals $1' \nabla 1'$, then the property follows immediately applying Ax. 9.
2. We will prove the first property, namely, that

$$(R \nabla S)\,;\pi = \mathit{Dom}\,(S)\,;R\,.$$

We then proceed as follows.

$$\begin{aligned}(R\nabla S)\,;\pi &= (R\nabla S)\,;(1\text{'}\nabla 1)^{\smile} && \text{(by Def. }\pi)\\ &= R;\breve{1\text{'}}\cdot S;\breve{1} && \text{(by Ax. 9)}\\ &= R\cdot S;1 && \text{(by Thms. 2.3.6 and 2.3.2)}\\ &= R\cdot \mathit{Dom}\,(S)\,;1 && \text{(by Thm. 2.3.14)}\\ &= \mathit{Dom}\,(S)\,;R. && \text{(by Thm. 2.3.8)}\end{aligned}$$

The case with ρ follows in a similar way.

3.

$$\begin{aligned}R;(S\nabla T) &= R;(S;\breve{\pi}\cdot T;\breve{\rho}) && \text{(by Ax. 8)}\\ &\leq (R;S;\breve{\pi})\cdot(R;T;\breve{\rho}) && \text{(by Thm. 2.3.16)}\\ &= (R;S)\,\nabla\,(R;T)\,. && \text{(by Ax. 8)}\end{aligned}$$

4.

$$\begin{aligned}F;(R\nabla S) &= F;(R;\breve{\pi}\cdot S;\breve{\rho}) && \text{(by Ax. 8)}\\ &= (F;R;\breve{\pi})\cdot(F;S;\breve{\rho}) && \text{(by Thm. 2.3.17)}\\ &= (F;R)\,\nabla\,(F;S)\,. && \text{(by Ax. 8)}\end{aligned}$$

5.

$$\begin{aligned}F;R\ \nabla\ S &= F;R;\breve{\pi}\cdot S;\breve{\rho} && \text{(by Ax. 8)}\\ &= F;(R;\breve{\pi}\cdot S;\breve{\rho}) && \text{(by Thm. 2.3.22)}\\ &= F;(R\nabla S)\,. && \text{(by Ax. 8)}\end{aligned}$$

6.

$$\begin{aligned}(R\nabla S)\cdot(T\nabla U) &= R;\breve{\pi}\cdot S;\breve{\rho}\cdot T;\breve{\pi}\cdot U;\breve{\rho} && \text{(by Ax. 8)}\\ &= R;\breve{\pi}\cdot T;\breve{\pi}\cdot S;\breve{\rho}\cdot U;\breve{\rho} && \text{(by BA)}\\ &= (R\cdot T)\,;\breve{\pi}\cdot(S\cdot U)\,;\breve{\rho} && \text{(by Thm. 2.3.20)}\\ &= (R\cdot T)\,\nabla\,(S\cdot U)\,. && \text{(by Ax. 8)}\end{aligned}$$

7.

$$
\begin{aligned}
(R\otimes S)^{\smile} &= (\pi;R \nabla \rho;S)^{\smile} && \text{(by (3.5))}\\
&= (\pi;R;\breve{\pi} \cdot \rho;S;\breve{\rho})^{\smile} && \text{(by Ax. 8)}\\
&= (\pi;R;\breve{\pi})^{\smile}\cdot(\rho;S;\breve{\rho})^{\smile} && \text{(by Thm. 2.3.5)}\\
&= \breve{\breve{\pi}};\breve{R};\breve{\pi} \cdot \breve{\breve{\rho}};\breve{S};\breve{\rho} && \text{(by Ax. 6)}\\
&= \pi;\breve{R};\breve{\pi} \cdot \rho;\breve{S};\breve{\rho} && \text{(by Ax. 4)}\\
&= \left(\pi;\breve{R}\right) \nabla \left(\rho;\breve{S}\right) && \text{(by Ax. 8)}\\
&= \breve{R}\otimes\breve{S}. && \text{(by (3.5))}
\end{aligned}
$$

8.

$$
\begin{aligned}
(R\otimes S)\cdot(T\otimes U) &= ((\pi;R) \nabla (\rho;S)) \cdot ((\pi;T) \nabla (\rho;U)) && \text{(by (3.5))}\\
&= \pi;R;\breve{\pi} \cdot \rho;S;\breve{\rho} \cdot \pi;T;\breve{\pi} \cdot \rho;U;\breve{\rho} && \text{(by Ax. 8)}\\
&= \pi;(R\cdot T);\breve{\pi} \cdot \rho;(S\cdot U);\breve{\rho} &&\\
& && \text{(by Thms. 2.3.17, 2.3.20)}\\
&= (\pi;(R\cdot T)) \nabla (\rho;(S\cdot U)) && \text{(by Ax. 8)}\\
&= (R\cdot T)\otimes(S\cdot U). && \text{(by (3.5))}
\end{aligned}
$$

9.

$$
\begin{aligned}
(R\nabla S);(T\otimes U) &= (R\nabla S);((T\otimes U)^{\smile})^{\smile} && \text{(by Ax. 4)}\\
&= (R\nabla S);(\breve{T}\otimes\breve{U})^{\smile} && \text{(by 7)}\\
&= (R\nabla S);(\pi;\breve{T} \nabla \rho;\breve{U})^{\smile} && \text{(by (3.5))}\\
&= \left(R;(\pi;\breve{T})^{\smile}\right)\cdot\left(S;(\rho;\breve{U})^{\smile}\right) && \text{(by Ax. 9)}\\
&= (R;T;\breve{\pi})\cdot(S;U;\breve{\rho}) && \text{(by Ax. 4, Ax. 6)}\\
&= (R;T)\nabla(S;U). && \text{(by Ax. 8)}
\end{aligned}
$$

10.

$$
\begin{aligned}
(R\otimes S);(T\otimes U) &= ((\pi;R) \nabla (\rho;S));(T\otimes U) && \text{(by (3.5))}\\
&= (\pi;R;T) \nabla (\rho;S;U) && \text{(by 9)}\\
&= (R;T)\otimes(S;U). && \text{(by (3.5))}
\end{aligned}
$$

11.

$$\begin{aligned}
(R+S)\otimes T &= (\pi;(R+S))\,\nabla\,(\rho;T) && \text{(by (3.5))}\\
&= \pi;(R+S);\breve{\pi}\cdot\rho;T;\breve{\rho} && \text{(by Ax. 8)}\\
&= (\pi;R;\breve{\pi}+\pi;S;\breve{\pi})\cdot\rho;T;\breve{\rho} && \text{(by Ax. 2)}\\
&= (\pi;R;\breve{\pi}\cdot\rho;T;\breve{\rho})+(\pi;S;\breve{\pi}\cdot\rho;T;\breve{\rho}) && \text{(BA)}\\
&= ((\pi;R)\,\nabla\,(\rho;T))+((\pi;S)\,\nabla\,(\rho;T)) && \text{(by Ax. 8)}\\
&= (R\otimes T)+(S\otimes T). && \text{(by (3.5))}
\end{aligned}$$

The other case follows in a similar way.

12.

$$\begin{aligned}
(R\otimes 1');\pi &= (\pi;R\,\nabla\,\rho);\pi && \text{(by (3.5))}\\
&= (\pi;R\,\nabla\,\rho);(1'\nabla 1)^{\smile} && \text{(by Def. }\pi)\\
&= \pi;R;\breve{1'}\cdot\rho;\breve{1} && \text{(by Ax. 9)}\\
&= \pi;R. && \text{(by Thm. 2.3.16)}
\end{aligned}$$

The case with ρ follows in a similar way.

13.

$$\begin{aligned}
\breve{\pi};\pi &= ((1'\nabla 1)^{\smile})^{\smile};(1'\nabla 1)^{\smile} && \text{(by Def. }\pi)\\
&= (1'\nabla 1);(1'\nabla 1)^{\smile} && \text{(by Ax. 4)}\\
&= 1';\breve{1'}\cdot 1;\breve{1} && \text{(by Ax. 9)}\\
&= 1';1'\cdot 1;1 && \text{(by Thms. 2.3.6 and 2.3.2)}\\
&= 1'\cdot 1 && \text{(by Ax. 5 and Thm. 2.3.3)}\\
&= 1'. && \text{(by BA)}
\end{aligned}$$

The proof for ρ is analogous.

14.

$$\begin{aligned}
\breve{\pi};\rho &= ((1'\nabla 1)^{\smile})^{\smile};(1\nabla 1')^{\smile} && \text{(by Defs. }\pi\text{ and }\rho)\\
&= (1'\nabla 1);(1\nabla 1')^{\smile} && \text{(by Ax. 4)}\\
&= (1';\breve{1})\cdot(1;\breve{1'}) && \text{(by Ax. 9)}\\
&= (1';1)\cdot(1;1') && \text{(by Thms. 2.3.2 and 2.3.6)}\\
&= 1\cdot 1 && \text{(by Ax. 5)}\\
&= 1. && \text{(by BA)}
\end{aligned}$$

15. In order to show that $Dom(\pi) = 1' \otimes 1'$ we will show that $\pi;1 = (1' \otimes 1');1$.

$$\begin{aligned}
\leq)\ \pi;1 &= (1' \otimes 1');\pi;1 && \text{(by 9)}\\
&\leq (1' \otimes 1');1;1 && \text{(by monotonicity)}\\
&= (1' \otimes 1');1. && \text{(by Thm. 2.3.3)}
\end{aligned}$$

$$\begin{aligned}
\geq)\ \pi;1 &= \pi;1;1 && \text{(by Thm. 2.3.3)}\\
&= (\pi;1 \cdot 1);1 && \text{(by BA)}\\
&\geq (\pi;\breve{\pi} \cdot \rho;\breve{\rho});1 && \text{(by monotonicity)}\\
&= (1' \otimes 1');1. && \text{(by (3.5))}
\end{aligned}$$

The proof for ρ is analogous.

16.

$$\begin{aligned}
Dom(\pi;R) &= (\pi;R;1) \cdot 1' && \text{(by Thm. 2.3.10)}\\
&= (\pi;Dom(R);1) \cdot 1' && \text{(by Thm. 2.3.14)}\\
&\leq (\pi;Dom(R) \cdot 1';1)\\
&\quad ;(1 \cdot Dom(R);\breve{\pi};1') && \text{(by (2.2))}\\
&= \pi;Dom(R);Dom(R);\breve{\pi} && \text{(by Ax. 5 and BA)}\\
&= \pi;(Dom(R) \cdot Dom(R));\breve{\pi} && \text{(by Thm. 2.3.7)}\\
&= \pi;Dom(R);\breve{\pi}. && \text{(BA)}
\end{aligned}$$

Also,

$$\begin{aligned}
Dom(\pi;R) &= (\pi;R;1) \cdot 1' && \text{(by Thm. 2.3.10)}\\
&\leq (\pi;1) \cdot 1' && \text{(by monotonicity)}\\
&= Dom(\pi) && \text{(by Thm. 2.3.10)}\\
&= Dom(\rho) && \text{(by 15)}\\
&= (\rho;\breve{\rho}) \cdot 1' && \text{(by Def. } Dom\text{)}\\
&\leq \rho;\breve{\rho}\,. && \text{(by monotonicity)}
\end{aligned}$$

Thus,

$$Dom(\pi;R) \ \leq\ (\pi;Dom(R);\breve{\pi}) \cdot (\rho;\breve{\rho}) = Dom(R) \otimes 1'\ .$$

Let us show now the other inclusion. First, let us note that

$$\begin{aligned}
Dom\,(R) \otimes 1' &= \pi ; Dom\,(R) \;\nabla\; \rho && \text{(by (3.5))}\\
&= \pi ; Dom\,(R) \,; \breve{\pi} \cdot \rho ; \breve{\rho} && \text{(by Ax. 8)}\\
&\leq \pi ; Dom\,(R) \,; 1 && \text{(by monotonicity)}\\
&= \pi ; R ; 1\ . && \text{(by Thm. 2.3.14)}
\end{aligned}$$

Second, note that by monotonicity $Dom\,(R) \otimes 1' \leq 1' \otimes 1'$, and since by Ax. 10, $1' \otimes 1' \leq 1'$, by transitivity $Dom\,(R) \otimes 1' \leq 1'$. Thus,

$$Dom\,(R) \otimes 1' \leq (\pi ; R ; 1) \cdot 1' = Dom\,(\pi ; R)\,.$$

17. First, note that

$$\begin{aligned}
\left(\breve{R} \otimes 1'\right) ; \breve{2} &= \pi ; \breve{R} \cdot \rho && \text{(by (3.5) and Ax. 9)}\\
&\leq \left((\pi \cdot \rho ; R) \,; \left(\breve{R} \cdot \breve{\pi} ; \rho\right)\right) \cdot \rho && \text{(by (2.2))}\\
&= \left((\pi \cdot \rho ; R) \,; \left(\breve{R} \cdot 1\right)\right) \cdot \rho && \text{(by 14)}\\
&\leq ((\pi \cdot \rho ; R) \,; 1) \cdot \rho && \text{(by monotonicity)}\\
&= (Dom\,(\pi \cdot \rho ; R) \,; 1) \cdot \rho && \text{(by Thm. 2.3.14)}\\
&= Dom\,(\pi \cdot \rho ; R) \,; \rho && \text{(by Thm. 2.3.8)}\\
&= Dom\,((\pi \;\nabla\; \rho ; R) ; \breve{2}) \,; \rho && \text{(by Ax. 9)}\\
&= Dom\,((1' \otimes R) ; \breve{2}) \,; \rho. && \text{(by (3.5))}
\end{aligned}$$

In order to prove the equality we will reason as follows. First, note that the relation $Dom\,((1' \otimes R) ; \breve{2}) \,; \rho$ is functional. Since we have already shown that $(\breve{R} \otimes 1') ; \breve{2} \leq Dom\,((1' \otimes R) ; \breve{2}) \,; \rho$, by Thm. 2.3.18 it suffices to show that

$$Dom\left((\breve{R} \otimes 1') ; \breve{2}\right) \geq Dom\,((1' \otimes R) ; \breve{2})\ .$$

We proceed as follows.

$$\begin{aligned}
Dom\left((\breve{R}\otimes 1');\breve{2}\right) &= Dom\left((\pi;\breve{R}\ \nabla\ \rho);\breve{2}\right) && \text{(by (3.5))}\\
&= Dom\left((\pi;\breve{R}\ \cdot\ \rho)^{\smile\smile}\right) && \text{(by Ax. 9 and Ax. 4)}\\
&= Dom\left(((\pi;\breve{R})^{\smile}\ \cdot\ \breve{\rho})^{\smile}\right) && \text{(by Thm. 2.3.5)}\\
&= Dom\left((R;\breve{\pi}\ \cdot\ \breve{\rho})^{\smile}\right) && \text{(by Ax. 6 and Ax. 4)}\\
&= Dom\left((R\nabla 1')^{\smile}\right) && \text{(by Ax. 8)}\\
&= Ran\,(R\nabla 1')\,. && \text{(by Def. } Dom\text{)}
\end{aligned}$$

We also have

$$\begin{aligned}
Dom\left((1'\otimes R);\breve{2}\right) &= Dom\left((\pi\ \nabla\ \rho;R);\breve{2}\right) && \text{(by (3.5))}\\
&= Dom\left(((\pi\ \cdot\ \rho;R)^{\smile})^{\smile}\right) && \text{(by Ax. 9 and Ax. 4)}\\
&= Dom\left((\breve{\pi}\ \cdot\ (\rho;R)^{\smile})^{\smile}\right) && \text{(by Thm. 2.3.5)}\\
&= Dom\left((\breve{\pi}\ \cdot\ \breve{R};\breve{\rho})^{\smile}\right) && \text{(by Ax. 6)}\\
&= Dom\left((1'\nabla\breve{R})^{\smile}\right) && \text{(by Ax. 8)}\\
&= Ran(1'\nabla\breve{R}). && \text{(by Def. } Dom\text{)}
\end{aligned}$$

We will finally show that $Ran\,(R\nabla 1')\ \geq\ Ran(1'\nabla\breve{R})$.

$$\begin{aligned}
Ran\,(R\nabla 1') &= (1;(R\nabla 1'))\cdot 1' && \text{(by Thm. 2.3.10)}\\
&= (1;\,(R;\breve{\pi}\ \cdot\ \breve{\rho}))\cdot 1' && \text{(by Ax. 8)}\\
&\geq\ \left(1;\breve{R};\,(R;\breve{\pi}\ \cdot\ \breve{\rho})\right)\cdot 1' && \text{(by monotonicity)}\\
&= \left(1;\,\left(\breve{R}\cdot 1\right);(R;\breve{\pi}\ \cdot\ \breve{\rho})\right)\cdot 1' && \text{(BA)}\\
&= \left(1;\,\left(\breve{R}\cdot(\breve{\pi};\rho)\right);(R;\breve{\pi}\ \cdot\ \breve{\rho})\right)\cdot 1' && \text{(by 14)}\\
&\geq\ \left(1;\,\left(\breve{\pi}\ \cdot\ \breve{R};\breve{\rho}\right)\right)\cdot 1' && \text{(by (2.2))}\\
&= \left(1;\,\left(1'\nabla\breve{R}\right)\right)\cdot 1' && \text{(by Ax. 8)}\\
&= Ran\left(1'\nabla\breve{R}\right). && \text{(by Thm. 2.3.10)}
\end{aligned}$$

18. In order to prove the equality we will prove both inclusions.

$$\begin{aligned}
&\breve{\pi} \cdot 1;(1'\nabla F) \\
&\quad\geq (1'\nabla 1) \cdot 1';(1'\nabla F) && \text{(by Def. } \pi \text{ and monotonicity)} \\
&\quad= (1'\nabla 1) \cdot (1'\nabla F) && \text{(by Ax. 5)} \\
&\quad= (1' {\cdot} 1' \;\nabla\; 1 {\cdot} F) && \text{(by 6)} \\
&\quad= 1'\nabla F. && \text{(by BA)}
\end{aligned}$$

$$\begin{aligned}
&\breve{\pi} \cdot 1;(1'\nabla F) \\
&\quad\leq \left(1 \cdot (1'\nabla 1);(1'\nabla F)^{\smile}\right);(1'\nabla F \cdot (1;(1'\nabla 1))) && \text{(by (2.2))} \\
&\quad= (1'\nabla 1);(1'\nabla F)^{\smile};(1'\nabla F \cdot (1;(1'\nabla 1))) && \text{(by BA)} \\
&\quad= (1'\nabla 1);(1'\nabla F)^{\smile};(1'\nabla F) && \text{(by monotonicity)} \\
&\quad= \left(1' \cdot 1;\breve{F}\right);(1'\nabla F) && \text{(by Ax. 9)} \\
&\quad= \left(1' \cdot 1;\breve{F}\right)^{\smile};(1'\nabla F) && \text{(by Thm. 2.3.6)} \\
&\quad= \left(\breve{1'} \cdot \left(1;\breve{F}\right)^{\smile}\right);(1'\nabla F) && \text{(by Thm. 2.3.5)} \\
&\quad= \left(1' \cdot \breve{\breve{F}};\breve{1}\right);(1'\nabla F) && \text{(by Thm. 2.3.6 and Ax. 6)} \\
&\quad= (1' \cdot F;1);(1'\nabla F) && \text{(by Thm. 2.3.2 and Ax. 4)} \\
&\quad= \mathit{Dom}(F);(1'\nabla F) && \text{(by Thm. 2.3.10)} \\
&\quad= \mathit{Dom}(F) \;\nabla\; \mathit{Dom}(F);F && \text{(by 4)} \\
&\quad= \mathit{Dom}(F)\nabla F && \text{(by Thm. 2.3.11)} \\
&\quad= 1'\nabla F. && \text{(by 5)}
\end{aligned}$$

19. Since the proofs of both properties are analogous, we will focus on the first one. In order to prove the equality we will prove both inclusions. Note first that by Thms. 2.3.8 and 2.3.14,

$$\mathit{Dom}\left((\breve{I}\otimes R);\breve{2}\right);\pi = (\breve{I}\otimes R);\breve{2};1 \cdot \pi\ .$$

$$\begin{aligned}
(1'\otimes R;I);\breve{2} &= \pi \cdot \rho;R;I && \text{(by (3.5) and 1)} \\
&\leq \pi. && \text{(by BA)}
\end{aligned}$$

Also,

$$\begin{aligned}
&(1'\otimes R;I);\breve{2} \\
&\quad = \pi \cdot \rho;R;I && \text{(by (3.5) and 1)} \\
&\quad = \pi \cdot \rho;R;I;Ran\,(I) && \text{(by Thm. 2.3.11)} \\
&\quad = \pi;Ran\,(I) \cdot \rho;R;I && \text{(by Thms. 2.3.22 and 2.3.20)} \\
&\quad = \pi;(\breve{I};I \cdot 1') \cdot \rho;R;I && \text{(by Def. } Dom\text{)} \\
&\quad \leq \pi;\breve{I};I \cdot \rho;R;I && \text{(by monotonicity)} \\
&\quad = (\pi;\breve{I} \cdot \rho;R);I && \text{(by Thm. 2.3.20)} \\
&\quad \leq (\pi;\breve{I} \cdot \rho;R);1 && \text{(by monotonicity)} \\
&\quad = (\breve{I}\otimes R);\breve{2};1. && \text{(by (3.5) and 1)}
\end{aligned}$$

Thus,

$$(1'\otimes R;I)\,;\breve{2} \leq Dom\left(\left(\breve{I}\otimes R\right);\breve{2}\right);\pi\ .$$

Let us now prove the other inclusion. For this, note that by (3.5) and 1,

$$(1' \otimes R;I)\,;\breve{2} = \pi \cdot \rho;R;I\ .$$

Then,

$$\begin{aligned}
Dom\left(\left(\breve{I}\otimes R\right);\breve{2}\right);\pi &\leq 1';\pi && \text{(by monotonicity)} \\
&= \pi. && \text{(by Ax. 5)}
\end{aligned}$$

Also,

$$\begin{aligned}
&Dom((\breve{I}\otimes R);\breve{2});\pi \\
&\quad = Dom(\pi;\breve{I} \cdot \rho;R);\pi && \text{(by (3.5) and 1)} \\
&\quad = \left(\left((\pi;\breve{I} \cdot \rho;R);(\pi;\breve{I} \cdot \rho;R)^{\smile}\right) \cdot 1'\right);\pi && \text{(by Def. } Dom\text{)} \\
&\quad = \left(\left((\pi;\breve{I} \cdot \rho;R);(I;\breve{\pi} \cdot \breve{R};\breve{\rho})\right) \cdot 1'\right);\pi \\
&\qquad && \text{(by Thm. 2.3.5 and Ax. 6)} \\
&\quad \leq \rho;R;I;\breve{\pi};\pi && \text{(by monotonicity)} \\
&\quad \leq \rho;R;I;1' && \text{(by 13)} \\
&\quad = \rho;R;I. && \text{(by Ax. 5)}
\end{aligned}$$

Thus,

$$Dom\left(\left(\breve{I}\otimes R\right);\breve{2}\right);\pi \leq (1'\otimes\ R;I);\breve{2}.$$

20.

$$\begin{aligned}
&(1'\otimes 1');\overline{R\otimes S};(1'\otimes 1')\\
&\quad = (1'\otimes 1');\overline{\pi;R;\breve{\pi}\cdot\rho;S;\breve{\rho}};(1'\otimes 1') && \text{(by (3.5) and Ax. 8)}\\
&\quad = (1'\otimes 1');\left(\overline{\pi;R;\breve{\pi}}+\overline{\rho;S;\breve{\rho}}\right);(1'\otimes 1') && \text{(by BA)}\\
&\quad = (1'\otimes 1');\overline{\pi;R;\breve{\pi}};(1'\otimes 1')\\
&\qquad + (1'\otimes 1');\overline{\rho;S;\breve{\rho}};(1'\otimes 1') && \text{(by Ax. 2)}\\
&\quad = \pi;\overline{R};\breve{\pi}+\rho;\overline{S};\breve{\rho} && \text{(by 15 and Thms. 2.3.19, 2.3.21)}\\
&\quad = \left(\pi;\overline{R};\breve{\pi}\cdot 1\right)+\left(1\cdot\rho;\overline{S};\breve{\rho}\right) && \text{(by BA)}\\
&\quad = \left(\pi;\overline{R};\breve{\pi}\cdot Dom(\pi);1;Dom(\pi)\right)\\
&\qquad + \left(Dom(\rho);1;Dom(\rho)\cdot\rho;\overline{S};\breve{\rho}\right) && \text{(by Thm. 2.3.22)}\\
&\quad = \left(\pi;\overline{R};\breve{\pi}\cdot Dom(\rho);1;Dom(\rho)\right)\\
&\qquad + \left(Dom(\pi);1;Dom(\pi)\cdot\rho;\overline{S};\breve{\rho}\right) && \text{(by 15)}\\
&\quad = \left(\pi;\overline{R};\breve{\pi}\cdot\rho;1;\breve{\rho}\right)+\left(\pi;1;\breve{\pi}\cdot\rho;\overline{S};\breve{\rho}\right) && \text{(by Thm. 2.3.14)}\\
&\quad = \left(\overline{R}\otimes 1\right)+\left(1\otimes\overline{S}\right). && \text{(by (3.5) and Ax. 8)}
\end{aligned}$$

□

This page is intentionally left blank

Chapter 4

Representability and Independence

Abstract fork algebras arose in the search for an abstract formalism suitable for systems specification and verification. As we will see in further chapters, their expressive power allows us to cope with different specification formalisms. Also, the calculus of abstract fork algebras with its simple equational rules allows non experts to understand and use the calculus. If specifications are to be written as formulas in the language of abstract fork algebras, we need an easily understandable semantics for these specifications. Obvious candidates are the abstract fork algebras, but it is not at all clear what these algebras look like. Better candidates to play this role are the proper fork algebras. They are particularly adequate because their universe made of binary relations and their simple operators can be understood even by non mathematicians. The previous remarks show that it is important to determine the relationship between proper and abstract fork algebras. The ideal situation would be for proper and abstract fork algebras to be the *same* thing. Since abstract fork algebras are models of a set of equations, the class is closed under isomorphisms, and therefore, the best we can expect is for abstract fork algebras to be *isomorphic* to proper ones. This is known as the *representability* problem for abstract fork algebras. The solution to this problem was obtained independently by Frias–Haeberer–Veloso [M. Frias et al. (1997)b] and by Gyuris [V. Gyuris (1995)]. In Section 4.1, and using as a central result a theorem due to Tarski [A. Tarski (1953)] on the representability of quasi-projective relation algebras (and generalized by Maddux in [R. Maddux (1978)]), we will present the representability theorem for abstract fork algebras.

As a corollary of the representability theorem, the axioms for abstract fork algebras provide a finite (and equational) axiomatization for proper fork algebras. In Section 4.2 we prove the independence of these axioms (i.e., we prove that none of the axioms can be deleted from the axiomatization).

4.1 Representability of Abstract Fork Algebras

Definition 4.1 We define the class of *representable fork algebras* as **I** PFA, the closure under isomorphism of proper fork algebras.

The class of representable fork algebras will be denoted by RFA.

In the next lemma we prove that relations π and ρ are a pair of quasi-projections as defined in [A. Tarski et al. (1987), p. 96].

Lemma 4.1 *The relations π and ρ are functional. Moreover, the equation $\breve{\pi};\rho = 1$ holds in every abstract fork algebra.*

Proof By Thms. 3.2.13 and 3.2.14. □

Relation algebras with projection elements have been widely studied. In [A. Tarski et al. (1987)], relation algebras having functional elements A and B satisfying $\breve{A};B = 1$ are called quasi-projective relation algebras by Tarski and Givant, and Maddux [R. Maddux (1989)] calls structures $\langle \mathfrak{R}, A, B \rangle$ ($\mathfrak{R}$ a relation algebra), pairing relation algebras. From Lemma 4.1 we immediately obtain the following corollary.

Corollary 4.1 *The relation algebra reduct of any abstract fork algebra is a quasi-projective relation algebra. Moreover, if we call $\mathfrak{R}$ the relation algebra reduct of a given abstract fork algebra, then any structure $\langle \mathfrak{R}, \pi, \rho \rangle$ is a pairing relation algebra.*

As the next step we will prove that the axioms characterizing abstract fork algebras are satisfied in any proper fork algebra, thus establishing that RFA $\subseteq$ AFA.

Theorem 4.1 RFA $\subseteq$ AFA.

Proof By Thm. 3.1, PFA = **I S P** FullPFA. Since the axioms of AFA are equations and equations are preserved under **I** (isomorphisms), **S** (subalgebras) and **P** (products), it suffices to show that for every $\mathfrak{A} \in$ FullPFA, $\mathfrak{A}$ satisfies axioms Ax. 1–Ax. 10. Therefore, for the remaining part of this theorem, let $\mathfrak{A} \in$ FullPFA. Notice that the reduct of $\mathfrak{A}$ to the similarity type $\langle \cup, \cap, ^{-}, \emptyset, U \times U, \circ, Id, ^{\smile} \rangle$ is an algebra of binary relations, thus satisfying Ax. 1–Ax. 7 characterizing RA. To shorten notation we will denote the relation $U \times U$ by V.

In order to show that Ax. 8 holds in $\mathfrak{A}$, we will prove that for any binary relations $R, S \in A$, $R \underline{\nabla} S = (R \circ (Id \underline{\nabla} V)) \cap (S \circ (V \underline{\nabla} Id))$.

$$\begin{array}{ll}
 & x R \underline{\nabla} S y \\
\iff & \{\text{by def. } \underline{\nabla}\} \\
 & \exists u, v\, (xRu \wedge xSv \wedge y = *(u,v)) \\
\iff & \{\text{by def. } Id \text{ and } V\} \\
 & \exists u, v (xRu \wedge uIdu \wedge uVv \wedge xSv \wedge vVu \wedge vIdv \wedge y = *(u,v)) \\
\iff & \{\text{by def. } \underline{\nabla}\} \\
 & \exists u, v\, (xRu \wedge uId \underline{\nabla} Vy \wedge xSv \wedge vV \underline{\nabla} Idy) \\
\iff & \{\text{elementary logic}\} \\
 & \exists u\, (xRu \wedge uId \underline{\nabla} Vy) \wedge \exists v\, (xSv \wedge vV \underline{\nabla} Idy) \\
\iff & \{\text{by def. } \circ\} \\
 & xR \circ (Id \underline{\nabla} V) y \ \wedge \ xS \circ (V \underline{\nabla} Id) y \\
\iff & \{\text{by def. } \cap\} \\
 & x(R \circ (Id \underline{\nabla} V)) \cap (S \circ (V \underline{\nabla} Id)) y\ .
\end{array}$$

In order to prove that Ax. 9 holds in $\mathfrak{A}$, we will prove that for any binary relations R, S, T and Q, $(R \underline{\nabla} S) \circ (T \underline{\nabla} Q)^{\smile} = (R \circ \breve{T}) \cap (S \circ \breve{Q})$.

$$\begin{array}{ll}
 & x(R \underline{\nabla} S) \circ (T \underline{\nabla} Q)^{\smile} y \\
\iff & \{\text{by def. } \circ\} \\
 & \exists u\, (xR \underline{\nabla} Su \wedge u(T \underline{\nabla} Q)^{\smile} y) \\
\iff & \{\text{by def. } ^{\smile}\} \\
 & \exists u\, (xR \underline{\nabla} Su \wedge yT \underline{\nabla} Qu) \\
\iff & \{\text{by def. } \underline{\nabla}\} \\
 & \exists u, v_1, v_2, w_1, w_2 (xRv_1 \wedge xSw_1 \wedge u = *(v_1, w_1) \\
 & \quad \wedge\ yTv_2 \wedge yQw_2 \wedge u = *(v_2, w_2)) \\
\iff & \{\text{by } * \text{ injective}\} \\
 & \exists v, w\, (xRv \wedge xSw \wedge yTv \wedge yQw) \\
\iff & \{\text{by elementary logic and def. } ^{\smile}\} \\
 & \exists v, w \left(xRv \wedge v\breve{T}y \wedge xSw \wedge w\breve{Q}y\right) \\
\iff & \{\text{by def. } \circ\} \\
 & xR \circ \breve{T} y \wedge xS \circ \breve{Q} y \\
\iff & \{\text{by def. } \cap\} \\
 & x(R \circ \breve{T}) \cap (S \circ \breve{Q}) y\ .
\end{array}$$

Regarding Ax. 10, in order to show that it holds in $\mathfrak{A}$, we will show that

$$(Id \underline{\nabla} V)^{\smile} \underline{\nabla} (V \underline{\nabla} Id)^{\smile} \subseteq Id \ .$$

$$\begin{array}{ll} & x (Id \underline{\nabla} V)^{\smile} \underline{\nabla} (V \underline{\nabla} Id)^{\smile} y \\ \Rightarrow & \{ \text{by def. } \underline{\nabla} \} \\ & \exists u, v (x (Id \underline{\nabla} V)^{\smile} u \wedge x (V \underline{\nabla} Id)^{\smile} v \wedge y = *(u, v)) \\ \Rightarrow & \{ \text{by def. } {}^{\smile} \} \\ & \exists u, v (u Id \underline{\nabla} V x \wedge v V \underline{\nabla} Id x \wedge y = *(u, v)) \\ \Rightarrow & \{ \text{by def. } \underline{\nabla} \} \\ & \exists u, v, u', v' (u Id u' \wedge v Id v' \wedge x = *(u', v') \wedge y = *(u, v)) \\ \Rightarrow & \{ \text{by def. } Id \} \\ & \exists u, v, u', v' (u = u' \wedge v = v' \wedge x = *(u', v') \wedge y = *(u, v)) \\ \Rightarrow & \{ \text{by } * \text{ function} \} \\ & x = y \ . \end{array}$$

□

Lemma 4.2 *Consider $\mathfrak{A} \in$ SAFA with relation algebra reduct $\mathfrak{A}'$. Given a relation algebra homomorphism $h : \mathfrak{A}' \longrightarrow \mathfrak{B}$ into a square algebra of binary relations $\mathfrak{B}$, there exists an expansion of $\mathfrak{B}$ to $\mathfrak{B}^* \in$ SPFA so that $h : \mathfrak{A} \longrightarrow \mathfrak{B}^*$ is a fork algebra homomorphism.*

Proof The algebra $\mathfrak{B}$ consists of relations on a set U, with largest element $V = U \times U$.

Since the quasi-projections $\pi, \rho \in A'$, $h(\pi), h(\rho) \in B$.

By Lemma 4.1, $\mathfrak{A}'$ satisfies:

$$\breve{\pi} ; \rho = 1, \qquad \breve{\pi} ; \pi \leq 1', \qquad \breve{\rho} ; \rho \leq 1' \ .$$

Thus,

$$\begin{array}{l} (h(\pi))^{\smile} \circ h(\rho) = h(1) = V, \\ h(\pi) \text{ is a functional binary relation,} \\ h(\rho) \text{ is a functional binary relation.} \end{array}$$

We define the relation $F \subseteq (U \times U) \times U$ by

$$\langle \langle a, b \rangle, c \rangle \in F \qquad \Longleftrightarrow \qquad \langle c, a \rangle \in h(\pi) \text{ and } \langle c, b \rangle \in h(\rho) \ .$$

(1) $V = Dom(F)$.
Given $\langle a, b \rangle \in V$, since $V \subseteq (h(\pi))^{\smile} \circ h(\rho)$ there exists $c \in U$ such that $\langle a, c \rangle \in (h(\pi))^{\smile}$ and $\langle c, b \rangle \in h(\rho)$. Thus, $\langle \langle a, b \rangle, c \rangle \in F$.

(2) F is functional on V.
Let $\langle\langle a,b\rangle, x\rangle, \langle\langle a,b\rangle, y\rangle \in F$. Then $\langle x,a\rangle \in h(\pi)$, $\langle x,b\rangle \in h(\rho)$, $\langle y,a\rangle \in h(\pi)$ and $\langle y,b\rangle \in h(\rho)$.
Thus, $\langle x,y\rangle \in (h(\pi) \circ (h(\pi))^{\smile}) \cap (h(\rho) \circ (h(\rho))^{\smile})$.
But, by Ax. 8 and Ax. 10 in Def. 3.4, $\mathfrak{A}'$ satisfies $(\pi;\breve{\pi}) \cdot (\rho;\breve{\rho}) \leq 1$', thus

$$(h(\pi) \circ (h(\pi))^{\smile}) \cap (h(\rho) \circ (h(\rho))^{\smile}) \subseteq h(1') = Id.$$

Hence, $x = y$.

(3) F is injective on V.
Consider $\langle a,b\rangle, \langle c,d\rangle \in V$ such that $\langle\langle a,b\rangle, z\rangle, \langle\langle c,d\rangle, z\rangle \in F$ for some $z \in U$.
Then $\langle z,a\rangle, \langle z,c\rangle \in h(\pi)$ and $\langle z,b\rangle, \langle z,d\rangle \in h(\rho)$. Since $h(\pi)$ and $h(\rho)$ are functional binary relations, $a = c$ and $b = d$.

Hence, the restriction of F to V gives a well-defined injective function $* : V \longrightarrow U$ such that

$$*(a,b) = c \qquad \Longleftrightarrow \qquad \langle c,a\rangle \in h(\pi) \text{ and } \langle c,b\rangle \in h(\rho)\ . \tag{4.1}$$

We expand $\mathfrak{B}$ to $\mathfrak{B}^* \in \mathsf{PFA}$ by defining

$$R \underline{\nabla} S = \{ \langle x, *(y,z)\rangle : \langle x,y\rangle \in R \ \wedge \ \langle x,z\rangle \in S \}.$$

In order to show that $\mathfrak{B}^*$ is closed under $\underline{\nabla}$, let us prove that

$$h(R) \underline{\nabla} h(S) = (h(R) \circ h(\pi)^{\smile}) \cap (h(S) \circ h(\rho)^{\smile})\ . \tag{4.2}$$

$x h(R) \underline{\nabla} h(S) y$
$\Leftrightarrow$ { by def. $\underline{\nabla}$ }
$\exists u, v (x h(R) u \wedge x h(S) v \wedge y = *(u,v))$
$\Leftrightarrow$ { by (4.1) }
$\exists u, v (x h(R) u \wedge u h(\pi)^{\smile} y \wedge x h(S) v \wedge v h(\rho)^{\smile} y \wedge y = *(u,v))$
$\Leftrightarrow$ { by def. $\circ$ }
$x h(R) \circ h(\pi)^{\smile} y \wedge x h(S) \circ h(\rho)^{\smile} y$
$\Leftrightarrow$ { by def. $\cap$ }
$x (h(R) \circ h(\pi)^{\smile}) \cap (h(S) \circ h(\rho)^{\smile}) y$.

We will now see that h is a fork algebra homomorphism from $\mathfrak{A}$ into $\mathfrak{B}^*$ (i.e., we will show that h preserves fork).

Consider $r, s \in A$.

$$
\begin{array}{cl}
 & h(r\nabla s) \\
= & \{\text{by Ax. } 9\} \\
 & h((r;\breve{\pi})\cdot(s;\breve{\rho})) \\
= & \{h \text{ is a relation algebra homomorphism}\} \\
 & (h(r)\circ(h(\pi))^{\smile})\cap(h(s)\circ(h(\rho))^{\smile}) \\
= & \{\text{by } (4.2)\} \\
 & h(r)\underline{\nabla}h(s)\ .
\end{array}
$$
□

Lemma 4.3 AFA = **I S P** SAFA.

Proof In [B. Jónsson et al. (1952), Thm. 4.14] Jónsson and Tarski proved that simple relation algebras are subdirectly indecomposable. This result also holds for fork algebras, since a subdirect decomposition of a fork algebra induces a subdirect decomposition of its relation algebra reduct. Joining this with Birkhoff's theorem on the decomposability of arbitrary algebras in terms of subdirectly indecomposable ones [G. Birkhoff (1944)], we prove inclusion $\subseteq$. Inclusion $\supseteq$ follows because AFA is a variety, SAFA $\subseteq$ AFA, and equations are preserved by **I**, **S** and **P**. □

In the proof of Thm. 4.2 we give below, we use the result announced in [A. Tarski (1953)] on the representability of quasi-projective relation algebras. This theorem is one of the central topics in algebraic logic, and different kinds of proofs for it can be found in [A. Tarski et al. (1987), Thm. 8.4(iii)], [R. Maddux (1989)] and elsewhere.

Theorem 4.2 *Given* $\mathfrak{A} \in$ AFA, *there exists* $\mathfrak{B} \in$ PFA *isomorphic to* $\mathfrak{A}$.

Proof In view of Lemmas 3.1 and 4.3, it suffices to prove that simple fork algebras are representable.

Given $\mathfrak{A} \in$ SAFA, by Cor. 4.1 its relation algebra reduct $\mathfrak{A}'$ is a simple quasi-projective relation algebra. Thus, since quasi-projective relation algebras are representable [A. Tarski et al. (1987), p. 242], there is a relation algebra isomorphism $h : \mathfrak{A}' \to \mathfrak{B}$ onto an algebra of binary relations $\mathfrak{B}$. Since $\mathfrak{A}'$ is simple, we can assume that $\mathfrak{B}$ is square [A. Tarski et al. (1987), p. 239].

By Lemma 4.2, $\mathfrak{B}$ can be expanded to $\mathfrak{B}^* \in$ SPFA, in such way that $h : \mathfrak{A} \to \mathfrak{B}^*$ is a fork algebra isomorphism. □

Theorem 4.3 AFA = RFA.

Proof It follows from Thm. 4.2 that AFA $\subseteq$ **I**PFA = RFA. By Thm. 4.1, RFA $\subseteq$ AFA. □

A direct consequence of Thm. 4.3 is that the classes of proper and abstract fork algebras share the same first-order theory. This result on the elementary equivalence of the classes is extremely useful in our setting, since first-order formulas in the language of fork algebras now have a clear meaning when being considered as assertions about binary relations (or programs). It will be shown in Section 7.5 that, while equations suffice to express algorithms, first-order formulas about relations can be used to describe design strategies for program development. This adds a new dimension to the development of algorithms within fork algebras.

4.2 Independence of the Axiomatization of Fork

The fact that the class of fork algebras is a finitely based variety leaves open the problem of whether the axiomatization adopted has superfluous axioms. This is known as the *independence* of the axioms that characterize fork. Avoiding superfluous axioms is important because fewer axioms make the calculus easier to understand. Also, it may have a positive impact in the performance of semi-automatic theorem provers. In this section we will show that the axioms that characterize fork (Ax. 8–Ax. 10 from Def. 3.4) are independent. In order to show the independence of a finite axiomatization $A = \{ A_1, \ldots, A_n \}$, it suffices to show that for each i, $1 \leq i \leq n$, there exists an algebra $\mathfrak{A}_i$ such that

$$\mathfrak{A}_i \models A \setminus \{ A_i \} \quad \text{and} \quad \mathfrak{A}_i \not\models A_i. \tag{4.3}$$

In the particular case of fork algebras, given proper subsets of the axiomatization we will construct algebras satisfying (4.3) by analyzing the meaning of the missing axiom towards the representability of fork algebras.

In what follows, we will denote the set containing formulas Ax. 8–Ax. 10 from Def. 3.4 by Axm_∇.

Theorem 4.4 *The set of axioms Axm_∇ defining fork is independent.*

Proof In order to show that Ax. 8–Ax. 10 are independent, we will present algebras of binary relations $\mathfrak{A}_i$, $1 \leq i \leq 3$, and operations $\underline{\nabla}_i$, $1 \leq i \leq 3$, such that* $\langle \mathfrak{A}_i, \underline{\nabla}_i \rangle \models Axm_\nabla \setminus \{ Axm.\ i \}$ but $\langle \mathfrak{A}_i, \underline{\nabla}_i \rangle \not\models Axm.\ i$. In the proof of Lemma 4.2, Ax. 8–Ax. 10 are used in order to

*Given an algebra $\mathfrak{A}$ and an operation $\oplus : A^n \to A$, by $\langle \mathfrak{A}, \oplus \rangle$ we denote the extension of the algebra $\mathfrak{A}$ obtained by adding the operation $\oplus$.

prove that the relation F is indeed an injective function as required (see Def. 3.1). Thus, each axiom imposes some properties on this relation F.

In order to prove that Ax. 10 is independent of Ax. 8 and Ax. 9, notice that Ax. 10 is used in Lemma 4.2 to show that F is functional. Let us define $\mathfrak{A}_3$ as $\mathfrak{Re}(\mathbb{N}^+)$, the full algebra of binary relations on the set $\mathbb{N}^+$ of all positive natural numbers, and let $\underline{\nabla}_3$ be defined for $R, S \subseteq \mathbb{N}^+ \times \mathbb{N}^+$ by

$$R\underline{\nabla}_3 S = \{\langle m, 2^n * 3^p\rangle : mRn \ \wedge \ mSp\} \cup \{\langle m, 5^n * 7^p\rangle : mRn \ \wedge \ mSp\} \ . \tag{4.4}$$

In (4.4), F is the (non functional) injective binary relation

$$\{\langle\langle m,n\rangle, 2^m * 3^n\rangle : m,n \in \mathbb{N}^+\} \cup \{\langle\langle m,n\rangle, 5^m * 7^n\rangle : m,n \in \mathbb{N}^+\} .$$

It is clear that $\mathfrak{A}_3$ is closed under $\underline{\nabla}_3$. Let V be a shorthand for the binary relation $\mathbb{N}^+ \times \mathbb{N}^+$. Notice that

$$Id\underline{\nabla}_3 V = \{\langle m, 2^m * 3^n\rangle : m,n \in \mathbb{N}^+\} \cup \{\langle m, 5^m * 7^n\rangle : m,n \in \mathbb{N}^+\}$$

and

$$V\underline{\nabla}_3 Id = \{\langle m, 2^n * 3^m\rangle : m,n \in \mathbb{N}^+\} \cup \{\langle m, 5^n * 7^m\rangle : m,n \in \mathbb{N}^+\} .$$

Then, given binary relations $R, S \subseteq \mathbb{N}^+ \times \mathbb{N}^+$,

$$R \circ (Id\underline{\nabla}_3 V) = \{\langle m, 2^n * 3^p\rangle : p \in \mathbb{N}^+ \ \wedge \ mRn\} \cup \{\langle m, 5^n * 7^p\rangle : p \in \mathbb{N}^+ \ \wedge \ mRn\}$$

and

$$S \circ (V\underline{\nabla}_3 Id) = \{\langle m, 2^n * 3^p\rangle : n \in \mathbb{N}^+ \ \wedge \ mSp\} \cup \{\langle m, 5^n * 7^p\rangle : n \in \mathbb{N}^+ \ \wedge \ mSp\} \ .$$

Thus,

$$\begin{aligned} R \circ (Id\underline{\nabla}_3 V) \ \cap \ S \circ (V\underline{\nabla}_3 Id) &= \{\langle m, 2^n * 3^p\rangle : mRn \ \wedge \ mSp\} \\ &\quad \cup \{\langle m, 5^n * 7^p\rangle : mRn \ \wedge \ mSp\} \\ &= R\underline{\nabla}_3 S \ . \end{aligned}$$

Then, $\langle \mathfrak{A}_3, \nabla_3\rangle \models Ax.\ 8$.

Given binary relations $T, Q \subseteq \mathbb{N}^+ \times \mathbb{N}^+$,

$$(T \underline{\nabla}_3 Q)^{\smile} = \{\, \langle 2^n * 3^p, m \rangle : mTn \ \wedge \ mQp \} \\ \cup \{\, \langle 5^n * 7^p, m \rangle : mTn \ \wedge \ mQp \} \ .$$

Then,

$$\begin{aligned} &(R \underline{\nabla}_3 S) \circ (T \underline{\nabla}_3 Q)^{\smile} \\ &\quad = \{\, \langle a,b \rangle : \exists c, d\, (aRc \ \wedge \ aSd \ \wedge \ bTc \ \wedge \ bQd) \} \\ &\quad = \{\, \langle a,b \rangle : \exists c\, (aRc \ \wedge \ bTc) \ \wedge \ \exists d\, (aSd \ \wedge \ bQd) \} \\ &\quad = \left\{ \langle a,b \rangle : \exists c (aRc \ \wedge \ c\breve{T}a) \ \wedge \ \exists d (aSd \ \wedge \ d\breve{Q}b) \right\} \\ &\quad = \left\{ \langle a,b \rangle : \exists c (aRc \ \wedge \ c\breve{T}a) \right\} \cap \left\{ \langle a,b \rangle : \exists d (aSd \ \wedge \ d\breve{Q}b) \right\} \\ &\quad = R \circ \breve{T} \ \cap S \circ \breve{Q} \ . \end{aligned}$$

Thus, $\langle \mathfrak{A}_3, \underline{\nabla}_3 \rangle \models Ax.\ 9$.

Let us show now that Ax. 10 does not hold in the structure $\langle \mathfrak{A}_3, \underline{\nabla}_3 \rangle$.

By definition of $\underline{\nabla}_3$,

$$(Id \underline{\nabla}_3 V)^{\smile} = \left\{ \langle 2^m * 3^n, m \rangle : m, n \in \mathbb{N}^+ \right\} \\ \cup \left\{ \langle 5^m * 7^n, m \rangle : m, n \in \mathbb{N}^+ \right\}$$

and

$$(V \underline{\nabla}_3 Id)^{\smile} = \left\{ \langle 2^n * 3^m, m \rangle : m, n \in \mathbb{N}^+ \right\} \\ \cup \left\{ \langle 5^n * 7^m, m \rangle : m, n \in \mathbb{N}^+ \right\} \ .$$

Then, the relation $\left\{ \langle 2^m * 3^n, 5^m * 7^n \rangle : m, n \in \mathbb{N}^+ \right\}$ is contained in $(Id \underline{\nabla}_3 V)^{\smile} \underline{\nabla}_3 (V \underline{\nabla}_3 Id)^{\smile}$. Thus, $(Id \underline{\nabla}_3 V)^{\smile} \underline{\nabla}_3 (V \underline{\nabla}_3 Id)^{\smile} \not\leq Id$.

Let us now prove that Ax. 9 is independent from Ax. 8 and Ax. 10.

In the proof of Lemma 4.2, Ax. 9 is used when proving that F is a total and injective binary relation. Since only Ax. 9 is used when proving these properties, violating any of them should provide good examples of algebras satisfying Ax. 8 and Ax. 10 but not Ax. 9. In order to violate the injectivity, let us consider the relation $F \subseteq (\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$ defined by

$$F = \{\, \langle \langle x, y \rangle, 0 \rangle : x, y \in \mathbb{N} \} \ .$$

If we define $\mathfrak{A}_2 := \mathfrak{Re}(\mathbb{N})$ and define for arbitrary binary relations $R, S \subseteq \mathbb{N} \times \mathbb{N}$

$$R \underline{\nabla}_2 S = \{ \langle m, F(n,p) \rangle : mRn \ \wedge \ mSp \},$$

it is trivial to show that

(1) $R\underline{\nabla}_2 S = \{ \langle m, 0 \rangle : m \in \mathsf{dom}(R) \ \wedge \ m \in \mathsf{dom}(S) \}$,
(2) $\langle \mathfrak{A}_2, \underline{\nabla}_2 \rangle \models \{ Ax.\ 8, Ax.\ 10 \}$ but $\langle \mathfrak{A}_2, \nabla_2 \rangle \not\models Ax.\ 9$.

In a similar way, if we want to violate the totality of F, we can define

$$F = \{ \langle \langle m, m \rangle, m \rangle : m \in \mathbb{N} \} \ .$$

In this case it is easy to check that

(1) $R\underline{\nabla}_2 S = R \cap S$,
(2) $\langle \mathfrak{A}_2, \underline{\nabla}_2 \rangle \models \{ Ax.\ 8, Ax.\ 10 \}$, but $\langle \mathfrak{A}_2, \underline{\nabla}_2 \rangle \not\models Ax.\ 9$.

But, probably, the easiest way to prove that Ax. 9 is independent of the other axioms is defining[†] $F = \emptyset$. In this case, $R\underline{\nabla}_2 S = \emptyset$ for all binary relations $R, S \subseteq \mathbb{N} \times \mathbb{N}$. It is trivial to prove that $\langle \mathfrak{A}_2, \underline{\nabla}_2 \rangle \models Ax.\ 8$ and $\langle \mathfrak{A}_2, \underline{\nabla}_2 \rangle \models Ax.\ 10$. On the other hand, $(V\underline{\nabla}_2 V) \circ (V\underline{\nabla}_2 V)^{\smile} = \emptyset \neq V = (V \circ \breve{V}) \cap (V \circ \breve{V})$. Thus, $\langle \mathfrak{A}_2, \underline{\nabla}_2 \rangle \not\models Ax.\ 9$.

Finally, let us consider the most interesting case, in which we prove that Ax. 8 is independent from Ax. 9 and Ax. 10. In Lemma 4.2, Ax. 8 is used in order to show that ∇ is definable in terms of the relations $(1' \nabla 1)^{\smile}$ and $(1 \nabla 1')^{\smile}$. This suggests that $\underline{\nabla}_1$ can be defined by cases, giving ad-hoc definitions for $Id\underline{\nabla}_1 V$ and $V\underline{\nabla}_1 Id$.

Let $U = \{ a \}$, and let $\langle U^\star, \star \rangle$ be the free groupoid with set of generators U. Let $\mathfrak{A}_1 := \mathfrak{Re}(U^\star)$. The binary operation $\underline{\nabla}$ defined by $R\underline{\nabla}S = \{ \langle x, y \star z \rangle : xRy \ \wedge \ xSz \}$ satisfies axioms Ax. 8–Ax. 10. Let us define the operation $\underline{\nabla}_1$ as follows:

$$R\underline{\nabla}_1 S = \begin{cases} (R\underline{\nabla}S) \cup Id_a & \text{if } R = Id \text{ and } S = V, \\ R\underline{\nabla}S & \text{otherwise .} \end{cases}$$

The relation Id_a is defined by $Id_a = \{ \langle a, a \rangle \}$. Let us start by showing that, in effect, $\langle \mathfrak{A}_1, \underline{\nabla}_1 \rangle \not\models Ax.\ 8$. Notice first that $Id\underline{\nabla}_1 V = (Id\underline{\nabla}V) \cup Id_a$.

[†]This was called to my attention by Paulo Veloso in a private communication.

On the other hand,

$$\begin{aligned}(Id \circ (Id \underline{\nabla}_1 V)) \cap (V \circ (V \underline{\nabla}_1 Id)) \\ &= (Id \underline{\nabla}_1 V) \cap (V \circ (V \underline{\nabla}_1 Id)) \\ &= ((Id \underline{\nabla} V) \cup Id_a) \cap (V \circ (V \underline{\nabla} Id)) \\ &= \underbrace{[(Id \underline{\nabla} V) \cap (V \circ (V \underline{\nabla} Id))]}_{A} \cup \underbrace{[Id_a \cap (V \circ (V \underline{\nabla} Id))]}_{B} .\end{aligned}$$

Let us analyze relations A and B. Regarding A, since $V \circ (V \underline{\nabla} Id) \supseteq Id \underline{\nabla} V$, $A = Id \underline{\nabla} V$. For relation B, since the range of $1'_a$ is contained in U, and the range of $V \circ (V \underline{\nabla} Id)$ is contained in $U^\star \setminus U$, $B = \emptyset$. Thus,

$$(Id \circ (Id \underline{\nabla}_1 V)) \cap (V \circ (V \underline{\nabla}_1 Id)) = Id \underline{\nabla} V .$$

Since $Id \underline{\nabla} V \neq (Id \underline{\nabla} V) \cup Id_a$, $\langle \mathfrak{A}_1, \underline{\nabla}_1 \rangle \not\models Ax.\ 8$. It only remains to be shown that axioms Ax. 9 and Ax. 10 are satisfied in $\langle \mathfrak{A}_1, \underline{\nabla}_1 \rangle$.

That Ax. 10 is satisfied is easily checked as follows:

$$\begin{aligned}(Id \underline{\nabla}_1 V)^{\smile} \underline{\nabla}_1 (V \underline{\nabla}_1 Id)^{\smile} &= ((Id \underline{\nabla} V) \cup Id_a)^{\smile} \underline{\nabla}_1 (V \underline{\nabla} Id)^{\smile} \\ &= ((Id \underline{\nabla} V) \cup Id_a)^{\smile} \underline{\nabla} (V \underline{\nabla} Id)^{\smile} .\end{aligned}$$

Since $\underline{\nabla}$ is additive,

$$\begin{aligned}((Id \underline{\nabla} V) \cup Id_a)^{\smile} \underline{\nabla} (V \underline{\nabla} Id)^{\smile} \\ &= \underbrace{[(Id \underline{\nabla} V)^{\smile} \underline{\nabla} (V \underline{\nabla} Id)^{\smile}]}_{A} \cup \underbrace{[Id_a \underline{\nabla} (V \underline{\nabla} Id)^{\smile}]}_{B} .\end{aligned}$$

By Ax. 10, $A \leq Id$. For relation B, since the domain of Id_a is contained in U and the domain of $(V \underline{\nabla} Id)^{\smile}$ is contained in $U^\star \setminus U$, $B = \emptyset$. Thus,

$$(Id \underline{\nabla}_1 V)^{\smile} \underline{\nabla}_1 (V \underline{\nabla}_1 Id)^{\smile} \subseteq Id ,$$

as was to be shown.

Let us check at last that Ax. 9 is also satisfied. Notice that by definition of $\underline{\nabla}_1$, we can always write $R \underline{\nabla}_1 S$ as $(R \underline{\nabla} S) \cup \alpha$, where α can take the values Id_a or $\emptyset$. Given $R, S, T, Q \subseteq U^\star \times U^\star$,

$$\begin{aligned}
&(R \underline{\nabla}_1 S) \circ (T \underline{\nabla}_1 Q)^{\smile} \\
&\qquad = ((R \underline{\nabla} S) \cup \alpha) \circ ((T \underline{\nabla} Q) \cup \beta)^{\smile} && \text{(by Def. } \underline{\nabla}_1\text{)} \\
&\qquad = (R \underline{\nabla} S) \circ (T \underline{\nabla} Q)^{\smile} \cup \alpha \circ (T \underline{\nabla} Q)^{\smile} \\
&\qquad\quad \cup (R \underline{\nabla} S) \circ \breve{\beta} \cup \alpha \circ \breve{\beta} . && (\underline{\nabla} \text{ additive})
\end{aligned}$$

Since $\mathsf{ran}\,(\alpha)$ and $\mathsf{dom}\,((T \underline{\nabla}_1 Q)^{\smile})$ are disjoint, $\alpha \circ (T \underline{\nabla} Q)^{\smile} = \emptyset$. Also, since $\mathsf{ran}\,(R \underline{\nabla} S)$ and $\mathsf{dom}(\breve{\beta})$ are disjoint, $(R \underline{\nabla} S) \circ \breve{\beta} = \emptyset$. Thus

$$\begin{aligned}
(R \underline{\nabla}_1 S) \circ (T \underline{\nabla}_1 Q)^{\smile} &= (R \underline{\nabla} S) \circ (T \underline{\nabla} Q)^{\smile} \cup \alpha \circ \breve{\beta} \\
&\qquad\qquad \text{(by previous discussion)} \\
&= \left(R \circ \breve{T} \cap S \circ \breve{Q}\right) \cup \alpha \circ \breve{\beta} . \qquad \text{(by Ax. 9)}
\end{aligned}$$

The following table shows the pairs of values $\langle \alpha, \beta \rangle$ for all possible values of R, S, T and Q.

$\langle T, Q \rangle$ / $\langle R, S \rangle$	$\langle Id, V \rangle$	other
$\langle Id, V \rangle$	$\langle Id_a, Id_a \rangle$	$\langle Id_a, \emptyset \rangle$
other	$\langle \emptyset, Id_a \rangle$	$\langle \emptyset, \emptyset \rangle$

In the case $\langle R, S \rangle = \langle Id, V \rangle$ and $\langle T, Q \rangle = \langle Id, V \rangle$, $\alpha \circ \breve{\beta} = Id_a \circ Id_a = Id_a$. Then

$$\left(R \circ \breve{T} \cap S \circ \breve{Q}\right) \cup \alpha \circ \breve{\beta} = (Id \cap V) \cup Id_a = Id = R \circ \breve{T} \cap S \circ \breve{Q} .$$

For the remaining cases, notice that $\alpha \circ \breve{\beta} = \emptyset$, and thus

$$\left(R \circ \breve{T} \cap S \circ \breve{Q}\right) \cup \alpha \circ \breve{\beta} = R \circ \breve{T} \cap S \circ \breve{Q} .$$

□

Chapter 5

Interpretability of Classical First-Order Logic

Classical first-order logic is a formalism suitable for the specification of certain views of systems. It has a good expressive power and is relatively easy to understand by non mathematicians. Since we are establishing the foundations of a calculus for system specification and verification using relational methods, it seems natural to study the relationship between classical first-order logic and the fork algebra calculus. Actually, as part of the software development process we will translate (interpret) classical first-order specifications into relational specifications. Such translation has to be well-behaved in some sense. A possible way to formulate this good behavior is by requiring the translation to be semantics-preserving. In this chapter we will define the translation from classical first-order logic to the fork algebra calculus, and prove the semantics-preservation theorem.

5.1 Basic Definitions

Tarski defined in [A. Tarski (1941)] the elementary theory of binary relations as a logical counterpart of the class of algebras of binary relations. In a similar way we will define an elementary theory of fork relations (ETFR for short) having as target the definition of the class of proper fork algebras.

Definition 5.1 Given a set of constant symbols C and a set of function symbols with arity F, the set of *individual terms on C and F* (denoted by *IndTerm*(C, F)) is the smallest set A satisfying:

(1) $\mathit{IndVar} \cup C \subseteq A$,
(2) If $f \in F$ has arity k and $t_1, \ldots, t_k \in A$, then $f(t_1, \ldots, t_k) \in A$.

Definition 5.2 Given a set of constant relation symbols P, the set of *relation designations* on P (denoted by $RelDes(P)$) is the smallest set A satisfying:

(1) $RelVar \cup \{0, 1, 1'\} \cup P \subseteq A$,
(2) If $R, S \in A$, then $\left\{\overline{R}, \breve{R}, R{+}S, R{\cdot}S, R;S, R\nabla S\right\} \subseteq A$.

Definition 5.3 Let u be a symbol, then $u^\star$ is the smallest set A satisfying:

(1) $u \in A$,
(2) if $x, y \in A$, then the expression $\star(x, y)$ also belongs to A.

Elements of $u^\star$ are called arities. The arity $\star(u, \star(u, \cdots))$ with k occurrences of u will be denoted by the number k. We will in general write $u \star \cdots \star u$ (k occurrences of u) instead of $\star(u, \star(u, \cdots))$. For instance,

$$4 = \star(u, \star(u, \star(u, u))) = u \star u \star u \star u.$$

Definition 5.4 Given a set of constant symbols C and a set of function symbols F, by $IndTerm(C, F)^\star$, we denote the smallest set A satisfying:

(1) $IndTerm(C, F) \subseteq A$,
(2) If $t_1, t_2 \in A$, then $\star(t_1, t_2) \in A$.

Definition 5.5 Given $t \in IndTerm(C, F)^\star$, the *arity* of term t (denoted by $arity(t)$) is inductively defined as follows:

(1) If $t \in IndTerm(C, F)$, then $arity(t) = u$,
(2) If $t_1, t_2 \in IndTerm(C, F)^\star$ and $t = \star(t_1, t_2)$, then

$$arity(t) = \star(arity(t_1), arity(t_2)) .$$

Definition 5.6 Given a set of constant symbols C, a set of function symbols F and a set of constant relation symbols P, the set of atomic formulas of ETFR is the smallest set A satisfying:

(1) $R = S \in A$ whenever $R, S \in RelDes(P)$,
(2) $t_1 R t_2 \in A$ whenever $t_1, t_2 \in IndTerm(C, F)^\star$ and $R \in RelDes(P)$.

From the atomic formulas, compound formulas are built as in first-order logic, with quantifiers applied only to individual variables. Notice that once the sets C, F and P are fixed, a unique set of formulas is characterized. We will denote this set by $ForETFR(C, F, P)$.

Definition 5.7 Given a set of constant symbols C, a set of function symbols F and a set of relation constant symbols P, we define the formalism $\mathsf{ETFR}(C, F, P)$ as follows:

Formulas: *ForETFR*(C, F, P).
Inference rules: Same as in ETBR.
Axioms: Extend the axioms of ETBR by adding formulas (3.2) and (3.3).

Much the same as Tarski defined his calculus of relations from the elementary theory of binary relations, we will define a calculus of fork relations (denoted by CFR) from the elementary theory of fork relations.

Definition 5.8 Given a set of relation constant symbols P, we define the formalism $\mathsf{CFR}(P)$ as follows:

Formulas: Those formulas from the ETFR in which neither individual variables nor constant symbols occur (i.e., Boolean combinations of equalities between relational designations). The set of formulas for $\mathsf{CFR}(P)$ will be denoted by *ForCFR*(P).
Inference rules: Same as in CR.
Axioms: Extend the axioms of CR by adding formulas (Ax. 8)–(Ax. 10) from Def. 3.4.

In this and the remaining sections, given sets C, F and P we will denote by $FOLE(C, F, P)$ the first-order logic with equality on the language with set of constant symbols C, set of function symbols F and set of predicate symbols P.

5.2 Interpreting *FOLE*

In order to fulfill the task of interpreting *FOLE*, we will perform an intermediate step. First we will show how to interpret *FOLE* into ETFR, and after doing this we will show how to interpret ETFR into CFR.

The algebraization of logics is a field of extensive and active work. In the remaining part of this chapter and the next one we will show how fork algebras can be used to interpret classical first-order logic, as well as many non-classical logics. The reader interested in the algebraization of logics should consider reading the book [L. Henkin et al. (1985)] (in particular Section 4.3, which studies the connections between cylindric algebras and logic, and Ch. 5, in which other algebraizations are presented), and also the book [P. Halmos (1962)]. Also fundamental are the works of the Budapest school, specially the papers [H. Andréka et al. (1994); H. Andréka et al. (1993); H. Andréka et al. (1981); I. Németi (1991)]. Finally, the work of Blok and Pigozzi (see [W. Blok et al. (1989)] and the references therein) is a very valuable source of results in algebraic logic.

Notice that in $FOLE(C,F,P)$ there is a standard notion of *arity* for function and predicate symbols. We will also assume that function and constant relation symbols from $\mathsf{ETFR}(C,F,P)$ have an associated arity. Arity of functions is defined as usual. For constant relation symbols the arity is defined as a pair $\langle a_1, a_2 \rangle$ where $a_1, a_2 \in u^\star$. Arity a_1 is called the *input arity* and a_2 is called the *output arity*. The reason for doing this is that in in the process of problem specification we will convert first-order predicates into input-output binary relations. To this syntactic definition of arity also corresponds a semantic notion.

Definition 5.9 Let $\mathfrak{A} \in \mathsf{PFAU}$. We define $arity : U_{\mathfrak{A}} \longrightarrow u^\star$ by:

(1) If $e \in Urel_{\mathfrak{A}}$, $arity(e) = u$,
(2) If $e = *(e_1, e_2)$, $arity(e) = \star(arity(e_1), arity(e_2))$.

Note that the *arity* function is partial, since there might be elements in $U_{\mathfrak{A}}$ that are not finitely generated from urelements and for which Def. 5.9 does not produce a well defined arity in $u^\star$.

Definition 5.10 Given a PFAU $\mathfrak{A}$, a binary relation $R \in \mathfrak{A}$ has arity $\langle a_1, a_2 \rangle$ $(a_1, a_2 \in u^\star)$ if

$$R \subseteq \{ \langle x, y \rangle \in U_{\mathfrak{A}} \times U_{\mathfrak{A}} : arity(x) = a_1 \wedge arity(y) = a_2 \} .$$

Given objects $a_1, \ldots, a_k, a_1{}', \ldots, a_k{}'$ and a finite set $A = \{ a_1, \ldots, a_k \}$, by A' we denote the set $\{ a_1{}', \ldots, a_k{}' \}$.

Given a finite set of symbols $A = \{ a_1, \ldots, a_k \}$ and a structure $\mathcal{A}$, by $A^{\mathcal{A}}$ we denote the set $\{ a_1{}^{\mathcal{A}}, \ldots, a_k{}^{\mathcal{A}} \}$ of the interpretations of the symbols in the structure $\mathcal{A}$.

We will define the semantics of $\mathsf{ETFR}(C,F,P)$ in terms of square proper fork algebras with urelements. The definition is as follows.

Definition 5.11 An adequate structure for $\mathsf{ETFR}(C,F,P)$ is a structure $\mathcal{A} = \langle \mathfrak{A}, C^{\mathcal{A}}, F^{\mathcal{A}}, P^{\mathcal{A}} \rangle$ satisfying:

(1) $\mathfrak{A} \in \mathsf{SPFAU}$,
(2) For each $c \in C$, $c^{\mathcal{A}} \in Urel_{\mathfrak{A}}$,

(3) For each $f \in F$ of arity k, $f^{\mathcal{A}} : \mathit{Urel}_{\mathfrak{A}}{}^{k} \to \mathit{Urel}_{\mathfrak{A}}$,
(4) For each $p \in P$ of arity $\langle a_1, a_2 \rangle$, $p^{\mathcal{A}} \in \mathfrak{A}$ has arity $\langle a_1, a_2 \rangle$.

Definition 5.12 Given $\mathfrak{A} \in \mathsf{PFAU}$, a mapping $\nu : \mathit{IndVar} \to \mathit{Urel}_{\mathfrak{A}}$ is called a *valuation of individual variables.* If $x \in \mathit{IndVar}$, by $\nu[x/a]$ we denote the valuation defined by:

$$\nu[x/a](y) = \begin{cases} \nu(y) & \text{if } y \neq x, \\ a & \text{if } y = x \ . \end{cases}$$

Definition 5.13 Given a structure $\mathcal{A} = \left\langle \mathfrak{A}, C^{\mathcal{A}}, F^{\mathcal{A}}, P^{\mathcal{A}} \right\rangle$ adequate for $\mathsf{ETFR}(C, F, P)$ and a valuation of individual variables ν, we define the mapping V_ν giving meaning in $\mathcal{A}$ to terms of $\mathit{IndTerm}(C, F)^{\star}$ as follows:

(1) $V_\nu(x) = \nu(x)$, for each individual variable x
(2) $V_\nu(c) = c^{\mathcal{A}}$, for each $c \in C$
(3) $V_\nu(f(t_1, \ldots, t_k)) = f^{\mathcal{A}}(V_\nu(t_1), \ldots, V_\nu(t_k))$ for each $f \in F$
(4) $V_\nu(\star(t, t')) = *(V_\nu(t), V_\nu(t'))$

Definition 5.14 Let $\mathcal{A} = \left\langle \mathfrak{A}, C^{\mathcal{A}}, F^{\mathcal{A}}, P^{\mathcal{A}} \right\rangle$ be an adequate structure for $\mathsf{ETFR}(C, F, P)$, and let $m : \mathit{RelVar} \to A$. The pair $\langle \mathcal{A}, m \rangle$ is called a *model* of $\mathsf{ETFR}(C, F, P)$. Mapping m extends homomorphically to compound relational designations. In order to simplify the notation, m will also denote the homomorphic extension.

Definition 5.15 Given $\varphi \in \mathit{ForETFR}(C, F, P)$ and a valuation for the individual variables ν, we say that ν satisfies the formula φ in the model $\mathcal{M} = \left\langle \left\langle \mathfrak{A}, C^{\mathcal{M}}, F^{\mathcal{M}}, P^{\mathcal{M}} \right\rangle, m \right\rangle$ (denoted by $\mathcal{M}, \nu \models_{\mathsf{ETFR}} \varphi$) whenever:

- If $\varphi = t_1 p t_2$ with $p \in \mathit{RelDes}(P)$ and $t_1, t_2 \in \mathit{IndTerm}(C, F)^{\star}$,

$$\mathcal{M}, \nu \models_{\mathsf{ETFR}} \varphi \text{ iff } \langle V_\nu(t_1), V_\nu(t_2) \rangle \in m(p) \ .$$

- If $\varphi = \neg\alpha$,

$$\mathcal{M}, \nu \models_{\mathsf{ETFR}} \varphi \quad \text{iff} \quad \mathcal{M}, \nu \not\models_{\mathsf{ETFR}} \alpha \ .$$

- If $\varphi = \alpha \vee \beta$,

$$\mathcal{M}, \nu \models_{\mathsf{ETFR}} \varphi \quad \text{iff} \quad \mathcal{M}, \nu \models_{\mathsf{ETFR}} \alpha \text{ or } \mathcal{M}, \nu \models_{\mathsf{ETFR}} \beta \ .$$

- If $\varphi = \alpha \wedge \beta$,

$$\mathcal{M}, \nu \models_{\mathsf{ETFR}} \varphi \quad \text{iff} \quad \mathcal{M}, \nu \models_{\mathsf{ETFR}} \alpha \text{ and } \mathcal{M}, \nu \models_{\mathsf{ETFR}} \beta \ .$$

– If $\varphi = \exists x\alpha$,

$$\mathcal{M}, \nu \models_{\mathsf{ETFR}} \exists x\alpha \quad \text{iff} \quad \text{there exists } a \in Urel_{\mathfrak{A}} \text{ such that } \mathcal{M}, \nu[x/a] \models_{\mathsf{ETFR}} \alpha \ .$$

– If $\varphi = \forall x\alpha$,

$$\mathcal{M}, \nu \models_{\mathsf{ETFR}} \forall x\alpha \quad \text{iff} \quad \text{for each } a \in Urel_{\mathfrak{A}},\ \mathcal{M}, \nu[x/a] \models_{\mathsf{ETFR}} \alpha \ .$$

The following mapping translates formulas from $FOLE(C, F, P)$ into formulas of $\mathsf{ETFR}(C, F, P')$. We will denote by $ForFOLE(C, F, P)$ the set of formulas in $FOLE(C, F, P)$.

In order to simplify proofs, from here on given a first-order predicate symbol $p \in P$ with arity k, we will assume that the arity of $p' \in P'$ is $\langle n, m\rangle$ $(n, m \in \mathbb{N})$ with $n + m = k$. Moreover, we will assume that the first n parameters from p will be input parameters of p' and the last m parameters from p will be output parameters of p'.

Definition 5.16 We define the translation T_∇ mapping formulas from $ForFOLE(C, F, P)$ to $ForETFR(C, F, P')$ inductively as follows:

(1) $T_\nabla(p(t_1, \ldots, t_n, t'_1, \ldots, t'_m)) = t_1 \star \cdots \star t_n p' t'_1 \star \cdots \star t'_m$, if $p \in P$ and p' has arity $\langle n, m\rangle$.
(2) $T_\nabla(t_1 = t_2) = t_1 1' t_2$,
(3) $T_\nabla(\neg\alpha) = \neg T_\nabla(\alpha)$,
(4) $T_\nabla(\alpha \vee \beta) = T_\nabla(\alpha) \vee T_\nabla(\beta)$,
(5) $T_\nabla(\alpha \wedge \beta) = T_\nabla(\alpha) \wedge T_\nabla(\beta)$,
(6) $T_\nabla(\exists x\alpha) = \exists x T_\nabla(\alpha)$,
(7) $T_\nabla(\forall x\alpha) = \forall x T_\nabla(\alpha)$.

Given a first-order term t and a valuation of variables ν into a $FOLE$ model $\mathfrak{A}$, by $\mathbf{V}_\nu(t)$ we denote the value of t under the valuation ν in the model $\mathfrak{A}$.

Lemma 5.1 *Let $\phi \in ForFOLE(C, F, P)$. Given a $FOLE(C, F, P)$ model $\mathfrak{A}$ with domain A, there exists a $\mathsf{ETFR}(C, F, P')$ model $\mathcal{B}$ such that for every valuation of the individual variables ν*

$$\mathfrak{A}, \nu \models_{FOLE} \phi \qquad \Longleftrightarrow \qquad \mathcal{B}, \nu \models_{\mathsf{ETFR}} T_\nabla(\phi) \ .$$

Proof Let $\mathfrak{B}$ be the full fork algebra with set of urelements A. Define, for $c \in C$ and $f \in F$, $c^{\mathcal{B}} = c^{\mathfrak{A}}$ and $f^{\mathcal{B}} = f^{\mathfrak{A}}$. Define, for $p \in P$ of arity n and $p' \in P'$ of arity $\langle r, s \rangle$ with $r + s = n$,

$$p'^{\mathcal{B}} = \left\{ \langle a_1 * \cdots * a_r, b_1 * \cdots * b_s \rangle : p^{\mathfrak{A}}(a_1, \ldots, a_r, b_1, \ldots, b_s) \right\} .$$

Let $m : \mathit{RelVar} \to B$ be arbitrary, and let $\mathcal{B} = \langle \langle \mathfrak{B}, C^{\mathcal{B}}, F^{\mathcal{B}}, P'^{\mathcal{B}} \rangle, m \rangle$. It is clear that

$$\forall t \in \mathit{IndTerm}(C, F),\ V_\nu(t) = \mathbf{V}_\nu(t) . \tag{5.1}$$

Let us proceed by induction on the complexity of the formula ϕ.

Given $p \in P$ of arity k, $p' \in P'$ of arity $\langle r, s \rangle$ and $t_1, \ldots, t_r, t'_1, \ldots, t'_s \in \mathit{IndTerm}(C, F)$,

$$\begin{array}{ll}
 & \mathfrak{A} \models_{FOLE} p(t_1, \ldots, t_r, t'_1, \ldots, t'_s) \\
\iff & \{\text{by Def. } \models_{FOLE}\} \\
 & \langle \mathbf{V}_\nu(t_1), \ldots, \mathbf{V}_\nu(t_r), \mathbf{V}_\nu(t'_1), \ldots, \mathbf{V}_\nu(t'_s) \rangle \in p^{\mathfrak{A}} \\
\iff & \left\{\text{by Def. } p'^{\mathcal{B}}\right\} \\
 & \langle \mathbf{V}_\nu(t_1) * \cdots * \mathbf{V}_\nu(t_r), \mathbf{V}_\nu(t'_1) * \cdots * \mathbf{V}_\nu(t'_s) \rangle \in p'^{\mathcal{B}} \\
\iff & \{\text{by } (5.1)\} \\
 & \langle V_\nu(t_1) * \cdots * V_\nu(t_r), V_\nu(t'_1) * \cdots * V_\nu(t'_s) \rangle \in p'^{\mathcal{B}} \\
\iff & \{\text{by Def. } \models_{\mathsf{ETFR}}\} \\
 & \mathcal{B}, \nu \models_{\mathsf{ETFR}} t_1 \star \cdots \star t_r p' t'_1 \star \cdots \star t'_s \\
\iff & \{\text{Def. } T_\nabla\} \\
 & \mathcal{B}, \nu \models_{\mathsf{ETFR}} T_\nabla(p(t_1, \ldots, t_r, t'_1, \ldots, t'_s)).
\end{array}$$

The remaining part of the inductive proof is simple and is left to the reader. □

Lemma 5.2 *Let $\phi \in \mathit{ForFOLE}(C, F, P)$. Let $\mathcal{A}$ be a $\mathsf{ETFR}(C, F, P')$ model. Then there exists a $\mathit{FOLE}(C, F, P)$ model $\mathfrak{B}$ such that for every valuation of the individual variables ν,*

$$\mathcal{A}, \nu \models_{\mathsf{ETFR}} T_\nabla(\phi) \quad \iff \quad \mathfrak{B}, \nu \models_{FOLE} \phi .$$

Proof Let $\mathcal{A} = \left\langle \left\langle \mathfrak{A}, C^{\mathcal{A}}, F^{\mathcal{A}}, P'^{\mathcal{A}} \right\rangle, m \right\rangle$. Let us define B, the universe of $\mathfrak{B}$, as $\mathit{Urel}_{\mathfrak{A}}$. For each $c \in C$ and $f \in F$, we define $c^{\mathfrak{B}} = c^{\mathcal{A}}$ and

$f^{\mathfrak{B}} = f^{\mathcal{A}}$. For each $p' \in P'$ of arity $\langle r, s\rangle$ we define

$$p^{\mathfrak{B}} = \left\{ \langle a_1, \ldots, a_r, b_1, \ldots, b_s\rangle : \langle a_1 * \cdots * a_r, b_1 * \cdots * b_s\rangle \in p'^{\mathcal{A}} \right\}.$$

Let $\mathfrak{B} = \left\langle B, C^{\mathfrak{B}}, F^{\mathfrak{B}}, P^{\mathfrak{B}} \right\rangle$. It is clear that

$$\forall t \in \mathit{IndTerm}(C, F),\ V_\nu(t) = \mathbf{V}_\nu(t)\ . \tag{5.2}$$

Let us proceed by induction on the complexity of the formula ϕ. If $\phi = p(t_1, \ldots, t_r, t'_1, \ldots, t'_s)$ with $p \in P$, then

$$\begin{array}{ll}
 & \mathcal{A}, \nu \models_{\mathsf{ETFR}} T_\nabla(p(t_1, \ldots, t_r, t'_1, \ldots, t'_s)) \\
\Longleftrightarrow & \{\text{by Def. } T_\nabla\} \\
 & \mathcal{A}, \nu \models_{\mathsf{ETFR}} t_1 \star \cdots \star t_r p' t'_1 \star \cdots \star t'_s \\
\Longleftrightarrow & \{\text{by Def. } \models_{\mathsf{ETFR}}\} \\
 & \langle V_\nu(t_1) * \cdots * V_\nu(t_r), V_\nu(t'_1) * \cdots * V_\nu(t'_r)\rangle \in p'^{\mathcal{A}} \\
\Longleftrightarrow & \{\text{by Def. } p^{\mathfrak{B}}\} \\
 & \langle V_\nu(t_1), \ldots, V_\nu(t_r), V_\nu(t'_1), \ldots, V_\nu(t'_r)\rangle \in p^{\mathfrak{B}} \\
\Longleftrightarrow & \{\text{by (5.2)}\} \\
 & \langle \mathbf{V}_\nu(t_1), \ldots, \mathbf{V}_\nu(t_r), \mathbf{V}_\nu(t'_1), \ldots, \mathbf{V}_\nu(t'_r)\rangle \in p^{\mathfrak{B}} \\
\Longleftrightarrow & \{\text{by Def. } \models_{FOLE}\} \\
 & \mathfrak{B}, \nu \models_{FOLE} p(t_1, \ldots, t_r, t'_1, \ldots, t'_s)\ .
\end{array}$$

The remaining part of the inductive proof is simple and is left to the reader. □

The next theorem proves the interpretability of classical first-order logic with equality into the elementary theory of fork relations. This will serve as an intermediate step in the proof of interpretability of classical first-order logic into the calculus of fork relations. Notice that since first-order predicate symbols do not divide arguments between input arguments and output arguments, this intermediate step arises naturally in the process of translating specifications to the calculus of fork relations.

By $\models_{FOLE} \phi$ we will denote the fact that formula ϕ is *valid* in *FOLE*. In a similar way we say that a formula ϕ from ETFR is *valid* in ETFR if for each ETFR model $\mathcal{A}$ and each valuation ν of the individual variables, $\mathcal{A}, \nu \models_{\mathsf{ETFR}} \phi$.

Theorem 5.1 *Let ϕ be a $FOLE(C, F, P)$ formula. Then*

$$\models_{FOLE} \phi \qquad \Longleftrightarrow \qquad \models_{\mathsf{ETFR}} T_\nabla(\phi)\ .$$

Proof
$\Rightarrow$) If $\not\models_{\mathsf{ETFR}} T_\nabla(\phi)$, then there exists an $\mathsf{ETFR}(C, F, P')$ model $\mathcal{A}$ and a valuation of individual variables ν such that $\mathcal{A}, \nu \not\models_{\mathsf{ETFR}} T_\nabla(\phi)$. Then, by Lemma 5.2, there exists a $FOLE(C, F, P)$ model $\mathfrak{B}$ such that $\mathfrak{B}, \nu \not\models_{FOLE} \phi$. Then, $\not\models_{FOLE} \phi$.
$\Leftarrow$) If $\not\models_{FOLE} \phi$, then there exists a $FOLE(C, F, P)$ model $\mathfrak{A}$ and a valuation of individual variables ν such that $\mathfrak{A}, \nu \not\models_{FOLE} \phi$. By Lemma 5.1 there exists a $\mathsf{ETFR}(C, F, P')$ structure $\mathcal{B}$ such that $\mathcal{B}, \nu \not\models_{\mathsf{ETFR}} T_\nabla(\phi)$. Then, $\not\models_{\mathsf{ETFR}} T_\nabla(\phi)$. □

In the remaining part of this section we will show that $\mathsf{ETFR}(C, F, P)$ can be interpreted into $\mathsf{CFR}(A)$ for a suitable set of constant relation symbols A. Finally, by exploiting the relationship which exists between $\mathsf{CFR}(A)$ and abstract fork algebras we will show how to reason algebraically in order to prove logical properties from $FOLE$ and ETFR. By $1'^{k}_{\mathsf{U}}$ we denote the relation $\underbrace{1'_{\mathsf{U}} \otimes \cdots \otimes 1'_{\mathsf{U}}}_{k \text{ times}}$.

Given sets C, F and P consisting of constant, function and relation symbols respectively, by K we denote the set $C' \cup F' \cup P'$. By $\mathsf{CFR}^+(K)$ we denote the extension of $\mathsf{CFR}(K)$ obtained by adding the following axioms:

(1) The formula

$$1;1'_{\mathsf{U}};1 = 1,$$

which implies that models of CFR^+ are abstract fork algebras with a nonempty set of urelements.

(2) For each $c \in C$, we add the following equations stating that c' is a *constant* relation having a urelement in its range:

$$\begin{aligned} &\breve{c'};c' + 1'_{\mathsf{U}} = 1'_{\mathsf{U}} \ (c' \text{ is functional}), \\ &1;c' = c' \ (c' \text{ is left-ideal}) \ , \\ &c';1 = 1 \ (c' \text{ is nonempty}). \end{aligned}$$

(3) For each $f \in F$ with arity k, we add equations stating that f' is a functional relation that takes k urelements as input and produces urelements as output:

$$\begin{aligned} &\breve{f'};f' + 1'_{\mathsf{U}} = 1'_{\mathsf{U}}, \\ &1'^{k}_{\mathsf{U}};f' = f'. \end{aligned}$$

(4) For each $p \in P$ with arity $\langle m, n \rangle$, the following equations stating that p' is a binary relation expecting m urelements as input and n urelements as output:

$$1\text{'}_{\mathsf{U}}^{m} ; p' ; 1\text{'}_{\mathsf{U}}^{n} = p' \ .$$

In (3) and (4) we are assuming that f' has input arity k and that p' has arity $\langle m, n \rangle$. We can generalize to arbitrary arities by rearranging parenthesis in a convenient way. For example, if p' is a relation constant whose arity is $\langle (u \star u) \star (u \star u), (u \star u) \star u \rangle$, we would impose the condition

$$(1\text{'}_{\mathsf{U}} \otimes 1\text{'}_{\mathsf{U}}) \otimes (1\text{'}_{\mathsf{U}} \otimes 1\text{'}_{\mathsf{U}}) p' (1\text{'}_{\mathsf{U}} \otimes 1\text{'}_{\mathsf{U}}) \otimes 1\text{'}_{\mathsf{U}} = p' \ .$$

Note that given a finite set K with constant, function and relation symbols, only a finite number of equations are introduced in (1)–(4) above.

In what follows, $t^{;n}$ is an abbreviation for $t; \cdots ;t$ (n times). For the sake of completeness, $t^{;0}$ is defined as 1'.

Prior to defining the mapping translating $\mathsf{ETFR}(C, F, P)$ formulas into $\mathsf{CFR}^{+}(K)$ formulas, we will translate individual terms to relation designations. This is necessary when translating atomic $\mathsf{ETFR}(C, F, P)$ formulas of the form $t R t'$, with $t, t' \in \mathit{IndTerm}(C, F)^{\star}$ and $R \in \mathit{RelDes}(P)$.

For the following definitions, σ will be a sequence of numbers sorted in increasing order. Intuitively, the sequence σ contains the indices of those individual variables that appear free in the formula (or term) being translated. By $\mathit{Ord}(n, \sigma)$ we denote the position of the index n in the sequence σ, by $[\sigma \oplus n]$ we denote the extension of the sequence σ with the index n, and by $\sigma(k)$ we denote the element in the k-th position of σ.

Definition 5.17 The mapping $\delta_\sigma : \mathit{IndTerm}(C, F) \to \mathit{RelDes}(C' \cup F')$, translating individual terms into relation designations, is defined inductively by the conditions:

(1) $\delta_\sigma(v_i) = \begin{cases} \rho^{;\mathit{Ord}(i,\sigma)-1} ; \pi & \text{if } i \text{ is not the last index in } \sigma, \\ \rho^{;\mathit{Length}(\sigma)-1} & \text{otherwise.} \end{cases}$

(2) $\delta_\sigma(c) = c'$ for each $c \in C$.

(3) $\delta_\sigma(f(t_1, \ldots, t_m)) = (\delta_\sigma(t_1) \nabla \cdots \nabla \delta_\sigma(t_m)) ; f'$ for each $f \in F$.

Before defining the mapping T_σ translating ETFR formulas, we need to define some auxiliary terms. Given a sequence σ such that $\mathit{Length}(\sigma) = l$, we define the term $\Delta_{\sigma,n}$ $(n < \omega)$ by the condition

$$\Delta_{\sigma,n} = \begin{cases} \delta_\sigma(v_{\sigma(1)})\nabla\cdots\nabla\delta_\sigma(v_{\sigma(k-1)})\nabla 1_{\mathsf{U}}\nabla\delta_\sigma(v_{\sigma(k)})\nabla\cdots\nabla\delta_\sigma(v_{\sigma(l)}) \\ \qquad\qquad\qquad\qquad\qquad\qquad \text{if } k = Ord(n, [\sigma\oplus n]) \leq l, \\ \delta_\sigma(v_{\sigma(1)})\nabla\cdots\nabla\delta_\sigma(v_{\sigma(l)})\nabla 1_{\mathsf{U}} \qquad \text{if } Ord(n, [\sigma\oplus n]) = l+1. \end{cases}$$

The term $\Delta_{\sigma,n}$ can be understood as a cylindrification [L. Henkin et al. (1971); L. Henkin et al. (1985)] in the k-th coordinate of an l-dimensional space.

For the next mapping to be correctly defined, we assume that atomic formulas of the form $R = S$ do not occur in the scope of a quantifier over individual variables. This is a reasonable assumption because, since atomic formulas of this form do not contain any individual variables, they can be promoted outside the scope of quantifiers. We will keep this assumption for the remaining part of the chapter.

Definition 5.18 We define the mapping T_σ translating $\mathsf{ETFR}(C, F, P)$ formulas to $\mathsf{CFR}^+(K)$ formulas as follows:

(1) $T_\sigma(R = S) \overset{\text{def}}{=} R = S$ $(R, S \in RelDes(P))$,

(2) $T_\sigma(t_1 \star\cdots\star t_r R t'_1 \star\cdots\star t'_s) \overset{\text{def}}{=}$

$$T'_\sigma(t_1 \star\cdots\star t_r R t'_1 \star\cdots\star t'_s) = 1\text{'}^k_{\mathsf{U}};1,$$

$(k = Length(\sigma)$, $t_i, t'_j \in IndTerm(C, F)$ for all i, j, $1 \leq i \leq r$, $1 \leq j \leq s$ and $R \in RelDes(P))$,

(3) $T_\sigma(\neg\alpha) \overset{\text{def}}{=} \neg T_\sigma(\alpha)$,

(4) $T_\sigma(\alpha\vee\beta) \overset{\text{def}}{=} T_\sigma(\alpha)\vee T_\sigma(\beta)$,

(5) $T_\sigma(\alpha\wedge\beta) \overset{\text{def}}{=} T_\sigma(\alpha)\wedge T_\sigma(\beta)$,

(6) $T_\sigma(\exists v_n\alpha) \overset{\text{def}}{=} T'_\sigma(\exists v_n\alpha) = 1\text{'}^k_{\mathsf{U}};1$, $(k = Length(\sigma))$,

(7) $T_\sigma(\forall v_n\alpha) \overset{\text{def}}{=} T'_\sigma(\forall v_n\alpha) = 1\text{'}^k_{\mathsf{U}};1$, $(k = Length(\sigma))$,

(8) $T'_\sigma(t_1 \star\cdots\star t_r R t'_1 \star\cdots\star t'_s) \overset{\text{def}}{=}$

$$\Big((\delta_\sigma(t_1)\nabla\cdots\nabla\delta_\sigma(t_r)) \ \nabla \ (\delta_\sigma(t'_1)\nabla\cdots\nabla\delta_\sigma(t'_s))\,;\breve{R}\Big)\,;\breve{2};1, \quad (5.3)$$

(9) $T'_\sigma(\neg\alpha) \overset{\text{def}}{=} \overline{T'_\sigma(\alpha)}$,

(10) $T'_\sigma(\alpha\vee\beta) \overset{\text{def}}{=} T'_\sigma(\alpha){+}T'_\sigma(\beta)$,

(11) $T'_\sigma(\alpha\wedge\beta) \overset{\text{def}}{=} T'_\sigma(\alpha){\cdot}T'_\sigma(\beta)$,

(12) $T'_\sigma(\exists v_n(\alpha)) \overset{\text{def}}{=} \Delta_{\sigma,n};T'_{[\sigma\oplus n]}(\alpha)$,
(13) $T'_\sigma(\forall v_n(\alpha)) \overset{\text{def}}{=} T'_\sigma(\neg\exists v_n\neg\alpha)$.

In (8) we are assuming that R has arity $\langle r,s\rangle$ $(r,s\in\mathbb{N})$. If we allow for arbitrary arities, then the parenthesis in formula (5.3) must be rearranged. For example, if R has arity $\langle (u\star u)\star(u\star u),(u\star u)\star u\rangle$, then

$$\begin{aligned}&T'_\sigma((t_1\star t_2)\star(t_3\star t_4)R(t_5\star t_6)\star t_7)\\&=\Big(((\delta_\sigma(t_1)\nabla\delta_\sigma(t_2))\ \nabla\ (\delta_\sigma(t_3)\nabla\delta_\sigma(t_4)))\ \nabla\ ((\delta_\sigma(t_5)\nabla\delta_\sigma(t_6))\ \nabla\delta_\sigma(t_7))\ ;\breve{R}\Big)\ ;\breve{\mathfrak{2}};1\ .\end{aligned}$$

Given a valuation of individual variables ν and a sequence of indices σ, by $s_{\nu,\sigma}$ we denote the object $a_1*\cdots*a_i*\cdots*a_n$ where:

(1) $n = \mathit{Length}(\sigma)$,
(2) $a_i = \nu(v_{\sigma(i)})$ for all i, $1\le i\le n$.

Given a formula or term α, by σ_α we denote the sequence of indices of variables with free occurrences in α, sorted in increasing order.

Lemma 5.3 *Let $\alpha\in \mathit{ForETFR}(C,F,P)$ not containing any atomic subformula of the form $R=S$, let $\sigma=\sigma_\alpha$ and let $\mathcal{A}$ be a* ETFR(C,F,P) *model. Then there is a* CFR$^+(K)$ *model $\mathcal{B}=\langle\langle\mathfrak{B},C^\mathcal{B}\cup F^\mathcal{B}\cup P^\mathcal{B}\rangle,m'\rangle$ such that*

$$\mathcal{A},\nu\models_{\mathsf{ETFR}}\alpha \qquad\Longleftrightarrow\qquad s_{\nu,\sigma}\in\mathsf{dom}\,(m'\,(T'_\sigma(\alpha)))\ .$$

Proof Assume $\mathcal{A}=\langle\langle\mathfrak{A},C^\mathcal{A},F^\mathcal{A},P^\mathcal{A}\rangle,m\rangle$. Let us define $\mathcal{B}$ as follows:

(1) Since $\mathfrak{A}\in\mathbf{S}\,\mathsf{FullPFAU}$, let $\mathfrak{B}\in\mathsf{FullPFAU}$ such that $\mathfrak{A}\in\mathbf{S}\{\mathfrak{B}\}$,
(2) $c^\mathcal{B}=\{\langle x,c^\mathcal{A}\rangle : x\in U_\mathfrak{A}\}$, for each $c\in C$,
(3) $f^\mathcal{B}=\{\langle a_1*\cdots*a_n,b\rangle : f^\mathcal{A}(a_1,\ldots,a_n)=b\}$, for each $f\in F$,
(4) $p^\mathcal{B}=p^\mathcal{A}$, for each $p\in P$,
(5) $m'(R)=m(R)$ for each $R\in \mathit{RelVar}$.

It is clear that individual terms denote functional relations. Given a functional binary relation f and an element a, by $[f](a)$ we denote that element b such that $\langle a,b\rangle\in f$. We will prove next that for any term $t\in \mathit{IndTerm}(C,F)$ and any sequence σ that extends σ_t, $V_\nu(t)=[m'\,(\delta_\sigma(t))](s_{\nu,\sigma})$.

Let $t=v_i\in \mathit{IndVar}$. If i is the last index in σ, then $V_\nu(v_i)=\nu(v_i)=[\rho^{\mathit{Length}(\sigma)-1}](s_{\nu,\sigma})=[m'\,(\delta_\sigma(v_i))](s_{\nu,\sigma})$. If i is not the last index in σ then $V_\nu(v_i)=\nu(v_i)=[\rho^{\mathit{Ord}(i,\sigma)-1};\pi](s_{\nu,\sigma})=[m'\,(\delta_\sigma(v_i))](s_{\nu,\sigma})$.

Let $t=c\in C$. $V_\nu(c)=c^\mathcal{A}=[c^\mathcal{B}](s_{\nu,\sigma})=[m'\,(c)](s_{\nu,\sigma})$.

Let $t = f(t_1,\ldots,t_k)$. $V_\nu(f(t_1,\ldots,t_k)) = f^{\mathcal{A}}(V_\nu(t_1),\ldots,V_\nu(t_k))$. By inductive hypothesis

$$V_\nu(t_i) = [m'(\delta_\sigma(t_i))](s_{\nu,\sigma}) \qquad \text{for all } i, 1 \leq i \leq k \,.$$

Then,

$$\begin{aligned}
&f^{\mathcal{A}}(V_\nu(t_1),\ldots,V_\nu(t_k))\\
&\quad= f^{\mathcal{A}}([m'(\delta_\sigma(t_1))](s_{\nu,\sigma}),\ldots,[m'(\delta_\sigma(t_k))](s_{\nu,\sigma}))\\
&\quad= [f^{\mathcal{B}}]([m'(\delta_\sigma(t_1))](s_{\nu,\sigma}) * \cdots * [m'(\delta_\sigma(t_k))](s_{\nu,\sigma}))\\
&\quad= [m'(((\delta_\sigma(t_1)\nabla \cdots \nabla\delta_\sigma(t_k))\,;f))](s_{\nu,\sigma})\\
&\quad= [m'(\delta_\sigma(f(t_1,\ldots,t_k)))](s_{\nu,\sigma}).
\end{aligned}$$

The remaining part of the proof follows by induction on the structure of the formula α. Notice that since $m(R) = m'(R)$ for all $R \in \mathit{RelVar}$, then for every $S \in \mathit{RelDes}(P)$ we have $m(S) = m'(S)$.

If $\alpha = t_1 \star \cdots \star t_r R t_1' \star \cdots \star t_s'$, then

$$\begin{aligned}
&\mathcal{A},\nu \models_{\mathsf{ETFR}} t_1 \star \cdots \star t_r R t_1' \star \cdots \star t_s'\\
&\quad\iff \langle V_\nu(t_1) * \cdots * V_\nu(t_r), V_\nu(t_1') * \cdots * V_\nu(t_s')\rangle \in m(R)\\
&\quad\iff \langle [m'(\delta_\sigma(t_1))](s_{\nu,\sigma}) * \cdots * [m'(\delta_\sigma(t_r))](s_{\nu,\sigma}),\\
&\qquad\quad [m'(\delta_\sigma(t_1'))](s_{\nu,\sigma}) * \cdots * [m'(\delta_\sigma(t_s'))](s_{\nu,\sigma})\rangle \in m(R)\\
&\quad\iff \langle [m'(\delta_\sigma(t_1))](s_{\nu,\sigma}) * \cdots * [m'(\delta_\sigma(t_r))](s_{\nu,\sigma}),\\
&\qquad\quad [m'(\delta_\sigma(t_1'))](s_{\nu,\sigma}) * \cdots * [m'(\delta_\sigma(t_s'))](s_{\nu,\sigma})\rangle \in m'(R)\\
&\quad\iff s_{\nu,\sigma} \in \mathsf{dom}\Big((m'(\delta_\sigma(t_1))\,\nabla \cdots \nabla m'(\delta_\sigma(t_r)))\\
&\qquad\quad \nabla\ ((m'(\delta_\sigma(t_1'))\,\nabla \cdots \nabla m'(\delta_\sigma(t_s')))\,;(m'(R))^{\smile})\,;\breve{2}\,;1\Big)\\
&\quad\iff s_{\nu,\sigma} \in \mathsf{dom}\Big(m'\Big(\Big((\delta_\sigma(t_1)\nabla \cdots \nabla\delta_\sigma(t_r))\\
&\qquad\quad \nabla\ \Big((\delta_\sigma(t_1')\nabla \cdots \nabla\delta_\sigma(t_s'))\,;\breve{R}\Big)\Big)\,;\breve{2}\,;1\Big)\Big)\\
&\quad\iff s_{\nu,\sigma} \in \mathsf{dom}\,(m'(T_\sigma'(t_1 \star \cdots \star t_r R t_1' \star \cdots \star t_s'))) \,.
\end{aligned}$$

This concludes the treatment for atomic formulas.

If $\alpha = \neg\beta$, then

$$\begin{aligned}
\mathcal{A}, \nu \models_{\mathsf{ETFR}} \neg\beta &\iff \mathcal{A}, \nu \not\models_{\mathsf{ETFR}} \beta \\
&\iff s_{\nu,\sigma} \notin \mathsf{dom}\,(m'\,(T'_\sigma(\beta))) \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\left(\overline{m'\,(T'_\sigma(\beta))}\right) \\
&\qquad\qquad \text{(by } m'\,(T'_\sigma(\beta)) \text{ right-ideal)} \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\left(m'\left(\overline{T'_\sigma(\beta)}\right)\right) \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\,(m'\,(T'_\sigma(\neg\beta)))\,.
\end{aligned}$$

If $\alpha = \beta \vee \gamma$, then

$$\begin{aligned}
&\mathcal{A}, \nu \models_{\mathsf{ETFR}} \beta \vee \gamma \\
&\iff \mathcal{A}, \nu \models_{\mathsf{ETFR}} \beta \text{ or } \mathcal{A}, \nu \models_{\mathsf{ETFR}} \gamma \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\,(m'\,(T'_\sigma(\beta))) \text{ or } s_{\nu,\sigma} \in \mathsf{dom}\,(m'\,(T'_\sigma(\gamma))) \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\,(m'\,(T'_\sigma(\beta)) \cup m'\,(T'_\sigma(\gamma))) \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\,(m'\,(T'_\sigma(\beta) + T'_\sigma(\gamma))) \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\,(m'\,(T'_\sigma(\beta \vee \gamma)))\,.
\end{aligned}$$

If $\alpha = \exists v_i \beta$, then

$$\begin{aligned}
&\mathcal{A}, \nu \models_{\mathsf{ETFR}} \exists v_i \beta \\
&\iff \text{there is } a \in \mathit{Urel}_{\mathfrak{A}} \text{ such that } \mathcal{A}, \nu[v_i/a] \models_{\mathsf{ETFR}} \beta \\
&\iff \text{there is } a \in \mathit{Urel}_{\mathfrak{A}} \text{ s.t. } s_{\nu[v_i/a],[\sigma\oplus i]} \in \mathsf{dom}\left(m'\left(T'_{[\sigma\oplus i]}(\beta)\right)\right) \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\left(m'\,(\Delta_{\sigma,i}) \circ m'\left(T'_{[\sigma\oplus i]}(\beta)\right)\right) \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\left(m'\left(\Delta_{\sigma,i} ; T'_{[\sigma\oplus i]}(\beta)\right)\right) \\
&\iff s_{\nu,\sigma} \in \mathsf{dom}\,(m'\,(T'_\sigma(\exists v_i \beta)))\,.
\end{aligned}$$

□

Lemma 5.4 *Let $\alpha \in ForETFR(C, F, P)$ not containing any atomic subformula of form $R = S$. Let $\sigma = \sigma_\alpha$ of length k. Let $\mathcal{A}$ be a* $\mathsf{ETFR}(C, F, P)$ *model. Then, there exists a model $\mathcal{B} = \langle \langle \mathfrak{B}, C^{\mathcal{B}} \cup F^{\mathcal{B}} \cup P^{\mathcal{B}} \rangle, m' \rangle$ for* $\mathsf{CFR}^+(K)$ *such that*

$$\mathcal{A} \models_{\mathsf{ETFR}} \alpha \qquad \iff \qquad \mathcal{B} \models_{\mathsf{CFR}^+} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}} ; 1\ .$$

Proof Let $\mathcal{B}$ be defined as in Lemma. 5.3.

$$\begin{aligned}\mathcal{A} \models_{\mathsf{ETFR}} \alpha &\iff \mathcal{A}, \nu \models_{\mathsf{ETFR}} \alpha \text{ for all } \nu \\ &\iff s_{\nu,\sigma} \in \mathsf{dom}\,(m'\,(T'_\sigma(\alpha))) \text{ for all } \nu \qquad \text{(by Lemma 5.3)} \\ &\iff m'\,(T'_\sigma(\alpha)) = 1\text{'}^k_{\mathsf{U}};1 \\ &\iff \mathcal{B} \models_{\mathsf{CFR^+}} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}};1.\end{aligned}$$

□

Lemma 5.5 *Let $\alpha \in ForETFR(C, F, P)$, let $\sigma = \sigma_\alpha$ of length k, and let $\mathcal{A}$ be a* ETFR(C, F, P) *model. Then, there exists a* CFR$^+(K)$ *model $\mathcal{B}$ such that*

$$\mathcal{A} \models_{\mathsf{ETFR}} \alpha \qquad \iff \qquad \mathcal{B} \models_{\mathsf{CFR^+}} T_\sigma(\alpha)\ .$$

Proof Assume $\mathcal{A} = \langle \langle \mathfrak{A}, C^{\mathcal{A}}, F^{\mathcal{A}}, P^{\mathcal{A}} \rangle, m \rangle$. Let $\mathcal{B}$ be defined as in Lemma 5.3.

If α is $R = S$ with $R, S \in RelDes(P)$, then

$$\begin{aligned}\mathcal{A} \models_{\mathsf{ETFR}} R = S &\iff m(R) = m(S) \\ &\iff m'(R) = m'(S) \\ &\iff \mathcal{B} \models_{\mathsf{CFR^+}} R = S \\ &\iff \mathcal{B} \models_{\mathsf{CFR^+}} T_\sigma(R = S).\end{aligned}$$

If α is $t_1 \star \cdots \star t_r R t'_1 \star \cdots \star t'_s$, then

$$\begin{aligned}&\mathcal{A} \models_{\mathsf{ETFR}} t_1 \star \cdots \star t_r R t'_1 \star \cdots \star t'_s \\ &\qquad \iff \mathcal{B} \models_{\mathsf{CFR^+}} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}};1 \qquad \text{(by Lemma 5.4)} \\ &\qquad \iff \mathcal{B} \models_{\mathsf{CFR^+}} T_\sigma(\alpha).\end{aligned}$$

If α is $\neg\beta$, then

$$\begin{aligned}\mathcal{A} \models_{\mathsf{ETFR}} \neg\beta &\iff \mathcal{A} \not\models_{\mathsf{ETFR}} \beta \\ &\iff \mathcal{B} \not\models_{\mathsf{CFR^+}} T_\sigma(\beta) \\ &\iff \mathcal{B} \models_{\mathsf{CFR^+}} \neg T_\sigma(\beta) \\ &\iff \mathcal{B} \models_{\mathsf{CFR^+}} T_\sigma(\neg\beta).\end{aligned}$$

If α is $\beta \vee \gamma$, then

$$\begin{aligned} \mathcal{A} \models_{\mathsf{ETFR}} \beta \vee \gamma &\iff \mathcal{A} \models_{\mathsf{ETFR}} \beta \text{ or } \mathcal{A} \models_{\mathsf{ETFR}} \gamma \\ &\iff \mathcal{B} \models_{\mathsf{CFR}^+} T_\sigma(\beta) \text{ or } \mathcal{B} \models_{\mathsf{CFR}^+} T_\sigma(\gamma) \\ &\iff \mathcal{B} \models_{\mathsf{CFR}^+} T_\sigma(\beta) \vee T_\sigma(\gamma) \\ &\iff \mathcal{B} \models_{\mathsf{CFR}^+} T_\sigma(\beta \vee \gamma). \end{aligned}$$

If α is $\exists v_i \beta$, then

$$\begin{aligned} &\mathcal{A} \models_{\mathsf{ETFR}} \exists v_i \beta \\ &\iff \mathcal{B} \models_{\mathsf{CFR}^+} T'_\sigma(\exists v_i \beta) = 1\text{'}^k_{\mathsf{U}};1 \qquad \text{(by Lemma 5.4)} \\ &\iff \mathcal{B} \models_{\mathsf{CFR}^+} T_\sigma(\exists v_i \beta). \end{aligned}$$

□

Definition 5.19 Let R be a set of constant relation symbols. A $\mathsf{CFR}^+(R)$ model $\mathcal{A} = \langle \langle \mathfrak{A}, R^{\mathcal{A}} \rangle, m \rangle$ is called *square* if $\mathfrak{A} \in \mathsf{SPFAU}$.

Given $\mathfrak{A} \in \mathsf{PFAU}$, $s = a_1 * \cdots * a_k$ ($a_i \in Urel_{\mathfrak{A}}$ for all i, $1 \leq i \leq k$) and a sequence σ of indices increasingly sorted and of length k, by $\nu_{s,\sigma}$ we denote the set of valuations of individual variables ν satisfying $\nu(v_{\sigma(i)}) = a_i$. Valuations in $\nu_{s,\sigma}$ agree in all those variables whose indices occur in σ.

Lemma 5.6 *Let $\alpha \in ForETFR(C, F, P)$ not containing any atomic subformula of form $R = S$, let $\sigma = \sigma_\alpha$ and let $\mathcal{A}$ be the square $\mathsf{CFR}^+(K)$ model $\langle \langle \mathfrak{A}, C^{\mathcal{A}} \cup F^{\mathcal{A}} \cup P^{\mathcal{A}} \rangle, m \rangle$. Then, there exists a $\mathsf{ETFR}(C, F, P)$ model $\mathcal{B}$ such that*

$$s \in \mathsf{dom}\,(m\,(T'_\sigma(\alpha))) \qquad \iff \qquad \mathcal{B}, \nu \models_{\mathsf{ETFR}} \alpha \textit{ for all } \nu \in \nu_{s,\sigma} .$$

Proof Let $\mathcal{B}$ be constructed as follows:

(1) Since $\mathfrak{A} \in \mathsf{S\,FullPFAU}$, let $\mathfrak{B} \in \mathsf{FullPFAU}$ such that $\mathfrak{A} \in \mathsf{S}\{\mathfrak{B}\}$.
(2) For each $c \in C$, if $c^{\mathcal{A}} = \{\langle x, a \rangle : x \in U_{\mathfrak{A}}\}$, define $c^{\mathcal{B}} = a$.
(3) For each $f \in F$ of arity k,

$$f^{\mathcal{B}}(a_1, \ldots, a_k) = b \qquad \iff \qquad \langle a_1 * \cdots * a_k, b \rangle \in f^{\mathcal{A}} .$$

(4) For each $p \in P$ we define $p^{\mathcal{B}} = p^{\mathcal{A}}$.
(5) We finally define $m'(R) = m(R)$ for all $R \in RelVar$.

We will first prove, as an auxiliary result, that for every term t,

$$[m\,(\delta_\sigma(t))]\,(s) = V_\nu(t) \qquad \text{for all } \nu \in \nu_{s,\sigma} .$$

If $t = v_i \in \mathit{IndVar}$ and i is the last index in σ,

$$\begin{aligned}[m\left(\delta_\sigma(v_i)\right)](s) &= \left[m\left(\rho^{Length(\sigma)-1}\right)\right](s)\\ &= s_{Length(\sigma)}\\ &= V_\nu(v_i) \text{ for all } \nu \in \nu_{s,\sigma}.\end{aligned}$$

If $t = v_i \in \mathit{IndVar}$ and i is not the last index in σ,

$$\begin{aligned}[m\left(\delta_\sigma(v_i)\right)](s) &= \left[m\left(\rho^{Ord(i,\sigma)-1};\pi\right)\right](s)\\ &= s_{Ord(i,\sigma)}\\ &= V_\nu(v_i) \text{ for all } \nu \in \nu_{s,\sigma}.\end{aligned}$$

If $t = c \in C$, then

$$\begin{aligned}[m\left(\delta_\sigma(c)\right)](s) &= \left[c^{\mathcal{A}}\right](s)\\ &= c^{\mathcal{B}}\\ &= V_\nu(c) \text{ for all } \nu \in \nu_{s,\sigma}.\end{aligned}$$

If $t = f(t_1, \dots t_k)$ with $f \in F$, then

$$\begin{aligned}[m\left(\delta_\sigma(f(t_1,\dots t_k))\right)](s) &= \left[\left(m\left(\delta_\sigma(t_1)\right) \nabla \cdots \nabla m\left(\delta_\sigma(t_k)\right)\right);f^{\mathcal{A}}\right](s)\\ &= \left[f^{\mathcal{A}}\right]\left([m\left(\delta_\sigma(t_1)\right)](s) * \cdots * [m\left(\delta_\sigma(t_k)\right)](s)\right)\\ &= \left[f^{\mathcal{A}}\right]\left(V_\nu(t_1) * \cdots * V_\nu(t_k)\right) \text{ for all } \nu \in \nu_{s,\sigma}\\ &= f^{\mathcal{B}}\left(V_\nu(t_1),\dots,V_\nu(t_k)\right) \text{ for all } \nu \in \nu_{s,\sigma}\\ &= V_\nu\left(f\left(t_1,\dots,t_k\right)\right) \text{ for all } \nu \in \nu_{s,\sigma}.\end{aligned}$$

Let us now prove the main result by induction on the structure of formula α, i.e., we will prove that

$$s \in \mathsf{dom}\left(m\left(T'_\sigma(\alpha)\right)\right) \qquad \Longleftrightarrow \qquad \mathcal{B}, \nu \models_{\mathsf{ETFR}} \alpha \text{ for all } \nu \in \nu_{s,\sigma}\ .$$

If $\alpha = t_1 \star \cdots \star t_m R t'_1 \star \cdots \star t'_n$, then

$$\begin{aligned}& s \in \mathsf{dom}\left(m\left(T'_\sigma(t_1 \star \cdots \star t_m R t'_1 \star \cdots \star t'_n)\right)\right)\\ &\Longleftrightarrow s \in \mathsf{dom}\Big(m\Big(\big(\left(\delta_\sigma(t_1) \nabla \cdots \nabla \delta_\sigma(t_m)\right)\\ &\qquad\qquad \nabla \ \left(\delta_\sigma(t'_1) \nabla \cdots \nabla \delta_\sigma(t'_n)\right);\breve{R}\big);\breve{2};1\Big)\Big)\\ &\Longleftrightarrow \langle [m\left(\delta_\sigma(t_1)\right)](s) * \cdots * [m\left(\delta_\sigma(t_m)\right)](s),\\ &\qquad\qquad [m\left(\delta_\sigma(t'_1)\right)](s) * \cdots * [m\left(\delta_\sigma(t'_n)\right)](s)\rangle \in m(R)\end{aligned}$$

$$\begin{aligned}
&\iff \langle [m\,(\delta_\sigma(t_1))]\,(s) * \cdots * [m\,(\delta_\sigma(t_m))]\,(s),\\
&\qquad\qquad [m\,(\delta_\sigma(t'_1))]\,(s) * \cdots * [m\,(\delta_\sigma(t'_n))]\,(s)\rangle \in m'\,(R)\\
&\iff \langle V_\nu(t_1) * \cdots * V_\nu(t_m),\\
&\qquad\qquad V_\nu(t'_1) * \cdots * V_\nu(t'_n)\rangle \in m'\,(R) \text{ for all } \nu \in \nu_{s,\sigma}\\
&\iff \mathcal{B}, \nu \models_{\mathsf{ETFR}} t_1 \star \cdots \star t_m R t'_1 \star \cdots \star t'_n \text{ for all } \nu \in \nu_{s,\sigma}.
\end{aligned}$$

If $\alpha = \neg\beta$, then

$$\begin{aligned}
s \in \mathsf{dom}\,(m\,(T'_\sigma(\neg\beta))) &\iff s \in \mathsf{dom}\left(\overline{m\,(T'_\sigma(\beta))}\right)\\
&\iff s \notin \mathsf{dom}\,(m\,(T'_\sigma(\beta))) \qquad \text{(by } m\,(T'_\sigma(\beta)) \text{ right-deal)}\\
&\iff \mathcal{B}, \nu \not\models_{\mathsf{ETFR}} \beta \text{ for some } \nu \in \nu_{s,\sigma}\\
&\iff \mathcal{B}, \nu \not\models_{\mathsf{ETFR}} \beta \text{ for all } \nu \in \nu_{s,\sigma}\\
&\iff \mathcal{B}, \nu \models_{\mathsf{ETFR}} \neg\beta \text{ for all } \nu \in \nu_{s,\sigma}.
\end{aligned}$$

If $\alpha = \beta \vee \gamma$, then

$$\begin{aligned}
&s \in \mathsf{dom}\,(m\,(T'_\sigma(\beta \vee \gamma)))\\
&\quad\iff s \in \mathsf{dom}\,(m\,(T'_\sigma(\beta))) \text{ or } s \in \mathsf{dom}\,(m\,(T'_\sigma(\gamma)))\\
&\quad\iff \mathcal{B}, \nu \models_{\mathsf{ETFR}} \beta \text{ for all } \nu \in \nu_{s,\sigma} \text{ or } \mathcal{B}, \nu \models_{\mathsf{ETFR}} \gamma \text{ for all } \nu \in \nu_{s,\sigma}\\
&\quad\iff \mathcal{B}, \nu \models_{\mathsf{ETFR}} \beta \vee \gamma \text{ for all } \nu \in \nu_{s,\sigma}.
\end{aligned}$$

If $\alpha = \exists v_i\beta$, given $s = a_1 * \cdots * a_k$ we will denote by $s_{i,a}$ the element $a_1 * \cdots * a_{i-1} * a * a_i * \cdots * a_k$. Then,

$$\begin{aligned}
&s \in \mathsf{dom}\,(m\,(T'_\sigma(\exists v_i\beta)))\\
&\quad\iff s \in \mathsf{dom}\left(m\,(\Delta_{\sigma,i})\,;m\left(T'_{[\sigma\oplus i]}(\beta)\right)\right)\\
&\quad\iff \text{there exists } a \in \mathit{Urel}_{\mathfrak{A}} \text{ s.t. } \langle s, s_{i,a}\rangle \in m\,(\Delta_{\sigma,i}) \text{ and}\\
&\qquad\quad s_{i,a} \in \mathsf{dom}\left(m\left(T'_{[\sigma\oplus i]}(\beta)\right)\right)\\
&\quad\iff \text{there exists } a \in \mathit{Urel}_{\mathfrak{A}} \text{ s.t. } \langle s, s_{i,a}\rangle \in m\,(\Delta_{\sigma,i}) \text{ and}\\
&\qquad\quad \mathcal{B}, \nu \models_{\mathsf{ETFR}} \beta \text{ for all } \nu \in \nu_{s_{i,a},[\sigma\oplus i]}\\
&\quad\iff \text{there exists } a \in \mathit{Urel}_{\mathfrak{A}} \text{ s.t. } \mathcal{B}, \nu[v_i/a] \models_{\mathsf{ETFR}} \beta \text{ for all } \nu \in \nu_{s,\sigma}\\
&\quad\iff \mathcal{B}, \nu \models_{\mathsf{ETFR}} \exists v_i\beta \text{ for all } \nu \in \nu_{s,\sigma}.
\end{aligned}$$

□

Lemma 5.7 *Let $\alpha \in ForETFR(C, F, P)$ not containing any atomic subformula of form $R = S$. Let $\sigma = \sigma_\alpha$ of length k. Let $\mathcal{A}$ be the square*

$\mathsf{CFR}^+(K)$ *model* $\langle \langle \mathfrak{A}, C^{\mathcal{A}} \cup F^{\mathcal{A}} \cup P^{\mathcal{A}} \rangle, m \rangle$. *Then, there exists a model* $\mathcal{B}$ *for* $\mathsf{ETFR}(C, F, P)$ *such that*

$$m\,(T'_\sigma(\alpha)) = 1\text{'}^k_{\mathsf{U}} \qquad \Longleftrightarrow \qquad \mathcal{B} \models_{\mathsf{ETFR}} \alpha \ .$$

Proof Let $\mathcal{B}$ be constructed as in Lemma 5.6.

$$\begin{aligned}
& m\,(T'_\sigma(\alpha)) = 1\text{'}^k_{\mathsf{U}} \\
& \quad \Longleftrightarrow\ s \in \mathsf{dom}\,(m\,(T'_\sigma(\alpha))) \text{ for all } s \in \mathsf{dom}\,(1\text{'}^k_{\mathsf{U}}) \\
& \quad \Longleftrightarrow\ \mathcal{B}, \nu \models \alpha \text{ for all } \nu \in \nu_{s,\sigma} \text{ and } s \in \mathsf{dom}\,(1\text{'}^k_{\mathsf{U}}) \qquad \text{(by Lemma 5.6)} \\
& \quad \Longleftrightarrow\ \mathcal{B}, \nu \models \alpha \text{ for all } \nu \\
& \quad \Longleftrightarrow\ \mathcal{B} \models_{\mathsf{ETFR}} \alpha.
\end{aligned}$$

□

Lemma 5.8 *Let* $\alpha \in ForETFR(C, F, P)$, *let* $\sigma = \sigma_\alpha$ *of length* k, *and let* $\mathcal{A}$ *be a square* $\mathsf{CFR}^+(K)$ *model. Then, there exists a* $\mathsf{ETFR}(C, F, P)$ *model* $\mathcal{B}$ *such that*

$$\mathcal{A} \models_{\mathsf{CFR+}} T_\sigma(\alpha) \qquad \Longleftrightarrow \qquad \mathcal{B} \models_{\mathsf{ETFR}} \alpha \ .$$

Proof Let us assume that $\mathcal{A} = \langle \langle \mathfrak{A}, C^{\mathcal{A}} \cup F^{\mathcal{A}} \cup P^{\mathcal{A}} \rangle, m \rangle$. Let $\mathcal{B}$ be defined as in Lemma 5.6.

Notice that since $m = m'$, for all $R \in RelDesP$ we have $m(R) = m'(R)$.

If α is $R = S$ with $R, S \in RelDes(P)$, then

$$\begin{aligned}
\mathcal{A} \models_{\mathsf{CFR+}} T_\sigma(R = S) &\iff \mathcal{A} \models_{\mathsf{CFR+}} R = S \\
&\iff m(R) = m(S) \\
&\iff m'(R) = m'(S) \\
&\iff \mathcal{B} \models_{\mathsf{ETFR}} R = S.
\end{aligned}$$

If α is $t_1 \star \cdots \star t_m R t'_1 \star \cdots \star t'_n$, then

$$\begin{aligned}
& \mathcal{A} \models_{\mathsf{CFR+}} T_\sigma(t_1 \star \cdots \star t_m R t'_1 \star \cdots \star t'_n) \\
& \quad \Longleftrightarrow\ \mathcal{A} \models_{\mathsf{CFR+}} T'_\sigma(t_1 \star \cdots \star t_m R t'_1 \star \cdots \star t'_n) = 1\text{'}^k_{\mathsf{U}};1 \\
& \quad \Longleftrightarrow\ \mathcal{B} \models_{\mathsf{ETFR}} t_1 \star \cdots \star t_m R t'_1 \star \cdots \star t'_n. \qquad \text{(by Lemma 5.7)}
\end{aligned}$$

If α is $\neg\beta$, then

$$\begin{aligned} \mathcal{A} \models_{\mathsf{CFR}^+} T_\sigma(\neg\beta) &\iff \mathcal{A} \models_{\mathsf{CFR}^+} \neg T_\sigma(\beta) \\ &\iff \mathcal{A} \not\models_{\mathsf{CFR}^+} T_\sigma(\beta) \\ &\iff \mathcal{B} \not\models_{\mathsf{ETFR}} \beta \\ &\iff \mathcal{B} \models_{\mathsf{ETFR}} \neg\beta. \end{aligned}$$

If α is $\beta \vee \gamma$, then

$$\begin{aligned} \mathcal{A} \models_{\mathsf{CFR}^+} T_\sigma(\beta \vee \gamma) &\iff \mathcal{A} \models_{\mathsf{CFR}^+} T_\sigma(\beta) \vee T_\sigma(\gamma) \\ &\iff \mathcal{A} \models_{\mathsf{CFR}^+} T_\sigma(\beta) \text{ or } \mathcal{A} \models_{\mathsf{CFR}^+} T_\sigma(\gamma) \\ &\iff \mathcal{B} \models_{\mathsf{ETFR}} \beta \text{ or } \mathcal{B} \models_{\mathsf{ETFR}} \gamma \\ &\iff \mathcal{B} \models \beta \vee \gamma. \end{aligned}$$

If α is $\exists v_i \beta$, then

$$\begin{aligned} \mathcal{A} \models_{\mathsf{CFR}^+} T_\sigma(\exists v_i \beta) &\iff \mathcal{A} \models_{\mathsf{CFR}^+} T'_\sigma(\exists v_i \beta) = 1\text{'}^k_{\mathsf{U}};1 \\ &\iff \mathcal{B} \models_{\mathsf{ETFR}} \exists v_i \beta. \qquad \text{(by Lemma 5.7)} \end{aligned}$$

□

Theorem 5.2 *Let* $\alpha \in \mathsf{ETFR}(C, F, P)$ *and let* $\sigma = \sigma_\alpha$. *Then,*

$$\models_{\mathsf{ETFR}} \alpha \qquad \iff \qquad \models_{\mathsf{CFR}^+} T_\sigma(\alpha)\ .$$

Proof
$\Rightarrow$) If $\not\models_{\mathsf{CFR}^+} T_\sigma(\alpha)$, then there exists a model $\mathcal{A} = \langle \langle \mathfrak{A}, C^{\mathcal{A}}, F^{\mathcal{A}}, P^{\mathcal{A}} \rangle, m \rangle$ with $\mathfrak{A} \in \mathsf{SAFAU}$ such that $\mathcal{A} \not\models_{\mathsf{CFR}^+} T_\sigma(\alpha)$. From $\mathcal{A}$ we build a square model $\mathcal{A}'$ up as follows. Let $\mathfrak{A}' \in \mathsf{SPFAU}$ such that $\mathfrak{A} \cong \mathfrak{A}'$ and $h : A \to A'$ a fork algebra isomorphism ($\mathfrak{A}'$ and h exist by Thm. 4.3). For each $c \in C$, let $c^{\mathcal{A}'} = h(c^{\mathcal{A}})$. For each $f \in F$, let $f^{\mathcal{A}'} = h(f^{\mathcal{A}})$. For each $p \in P$, let $p^{\mathcal{A}'} = h(p^{\mathcal{A}})$. Finally, for each $R \in \mathit{RelVar}$ we define $m'(R) = h(m(R))$. It is clear that $\mathcal{A}' \not\models_{\mathsf{CFR}^+} T_\sigma(\alpha)$. Then, by Lemma 5.8 there exists a $\mathsf{ETFR}(C, F, P)$ model $\mathcal{B}$ such that $\mathcal{B} \not\models_{\mathsf{ETFR}} \alpha$. Then, $\not\models_{\mathsf{ETFR}} \alpha$.
$\Leftarrow$) If $\not\models_{\mathsf{ETFR}} \alpha$ then there exists a $\mathsf{ETFR}(C, F, P)$ model $\mathcal{A}$ such that $\mathcal{A} \not\models_{\mathsf{ETFR}} \alpha$. Then, by Lemma 5.5 there exists a $\mathsf{CFR}^+(C \cup F \cup P)$ model $\mathcal{B}$ such that $\mathcal{B} \not\models_{\mathsf{CFR}^+} T_\sigma(\alpha)$. Then, $\not\models_{\mathsf{CFR}^+} T_\sigma(\alpha)$. □

Theorem 5.2 shows that reasoning in ETFR can be replaced by reasoning in CFR^+. The reason why this is considered an algebraization of ETFR is because validity in CFR^+ reduces to the verification of the validity of a formula in SAFAU.

Theorem 5.3 *Let $\alpha \in \mathsf{ETFR}(C,F,P)$ and let $\sigma = \sigma_\alpha$ of length k. Then,*

$$\models_{\mathsf{ETFR}} \alpha \qquad \Longleftrightarrow \qquad \models_{\mathsf{CFR}^+} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}} \ .$$

Proof
$\Rightarrow$) If $\not\models_{\mathsf{CFR}^+} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}}$ then there exists a $\mathsf{CFR}^+(C \cup F \cup P)$ model $\mathcal{A} = \langle \langle \mathfrak{A}, C^{\mathcal{A}}, F^{\mathcal{A}}, P^{\mathcal{A}} \rangle, m \rangle$ with $\mathfrak{A} \in$ SAFAU such that $\mathcal{A} \not\models_{\mathsf{CFR}^+} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}}$. From $\mathcal{A}$ we build a proper model $\mathcal{A}'$ up as follows. Let $\mathfrak{A}' \in$ SPFAU such that $\mathfrak{A} \cong \mathfrak{A}'$ and $h : A \to A'$ a fork algebra isomorphism ($\mathfrak{A}'$ and h exist by Thm. 4.3). For each $c \in C$, let $c^{\mathcal{A}'} = h(c^{\mathcal{A}})$. For each $f \in F$, let $f^{\mathcal{A}'} = h(f^{\mathcal{A}})$. For each $p \in P$, let $p^{\mathcal{A}'} = h(p^{\mathcal{A}})$. Finally, for each $R \in \mathit{RelVar}$ we define $m'(R) = h(m(R))$. It is clear that $\mathcal{A}' \not\models_{\mathsf{CFR}^+} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}}$. Then, by Lemma 5.7 there exists a $\mathsf{ETFR}(C,F,P)$ model $\mathcal{B}$ such that $\mathcal{B} \not\models_{\mathsf{ETFR}} \alpha$. Then, $\not\models_{\mathsf{ETFR}} \alpha$.
$\Leftarrow$) If $\not\models_{\mathsf{ETFR}} \alpha$ then there exists a $\mathsf{ETFR}(C,F,P)$ model $\mathcal{A}$ such that $\mathcal{A} \not\models_{\mathsf{ETFR}} \alpha$. Then, by Lemma 5.4 there exists a $\mathsf{CFR}^+(C \cup F \cup P)$ model $\mathcal{B}$ such that $\mathcal{B} \not\models_{\mathsf{CFR}^+} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}}$. Then $\not\models_{\mathsf{CFR}^+} T'_\sigma(\alpha) = 1\text{'}^k_{\mathsf{U}}$. □

In order to obtain an algebraization of *FOLE* it suffices to compose the mappings T_∇ and T'_σ. The result of this composition is the mapping $T_{\nabla,\sigma} : \mathit{ForFOLE}(C,F,P) \to \mathit{ForCFR}(C \cup F \cup P)$. Recall that in order to apply the mapping T_∇ it is necessary to divide arguments of predicate symbols between input and output arguments. For the next theorem, and in order to simplify the notation, we will assume that all arguments are to be considered as input arguments.

(1) $T_{\nabla,\sigma}(p(t_1,\ldots,t_k)) = (\delta_\sigma(t_1)\nabla \cdots \nabla\delta_\sigma(t_k))\,;p'\,;1$,
(2) $T_{\nabla,\sigma}(\neg\alpha) = \overline{T_{\nabla,\sigma}(\alpha)}$,
(3) $T_{\nabla,\sigma}(\alpha \vee \beta) = T_{\nabla,\sigma}(\alpha) + T_{\nabla,\sigma}(\beta)$,
(4) $T_{\nabla,\sigma}(\exists v_n \alpha) = \Delta_{\sigma,n}\,;T_{\nabla,[\sigma \oplus n]}(\alpha)$.

By using the translation $T_{\nabla,\sigma}$ we can prove the following theorem.

Theorem 5.4 *Let $\alpha \in \mathit{ForFOLE}(C,F,P)$ and let $\sigma = \sigma_\alpha$. Then, if σ has length k,*

$$\models_{\mathit{FOLE}} \alpha \qquad \Longleftrightarrow \qquad \models_{\mathsf{CFR}^+} T_{\nabla,\sigma}(\alpha) = 1\text{'}^k_{\mathsf{U}}\,;1 \ .$$

Proof By Thm. 5.1,

$$\models_{FOLE} \alpha \qquad \Longleftrightarrow \qquad \models_{\mathsf{ETFR}} T_{\nabla}(\alpha) \ . \tag{5.4}$$

By Thm. 5.3,

$$\models_{\mathsf{ETFR}} T_{\nabla}(\alpha) \qquad \Longleftrightarrow \qquad \models_{\mathsf{CFR}^+} T'_{\sigma}(T_{\nabla}(\alpha)) = 1\text{'}^{k}_{\mathsf{U}};1 \ . \tag{5.5}$$

Joining (5.4) and (5.5) and the fact $T_{\nabla,\sigma}(\alpha) = T'_{\sigma}(T_{\nabla}(\alpha))$,

$$\models_{FOLE} \alpha \qquad \Longleftrightarrow \qquad \models_{\mathsf{CFR}^+} T_{\nabla,\sigma}(\alpha) = 1\text{'}^{k}_{\mathsf{U}};1 \ .$$

□

From $\mathsf{CFR}^+(P)$ we define the formalism $\mathsf{CFREQ}^+(P)$ as a restriction of $\mathsf{CFR}^+(P)$. $\mathsf{CFREQ}^+(P)$ is defined as follows.

Formulas: Equations from $ForCFR^+$.
Inference Rules: The following inference rules for equational logic:

(1) $\vdash_{\mathsf{CFREQ}^+} p = p$, for every $p \in RelDes(P)$,
(2) $p = q \vdash_{\mathsf{CFREQ}^+} q = p$, for every $p, q \in RelDes(P)$,
(3) $p = q, q = r \vdash_{\mathsf{CFREQ}^+} p = r$, for every $p, q, r \in RelDes(P)$,
(4) If $\vdash_{\mathsf{CFREQ}^+} p = q$, $r \in RelDes(P)$ contains the subterm p, and s is obtained from r by *replacement* of p by the term q, then $\vdash_{\mathsf{CFREQ}^+} r = s$,
(5) If $\vdash_{\mathsf{CFREQ}^+} p = q$, x is a variable (possibly occurring in p or q), and $r \in RelDesP$, then $\vdash_{\mathsf{CFREQ}^+} p[r/x] = q[r/x]$. That is, *substitution* of variables by terms is a valid inference rule.

Axioms: Set of equations characterizing the class of abstract fork algebras with a nonempty set of urelements.

Notice that in the axiomatization of CFREQ^+ we dropped the axiom requiring models to be simple.

Theorem 5.5 *Let e be an equation from* CFREQ^+. *Then,*

$$\models_{\mathsf{CFR}^+} e \qquad \Longleftrightarrow \qquad \vdash_{\mathsf{CFREQ}^+} e \ .$$

Proof By definition of the formalism CFR^+,

$$\models_{\mathsf{CFR}^+} e \text{ iff for all } \mathfrak{A} \in \mathsf{SAFAU},\ \mathfrak{A} \models e \ . \tag{5.6}$$

Since the variety generated by SAFAU is AFAU, SAFAU and AFAU share the same equational theory. Thus,

$$\text{for all } \mathfrak{A} \in \mathsf{SAFAU},\ \mathfrak{A} \models e \text{ iff for all } \mathfrak{A} \in \mathsf{AFAU},\ \mathfrak{A} \models e\ . \tag{5.7}$$

Then, by (5.6) and (5.7),

$$\models_{\mathsf{CFR}^+} e \text{ iff for all } \mathfrak{A} \in \mathsf{AFAU},\ \mathfrak{A} \models e\ . \tag{5.8}$$

By definition of CFREQ^+ and (5.8),

$$\models_{\mathsf{CFR}^+} e \quad \Longleftrightarrow \quad \models_{\mathsf{CFREQ}^+} e\ . \tag{5.9}$$

Since equational logic is complete [G. Birkhoff (1944)],

$$\models_{\mathsf{CFREQ}^+} e \quad \Longleftrightarrow \quad \vdash_{\mathsf{CFREQ}^+} e\ . \tag{5.10}$$

Finally, joining (5.9) and (5.10),

$$\models_{\mathsf{CFR}^+} e \quad \Longleftrightarrow \quad \vdash_{\mathsf{CFREQ}^+} e\ .$$

□

Theorem 5.6 *Let $\alpha \in ForFOLE(C, F, P)$ and let $\sigma = \sigma_\alpha$ of length k. Then,*

$$\models_{FOLE} \alpha \quad \Longleftrightarrow \quad \vdash_{\mathsf{CFREQ}^+} T_{\nabla,\sigma}(\alpha) = 1\text{'}^k_{\mathsf{U}};1\ .$$

Proof By Thm. 5.4,

$$\models_{FOLE} \alpha \quad \Longleftrightarrow \quad \models_{\mathsf{CFR}^+} T_{\nabla,\sigma}(\alpha) = 1\text{'}^k_{\mathsf{U}};1\ . \tag{5.11}$$

By Thm. 5.5,

$$\models_{\mathsf{CFR}^+} T_{\nabla,\sigma}(\alpha) = 1\text{'}^k_{\mathsf{U}};1 \quad \Longleftrightarrow \quad \vdash_{\mathsf{CFREQ}^+} T_{\nabla,\sigma}(\alpha) = 1\text{'}^k_{\mathsf{U}};1\ . \tag{5.12}$$

Thus, by (5.11) and (5.12),

$$\models_{FOLE} \alpha \quad \Longleftrightarrow \quad \vdash_{\mathsf{CFREQ}^+} T_{\nabla,\sigma}(\alpha) = 1\text{'}^k_{\mathsf{U}};1\ .$$

□

If in the preceding theorem α is a sentence, then $Length(\sigma_\alpha) = 0$. Then the following corollary holds.

Corollary 5.1 *If $\alpha \in ForFOLE(C, F, P)$ is a sentence, then*

$$\models_{FOLE} \alpha \quad \Longleftrightarrow \quad \vdash_{\mathsf{CFREQ}^+} T_{\nabla,\langle\rangle}(\alpha) = 1.$$

The result shown in Cor. 5.1 was already known for other algebraic systems closely related to fork algebras, as quasi-projective relation algebras and pairing relation algebras. The work on the interpretability of first-order theories in quasi-projective relation algebras was extensively developed by Tarski and Givant in [A. Tarski et al. (1987)], while the version for pairing relation algebras was developed by Maddux in [R. Maddux (1989)]. In [L. Henkin et al. (1985)], *FOLE* is algebraized using cylindric algebras. Cylindric algebras are a very natural algebraic counterpart of *FOLE*. The fact that they have a Boolean algebra reduct allows the propositional part of *FOLE* to be algebraized. Also, for each quantifier $\exists v_i$, a new operator c_i (called the i-th cylindrification) is defined. The axioms for the cylindrifications are natural translations of valid properties for the existential quantifiers. For example, the property $\exists v_i \exists v_j \alpha \Leftrightarrow \exists v_j \exists v_i \alpha$ corresponds to the cylindric algebra axiom $c_i c_j x = c_j c_i x$, for all pairs of indices $\langle i, j \rangle$. Finally, for each pair of indices $\langle i, j \rangle$, a constant element d_{ij} (called the i,j-diagonal element) is distinguished. Intuitively, d_{ij} characterizes algebraically the predicate $v_i = v_j$. Notice that an infinite number of axioms are required for axiomatizing the infinitely many cylindrifications and diagonal elements. The fact fork algebras have only finitely many operators and are axiomatized by a finite set of equations makes fork algebras more attractive in computer science, where this finiteness plays an essential part in the implementability of a calculus for program construction based on fork algebras.

Chapter 6

Algebraization of Non-Classical Logics

The results in this chapter were obtained jointly by Frias and Orlowska [M. Frias et al. (1997)c; M. Frias et al. (1997)d]. Equational reasoning based on substitution of equals by equals is the kind of manipulation that is performed in many information processing systems. The role of equational logics in the development of formal methods for computer science applications is increasingly recognized and various tools have been developed for modeling user's systems and carrying through designs within the equational framework [D. Gries (1995); D. Gries et al. (1993)].

The idea of relational formalization of logical systems was originated by Ewa Orlowska in [E. Orlowska (1988)] and further developed in [E. Orlowska (1992); E. Orlowska (1994); E. Orlowska (1995)].

Examples of relational formalisms for applied logics can also be found in Buszkowski and Orlowska [W. Buszkowski et al. (1996)], Demri and Orlowska [S. Demri et al. (1996); S. Demri et al. (1994)], Herment and Orlowska [M. Herment et al. (1995)] and elsewhere. The paradigm of relational formalization of logical systems is based on the principle of replacing any logic with a theory of a suitable class of algebras of relations [W. MacCaull (1997); E. Orlowska (1988); E. Orlowska (1992); E. Orlowska (1994); E. Orlowska (1995)]. In order to define such a theory for a given logic, the language of the logic is to be translated into a sufficiently expressive language of relational terms in a validity preserving manner, i.e., a logical formula α is valid if its translation $T(\alpha)$ is a term such that $T(\alpha) = 1$ holds in every relation algebra from the underlying class of algebras. However, since the class of representable relation algebras is not finitely axiomatizable [D. Monk (1964)], and the finitely axiomatizable class of relation algebras is not representable [R. Lyndon (1950)], the existing relational frameworks for non-classical logics suffer several disadvantages. The class of fork algebras is both finitely axiomatizable and representable, and hence a fork algebra formalism seems to be an appropriate candidate for relational formalization of non-classical logics.

The standard semantics of non-classical logics are usually defined in terms of frames [S. Kripke (1963); S. Kripke (1965)], that is, relational systems consisting of a set W of states and a family of accessibility relations in W. In any particular logical system the accessibility relations are assumed to satisfy some constraints. The meaning of a propositional formula is defined, first, by means of an assignment m of subsets of W to propositional variables, and second, by extending m to all the formulas of the language under consideration. In this way every formula α is interpreted as a subset of states, with the intuition that $m(\alpha)$ consists of those states in which α is true. The meaning $m(\alpha)$ of formulas built with the classical propositional connectives of negation, disjunction and conjunction is defined from the meanings of the subformulas of α by the well known interpretation of these connectives in terms of Boolean operations of complement, join, and meet, respectively. The meaning $m(\alpha)$ of formulas built with intentional operators, such as modal operators of possibility and necessity, is usually defined in terms of both the values of m for the subformulas of α and an accessibility relation, which is most often a binary or ternary relation in W.

The interpretability of a non-classical logic in the calculus CFREQ^+ is established by means of a deduction preserving translation of formulas of the logic into formulas of CFREQ^+. Under this translation, formulas, formerly understood as sets of states, and accessibility relations, receive a uniform representation as relations. The propositional connectives are transformed into relational operations. The constraints on accessibility relations are translated into relational equations. The major advantage of relational formalization is that it provides a uniform framework for representation of a broad class of applied logics and enables us to apply an equational proof theory to these logics.

In the first part of this chapter we will develop an equational formalism based on fork algebras that is capable of modeling a great variety of applied non-classical logics and of simulating non-classical means of reasoning. In the second part of the chapter we will define a Rasiowa-Sikorski style deduction system [H. Rasiowa et al. (1963)] for fork logic. We will then define a validity preserving translation from the language of intuitionistic logic and minimal intuitionistic logic into the language of fork logic. Next, we discuss three methods of intuitionistic reasoning within the framework of fork logic. The first method consists of extending the Rasiowa-Sikorski proof system of fork logic with some specific rules that reflect properties of the accessibility relation from Kripke models of logics based on intuitionism. The second method is based on a kind of relational deduction theorem that enables us to express derivability in fork logic of a term (representing a formula of a logic) from a finite number of terms (representing conditions on the accessibility relation). In this case the plain proof system for fork logic is an adequate deduction tool. The third method employs the equational theory of fork algebras. We extend this equational theory with equations that represent the required properties of the accessibility relation and treat them as specific axioms, as we will do with modal logics in the first part of the chapter.

The chapter is organized as follows. Section 6.2 presents the fork logic *FL*, a simplified version of CFREQ^+. In Sections 6.3–6.5 a wide variety of modal logics are algebraized. In Section 6.6 we discuss the fork algebraic formalization of modal logics determined by a Hilbert-style axiom system. In Section 6.7, propositional dynamic logic is algebraized. In Section 6.8 the logic *FL′* (a restriction of ETFR) is defined, and in Section 6.9 a Rasiowa-Sikorski style calculus for *FL′* is presented. In Sections 6.10–6.12 the calculus presented in Section 6.8 is used for algebraizing intuitionistic logic, minimal intuitionistic logic and a wide class of intermediate logics.

6.1 Basic Definitions and Properties

Theorem 6.1 *Let $\mathcal{V}$ be the variety generated by the class of* $At\mathsf{SFAU}$. *Then,* $\mathcal{V} = \mathsf{AFAU}$.

Proof Clearly, $\mathcal{V} \subseteq \mathsf{AFAU}$. Let us prove the other inclusion. Given an algebra $\mathfrak{A} \in \mathsf{AFAU}$, by the representation theorem (Thm. 4.3) $\mathfrak{A}$ is isomorphic to an algebra $\mathfrak{B} \in \mathsf{PFAU}$. By definition of PFAU and Thm. 3.1, $\mathfrak{B} \in \mathbf{I}\,\mathbf{S}\,\mathbf{P}\,\mathsf{FullPFAU}$. Since full proper fork algebras with urelements are atomic, simple and have urelements, and varieties are closed under isomorphisms, subalgebras and products, $\mathfrak{B} \in \mathcal{V}$. Finally, since $\mathfrak{A}$ is isomorphic to $\mathfrak{B}$, also $\mathfrak{A} \in \mathcal{V}$, which implies that AFAU $\subseteq \mathcal{V}$. □

Theorem 6.2 *Let $\mathcal{V}$ be the variety generated by* SPFAU. *Then $\mathcal{V} =$* AFAU.

Proof Similar to the proof of Thm. 6.1. □

Theorem 6.3 *The equational theories of At*SFAU *and* SPFAU *coincide with the equational theory of* AFAU.

Proof Is a direct consequence of Thms. 6.1 and 6.2. □

Definition 6.1 We say that an equation e is provable from a set of equations E in CFREQ$^+$ relativized to an equational theory Δ (denoted by $E \vdash_{\mathsf{CFREQ}^+,\Delta} e$, if $E \cup \Delta \vdash_{\mathsf{CFREQ}^+} e$.

The following definition gives an abstract characterization of urelements.

Definition 6.2 Given $\mathfrak{A} \in At\mathsf{FAU}$, $x \in A$ is called an *abstract urelement* if x is an atom and $x \leq 1'_{\mathsf{U}}$.

In the remaining part of this chapter we will deal with sentences from *FOLE*, therefore, as Cor. 5.1 shows, only very restricted equations are needed (i.e., just those equations in which the term in the right-hand side is 1). Thus, we will use a simplified version of CFREQ$^+$ that will be called *fork logic FL.*

6.2 The Fork Logic *FL*

In this section we introduce what we call *Fork Logic FL.* We also present a completeness theorem and a theorem on the interpretability of classical first-order logic in *FL.* This interpretability theorem will prove to be useful in Section 6.4 for the description of model constraints.

Definition 6.3 We define the alphabet of fork logic as the union of the sets described by the following conditions:

(1) A countable set *RelVar* of *relational variables.*
(2) The set of *logical symbols*: $+, \cdot, ;, \nabla, {}^{-}, {}^{\smile}, 1', 0, 1$.
(3) A countable set *RelConst* of *extralogical symbols* (i.e., relational constants whose meaning varies between models).

Definition 6.4 A finite sequence of symbols from the alphabet of fork logic is a fork formula iff it belongs to every set Ω satisfying:

(1) $\mathit{RelVar} \cup \mathit{RelConst} \cup \{\, 1', 0, 1 \,\} \subseteq \Omega$.
(2) If $R, S \in \Omega$ then $\left\{ \overline{R}, \breve{R}, R{+}S, R{\cdot}S, R;S, R\nabla S \right\} \subseteq \Omega$.

Definition 6.5 We say that a fork formula α is *provable* from a set of fork formulas Γ in an equational theory Δ (denoted by $\Gamma \vdash_{FL,\Delta} \alpha$) if

$$\{\, \gamma = 1 : \gamma \in \Gamma \,\} \vdash_{\mathsf{CFREQ^+},\Delta} \alpha = 1 \ .$$

Definition 6.6 A *fork model* is a structure $\langle \mathfrak{A}, m \rangle$ where $\mathfrak{A} \in \mathsf{AFAU}$, and m is the meaning function that assigns relations in A both to variables in *RelVar* and to the extralogical symbols in *RelConst*. It is clear how to extend m homomorphically to a function $m' : \Omega \to A$. For the sake of simplicity, we will use the name m for both mappings.

Definition 6.7 A fork model $\langle \mathfrak{A}, m \rangle$ is called *proper* if $\mathfrak{A} \in \mathsf{PFAU}$, and *simple* if $\mathfrak{A} \in \mathsf{SAFAU}$. Similarly, a fork model is called *atomic* if $\mathfrak{A} \in At\mathsf{FAU}$. The class of proper fork models will be denoted by PFM, and the class of simple fork models by SFM. The class of fork models simple and proper will be denoted by SPFM and the class of atomic fork models will be denoted by $At\mathsf{FM}$. The class of atomic and proper fork models will be denoted by $At\mathsf{PFM}$, and the class of atomic and simple proper fork models by $At\mathsf{SPFM}$.

Definition 6.8 A fork formula φ is said to be true in a fork model $\mathcal{F} = \langle \mathfrak{A}, m \rangle$ (denoted by $\mathcal{F} \models_{FL} \varphi$) if $m(\varphi) = 1$ holds in $\mathfrak{A}$. A fork formula φ is said to be true in a class of fork models $\mathcal{K}$, if for every member $\mathcal{F}$ of $\mathcal{K}$, $\mathcal{F} \models_{FL} \varphi$.

Definition 6.9 A fork formula φ is said to be valid in *FL* (denoted by $\models_{FL} \varphi$) if in every fork model $\mathcal{F}$ we have $\mathcal{F} \models_{FL} \varphi$.

Since PFA is a finitely based variety whose axioms are those for abstract fork algebras, the following theorem holds.

Theorem 6.4 *FL is strongly complete, i.e., given a fork formula R, a set Φ of fork formulas, and an equational theory Δ,*

$$\Phi \models_{FL,\Delta} R \text{ in } \mathsf{PFM} \qquad \Longleftrightarrow \qquad \Phi \vdash_{FL,\Delta} R \ .$$

Proof
$\Rightarrow$) If $\Phi \nvdash_{FL,\Delta} R$, then there exist $\mathfrak{A} \in \mathsf{AFAU}$ and a meaning function m such that

(1) for every equation $l = r \in \Delta$, $m(l) = m(r)$,

(2) for every $\varphi \in \Phi$, $m(\varphi) = 1$,
(3) $m(R) \neq 1$.

By Thm. 4.3, there exists $\mathfrak{B} \in$ PFAU and an isomorphism $h : \mathfrak{A} \to \mathfrak{B}$. Let m' be the meaning function defined by $m'(\alpha) = h(m(\alpha))$. Then,

(1) for every equation $l = r \in \Delta$, $m'(l) = m'(r)$,
(2) for every $\varphi \in \Phi$, $m'(\varphi) = 1$,
(3) $m'(R) \neq 1$.

Thus, it is not the case that $\Phi \models_{FL,\Delta} R$ in PFM.
$\Leftarrow$) The proof in this direction is simple and is left to the reader. □

As a consequence of Cor. 5.1, the following theorem on the interpretability of classical first-order theories in *FL* follows.

Theorem 6.5 *Any first-order theory is interpretable in FL, i.e., given a first-order theory Ψ and a sentence α, there exists a set of fork formulas F_Ψ (constructed from Ψ) and a fork formula t_α (constructed from α) such that*

$$\Psi \vdash \alpha \qquad \Longleftrightarrow \qquad F_\Psi \vdash_{FL} t_\alpha \; .$$

Proof Define F_Ψ as $\{ T_{\nabla,\sigma}(\psi) : \psi \in \Psi \}$ and t_X as $T_{\nabla,\sigma}(\alpha)$. Then apply Cor. 5.1. □

6.3 Modal Logics

In this section we present an introduction to modal logic. We begin by introducing the notion of frame, and then the notion of Kripke model. Using these structures we define satisfiability and validity in modal logics. For a thorough treatment of modal logic we direct the reader to [P. Blackburn et al. (2001)].

Definition 6.10 A *frame* is a structure $\langle W, R \rangle$ where W is a nonempty set of possible worlds and $R \subseteq W \times W$ is a binary accessibility relation between worlds.

Definition 6.11 Given a frame $\langle W, R \rangle$, a Kripke model is a structure $\mathfrak{M} = \langle W, R, m \rangle$ where m is a meaning function that assigns subsets of W to propositional variables.

Definition 6.12 The inductive definition of satisfiability describes the truth conditions depending on the complexity of formulas. For the atomic formulas (propositional variables) we have:

(at) $M, w \models p$ iff $w \in m(p)$, for any propositional variable p.

For formulas built with extensional operators such as classical negation, disjunction, conjunction or implication, their satisfiability at a possible world is completely determined by satisfiability of their subformulas at that world.

($\neg$) $M, w \models \neg\alpha$ iff not $M, w \models \alpha$,
($\vee$) $M, w \models \alpha \vee \beta$ iff $M, w \models \alpha$ or $M, w \models \beta$,
($\wedge$) $M, w \models \alpha \wedge \beta$ iff $M, w \models \alpha$ and $M, w \models \beta$,
($\rightarrow$) $M, w \models \alpha \rightarrow \beta$ iff $M, w \models \neg\alpha \vee \beta$.

For formulas defined from modal operators, such as $[R]$ (necessity) and $\langle R \rangle$ (possibility), we have

($[R]$) $M, w \models [R]\,\alpha$ iff for all $u \in W$, $\langle w, u \rangle \in R$ implies $M, u \models \alpha$,
($\langle R \rangle$) $M, w \models \langle R \rangle\,\alpha$ iff there is $u \in W$ s.t. $\langle w, u \rangle \in R$ and $M, u \models \alpha$.

For the sake of simplicity, we use the same symbol for the relational constant R that appears in modal operators and the accessibility relation that is denoted by this constant.

In various modal logics, the accessibility relation is assumed to satisfy certain conditions. If we call $FRM(C)$ the class of all those frames in which the accessibility relation satisfies a given set of conditions C, then the set of all the formulas valid in that class is called the logic $L(C)$. For example:

$$\begin{array}{lcl}
K & := & L(\emptyset), \\
T & := & L(\{\, \textit{reflexive} \,\}), \\
KB & := & L(\{\, \textit{symmetric} \,\}), \\
B & := & L(\{\, \textit{reflexive}, \textit{symmetric} \,\}), \\
K4 & := & L(\{\, \textit{transitive} \,\}), \\
KB4 & := & L(\{\, \textit{symmetric}, \textit{transitive} \,\}), \\
S4 & := & L(\{\, \textit{reflexive}, \textit{transitive} \,\}), \\
S5 & := & L(\{\, \textit{equivalence} \,\}), \textit{etc.}
\end{array}$$

Definition 6.13 Given a Kripke model $\mathfrak{M} = \langle W, R, m \rangle$ and a modal formula α, α is said to be *true* in $\mathfrak{M}$ if

$$\mathfrak{M}, w \models \alpha \qquad \text{for all } w \in W \ .$$

Definition 6.14 A modal formula α is called *valid* if it is true in all models.

6.4 Representation of Constraints in *FL*

In many nonclassical logics, accessibility relations in Kripke models must satisfy some properties or 'constraints'. We have already shown examples of constraints assumed in particular modal logics in the previous section. In this section we will show how these constraints can be captured in an abstract relational language by using fork logic.

When constraints are given as first-order formulas predicating about worlds, sometimes it is possible to capture these constraints by appealing to relation algebra concepts, without using fork logic. For example, in the case of the logic T, in which the accessibility relation must be reflexive, we can represent this fact in relation algebra with the condition

$$\overline{1'} + R = 1 \ . \tag{6.1}$$

If we look at (6.1) as a fork algebra equation, then a first advantage is given by the representation theorem (Thm. 4.3). It allows us to look at 1' as the diagonal relation, + as the union between sets, and 1 as the universal relation, property that is not shared in general by relation algebras. Another advantage of fork logic is its expressiveness, since fork logic allows us to express strictly more things than relation algebras. Recall that while relation algebra terms are adequate for interpreting just a three variable fragment of first-order logic (see [A. Tarski et al. (1987), pp. 76–87]), Thm. 6.5 guarantees that all of classical first-order logic can be interpreted in fork logic. An immediate consequence of this is that many developments which resort to algebras of binary relations can now be carried on just by resorting to the framework of abstract fork algebras.

In order to formalize the previous remarks, let Δ be a set of first-order constraints defining a set of Kripke models. We represent the set Δ in the logic FL by $\{T_{\nabla,\sigma}(\delta) : \delta \in \Delta\}$.

6.5 Interpretability of Modal Logics in *FL*

Throughout this section we will assume a fixed (but arbitrary) modal logic L. Since the notion of satisfiability has a finer granularity in modal than in classical logics because of the notion of satisfiability at a given world, we will define a similar notion for *FL*.

Definition 6.15 Given an atomic fork model $\langle \mathfrak{A}, m \rangle$, $x \in A$ is called a *relational world* if it is an atom satisfying $x \leq 1'_{\mathsf{U}}$.

Definition 6.16 Given an atomic fork model $\mathcal{F} = \langle \mathfrak{A}, m \rangle$, $\mathcal{F}$ satisfies the fork formula α in a relational world w (denoted by $\mathcal{F}, w \models_{FL} \alpha$), if in the fork algebra $\mathfrak{A}$ the inequality $w\,;m(\alpha) \neq 0$ is true.

In order to interpret L in *FL* we will proceed in the following way.

(1) We will define a mapping T_M from modal formulas to fork formulas.
(2) We will prove that given a Kripke model $\mathfrak{K} = \langle W, R, m \rangle$ and a formula α, there exists a fork model $\mathcal{F}$ (constructed from $\mathfrak{K}$) such that α is satisfied at a world w in $\mathfrak{K}$ iff $T_M(\alpha)$ is satisfied in the relational world $\{\langle w, w\rangle\}$ in $\mathcal{F}$.
(3) We will prove that given a fork model $\mathcal{F}$ for L and a modal formula α, there exists a Kripke model $\mathfrak{K}$ (constructed from $\mathcal{F}$) such that $T_M(\alpha)$ is satisfied at a given world w in $\mathcal{F}$ iff α is satisfied at w in $\mathfrak{K}$.

Definition 6.17 Before defining the mapping T_M, we define the mapping T'_M by:

(1) $T'_M(p_i) = P_i$, where p_i is a propositional variable and P_i is a relational variable,
(2) $T'_M(\neg\alpha) = 1'_{\mathsf{U}}\,;\overline{T'_M(\alpha)}$,
(3) $T'_M(\alpha \wedge \beta) = T'_M(\alpha) \cdot T'_M(\beta)$,
(4) $T'_M(\alpha \vee \beta) = T'_M(\alpha) + T'_M(\beta)$,
(5) $T'_M(\langle R\rangle\,\alpha) = R;T'_M(\alpha)$,
(6) $T'_M([R]\alpha) = T'_M(\neg\,\langle R\rangle\,\neg\alpha)$.

We finally define the mapping T_M by $T_M(\alpha) = T'_M(\alpha) + \overline{_{\mathsf{U}}1}$.

For the sake of simplicity, in Def. 6.17 (5) and (6), we assume that the constant R from the modal language is translated into a constant from the language of fork logic that is denoted by the same symbol.

Lemma 6.1 *Given a Kripke model $\mathfrak{K} = \langle W, R, m \rangle$ for L, a world $w \in W$, and a modal formula α, there exists $\mathcal{F} = \langle A, m' \rangle \in At\mathsf{PFM}$ constructed from $\mathfrak{K}$, and a relational world w' constructed from w such that*

$$\mathfrak{K}, w \models \alpha \quad \Longleftrightarrow \quad \mathcal{F}, w' \models_{FL} T'_M(\alpha) \ .$$

Proof Let $\mathfrak{A}$ be the full fork algebra with set of urelements W. $\mathfrak{A}$ is simple, proper and atomic. Let $m'(P_i)$ be the right-ideal relation with domain $m(p_i)$. Let w' be the relational world $\{ \langle w, w \rangle \}$. More generally, given $v \in W$, let $v' := \{ \langle v, v \rangle \}$. The remaining part of the proof proceeds by induction on the structure of the formula α.

$\alpha = p_i$:

$$\begin{array}{ll} & \mathfrak{K}, w \models \alpha \\ \Longleftrightarrow & \{\text{by def.} \models\} \\ & w \in m(p_i) \\ \Longleftrightarrow & \{\text{by def. } m'(P_i)\} \\ & w \in \mathsf{dom}\,(m'(P_i)) \\ \Longleftrightarrow & \{\text{by def. } w'\} \\ & w' ; m'(P_i) \neq 0 \\ \Longleftrightarrow & \{\text{by def.} \models_{FL}\} \\ & \mathcal{F}, w' \models_{FL} T'_M(\alpha). \end{array}$$

$\alpha = \neg\beta$:

$$\begin{array}{ll} & \mathfrak{K}, w \models \alpha \\ \Longleftrightarrow & \{\text{by def.} \models\} \\ & \mathfrak{K}, w \not\models \beta \\ \Longleftrightarrow & \{\text{by inductive hypothesis }\} \\ & \mathcal{F}, w' \not\models_{FL} T'_M(\beta) \\ \Longleftrightarrow & \{\text{by def.} \not\models_{FL}\} \\ & w' ; T'_M(\beta) = 0 \\ \Longleftrightarrow & \{\text{by properties of binary relations}\} \\ & w' ; 1\text{'}_{\mathsf{U}} ; \overline{T'_M(\beta)} \neq 0 \\ \Longleftrightarrow & \{\text{by def.} \models_{FL} \ \} \\ & \mathcal{F}, w' \models_{FL} T'_M(\alpha). \end{array}$$

$\alpha = \beta \vee \gamma$:

$$\begin{array}{cl}
 & \mathfrak{K}, w \models \alpha \\
\iff & \{\text{by def. } \models\} \\
 & \mathfrak{K}, w \models \beta \text{ or } \mathfrak{K}, w \models \gamma \\
\iff & \{\text{by inductive hypothesis}\} \\
 & \mathcal{F}, w' \models_{FL} T'_M(\beta) \text{ or } \mathcal{F}, w' \models_{FL} T'_M(\gamma) \\
\iff & \{\text{by def. } \models_{FL}\} \\
 & w' ; T'_M(\beta) \neq 0 \text{ or } w' ; T'_M(\gamma) \neq 0 \\
\iff & \{\text{by properties of binary relations}\} \\
 & w' ; (T'_M(\beta) + T'_M(\gamma)) \neq 0 \\
\iff & \{\text{by def. } \models_{FL}\} \\
 & \mathcal{F}, w' \models_{FL} T'_M(\alpha).
\end{array}$$

$\alpha = \langle R \rangle \beta$:

$$\begin{array}{cl}
 & \mathfrak{K}, w \models \alpha \\
\iff & \{\text{by def. } \models\} \\
 & \text{there exists } u \text{ such that } wRu \text{ and } \mathfrak{K}, u \models \beta \\
\iff & \{\text{by inductive hypothesis}\} \\
 & wRu \text{ and } \mathcal{F}, u' \models_{FL} T'_M(\beta) \\
\iff & \{\text{by def. } \models_{FL}\} \\
 & wRu \text{ and } u' ; T'_M(\beta) \neq 0 \\
\iff & \{\text{by properties of binary relations}\} \\
 & w' ; R ; T'_M(\beta) \neq 0 \\
\iff & \{\text{by def. } \models_{FL}\} \\
 & \mathcal{F}, w' \models_{FL} T'_M(\alpha).
\end{array}$$

□

Lemma 6.2 *Given $\mathcal{F} = \langle \mathfrak{A}, m \rangle \in At\mathsf{SPFM}$, a relational world w and a modal formula α, there exists a Kripke model $\mathfrak{K} = \langle W, R', m' \rangle$ (constructed from $\mathcal{F}$) and a world $w' \in W$ (constructed from w) such that*

$$\mathcal{F}, w \models_{FL} T'_M(\alpha) \qquad \iff \qquad \mathfrak{K}, w' \models \alpha\ .$$

Proof Let us take $W = Urel_{\mathfrak{A}}$, $R' = \{ \langle x, y \rangle \in W^2 : x ; m(R) ; y \neq 0 \}$, $w' = w$, and let m' be defined by $m'(p_i) = \{ u \in W : u ; m(P_i) \neq 0 \}$. In order to prove the theorem we will proceed by induction on the structure of the formula α.

$\alpha = p_i$:

$$\begin{array}{ll}
 & \mathcal{F}, w \models_{FL} T'_M(\alpha) \\
\Longleftrightarrow & \{\text{by def. } \models_{FL}\} \\
 & w;m(P_i) \neq 0 \\
\Longleftrightarrow & \{\text{by def. } m'\} \\
 & w' \in m'(p_i) \\
\Longleftrightarrow & \{\text{by def. } \models\} \\
 & \mathfrak{K}, w' \models \alpha.
\end{array}$$

$\alpha = \neg\beta$:

$$\begin{array}{ll}
 & \mathcal{F}, w \models_{FL} T'_M(\alpha) \\
\Longleftrightarrow & \{\text{by def. } T'_M \text{ and } \models_{FL}\} \\
 & w;1\text{'}_{\mathsf{U}};\overline{T'_M(\beta)} \neq 0 \\
\Longleftrightarrow & \left\{\text{by } Dom\,(T'_M(\beta)) \cdot Dom\left(\overline{T'_M(\beta)}\right) = 0 \text{ and } w \text{ atom}\right\} \\
 & w;T'_M(\beta) = 0 \\
\Longleftrightarrow & \{\text{by def. } \models_{FL}\} \\
 & \mathcal{F}, w \not\models_{FL} T'_M(\beta) \\
\Longleftrightarrow & \{\text{by inductive hypothesis}\} \\
 & \mathfrak{K}, w' \not\models \beta \\
\Longleftrightarrow & \{\text{by def. } \models\} \\
 & \mathfrak{K}, w' \models \alpha.
\end{array}$$

$\alpha = \beta \vee \gamma$:

$$\begin{array}{ll}
 & \mathcal{F}, w \models_{FL} T'_M(\alpha) \\
\Longleftrightarrow & \{\text{by def. } \models_{FL} \text{ and } T'_M\} \\
 & w;(T'_M(\beta)+T'_M(\gamma)) \neq 0 \\
\Longleftrightarrow & \{\text{by properties if binary relations }\} \\
 & w;T'_M(\beta) \neq 0 \text{ or } w;T'_M(\gamma) \neq 0 \\
\Longleftrightarrow & \{\text{by def. } \models_{FL}\} \\
 & \mathcal{F}, w \models_{FL} T'_M(\beta) \text{ or } \mathcal{F}, w \models_{FL} T'_M(\gamma) \\
\Longleftrightarrow & \{\text{by inductive hypothesis}\} \\
 & \mathfrak{K}, w' \models \beta \text{ or } \mathfrak{K}, w' \models \gamma \\
\Longleftrightarrow & \{\text{by def. } \models\} \\
 & \mathfrak{K}, w' \models \alpha.
\end{array}$$

$\alpha = \langle R \rangle \beta$:

$$\begin{array}{ll} & \mathcal{F}, w \models_{FL} T'_M(\alpha) \\ \iff & \{\text{by def. } \models_{FL}\} \\ & w;R;T'_M(\beta) \neq 0. \end{array}$$

Since $\mathfrak{A}$ is atomic, there is an atom $u \leq \mathit{Ran}(R) \cdot \mathit{Dom}(T'_M(\beta))$. Since $\mathit{Ran}(R) \leq 1'_{\mathsf{U}}$, $u \leq 1'_{\mathsf{U}}$, and therefore is a relational world. Thus,

$$\begin{array}{ll} & w;R;T'_M(\beta) \neq 0 \\ \iff & \{\text{by previous paragraph}\} \\ & w;R;u \neq 0 \text{ and } u;T'_M(\beta) \neq 0 \\ \iff & \{\text{by def. } \models_{FL}\} \\ & \langle w, u \rangle \in R' \text{ and } \mathcal{F}, u \models_{FL} T'_M(\beta) \\ \iff & \{\text{by inductive hypothesis}\} \\ & \langle w, u \rangle \in R' \text{ and } \mathfrak{K}, u \models \beta \\ \iff & \{\text{by def. } \models\} \\ & \mathfrak{K}, w' \models \alpha. \end{array}$$

□

From Lemmas 6.1 and 6.2, we obtain the following result on the interpretability of the modal logic K. In order to shorten notation, given a set of modal formulas Ψ, by $T_M(\Psi)$ we denote $\{T_M(\psi) : \psi \in \Psi\}$.

Theorem 6.6 *Given a set of modal formulas Ψ and a modal formula φ,*

$$\Psi \models_K \varphi \iff T_M(\Psi) \models_{FL} T_M(\varphi) \text{ in } At\mathsf{SPFM}.$$

Proof $\Rightarrow$) Let us suppose it is not the case that $T_M(\Psi) \models_{FL} T_M(\varphi)$ in $At\mathsf{SPFM}$. Then, there exists $\mathcal{F} = \langle \mathfrak{A}, m \rangle \in At\mathsf{SPFM}$ such that the set of equations $\{T_M(\psi) = 1 : \psi \in \Psi\}$ holds in $\mathfrak{A}$, but $T_M(\varphi) \neq 1$. Then, $\overline{\mathit{Dom}(T'_M(\varphi))} \cdot 1'_{\mathsf{U}} \neq 0$, and since $\mathfrak{A}$ is atomic, there exists an atom w such that $w \leq \overline{\mathit{Dom}(T'_M(\varphi))} \cdot 1'_{\mathsf{U}}$. Then, since $(\overline{\mathit{Dom}(T'_M(\varphi))} \cdot 1'_{\mathsf{U}});T'_M(\varphi) = 0$, we have $w;T'_M(\varphi) = 0$. Thus the fork model $\mathcal{F}$ and the relational world w satisfy $\mathcal{F}, w \models_{FL} T_M(\Psi)$ and $\mathcal{F}, w \not\models_{FL} T_M(\varphi)$. Then, by Lemma 6.2 there exists a Kripke model $\mathfrak{K} = \langle W, R', m' \rangle$ and $w' \in W$ such that w' satisfies Ψ in the model $\mathfrak{K}$, but does not satisfy φ, which contradicts the hypothesis. $\Leftarrow$) Let us suppose it is not the case that $\Psi \models_K \varphi$. Then, there exists a Kripke model $\mathfrak{K} = \langle W, R, m \rangle$ and $w \in W$ such that w satisfies Ψ in the model $\mathfrak{K}$, but φ is not satisfied. Then, by Lemma 6.1 there exists

$\mathcal{F} = (\mathfrak{A}, m') \in At\mathsf{SPFM}$ and a relational world w' such that $T_M(\Psi)$ holds at w' in $\mathcal{F}$, but $T_M(\varphi)$ does not, which contradicts the hypothesis. $\Box$

The following corollary shows the real strength of Thm. 6.6.

Corollary 6.1 *Given a set of modal formulas* $\Psi \cup \{\varphi\}$,

$$\Psi \models_K \varphi \qquad \Longleftrightarrow \qquad T_M(\Psi) \vdash_{FL} T_M(\varphi) \ .$$

Proof By Thm. 6.6, $\Psi \models_K \varphi$ iff $T_M(\Psi) \models_{FL} T_M(\varphi)$ in the class of $At\mathsf{SPFM}$. By Thm. 6.3, $T_M(\Psi) \models_{FL} T_M(\varphi)$ in the class of $At\mathsf{SPFM}$ iff $T_M(\Psi) \models_{FL} T_M(\varphi)$. Finally, by completeness of equational logic [G. Birkhoff (1944)], $T_M(\Psi) \models_{FL} T_M(\varphi)$ iff $T_M(\Psi) \vdash_{FL} T_M(\varphi)$. $\Box$

Theorem 6.6 is proved for the modal logic K, in which no constraints are imposed over the accessibility relation. The next theorem generalizes the previous result.

Theorem 6.7 *Given a modal logic* $L(\Gamma)$, *where* Γ *is a set (not necessarily finite) of first-order sentences,*

$$\models_{L(\Gamma)} \varphi \qquad \Longleftrightarrow \qquad T_{\nabla,\sigma}(\Gamma) \vdash_{FL} T_M(\varphi) \ .$$

Proof Follows directly from Cor. 5.1 and Cor. 6.1. $\Box$

6.6 A Proof Theoretical Approach

If the logic L under consideration has an axiomatic system that is complete with respect to the semantics of (the language of) L, then we can prove interpretability of L in fork logic in a proof theoretic style. As an example, let us show how the equational calculus of fork algebras can be used for proving interpretability of some specific modal logics.

Definition 6.18 The calculus for the modal logic K is given by the following axiom schemas and rules:

(K) $[R](A \to B) \to ([R]A \to [R]B)$.
(RN) $\{A\} \vdash_K [R]A$
(MP) $\{A, A \to B\} \vdash_K B$.

The following theorem shows that we can replace provability of any formula α in the logic K by provability of a fork logic formula obtained from α in fork logic.

Theorem 6.8 *For every modal sentence α, we have*

$$\vdash_K \alpha \qquad \Longleftrightarrow \qquad \vdash_{FL} T_M(\alpha)\ .$$

Proof

$\Rightarrow$) We will proceed by induction on the length of proofs. Let us have a proof of length 1, then α must be an instance of the axiom schema K.

$$\begin{aligned}
T_M(K) &= 1'_{\mathsf{U}};\overline{T'_M([R](A\to B))} + T'_M([R]A\to[R]B) + \overline{{}_{\mathsf{U}}1}\\
&= 1'_{\mathsf{U}};\overline{1'_{\mathsf{U}};\overline{R;1'_{\mathsf{U}};\overline{T'_M(A\to B)}}} + 1'_{\mathsf{U}};\overline{1'_{\mathsf{U}};\overline{R;1'_{\mathsf{U}};\overline{T'_M(A)}}}\\
&\quad + 1'_{\mathsf{U}};\overline{R;1'_{\mathsf{U}};\overline{\overline{T'_M(B)}}} + \overline{{}_{\mathsf{U}}1}\\
&= R;\overline{1'_{\mathsf{U}};\overline{T'_M(A)} + T'_M(B)} + 1'_{\mathsf{U}};R;\overline{T'_M(A)}\\
&\quad + 1'_{\mathsf{U}};\overline{R;\overline{\overline{T'_M(B)}}} + \overline{{}_{\mathsf{U}}1}\\
&= R;\left(T'_M(A)\cdot\overline{T'_M(B)}\right) + R;\overline{T'_M(A)} + 1'_{\mathsf{U}};\overline{R;\overline{\overline{T'_M(B)}}} + \overline{{}_{\mathsf{U}}1}\\
&= 1'_{\mathsf{U}};R;\left(\left(T'_M(A)\cdot\overline{T'_M(B)}\right) + \overline{T'_M(A)}\right) + 1'_{\mathsf{U}};\overline{R;\overline{\overline{T'_M(B)}}} + \overline{{}_{\mathsf{U}}1}\\
&\geq 1'_{\mathsf{U}};R;\overline{T'_M(B)} + 1'_{\mathsf{U}};\overline{R;\overline{\overline{T'_M(B)}}} + \overline{{}_{\mathsf{U}}1}\\
&= {}_{\mathsf{U}}1 + \overline{{}_{\mathsf{U}}1}\\
&= 1.
\end{aligned}$$

Since $T_M(K) \geq 1$, it must be $T_M(K) = 1$, as was to be proved.
If the length of the proof is greater than 1, then:

(1) α was obtained by RN from a formula β, then if $T_M(\beta) = 1$, we have

$$\begin{aligned}
T_M([R]\beta) &= 1'_{\mathsf{U}};\overline{R;1'_{\mathsf{U}};\overline{T'_M(\beta)}} + \overline{{}_{\mathsf{U}}1}\\
&= 1'_{\mathsf{U}};\overline{R;1'_{\mathsf{U}};\overline{{}_{\mathsf{U}}1}} + \overline{{}_{\mathsf{U}}1}\\
&= 1'_{\mathsf{U}};\overline{R;0} + \overline{{}_{\mathsf{U}}1}\\
&= {}_{\mathsf{U}}1+\overline{{}_{\mathsf{U}}1}\\
&= 1.
\end{aligned}$$

(2) α was obtained by MP from the formulas β and $\beta \to \alpha$. Then, if

$T_M(\beta) = 1$ and $T_M(\beta \to \alpha) = 1$, we have

$$\begin{aligned} T_M(\alpha) &= 0 \;+\; T'_M(\alpha) \;+\; \overline{{}_{\mathsf{U}}1} \\ &= 1\text{'}_{\mathsf{U}};\overline{{}_{\mathsf{U}}1} \;+\; T'_M(\alpha) \;+\; \overline{{}_{\mathsf{U}}1} \\ &= 1\text{'}_{\mathsf{U}};\overline{T'_M(\beta)} \;+\; T'_M(\alpha) \;+\; \overline{{}_{\mathsf{U}}1} \\ &= T_M(\beta \to \alpha) \\ &= 1. \end{aligned}$$

$\Leftarrow$) Let us suppose it is not the case that $\vdash_K \alpha$. Then, since the calculus is complete, there exists a Kripke model $\mathfrak{M} = (W, R, m)$ and a world $w \in W$ such that $\mathfrak{M}, w \not\models_K \alpha$. Thus, by Lemma 6.1, there exists a fork model $\mathcal{F} = \langle \mathfrak{A}, m' \rangle$ and a relational world w' such that $\mathcal{F}, w' \not\models_{FL} \alpha$, which contradicts the hypothesis. $\square$

As a second example, let us consider the version of Thm. 6.8 for the modal logic T.

Definition 6.19 The calculus for the modal logic T is obtained from the calculus for the logic K by adding the axiom schema

(T) $[R]A \to A$.

It is well known that the calculus T is complete with respect to those frames whose accessibility relation is reflexive. Reflexivity can be easily characterized as an equation in the calculus of relations. For instance, the equation $1\text{'}_{\mathsf{U}} \leq R$ says that R is reflexive when R is interpreted as a relation on the set of urelements. Nevertheless, in the following theorem we will use the characterization of reflexivity provided by the mapping $T_{\nabla,\sigma}$ (Def. 5.18)

$$T_{\nabla,\sigma}(\forall x\,(Rxx)) = \overline{1_{\mathsf{U}};\overline{\left(1\text{'} \;\nabla\; 1\text{'};\breve{R}\right);\breve{2};1}}\;.$$

Even though the equation obtained in this way is more complex, it is worth emphasizing that it was obtained automatically from the first-order definition of reflexivity.

Now we will show that both equations characterizing reflexivity are equivalent.

$$
\begin{array}{lr}
T_{\nabla,\sigma}\left(\forall x\,(Rxx)\right) = 1 & \\
\iff \overline{1_{\mathsf{U}};\overline{\overline{\left(1' \;\nabla\; 1';\breve{R}\right);\breve{2};1}}} = 1 & \text{(by applying } T_{\nabla,\sigma}\text{)} \\
\iff \overline{1_{\mathsf{U}};\overline{\overline{\left(1'\cdot\breve{R}\right);1}}} = 1 & \text{(by Ax. 5 and Thm. 3.2.1)} \\
\iff 1_{\mathsf{U}};\overline{\left(1'\cdot\breve{R}\right);1} = 0 & \text{(by BA)} \\
\iff ({}_{\mathsf{U}}1;1) \cdot \overline{\left(1'\cdot\breve{R}\right);1} = 0 & \text{(by Ax. 7)} \\
\iff {}_{\mathsf{U}}1 \leq \left(1'\cdot\breve{R}\right);1 & \text{(by BA)} \\
\iff 1'_{\mathsf{U}} \leq 1'\cdot\breve{R} & \text{(by prop. of right-ideals)} \\
\iff 1'_{\mathsf{U}} \leq \breve{R} & \text{(by BA)} \\
\iff 1'_{\mathsf{U}} \leq R. & \text{(by monotonicity of } \breve{}\text{)}
\end{array}
$$

Theorem 6.9 *For every modal formula α,*

$$\vdash_T \alpha \qquad \iff \qquad T_{\nabla,\sigma}\left(\forall x\,(Rxx)\right) \vdash_{FL} T_M(\alpha)\ .$$

Proof Let us show that $T_{\nabla,\sigma}\left(\forall x\,(Rxx)\right) \vdash_{FL} T_M\left([R]A \to A\right)$.

$$
\begin{aligned}
T_M([R]A \to A) &= 1'_{\mathsf{U}};\overline{1'_{\mathsf{U}};\overline{\overline{R;1'_{\mathsf{U}};\overline{T'_M(A)}}}} + T'_M(A) + \overline{{}_{\mathsf{U}}1} \\
&= R;\overline{T'_M(A)} + T'_M(A) + \overline{{}_{\mathsf{U}}1} \\
&\geq 1'_{\mathsf{U}};\overline{T'_M(A)} + T'_M(A) + \overline{{}_{\mathsf{U}}1} \\
&= 1'_{\mathsf{U}};\overline{T'_M(A)} + 1'_{\mathsf{U}};T'_M(A) + \overline{{}_{\mathsf{U}}1} \\
&= {}_{\mathsf{U}}1 + \overline{{}_{\mathsf{U}}1} \\
&= 1.
\end{aligned}
$$

□

Next, let us consider the logic B, whose accessibility relation is reflexive and symmetric. The following definition presents a calculus for B.

Definition 6.20 The calculus for the logic B is obtained from the calculus for T by adding the axiom

(B) $A \to [R]\langle R\rangle A$.

It is easy to show that applying the mapping $T_{\nabla,\sigma}$ to the first-order sentence $\forall x, y(Rxy \rightarrow Ryx)$ (which asserts the symmetry of R) we obtain an equation equivalent to the simpler relational equation $R = \breve{R}$.

Theorem 6.10 *For every modal formula α,*

$$\vdash_B \alpha \qquad \Longleftrightarrow \qquad T_{\nabla,\sigma}(\forall xy\,(Rxy \rightarrow Ryx)) \vdash_{FL} T_M(\alpha)\ .$$

Proof Let us show that the axiom schema B satisfies $T_M(B) = 1$. Applying the mapping T'_M, we obtain

$$\begin{array}{cl}
 & T'_M(A \rightarrow [R]\,\langle R\rangle\, A) \\
= & \{\text{by def. } T'_M\,\} \\
 & 1'_{\mathsf{U}};\overline{T'_M(A)} + \overline{1'_{\mathsf{U}};R;1'_{\mathsf{U}};\overline{R;T'_M(A)}} \\
= & \{\text{by } \mathsf{BA}\} \\
 & \overline{\overline{1'_{\mathsf{U}};\overline{\overline{T'_M(A)}} + \overline{1'_{\mathsf{U}};R;1'_{\mathsf{U}};\overline{R;T'_M(A)}}}} \\
= & \{\text{by } \mathsf{BA}\} \\
 & \overline{\overline{1'_{\mathsf{U}};\overline{\overline{T'_M(A)}}} \cdot \overline{\overline{1'_{\mathsf{U}};R;1'_{\mathsf{U}};\overline{R;T'_M(A)}}}} \\
= & \{\text{by Thm. 2.3.19 and } \mathsf{BA}\} \\
 & \overline{\left(\overline{{}_{\mathsf{U}}1} + 1'_{\mathsf{U}};T'_M(A)\right) \cdot \left(\overline{{}_{\mathsf{U}}1} + 1'_{\mathsf{U}};R;1'_{\mathsf{U}};\overline{R;T'_M(A)}\right)} \\
= & \{\text{by } \mathsf{BA},\ R = 1'_{\mathsf{U}};R;1'_{\mathsf{U}},\ T'_M = 1'_{\mathsf{U}};T'_M\ \} \\
 & \overline{\overline{{}_{\mathsf{U}}1} + \left((T'_M(A)) \cdot \left(R;\overline{R;T'_M(A)}\right)\right)} \\
= & \{\text{by } R \text{ symmetric}\} \\
 & \overline{\overline{{}_{\mathsf{U}}1} + \left((T'_M(A)) \cdot \left(R;\overline{\breve{R};T'_M(A)}\right)\right)} \\
\geq & \{\text{by (2.2)}\} \\
 & \overline{\overline{{}_{\mathsf{U}}1} + 1;(\overline{\breve{R};T'_M(A)} \cdot (\breve{R};T'_M(A)))} \\
= & \{\text{by } \mathsf{BA}\} \\
 & \overline{\overline{{}_{\mathsf{U}}1} + 1;0} \\
= & \{\text{by Thm. 2.3.1}\} \\
 & \overline{\overline{{}_{\mathsf{U}}1} + 0} \\
= & \{\text{by } \mathsf{BA}\ \} \\
 & {}_{\mathsf{U}}1.
\end{array}$$

Thus, $T_M(B) = T'_M(B) + \overline{{}_{\mathsf{U}}1} = {}_{\mathsf{U}}1 + \overline{{}_{\mathsf{U}}1} = 1.$ □

6.7 Interpretability of Propositional Dynamic Logic in *FL*

Propositional dynamic logic is considered as a programming logic, i.e., a logic suitable for asserting and proving properties of programs. Dynamic logic is a modal logic whose modal operators are determined by programs understood as binary relations in a set of computation states. For thorough presentations of dynamic logic see [D. Harel (1984); D. Harel et al. (2000)]. The following definitions provide a formal description of propositional dynamic logic.

Definition 6.21 Let us consider a set P_0 of *atomic programs*, and a set F_0 of *atomic dynamic formulas*. From these sets we will construct the sets F of dynamic formulas and P of compound programs.

F and P are the smallest sets satisfying the following conditions:

(1) $true \in F$, $false \in F$, $F_0 \subseteq F$,
(2) if $p \in F$ and $q \in F$ then $\neg p \in F$ and $(p \vee q) \in F$,
(3) if $p \in F$ and $\alpha \in P$ then $\langle \alpha \rangle\, p \in F$,
(4) $P_0 \subseteq P$,
(5) if $\alpha \in P$ and $\beta \in P$ then $(\alpha \cup \beta) \in P$, $(\alpha;\beta) \in P$ and $\alpha^* \in P$,
(6) if $p \in F$ then $p? \in P$.

Notice that in Def. 6.21 the sets F and P are defined by mutual recursion, i.e., in order to define F we assume the definition of P (in (3)) and in order to define P we assume that F is defined (in (6)).

Definition 6.22 A dynamic model is a triple $\mathfrak{D} = \langle\, W, \tau, \delta \,\rangle$ where W is a set of *states*, τ assigns subsets of W to atomic formulas, and δ assigns subsets of $W \times W$ to atomic programs. The mappings τ and δ are extended inductively to determine the meaning of compound formulas and programs as follows:

$\tau(true) = W$,
$\tau(false) = \emptyset$,
$\tau(\neg p) = \overline{\tau(p)}$,
$\tau(p \vee q) = \tau(p) \cup \tau(q)$,
$\tau(\langle \alpha \rangle\, p) = \{\, s \in W : \exists t\, (\langle s,t \rangle \in \delta(\alpha) \wedge t \in \tau(p))\,\}$,
$\delta(\alpha;\beta) = \{\, \langle s,t \rangle : \exists u\, (\langle s,u \rangle \in \delta(\alpha) \wedge \langle u,t \rangle \in \delta(\beta))\,\}$,
$\delta(\alpha \cup \beta) = \delta(\alpha) \cup \delta(\beta)$,
$\delta(p?) = \{\, \langle s,s \rangle : s \in \tau(p)\,\}$,
$\delta(\alpha^*) = \delta(\alpha)^*$ (the reflexive and transitive closure of $\delta(\alpha)$).

Propositional dynamic logic is know to have a complete Hilbert-style calculus. The calculus is given in the following definition. The proof of the completeness of the calculus is given in [D. Harel (1984)], Thm. 2.11, p. 515.

Definition 6.23 The calculus for propositional dynamic logic is given by the following axiom schemes and inference rules:

(A1) all instances of tautologies of the propositional calculus.
(A2) $\langle\alpha\rangle (p \vee q) \leftrightarrow (\langle\alpha\rangle p \vee \langle\alpha\rangle q)$.
(A3) $\langle\alpha;\beta\rangle p \leftrightarrow \langle\alpha\rangle \langle\beta\rangle p$.
(A4) $\langle\alpha \cup \beta\rangle p \leftrightarrow (\langle\alpha\rangle p \vee \langle\beta\rangle p)$.
(A5) $\langle\alpha^*\rangle p \leftrightarrow (p \vee \langle\alpha\rangle \langle\alpha^*\rangle p)$.
(A6) $\langle q?\rangle p \leftrightarrow p \wedge q$.
(A7) $[\alpha^*](p \to [\alpha]p) \to (p \to [\alpha^*]p)$.
(A8) $[\alpha](p \to q) \to ([\alpha]p \to [\alpha]q)$.

The inference rules for the calculus are, as in K, modus ponens and generalization.

The presence of the Kleene star operator in the language of dynamic logic requires a slight generalization of fork algebras in order to obtain the interpretability result.

Definition 6.24 A *closure* AFA (CAFA for short), is a structure $\langle \mathfrak{A}, * \rangle$ such that $\mathfrak{A} \in$ AFA, and $*$ satisfies the equations

$$R^* = 1' + R;R^*, \qquad \text{(Ax. 11)}$$

$$R^*;S;1 \leq S;1 + R^*;(\overline{S;1} \cdot R;S;1). \qquad \text{(Ax. 12)}$$

The second equation was added for technical reasons. It is needed in order to prove Thm. 6.11 below. In order to show its validity, let us analyze its meaning. The second-order formula

$$\forall R \forall S \forall x\, (\exists y\, (x R^* y \wedge y \in S) \Rightarrow (x \in S \vee \exists z\, (x R^* z \wedge z \notin S \wedge \exists w\, (z R w \wedge w \in S))))$$

expresses that if a finite path exists in the graph induced by R, which connects x with an element $y \in S$, then, either x is already in S, or from x

we can reach an object outside S which is R-next to an object in S. This is a desirable property of the operation *. Keeping in mind that right-ideal relations represent sets, it is easy to see that Ax. 12 in Def. 6.24 represents this second-order formula.

From closure fork algebras, it is an easy task to generalize fork logic to the so called closure fork logic (denoted by *CFL*). The construction of closure fork logic is analogous to the construction of fork logic.

In the forthcoming theorems the mappings T_{DL} and T_P defined below will play a central role.

Definition 6.25 In order to define the mappings T_{DL} and T_P from dynamic logic formulas and compound programs, respectively, into terms in the language of closure fork algebra with urelements, we first define the mappings T'_{DL} and T_P by mutual recursion.

$$
\begin{aligned}
&T'_{DL}(p_i) = P_i, (p_i \text{ an atomic formula.})\\
&T'_{DL}(true) = {}_{\mathsf{U}}1,\\
&T'_{DL}(false) = 0,\\
&T'_{DL}(\neg p) = 1'_{\mathsf{U}} ; \overline{T'_{DL}(p)},\\
&T'_{DL}(p \vee q) = T'_{DL}(p) + T'_{DL}(q),\\
&T'_{DL}(\langle R \rangle\, p) = T_P(R) ; T'_{DL}(p),\\
&T_P(R_i) = R_i, (R_i \text{ an atomic program.})\\
&T_P(R \cup S) = T_P(R) + T_P(S),\\
&T_P(R;S) = T_P(R) ; T_P(S),\\
&T_P(R^*) = T_P(R)^*,\\
&T_P(p?) = T'_{DL}(p) \cdot 1'_{\mathsf{U}}.
\end{aligned}
$$

Next, we define the mapping T_{DL} by $T_{DL}(\alpha) = T'_{DL}(\alpha) + \overline{{}_{\mathsf{U}}1}$.

In a similar way as in *FL*, we introduce the notion of *closure fork model.*

Definition 6.26 A *closure fork model* is a pair $\langle \mathfrak{A}, m \rangle$ where $\mathfrak{A}$ is closure abstract fork algebra with urelements (CAFAU), and m is the meaning function that assigns relations in A both to variables in *RelVar* and to the extralogical symbols*. It is clear how to extend m homomorphically to a function $m' : \Omega \to A$. For the sake of simplicity, we will use the name m for both mappings.

*In this case, the extralogical symbols are the atomic programs and the atomic formulas.

Lemma 6.3 *Given a dynamic model $D = \langle W, \tau, \delta \rangle$, there exists a closure fork model $\mathcal{C} = \langle \mathfrak{A}, m \rangle$ such that for any dynamic formula φ*

$$\mathsf{dom}\,(m(T'_{DL}(\varphi))) \quad = \quad \tau(\varphi)\ .$$

Proof Let R_i be an atomic program, and p_i an atomic formula. Let $\mathfrak{A}$ be the full fork algebra with set of urelements W. Let us define $m\,(P_i) = \{\, \langle s, x\rangle : s \in \tau(p_i)\,\}$ and $m\,(R_i) = \delta(R_i)$. The proof is by induction on the structure of the formula φ.

$\varphi = p_i$:

$$\begin{array}{ll} & \mathsf{dom}\,(m\,(T'_{DL}(p_i))) \\ = & \{\,\text{by def. } T'_{DL}\,\} \\ & \mathsf{dom}\,(m\,(P_i)) \\ = & \{\,\text{by def. } m\,\} \\ & \tau(p_i). \end{array}$$

$\varphi = true$:

$$\begin{array}{ll} & \mathsf{dom}\,(m\,(T'_{DL}(true))) \\ = & \{\,\text{by def. } T'_{DL}\,\} \\ & \mathsf{dom}\,({}_{U}1) \\ = & \{\,\text{by } W = Urel_{\mathfrak{A}}\,\} \\ & W \\ = & \{\,\text{by def. } \tau\,\} \\ & \tau(true). \end{array}$$

$\varphi = false$:

$$\begin{array}{ll} & \mathsf{dom}\,(m\,(T'_{DL}(false))) \\ = & \{\,\text{by def. } T'_{DL}\,\} \\ & \mathsf{dom}\,(0) \\ = & \{\,\text{by def. } 0\,\} \\ & \emptyset \\ = & \{\,\text{by def. } \tau\,\} \\ & \tau(false). \end{array}$$

$\varphi = \neg p$:

$$\begin{array}{cl} & \mathsf{dom}\,(m\,(T'_{DL}(\neg p))) \\ = & \{\text{by def. } T'_{DL}\} \\ & \mathsf{dom}\left(1\text{'}\cup;\overline{m\,(T'_{DL}(p))}\right) \\ = & \{\text{by } m\,(T'_{DL}(p)) \text{ right-ideal}\} \\ & W \setminus \mathsf{dom}\,(m\,(T'_{DL}(p))) \\ = & \{\text{by inductive hypothesis}\} \\ & W \setminus \tau(p) \\ = & \{\text{by def. } \tau\} \\ & \tau(\neg p). \end{array}$$

$\varphi = p \vee q$:

$$\begin{array}{cl} & \mathsf{dom}\,(m\,(T'_{DL}(p \vee q))) \\ = & \{\text{by def. } T'_{DL}\} \\ & \mathsf{dom}\,((m\,(T'_{DL}(p) + T'_{DL}(q)))) \\ = & \{\text{by def. } m\} \\ & \mathsf{dom}\,(m\,(T'_{DL}(p))) \cup \mathsf{dom}\,(m\,(T'_{DL}(q))) \\ = & \{\text{by inductive hypothesis}\} \\ & \tau(p) \cup \tau(q) \\ = & \{\text{by def. } \tau\} \\ & \tau(p \vee q). \end{array}$$

$\varphi = \langle Q \rangle\, p$: Let us prove by induction on the structure of Q that

$$m(T_P(Q)) = \delta(Q)\ .$$

$Q = R_i$:

$$\begin{array}{cl} & m\,(T_P(R_i)) \\ = & \{\text{by def. } T_P\} \\ & m\,(R_i) \\ = & \{\text{by def. } m\} \\ & \delta(R_i). \end{array}$$

$Q = R \cup S$:

$$\begin{array}{cl} & m\,(T_P(R \cup S)) \\ = & \{\text{by def. } T_P\} \end{array}$$

$$\begin{array}{cl} & m\,(T_P(R)+T_P(S)) \\ = & \{\text{by def. } m\} \\ & m\,(T_P(R)) \cup m\,(T_P(S)) \\ = & \{\text{by inductive hypothesis}\} \\ & \delta(R) \cup \delta(S) \\ = & \{\text{by def. } \delta\} \\ & \delta(R \cup S). \end{array}$$

$Q = R;S$:

$$\begin{array}{cl} & m\,(T_P(R;S)) \\ = & \{\text{by def. } T_P\} \\ & m\,(T_P(R);T_P(S)) \\ = & \{\text{by def. } m\} \\ & m\,(T_P(R)) \circ m\,(T_P(S)) \\ = & \{\text{by inductive hypothesis}\} \\ & \delta(R) \circ \delta(S) \\ = & \{\text{by def. } \delta\} \\ & \delta(R;S). \end{array}$$

$Q = R^*$:

$$\begin{array}{cl} & m\,(T_P(R^*)) \\ = & \{\text{by def. } T_P\} \\ & m\,(T_P(R)^*) \\ = & \{\text{by def. } m\} \\ & m\,(T_P(R))^* \\ = & \{\text{by inductive hypothesis}\} \\ & \delta(R)^* \\ = & \{\text{by def. } \delta\ \} \\ & \delta(R^*). \end{array}$$

$Q = q?$: By definition of T_P,

$$m\,(T_P(q?)) = (m\,(T'_{DL}(q)) \cdot 1'_{\mathsf{U}}) \ .$$

But,

$$m\,(T'_{DL}(q)) \cdot 1'_{\mathsf{U}} = \{\, \langle w, w \rangle \in W^2 : w \in \mathsf{dom}\,(m\,(T'_{DL}(q)))\,\} \ .$$

Since the complexity of q is strictly smaller that the complexity of φ, by inductive hypothesis

$$m\left(T'_{DL}(q)\right) \cdot 1\text{'}_{\mathsf{U}} = \left\{ \langle w, w \rangle \in W^2 : w \in \tau(q) \right\} = \delta\left(q?\right) .$$

Finally,

$$\begin{aligned}
& \mathsf{dom}\left(m(T'_{DL}(\langle Q \rangle\, p))\right) \\
= \; & \{\text{ by def. } T'_{DL} \;\} \\
& \mathsf{dom}\left(m\left(T_P(Q) ; T'_{DL}(p)\right)\right) \\
= \; & \{\text{ by def. } m\,\} \\
& \mathsf{dom}\left(m\left(T_P(Q)\right) \circ m\left(T'_{DL}(p)\right)\right) \\
= \; & \{\text{ by previous lemma}\} \\
& \mathsf{dom}\left(\delta(Q) \circ m\left(T'_{DL}(p)\right)\right) \\
= \; & \{\text{ by def. } \circ\,\} \\
& \{\, s \in W : \exists t \in W\left(\langle s,t \rangle \in \delta(Q) \;\wedge\; t \in \mathsf{dom}\left(m\left(T'_{DL}(p)\right)\right)\right)\} \\
= \; & \{\text{ by inductive hypothesis}\} \\
& \{\, s \in W : \exists t \in W\left(\langle s,t \rangle \in \delta(Q) \;\wedge\; t \in \tau(p)\right)\} \\
= \; & \{\text{ by def. } \tau\,\} \\
& \tau\left(\langle Q \rangle\, p\right) .
\end{aligned}$$

□

Theorem 6.11 *Given a dynamic formula* φ*,*

$$\vdash_{DL} \varphi \qquad \Longleftrightarrow \qquad \vdash_{CFL} T_{DL}(\varphi) \; .$$

Proof $\Rightarrow$) Proving that $\vdash_{CFL}\; T_{DL}(\varphi)$, is equivalent to proving that $T'_{DL}(\varphi) = {}_{\mathsf{U}}1$. The proof proceeds by induction on the length of proofs. If the proof has length 1, then φ must be an instance of one of the axioms. Let us analyze each one of the axiom schemes.
(A1) If φ is an instance of a propositional tautology, $T'_{DL}(\varphi) = {}_{\mathsf{U}}1$ can be easily proved.
(A2) By definition of $\leftrightarrow$,

$$\begin{aligned}
&\langle \alpha \rangle\, (p \vee q) \;\leftrightarrow\; (\langle \alpha \rangle\, p \vee \langle \alpha \rangle\, q) \\
&\quad = \left(\langle \alpha \rangle\, (p \vee q) \rightarrow (\langle \alpha \rangle\, p \vee \langle \alpha \rangle\, q)\right) \;\wedge\; \left(\langle \alpha \rangle\, (p \vee q) \leftarrow (\langle \alpha \rangle\, p \vee \langle \alpha \rangle\, q)\right) .
\end{aligned}$$

Since $T'_{DL}\left(P \wedge Q\right) = T'_{DL}\left(P\right) \cdot T'_{DL}\left(Q\right)$, it suffices to show that

$$T'_{DL}\left(\langle \alpha \rangle\, (p \vee q) \rightarrow (\langle \alpha \rangle\, p \vee \langle \alpha \rangle\, q)\right) = {}_{\mathsf{U}}1 \qquad \text{and}$$
$$T'_{DL}\left(\langle \alpha \rangle\, (p \vee q) \leftarrow (\langle \alpha \rangle\, p \vee \langle \alpha \rangle\, q)\right) = {}_{\mathsf{U}}1 \; .$$

$$\begin{aligned}
& T'_{DL}\left(\langle\alpha\rangle(p\vee q)\rightarrow(\langle\alpha\rangle p\vee\langle\alpha\rangle q)\right)\\
= \;& \{\text{by def.} \rightarrow\}\\
& 1'_{\mathsf{U}};\overline{T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))} + T_P(\alpha);T'_{DL}(p) + T_P(\alpha);T'_{DL}(q)\\
= \;& \{\text{by Ax. 2}\}\\
& 1'_{\mathsf{U}};\overline{T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))} + 1'_{\mathsf{U}};T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))\\
= \;& \{\text{by Ax. 2}\}\\
& 1'_{\mathsf{U}};\left(\overline{T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))} + T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))\right)\\
= \;& \{\text{by } \mathsf{BA}\}\\
& {}_{\mathsf{U}}1.
\end{aligned}$$

$$\begin{aligned}
& T'_{DL}\left(\langle\alpha\rangle(p\vee q)\leftarrow(\langle\alpha\rangle p\vee\langle\alpha\rangle q)\right)\\
= \;& \{\text{by def.} \leftarrow\}\\
& 1'_{\mathsf{U}};\overline{T_P(\alpha);T'_{DL}(p) + T_P(\alpha);T'_{DL}(q)}\\
& \quad + T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))\\
= \;& \{\text{by Ax. 2}\}\\
& 1'_{\mathsf{U}};\overline{T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))} + 1'_{\mathsf{U}};T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))\\
= \;& \{\text{by Ax. 2}\}\\
& 1'_{\mathsf{U}};\left(\overline{T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))} + T_P(\alpha);(T'_{DL}(p)+T'_{DL}(q))\right)\\
= \;& \{\text{by } \mathsf{BA}\}\\
& {}_{\mathsf{U}}1.
\end{aligned}$$

(A3)

$$\begin{aligned}
& T'_{DL}\left(\langle\alpha;\beta\rangle p \leftrightarrow \langle\alpha\rangle\langle\beta\rangle p\right)\\
= \;& \{\text{by def.} \leftrightarrow\}\\
& T'_{DL}\left((\langle\alpha;\beta\rangle p\rightarrow\langle\alpha\rangle\langle\beta\rangle p) \wedge (\langle\alpha;\beta\rangle p\leftarrow\langle\alpha\rangle\langle\beta\rangle p)\right)\\
= \;& \{\text{by defs.} \rightarrow, \leftarrow \text{ and } T'_{DL}\}\\
& \left(1'_{\mathsf{U}};\overline{T_P(\alpha;\beta);T'_{DL}(p)} + 1'_{\mathsf{U}};T_P(\alpha);T_P(\beta);T'_{DL}(p)\right)\\
& \quad\cdot\left(1'_{\mathsf{U}};\overline{T_P(\alpha);T_P(\beta);T'_{DL}(p)} + 1'_{\mathsf{U}};T_P(\alpha;\beta);T'_{DL}(p)\right)\\
= \;& \{\text{by def. } T_P\}\\
= \;& \{\text{by Ax. 2 and } \mathsf{BA}\}\\
& {}_{\mathsf{U}}1\cdot{}_{\mathsf{U}}1\\
= \;& \{\mathsf{BA}\}\\
& {}_{\mathsf{U}}1.
\end{aligned}$$

(A4)

$$\begin{aligned}
& T'_{DL}\left(\langle\alpha\cup\beta\rangle p\leftrightarrow(\langle\alpha\rangle p\vee\langle\beta\rangle p)\right)\\
= \;& \{\text{by def.} \leftrightarrow\}\\
& T'_{DL}\left((\langle\alpha\cup\beta\rangle p\rightarrow(\langle\alpha\rangle p\vee\langle\beta\rangle p)) \wedge (\langle\alpha\cup\beta\rangle p\leftarrow(\langle\alpha\rangle p\vee\langle\beta\rangle p))\right)
\end{aligned}$$

$$
\begin{aligned}
= \; & \{\text{by def. } T'_{DL}\} \\
& \Big(1'\mathsf{U}\,;\overline{T_P(\alpha \cup \beta)\,;T'_{DL}(p)} + T_P(\alpha)\,;T'_{DL}(p) + T_P(\beta)\,;T'_{DL}(p)\Big) \\
& \cdot \Big(1'\mathsf{U}\,;\overline{T_P(\alpha)\,;T'_{DL}(p) + T_P(\beta)\,;T'_{DL}(p)} + T_P(\alpha \cup \beta)\,;T'_{DL}(p)\Big) \\
= \; & \{\text{by def. } T_P\} \\
& \Big(1'\mathsf{U}\,;\overline{T_P(\alpha \cup \beta)\,;T'_{DL}(p)} + 1'\mathsf{U}\,;T_P(\alpha \cup \beta)\,;T'_{DL}(p)\Big) \\
& \quad \cdot \Big(1'\mathsf{U}\,;\overline{T_P(\alpha \cup \beta)\,;T'_{DL}(p)} + 1'\mathsf{U}\,;T_P(\alpha \cup \beta)\,;T'_{DL}(p)\Big) \\
= \; & \{\text{by Ax. 2 and } \mathsf{BA}\} \\
& \mathsf{U}1 \cdot \mathsf{U}1 \\
= \; & \{\text{by } \mathsf{BA}\} \\
& \mathsf{U}1.
\end{aligned}
$$

(A5)

$$
\begin{aligned}
& T'_{DL}(\langle \alpha^* \rangle p \leftrightarrow (p \vee \langle \alpha \rangle \langle \alpha^* \rangle p)) \\
= \; & \{\text{by def. } \leftrightarrow\} \\
& T'_{DL}(((\langle \alpha^* \rangle p \rightarrow (p \vee \langle \alpha \rangle \langle \alpha^* \rangle p)) \wedge ((\langle \alpha^* \rangle p \leftarrow (p \vee \langle \alpha \rangle \langle \alpha^* \rangle p))) \\
= \; & \{\text{by defs. } \rightarrow, \leftarrow \text{ and } T'_{DL}\} \\
& \Big(1'\mathsf{U}\,;\overline{T_P(\alpha^*)\,;T'_{DL}(p)} + T'_{DL}(p) + T_P(\alpha)\,;T_P(\alpha^*)\,;T'_{DL}(p)\Big) \\
& \quad \cdot \Big(1'\mathsf{U}\,;\overline{T'_{DL}(p) + T_P(\alpha)\,;T_P(\alpha^*)\,;T'_{DL}(p)} + T_P(\alpha^*)\,;T'_{DL}(p)\Big) \\
= \; & \{\text{by def. } T_P\} \\
& \Big(1'\mathsf{U}\,;\overline{T_P(\alpha)^*\,;T'_{DL}(p)} + T'_{DL}(p) + T_P(\alpha)\,;T_P(\alpha)^*\,;T'_{DL}(p)\Big) \\
& \quad \cdot \Big(1'\mathsf{U}\,;\overline{T'_{DL}(p) + T_P(\alpha)\,;T_P(\alpha)^*\,;T'_{DL}(p)} + T_P(\alpha)^*\,;T'_{DL}(p)\Big) \\
= \; & \{\text{by Ax. 2}\} \\
& \Big(1'\mathsf{U}\,;\overline{T_P(\alpha)^*\,;T'_{DL}(p)} + \big(1' + T_P(\alpha)\,;T_P(\alpha)^*\big)\,;T'_{DL}(p)\Big) \\
& \quad \cdot \Big(1'\mathsf{U}\,;\overline{\big(1' + T_P(\alpha)\,;T_P(\alpha)^*\big)\,;T'_{DL}(p)} + T_P(\alpha)^*\,;T'_{DL}(p)\Big) \\
= \; & \{\text{by Ax. 11}\} \\
& \Big(1'\mathsf{U}\,;\overline{T_P(\alpha)^*\,;T'_{DL}(p)} + 1'\mathsf{U}\,;T_P(\alpha)^*\,;T'_{DL}(p)\Big) \\
& \quad \cdot \Big(1'\mathsf{U}\,;\overline{T_P(\alpha)^*\,;T'_{DL}(p)} + 1'\mathsf{U}\,;T_P(\alpha)^*\,;T'_{DL}(p)\Big) \\
= \; & \{\text{by Ax. 2 and } \mathsf{BA}\} \\
& \mathsf{U}1 \cdot \mathsf{U}1 \\
= \; & \{\text{by } \mathsf{BA}\} \\
& \mathsf{U}1.
\end{aligned}
$$

(A6)

$$\begin{aligned}
& T'_{DL}\left(\langle p?\rangle\, q \;\leftrightarrow\; p \wedge q\right) \\
= \;& \{\text{by def. } \leftrightarrow\} \\
& T'_{DL}\left(\left(\langle p?\rangle\, q \rightarrow p \wedge q\right) \;\wedge\; \left(\langle p?\rangle\, q \leftarrow p \wedge q\right)\right) \\
= \;& \{\text{by defs. } \rightarrow,\ \leftarrow \text{ and } T'_{DL}\} \\
& \left(1\text{'}_{\mathsf{U}};\overline{T_P\,(p?)\,;T'_{DL}\,(q)} \;+\; \left(T'_{DL}\,(p)\cdot T'_{DL}\,(q)\right)\right) \\
& \quad\cdot\left(1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)\cdot T'_{DL}\,(q)} \;+\; \left(T_P\,(p?)\,;T'_{DL}\,(q)\right)\right) \\
= \;& \{\text{by def. } T_P\} \\
& \left(1\text{'}_{\mathsf{U}};\overline{(T'_{DL}\,(p)\cdot 1\text{'}_{\mathsf{U}});T'_{DL}\,(q)} \;+\; \left(T'_{DL}\,(p)\cdot T'_{DL}\,(q)\right)\right) \\
& \quad\cdot\left(1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)\cdot T'_{DL}\,(q)} \;+\; \left((T'_{DL}\,(p)\cdot 1\text{'}_{\mathsf{U}});T'_{DL}\,(q)\right)\right) \\
= \;& \{\text{by } T'_{DL}\,(p) \text{ right-ideal}\} \\
& \left(1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)\cdot T'_{DL}\,(q)} \;+\; \left(T'_{DL}\,(p)\cdot T'_{DL}\,(q)\right)\right) \\
& \quad\cdot\left(1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)\cdot T'_{DL}\,(q)} \;+\; \left(T'_{DL}\,(p)\cdot T'_{DL}\,(q)\right)\right) \\
= \;& \{\text{by Ax. } 2\} \\
& 1\text{'}_{\mathsf{U}};\left(\overline{T_P\,(p?)\,;T'_{DL}\,(q)} \;+\; \left(T_P\,(p?)\,;T'_{DL}\,(q)\right)\right) \\
& \quad\cdot\; 1\text{'}_{\mathsf{U}};\left(\overline{T_P\,(p?)\,;T'_{DL}\,(q)} \;+\; \left(T_P\,(p?)\,;T'_{DL}\,(q)\right)\right) \\
= \;& \{\text{by } \mathsf{BA}\} \\
& {}_{\mathsf{U}}1\cdot{}_{\mathsf{U}}1 \\
= \;& \{\text{by } \mathsf{BA}\} \\
& {}_{\mathsf{U}}1.
\end{aligned}$$

(A7)

$$\begin{aligned}
& T'_{DL}\left([\alpha^*]\,(p \rightarrow [\alpha]p) \rightarrow (p \rightarrow [\alpha^*]p)\right) \\
& \quad = 1\text{'}_{\mathsf{U}};\overline{1\text{'}_{\mathsf{U}};\overline{T_P\,(\alpha^*)\,;\overline{1\text{'}_{\mathsf{U}};1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)}+ 1\text{'}_{\mathsf{U}};\overline{T_P\,(\alpha)\,;1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)}}}}} \\
& \qquad + 1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)} + 1\text{'}_{\mathsf{U}};\overline{T_P\,(\alpha^*)\,;1\text{'}_{\mathsf{U}};\overline{\overline{T'_{DL}\,(p)}}} \\
& \quad = 1\text{'}_{\mathsf{U}};T_P\,(\alpha^*)\,;\overline{1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)} + 1\text{'}_{\mathsf{U}};\overline{T_P\,(\alpha)\,;\overline{\overline{T'_{DL}\,(p)}}}} \\
& \qquad + 1\text{'}_{\mathsf{U}};\overline{T'_{DL}\,(p)} + 1\text{'}_{\mathsf{U}};\overline{T_P\,(\alpha^*)\,;\overline{\overline{T'_{DL}\,(p)}}} && \text{(by Thm. 2.3.19)} \\
& \quad = T_P\,(\alpha^*)\,;\left(T'_{DL}(p)\;\cdot\;T_P\,(\alpha)\,;\overline{T'_{DL}(p)}\right) \\
& \qquad + 1\text{'}_{\mathsf{U}};\overline{T'_{DL}(p)} + 1\text{'}_{\mathsf{U}};\overline{T_P\,(\alpha^*)\,;\overline{\overline{T'_{DL}(p)}}}. && \text{(by Thm. 2.3.19)}
\end{aligned}$$

Now,

$$T_P(\alpha^*) ; \left(T'_{DL}(p) \cdot T_P(\alpha) ; \overline{T'_{DL}(p)}\right) + 1'_U ; \overline{T'_{DL}(p)} + \overline{1'_U ; T_P(\alpha^*) ; \overline{T'_{DL}(p)}} = {}_U1$$

iff (by elementary Boolean algebra)

$$T_P(\alpha^*) ; \overline{T'_{DL}(p)} \leq T_P(\alpha^*) ; \left(T'_{DL}(p) \cdot T_P(\alpha) ; \overline{T'_{DL}(p)}\right) + 1'_U ; \overline{T'_{DL}(p)}$$

iff (by properties of right-ideal relations)

$$T_P(\alpha^*) ; \overline{T'_{DL}(p) ;1;1} \leq T_P(\alpha^*) ; \left(T'_{DL}(p) ;1;1 \cdot T_P(\alpha) ; \overline{T'_{DL}(p) ;1;1}\right) + 1'_U ; \overline{T'_{DL}(p) ;1;1}$$

iff (by properties of right-ideal relations)

$$T_P(\alpha^*) ; \overline{T'_{DL}(p) ;1};1 \leq T_P(\alpha^*) ; \left(\overline{\overline{T'_{DL}(p) ;1};1} \cdot T_P(\alpha) ; \overline{T'_{DL}(p) ;1};1\right) + 1'_U ; \overline{T'_{DL}(p) ;1};1.$$

If we call q the term $\overline{T'_{DL}(p) ;1}$, the last equation is equivalent to

$$T_P(\alpha)^* ;q;1 \leq T_P(\alpha)^* ; \left(\overline{q;1} \cdot T_P(\alpha) ;q;1\right) + q;1,$$

which is an instance of Ax. 12 in Def. 6.24.
(A8) That $T'_{DL}([\alpha](p \rightarrow q) \rightarrow ([\alpha]p \rightarrow [\alpha]q)) = {}_U1$, follows from the proof of Thm. 6.8.

If the proof has length greater than 1, then φ was obtained by applying either modus ponens or generalization. That these rules preserve provability in fork algebras was already proved in Thm. 6.8.
$\Leftarrow$) Let us assume that $\vdash_{CFL} T_{DL}(\varphi)$, but $\not\vdash_{DL} \varphi$. Then there exists a dynamic model $\mathfrak{M} = (W, \tau, \delta)$ in which φ does not hold. By applying Lemma 6.3, from $\mathfrak{M}$ we can construct a closure fork model $\mathcal{F} = \langle \mathfrak{A}, m \rangle$ satisfying:

$$\mathsf{dom}\,(m\,(T_{DL}\,(\varphi))) \quad = \quad \tau(\varphi)\ .$$

Since φ is not valid in $\mathfrak{M}$, must be $\tau(\varphi) \neq W$, and thus, there exists an element from W which is not in $\mathsf{dom}\,(m\,(T_{DL}(\varphi)))$. Then, in $\mathfrak{A}$, the equation $T_{DL}(\varphi) = 1$ does not hold, and thus it is not provable, which leads to a contradiction. □

6.8 The Fork Logic FL'

The logic FL' (to be used in the remaining part of this chapter) is related both to FL and to ETFR, as follows from the definitions to be presented next.

6.8.1 *Syntax of FL'*

We will assume that there are infinite disjoint sets UreVar and $\mathit{CompVar}$ such that $\mathit{IndVar} = \mathit{UreVar} \cup \mathit{CompVar}$. Intuitively, variables from UreVar will range over urelements, while those in $\mathit{CompVar}$ will range over arbitrary elements in the base of fork algebras.

Given a set of constant relation symbols P, we define the set of formulas of the logic FL' (denoted by $\mathit{ForkFor}(P)$) as the set

$$\{\, t_1 R t_2 : t_1, t_2 \in \mathit{IndTerm}(\emptyset, \emptyset)^\star \text{ and } R \in \mathit{RelDes}(P) \,\}\,.$$

Notice that any set P of constant relation symbols determines a language. These languages will be referred to as fork languages. We will denote the fork language on the set of constant relation symbols P by $\mathcal{L}(P)$.

Notice also that the formulas in $\mathcal{L}(P)$ correspond to a subset of the atomic formulas in $\mathit{ForETFR}(\emptyset, \emptyset, P)$ (cf. Def. 5.6).

6.8.2 *Semantics of FL'*

Because of the relationship between $\mathcal{L}(P)$ and $\mathit{ForETFR}(\emptyset, \emptyset, P)$, the semantics of FL' is naturally defined. For example, an adequate structure for $\mathcal{L}(P)$ will be an adequate structure for ETFR$(\emptyset, \emptyset, P)$ as defined in Def. 5.11. In a similar way, the notion of model follows Def. 5.14. Since the language of FL' is simpler than the language of $\mathit{ForETFR}(\emptyset, \emptyset, P)$, we will present a simplified version of the definition of ETFR model that we will use in the remaining part of the chapter.

Definition 6.27 A fork model for a language $\mathcal{L}(X)$ is a structure $\mathcal{F} = \langle \mathfrak{A}, m \rangle$ such that

(1) $\mathfrak{A} \in$ SPFAU,
(2) $m : \mathit{RelVar} \cup X \to A$ is the meaning function, and
(3) $m(1') = \mathit{Id}$.

Clearly m extends homomorphically to a function $m' : RelDes(X) \to A$. For the sake of simplicity we will denote both m and m' by m. Notice that in particular $m(0) = \emptyset$ and $m(1) = V$, the greatest relation of the fork algebra $\mathfrak{A}$. Notice that the class of FL' models corresponds to SPFM.

The notion of valuation differs from Def. 5.12, though. This is due to the partition of *IndVar* into the sets *UreVar* and *CompVar*.

Definition 6.28 Given $\mathcal{F} = \langle \mathfrak{A}, m \rangle \in$ SPFM, a valuation over $\mathcal{F}$ is a mapping $\nu : IndVar \to U_{\mathfrak{A}}$ satisfying

(1) $\nu(x) \in Urel_{\mathfrak{A}}$ if $x \in UreVar$,
(2) $\nu(x) \in U_{\mathfrak{A}}$ if $x \in CompVar$.

Every valuation ν extends homomorphically to a mapping $\nu' : IndTerm \to U_{\mathfrak{A}}$. We will denote both ν and ν' by ν.

The notion of satisfiability of a formula by a valuation is just a simplification of Def. 5.15

Definition 6.29 A fork formula $t_1 R t_2$ is satisfied in $\mathcal{F} = \langle \mathfrak{A}, m \rangle \in$ SPFM by a valuation ν (denoted by $\mathcal{F}, \nu \models_{FL'} t_1 R t_2$) if $\langle V_\nu(t_1), V_\nu(t_2) \rangle \in m(R)$.

Definition 6.30 A fork formula $t_1 R t_2$ is true in $\mathcal{F} \in$ SPFM (denoted by $\mathcal{F} \models_{FL'} t_1 R t_2$) if for every valuation ν, $\mathcal{F}, \nu \models_{FL'} t_1 R t_2$.

Definition 6.31 A fork formula $t_1 R t_2$ is valid in FL' (denoted by $\models_{FL'} t_1 R t_2$) if it is true in every $\mathcal{F} \in$ SPFM.

This notion of validity extends in a natural way to sequences of formulas $\gamma_1, \gamma_2, \ldots, \gamma_k$.

Definition 6.32 A sequence of formulas $\gamma_1, \gamma_2, \ldots, \gamma_k$ is valid if for every fork model $\mathcal{F}$ and every valuation ν over $\mathcal{F}$, there exists i, $1 \leq i \leq k$, such that $\mathcal{F}, \nu \models_{FL'} \gamma_i$.

Finally, given sequences of formulas $\Gamma_1, \ldots, \Gamma_n$, we define:

Definition 6.33 The family of sequences of formulas $(\Gamma_i)_{1 \leq i \leq n}$ is valid if for all i, $1 \leq i \leq n$, the sequence Γ_i is valid.

6.9 A Rasiowa–Sikorski Calculus for *FL'*

The original Rasiowa–Sikorski proof system presented in [H. Rasiowa et al. (1963)] refers to the classical predicate logic. The system is designed for verification of validity of formulas of this logic. It consists of a pair of rules for each propositional connective and each quantifier. Every pair of rules, in turn, consists of a 'positive' rule and a 'negative' rule. A positive (resp. negative) rule exhibits the logical behavior of the underlying connective or quantifier (negated connective or negated quantifier). For example, the rules for conjunction are the following.

$$\frac{\Gamma, \alpha \wedge \beta, \Delta}{\Gamma, \alpha, \Delta \quad \Gamma, \beta, \Delta} \ (P\wedge)$$

$$\frac{\Gamma, \neg(\alpha \wedge \beta), \Delta}{\Gamma, \neg\alpha, \neg\beta, \Delta} \ (N\wedge)$$

The system operates in a top-down manner. Application of a rule results in the decomposition of a given formula into the formulas that are the arguments of a respective connective or quantifier. In general, the rules apply to finite sequences of formulas. To apply a rule we choose a formula in a sequence that is to be decomposed and we replace it by its components, thus obtaining either a single new sequence (for 'or'-like connectives) or a pair of sequences (for 'and'-like connectives). In the process of decomposition we form a tree whose nodes consist of finite sequences of formulas. We stop applying rules to the formulas in a node after obtaining an axiom sequence (appropriately defined) or when none of the rules is applicable to the formulas in this node. If the decomposition tree of a given formula is finite, then its validity can be syntactically recognized from the form of the sequences appearing in the leaves of the tree. In the present section we define a Rasiowa-Sikorski style system for the fork logic *FL'*. In [R. Maddux (1983)] Maddux presented a sequent calculus for relation algebras. The system we present here is an extension of the proof system presented in Orlowska [E. Orlowska (1988); E. Orlowska (1995)]. The system consists of a positive and a negative decomposition rule for each relational operation from the language of fork logic, and also of specific rules that reflect properties of the injective function $\star$ and the relational constant 1'. This calculus can be considered a proof system for fork algebras in the sense that whenever we want to prove an equation $R = 1$, it suffices to prove in the calculus the formula xRy.

6.9.1 *The Deduction System for FL′*

In this subsection we will present the rules of the sequent calculus *FLC* for the fork logic *FL′*. Since we are dealing with fork algebras with urelements (required in order to interpret first-order theories), the calculus we present is more involved than a calculus for fork algebras when no assumption is done on the existence of urelements.

$$\frac{\Gamma, xR{+}Sy, \Delta}{\Gamma, xRy, xSy, \Delta}\ (P+) \qquad \frac{\Gamma, x\overline{R{+}S}y, \Delta}{\Gamma, x\overline{R}y, \Delta \quad \Gamma, x\overline{S}y, \Delta}\ (N+)$$

$$\frac{\Gamma, xR{\cdot}Sy, \Delta}{\Gamma, xRy, \Delta \quad \Gamma, xSy, \Delta}\ (P\cdot) \qquad \frac{\Gamma, x\overline{R{\cdot}S}y, \Delta}{\Gamma, x\overline{R}y, x\overline{S}y, \Delta}\ (N\cdot)$$

$$\frac{\Gamma, xR;Sy, \Delta}{\Gamma, xRz, \Delta, xR;Sy \quad \Gamma, zSy, \Delta, xR;Sy}\ (P;) \quad \frac{\Gamma, x\overline{R;S}y, \Delta}{\Gamma, x\overline{R}z_1, z_1\overline{S}y, \Delta \quad \Gamma, x\overline{R}z_2, z_2\overline{S}y, \Delta}\ (N;)$$

$$\frac{\Gamma, x\overline{\overline{R}}y, \Delta}{\Gamma, xRy, \Delta}\ (N^-)$$

$$\frac{\Gamma, x\breve{R}y, \Delta}{\Gamma, yRx, \Delta}\ (P^{\smile}) \qquad \frac{\Gamma, x\overline{\breve{R}}y, \Delta}{\Gamma, y\overline{R}x, \Delta}\ (N^{\smile})$$

$$\frac{\Gamma, xR\nabla Sy, \Delta}{\Gamma, y1\text{'}u \star v, \Delta, xR\nabla Sy \quad \Gamma, xRu, \Delta, xR\nabla Sy \quad \Gamma, xSv, \Delta, xR\nabla Sy}\ (P\nabla)$$

$$\frac{\Gamma, x\overline{R\nabla S}y, \Delta}{\begin{array}{cc}\Gamma, y0\text{'}u_1 \star v_1, x\overline{R}u_1, x\overline{S}v_1, \Delta & \Gamma, y0\text{'}u_2 \star v_2, x\overline{R}u_2, x\overline{S}v_2, \Delta \\ \Gamma, y0\text{'}u_3 \star v_3, x\overline{R}u_3, x\overline{S}v_3, \Delta & \Gamma, y0\text{'}u_4 \star v_4, x\overline{R}u_4, x\overline{S}v_4, \Delta\end{array}}\ (N\nabla)$$

$$\frac{\Gamma, x_1 \star x_2 1\text{'}y_1 \star y_2, \Delta}{\Gamma, x_1 1\text{'}y_1, \Delta \quad \Gamma, x_2 1\text{'}y_2, \Delta}\ (P1\text{'}) \qquad \frac{\Gamma, x_1 \star x_2 0\text{'}y_1 \star y_2, \Delta}{\Gamma, x_1 0\text{'}y_1, x_2 0\text{'}y_2, \Delta}\ (N1\text{'})$$

$$\frac{\Gamma, xRy, \Delta}{\Gamma, x1\text{'}z, xRy, \Delta \quad \Gamma, zRy, xRy, \Delta}\ (1\text{'}_a)$$

$$\frac{\Gamma, xRy, \Delta}{\Gamma, xRz, xRy, \Delta \quad \Gamma, z1\text{'}y, xRy, \Delta}\ (1\text{'}_b)$$

$$\frac{\Gamma, x1\text{'}y, \Delta}{\Gamma, y1\text{'}x, \Delta, x1\text{'}y}\ (Sym)$$

$$\frac{\Gamma, x1\text{'}y, \Delta}{\Gamma, x1\text{'}z, \Delta, x1\text{'}y \quad \Gamma, z1\text{'}y, \Delta, x1\text{'}y}\ (Trans)$$

$$\frac{\Gamma, x1\text{'}y, \Delta}{\Gamma, x \star u1\text{'}y \star v, \Delta, x1\text{'}y \quad \Gamma, x \star u0\text{'}y \star v, \Delta, x1\text{'}y}\ (Cut)$$

$$\frac{\Gamma}{\Gamma, x1\text{'}y}\ (U)$$

In rule $(P;)$, $z \in \mathit{IndTerm}$[†] is arbitrary. In rule $(N;)$, $z_1 \in \mathit{UreVar}$ and $z_2 \in \mathit{CompVar}$. In rule $(P\nabla)$, $u, v \in \mathit{IndTerm}$ are arbitrary. In rule $(N\nabla)$, $u_1, u_2, v_1, v_3 \in \mathit{UreVar}$ and $u_3, u_4, v_2, v_4 \in \mathit{CompVar}$. In rules (1'_a) and (1'_b), $z \in \mathit{IndVar}$ is arbitrary. In rule (Trans), $z \in \mathit{IndTerm}$ is arbitrary. In rule (Cut), $u, v \in \mathit{IndTerm}$ are arbitrary. Finally, in rule (U), $x \in \mathit{UreVar}$ and $y \in \mathit{IndTerm} \setminus \mathit{UreVar}$.

Definition 6.34 A fork formula $t_1 R t_2$ is called *indecomposable* if it satisfies either of the following conditions.

(1) $R \in \mathit{RelVar} \cup \mathit{RelConst}$,
(2) $R = \overline{S}$ and $S \in \mathit{RelVar} \cup \mathit{RelConst}$,
(3) $R \in \{1\text{'}, 0\text{'}\}$.

Definition 6.35 A sequence of formulas Γ is called *indecomposable* if all the formulas in Γ are indecomposable.

Definition 6.36 A sequence of formulas Γ is called *fundamental* if either of the following is true.

(1) Γ contains simultaneously the formulas $t_1 R t_2$ and $t_1 \overline{R} t_2$, for some $t_1, t_2 \in \mathit{IndTerm}$ and $R \in \mathit{RelDes}$.
(2) Γ contains the formula $t\, 1\text{'}\, t$ for some $t \in \mathit{IndTerm}$.

Definition 6.37 Let T be a tree satisfying:

(1) Each node contains a finite sequence of fork formulas.
(2) If the sequences of fork formulas $\Delta_1, \ldots, \Delta_k$ are the immediate successors of the sequence of fork formulas Γ, then there exists an instance of a rule from FLC of form

$$\frac{\Gamma}{\Delta_1 \quad \Delta_2 \quad \cdots \quad \Delta_k}.$$

Then, T is a *proof tree.*

A branch in a proof tree is called *closed* if it ends in a fundamental sequence.

Definition 6.38 A formula $t_1 R t_2$ is *provable* in the calculus FLC iff there exists a proof tree T satisfying:

[†]In order to simplify the notation, from here on we will refer to the set $\mathit{IndTerm}(\emptyset, \emptyset)^\star$ by $\mathit{IndTerm}$.

(1) T is finite,
(2) $t_1 R t_2$ is the root of T,
(3) Each leaf of T contains a fundamental sequence.

6.9.2 *Soundness and Completeness of the Calculus FLC*

Theorem 6.12 *The calculus FLC is sound with respect to FL′.*

Proof The proof proceeds in two steps. First, we prove that for any rule the upper sequence of the rule is valid if and only if all the lower sequences are valid. This property of the rules will be referred to as their admissibility. Once the first step is established, the second step is an induction on the structure of the proof tree as follows:

(1) If the tree has height 1 (i.e., the root is a fundamental sequence), then it is trivially valid.
(2) Assume that if the tree has height less than or equal to n, then the fact that all leaves contain fundamental sequences implies that the sequence in the root is valid.
(3) Let T be a tree with height $n + 1$. If the transition from the root to the nodes in the first level was obtained applying a rule $\mathcal{R}$ of the form

$$\frac{\Gamma}{\Gamma_1 \;\; \Gamma_2 \;\; \cdots \;\; \Gamma_k},$$

let us call T_i $(1 \leq i \leq k)$ the subtree of T with root Γ_i. Since for all i the height of T_i is less or equal than n and all the leaves contain fundamental sequences, the root of each T_i must contain a valid sequence. Since rules preserve validity in both directions, then the sequence Γ must be valid, as was to be proved.

Let us show as an example that the rule $(P\nabla)$ is admissible. The admissibility of the remaining rules is proved in a similar way.

Let us consider a sequence of fork formulas $\Gamma, t_1 R\nabla S t_2, \Delta$ from a language $\mathcal{L}(X)$. Let $\mathcal{M} = \langle \mathfrak{A}, m \rangle$ be a fork model, and let ν be a valuation over $\mathcal{M}$.

If $\mathcal{M}, \nu \models_{FL'} \Gamma, t_1 R\nabla S t_2, \Delta$, then the following three possibilities arise:

(1) $\mathcal{M}, \nu \models_{FL'} \gamma$, with $\gamma \in \Gamma$,
(2) $\mathcal{M}, \nu \models_{FL'} \delta$, with $\delta \in \Delta$,

(3) $\mathcal{M}, \nu \models_{FL'} t_1 R \nabla S t_2$.

If (1) or (2) are true, then it is immediate that the three sequences in the lower part of the rule $(P\nabla)$ are satisfied in the fork model $\mathcal{M}$ by the valuation ν. If (3) is true, then, since the fork formula $t_1 R \nabla S t_2$ is repeated in the three sequences in the lower part of the rule, then these sequences are also satisfied in the fork model $\mathcal{M}$ by the valuation ν.

On the other hand, if

- $\mathcal{M}, \nu \models_{FL'} \Gamma, t_2 1\text{'} u \star v, \Delta, t_1 R \nabla S t_2$,
- $\mathcal{M}, \nu \models_{FL'} \Gamma, t_1 R u, \Delta, t_1 R \nabla S t_2$, and
- $\mathcal{M}, \nu \models_{FL'} \Gamma, t_1 S v, \Delta, t_1 R \nabla S t_2$,

then the following four possibilities arise:

(1) $\mathcal{M}, \nu \models_{FL'} \gamma$ with $\gamma \in \Gamma$,
(2) $\mathcal{M}, \nu \models_{FL'} \delta$ with $\delta \in \Delta$,
(3) $\mathcal{M}, \nu \models_{FL'} t_1 R \nabla S t_2$,
(4) $\mathcal{M}, \nu \models_{FL'} t_2 1\text{'} u \star v$, $\mathfrak{M}, \nu \models_{FL'} t_1 R u$, and $\mathfrak{M}, \nu \models_{FL'} t_1 S v$.

If (1), (2) or (3) are true then clearly the sequence of fork formulas $\Gamma, t_1 R \nabla S t_2, \Delta$ is satisfied in the fork model $\mathcal{M}$ by the valuation ν. If (4) is true, then, by definition of fork (Def. 3.1), $\mathcal{M}, \nu \models_{FL'} t_1 R \nabla S t_2$ and thus, $\mathcal{M}, \nu \models_{FL'} \Gamma, t_1 R \nabla S t_2, \Delta$. □

Definition 6.39 A proof tree T of a sequence of formulas Γ is called *saturated* if, intuitively, all the applicable rules were applied in the open branches. Formally speaking, a proof tree of Γ is called saturated if for every open branch B, the following conditions are satisfied:

(1) If $xR{+}Sy \in B$, then both $xRy \in B$ and $xSy \in B$ by an application of rule $(P+)$.
(2) If $x\overline{R{+}S}y \in B$, then either $x\overline{R}y \in B$ or $x\overline{S}y \in B$ by an application of rule $(N+)$.
(3) If $xR{\cdot}Sy \in B$, then either $xRy \in B$ or $xSy \in B$ by an application of rule $(P\cdot)$.
(4) If $x\overline{R{\cdot}S}y \in B$, then both $x\overline{R}y \in B$ and $x\overline{S}y \in B$ by an application of rule $(N\cdot)$.
(5) If $x\overline{\overline{R}}y \in B$, then $xRy \in B$ by an application of rule (N^{-}).
(6) If $xR;Sy \in B$, then for all $t \in$ *IndTerm*, either $xRt \in B$ or $tSy \in B$ by an application of rule $(P;)$.

(7) If $x\overline{R;S}y \in B$, then for some $z \in \mathit{IndVar}$ both $x\overline{R}z \in B$ and $z\overline{S}y \in B$ by an application of rule $(N;)$.
(8) If $x\breve{R}y \in B$, then $yRx \in B$ by an application of rule $(P^{\smile})$.
(9) If $x\overline{\breve{R}}y \in B$, then $y\overline{R}x \in B$ by an application of rule $(N^{\smile})$.
(10) If $x \star y1\text{'}u \star v \in B$, then either $x1\text{'}u \in B$ or $y1\text{'}v \in B$ by an application of rule $(P1\text{'})$.
(11) If $x \star y0\text{'}u \star v \in B$ then both $x0\text{'}u \in B$ and $y0\text{'}v \in B$ by an application of rule $(N1\text{'})$.
(12) If $xRy \in B$, then for all $z \in \mathit{IndVar}$ either $x1\text{'}z \in B$ or $zRy \in B$ by an application of rule (1'_a).
(13) If $xRy \in B$, then for all $z \in \mathit{IndVar}$ either $xRz \in B$ or $z1\text{'}y \in B$ by an application of rule (1'_b).
(14) If $x1\text{'}y \in B$, then $y1\text{'}x \in B$ by an application of rule *Sym*.
(15) If $x1\text{'}y \in B$, then for all $z \in \mathit{IndTerm}$ either $x1\text{'}z \in B$ or $z1\text{'}y \in B$ by an application of rule (Trans).
(16) If $x1\text{'}y \in B$, then for all $u, v \in \mathit{IndTerm}$ either $x \star u1\text{'}y \star v \in B$ or $x \star u0\text{'}y \star v \in B$ by an application of rule (Cut).
(17) If $xR\nabla Sy \in B$, then for all $u, v \in \mathit{IndTerm}$ one of the formulas $y1\text{'}u \star v$, xRu or xSv is in B by an application of rule $(P\nabla)$.
(18) If $x\overline{R\nabla S}y \in B$, then there are $u, v \in \mathit{IndVar}$ such that the formulas $y0\text{'}u \star v$, $x\overline{R}u$ and $x\overline{S}v$ are in B by an application of rule $(N\nabla)$.
(19) For all $x \in \mathit{UreVar}$ and $y \in \mathit{IndTerm} \setminus \mathit{UreVar}$, $x1\text{'}y \in B$ by an application of rule (U).

Definition 6.40 Given $X \subseteq \mathit{RelConst}$, we define the order of $R \in \mathit{RelDes}(X)$ (denoted by $o(R)$) by the conditions:

(1) $o(R) = 1$ if $R \in \mathit{RelConst} \cup \mathit{RelVar} \cup \{1\text{'}\}$,
(2) $o(R) = o(S) + 1$ if $R = \overline{S}$ or $R = \breve{S}$,
(3) $o(R) = \max\{o(S), o(T)\} + 1$ if $R = S{+}T$, $R = S{\cdot}T$, $R = S;T$, or $R = S\nabla T$.

Theorem 6.13 *The calculus FLC is complete with respect to FL', i.e., if a formula tRt' is valid in FL', then it is provable in FLC.*

Proof Assume tRt' is not provable in *FLC*. Then, no proof tree exists that provides a proof for tRt'. In particular, no saturated tree with root tRt' provides a proof. Therefore, if T is a saturated tree, there must exist an infinite branch B in T.

Let $\equiv$ be the binary relation on *IndTerm* defined by

$$x \equiv y \qquad \Longleftrightarrow \qquad x 1\text{'} y \notin B \ .$$

Let us prove that $\equiv$ is an equivalence relation.

Since for all $t \in \mathit{IndTerm}$ $t 1\text{'} t \notin B$ (otherwise B would contain a fundamental sequence), $\equiv$ is reflexive.

If $t_1, t_2 \in \mathit{IndTerm}$ satisfy $t_1 \equiv t_2$ (or equivalently $t_1 1\text{'} t_2 \notin B$), then $t_2 \equiv t_1$. Otherwise, if $t_2 1\text{'} t_1 \in B$, then, by application of the rule (*Sym*) $t_1 1\text{'} t_2 \in B$, which is a contradiction.

If $t_1 \equiv t_2$ and $t_2 \equiv t_3$ ($t_1 1\text{'} t_2 \notin B$ and $t_2 1\text{'} t_3 \notin B$), let us show that $t_1 \equiv t_3$. If $t_1 \not\equiv t_3$, then $t_1 1\text{'} t_3 \in B$. Thus, by one application of the rule (*Trans*) either $t_1 1\text{'} t_2 \in B$ or $t_2 1\text{'} t_3 \in B$, which is a contradiction.

Let $\mathfrak{A}$ be the FullPFA with underlying domain $\{ |x| : x \in \mathit{IndTerm} \}$ and pairing function $\star$ defined by $|x| \star |y| = |x \star y|$. If $|x_1| = |x_2|$ and $|y_1| = |y_2|$ then must be $|x_1 \star y_1| = |x_2 \star y_2|$. Otherwise, if $|x_1 \star y_1| \neq |x_2 \star y_2|$, then $x_1 \star y_1 1\text{'} x_2 \star y_2 \in B$. Applying rule ($P1$') either $x_1 1\text{'} x_2 \in B$ or $y_1 1\text{'} y_2 \in B$, which is a contradiction. Then, $\star$ is a well-defined function.

Let us check that $\star$ is injective. If $|t_1| \star |t_2| = |t_3| \star |t_4|$ then, by definition of $\star$, $|t_1 \star t_2| = |t_3 \star t_4|$. Thus, $t_1 \star t_2 1\text{'} t_3 \star t_4 \notin B$. If $|t_1| \neq |t_3|$ then $t_1 1\text{'} t_3 \in B$. Applying the rule (*Cut*) either $t_1 \star t_2 1\text{'} t_3 \star t_4 \in B$ or $t_1 \star t_2 0\text{'} t_3 \star t_4 \in B$. Since $t_1 \star t_2 1\text{'} t_3 \star t_4 \notin B$, then $t_1 \star t_2 0\text{'} t_3 \star t_4 \in B$. Applying rule ($N1$') yields that $t_1 0\text{'} t_3 \in B$, thus B would be a closed branch, which contradicts our assumptions. We then conclude that $|t_1| = |t_3|$. In a similar way we prove that $|t_2| = |t_4|$.

Notice that $\mathit{Urel}_{\mathfrak{A}} = \{ |x| : x \in \mathit{UreVar} \}$, because if $|x| = |t_1| \star |t_2|$ then $|x| = |t_1 \star t_2|$. Then, $x 1\text{'} t_1 \star t_2 \notin B$. By applying rule ($U$) we arrive at a contradiction. Thus, $\mathfrak{A} \in$ SPFAU.

Let us define, for $R \in \mathit{RelVar} \cup \mathit{RelConst}$,

$$\langle |t_1|, |t_2| \rangle \in m(R) \qquad \Longleftrightarrow \qquad t_1 R t_2 \notin B \ .$$

Let us check that m is well defined. Let us see that whenever $t_1 \equiv t_3$ and $t_2 \equiv t_4$, if $\langle |t_1|, |t_2| \rangle \in m(R)$ then $\langle |t_3|, |t_4| \rangle \in m(R)$. Since $\langle |t_1|, |t_2| \rangle \in m(R)$, $t_1 R t_2 \notin B$. If $\langle |t_3|, |t_4| \rangle \notin m(R)$, then $t_3 R t_4 \in B$. Applying rule (1'_a) implies that either $t_3 1\text{'} t_1 \in B$ or $t_1 R t_4 \in B$. If $t_3 1\text{'} t_1 \in B$, applying rule (*Sym*) implies that $t_1 1\text{'} t_3 \in B$. Since $t_1 1\text{'} t_3 \notin B$, then $t_1 R t_4 \in B$. Applying rule (1'_b) implies that either $t_1 R t_2 \in B$ or $t_4 1\text{'} t_2 \in B$. If $t_4 1\text{'} t_2 \in B$, one application of rule (*Sym*) would imply that $t_2 1\text{'} t_4 \in B$, which is not the case. Thus, $t_1 R t_2 \in B$ which also leads to a contradiction. We have then shown that $\langle |t_3|, |t_4| \rangle \in m(R)$.

Therefore the structure $\mathcal{M} = \langle \mathfrak{A}, m \rangle$ belongs to SPFM.

Let ν be the valuation defined by $\nu(x) = |x|$, for $x \in \mathit{IndVar}$. Let us show by induction that $V_\nu(t) = |t|$ for all $t \in \mathit{IndTerm}$. By definition it is true for variables. If $t = t_1 \star t_2$, $V_\nu(t) = V_\nu(t_1 \star t_2) = V_\nu(t_1) \star V_\nu(t_2) = |t_1| \star |t_2| = |t_1 \star t_2|$.

Let us define

$$S = \{ \alpha \in \mathit{ForkFor} : \mathcal{M}, \nu \models_{FL'} \alpha \ \wedge \ \alpha \in B \} .$$

Notice that since $t R t'$ is valid, $\mathcal{M}, \nu \models_{FL'} t R t'$, and thus $S \neq \emptyset$. Then, since the set S is well-ordered by o, by Zorn's lemma S has a minimum element α'.

Notice that α' cannot have the shape $t_1 1' t_2$ because, since $\alpha' \in S$, $\mathfrak{A}, \nu \models_{FL'} t_1 1' t_2$. Then, it must be $|t_1| = |t_2|$, which implies $t_1 1' t_2 \notin B$, a contradiction.

Notice also that α' cannot have any of the following shapes:

$$\begin{array}{ccc} t_1 \overline{\overline{R}} t_2, & t_1 \breve{R} t_2, & t_1 \overline{\breve{R}} t_2, \\ t_1 R{+}S t_2, & t_1 \overline{R{+}S} t_2, & t_1 R{\cdot}S t_2, \\ t_1 \overline{R{\cdot}S} t_2, & t_1 \overline{R;S} t_2, & t_1 \overline{R \nabla S} t_2, \end{array}$$

because in any of this cases a formula α'' appears in S satisfying $o(\alpha'') < o(\alpha')$, contradicting the minimality of α'.

If $\alpha' = t_1 R;S t_2$, by definition of the saturated tree there exists a level in B in which we have a derivation with shape

$$\frac{\Gamma_1', \alpha', \Gamma_2'}{\Gamma_1', t_1 R z, \Gamma_2', \alpha' \quad \Gamma_1', z S t_2, \Gamma_2', \alpha'} \ (P;)$$

and z satisfies $\mathcal{M}, \nu \models_{FL'} t_1 R z$ and $\mathcal{M}, \nu \models_{FL'} z S t_2$. Therefore, there exists $\alpha'' \in S$ with $o(\alpha'') < o(\alpha')$.

If $\alpha' = t_1 R \nabla S t_2$, by definition of the saturated tree there exists a level in B in which we have a derivation with shape

$$\frac{\Gamma_1', \alpha', \Gamma_2'}{\Gamma_1', t_2 1' u \star v, \Gamma_2', \alpha' \quad \Gamma_1', t_1 R u, \Gamma_2', \alpha' \quad \Gamma_1', t_1 S v, \Gamma_2', \alpha'} \ (P\nabla) ,$$

and u and v satisfy $\mathcal{M}, \nu \models_{FL'} t_1 1' u \star v$, $\mathcal{M}, \nu \models_{FL'} t_1 R u$, and $\mathcal{M}, \nu \models_{FL'} t_2 S v$. Therefore there exists $\alpha'' \in S$ with $o(\alpha'') < o(\alpha')$.

From the previous arguments, it follows that α' must be indecomposable.

If $\alpha' = t_1 Q t_2$ for some $Q \in \mathit{RelVar} \cup \mathit{RelConst}$, and $t_1, t_2 \in \mathit{IndTerm}$, then, since $\mathcal{M}, \nu \models_{FL'} \alpha'$, $\langle |t_1|, |t_2| \rangle \in m(Q)$, but this is so if and only if (by definition of m), $t_1 Q t_2 \notin B$, which leads to a contradiction.

If $\alpha' = t_1 \overline{Q} t_2$ for some $Q \in \mathit{RelVar} \cup \mathit{RelConst}$, and $t_1, t_2 \in \mathit{IndTerm}$, then, since $\mathcal{M}, \nu \models_{FL'} \alpha'$, $\langle |t_1|, |t_2| \rangle \notin m(Q)$ or, equivalently, $t_1 Q t_2 \in B$. Since $t_1 \overline{Q} t_2 \in B$ too, B is closed, which is a contradiction.

If $\alpha' = t_1 0' t_2$ for $t_1, t_2 \in \mathit{IndTerm}$, then, since $\mathcal{A}, \nu \models_{FL'} \alpha'$, $\langle |t_1|, |t_2| \rangle \in m(0')$, and thus $|t_1| \neq |t_2|$. This implies that $t_1 1' t_2 \in B$ and also that B is closed, which is a contradiction. □

6.9.3 *Examples of Proofs in the Calculus FLC*

As an exercise let us show that some valid properties of fork algebras are provable in the calculus *FLC*. As a general practice we will sometimes omit some formulas when passing from a level to the level below, provided the formulas are not required to obtain the fundamental sequences. This will not affect the soundness of the calculus, and will simplify reading the proofs.

Let us prove that $(R \nabla S)\,;(T \nabla Q)^{\smile} \leq R;\breve{T} \cdot S;\breve{Q}$. In order to start the derivation, we need first to convert the formula into an equation of the form $t = 1$. Notice that in general, $R \leq S \iff \overline{R} + S = 1$. Then,

$$\dfrac{\dfrac{x\overline{(R\nabla S)\ ;\ (T\nabla Q)^{\smile}} + R;\breve{T}\cdot S;\breve{Q}y}{x\overline{(R\nabla S)\ ;\ (T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y}\ (P+)}{\underbrace{x\overline{R\nabla S}z_1, z_1\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y}_{\Sigma_1}\quad \underbrace{x\overline{R\nabla S}z_2, z_2\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y}_{\Sigma_2}}\ (N;)$$

In the sequence Σ_1, $z_1 \in \mathit{UreVar}$, while in Σ_2, $z_2 \in \mathit{CompVar}$. Let us analyze each sequence.

If we apply the rule $(N\nabla)$ on sequence Σ_1, then we obtain the following four sequences

(1) $z_1 0' u_1 \star v_1, x\overline{R}u_1, x\overline{S}v_1, z_1\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y$, with u_1 and v_1 from *UreVar*,

(2) $z_1 0' u_2 \star v_2, x\overline{R}u_2, x\overline{S}v_2, z_1\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y$, with u_2 from *UreVar* and v_2 from *CompVar*,

(3) $z_1\,0'\,u_3 \star v_3, x\overline{R}u_3, x\overline{S}v_3, z_1\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y$, with u_3 from *CompVar* and v_3 from *UreVar*,

(4) $z_1\,0'\,u_4 \star v_4, x\overline{R}u_4, x\overline{S}v_4, z_1\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y$, with the individual variables u_4 and v_4 from *CompVar*.

Any of the four branches is closed by applying the rule (U) once in each branch, adding the fork formula $z_1 1'\, u_i \star v_i$, $1 \le i \le 4$.

Regarding branch Σ_2, applying rule $(N\nabla)$ we obtain (as with Σ_1) the following four sequences

(1) $z_2\,0'\,u_1 \star v_1, x\overline{R}u_1, x\overline{S}v_1, z_2\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y$, with u_1 and v_1 from *UreVar*,

(2) $z_2\,0'\,u_2 \star v_2, x\overline{R}u_2, x\overline{S}v_2, z_2\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y$, with u_2 from *UreVar* and v_2 from *CompVar*,

(3) $z_2\,0'\,u_3 \star v_3, x\overline{R}u_3, x\overline{S}v_3, z_2\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y$, with u_3 from *CompVar* and v_3 from *UreVar*,

(4) $z_2\,0'\,u_4 \star v_4, x\overline{R}u_4, x\overline{S}v_4, z_2\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y$, with the individual variables u_4 and v_4 from *CompVar*.

We then proceed in the same way with the four branches, as follows.

$$\cfrac{\cfrac{z_2\,0'\,u_i \star v_i, x\overline{R}u_i, x\overline{S}v_i, z_2\overline{(T\nabla Q)^{\smile}}y, xR;\breve{T}\cdot S;\breve{Q}y\ (N^{\smile})}{z_2\,0'\,u_i \star v_i, x\overline{R}u_i, x\overline{S}v_i, y\overline{T\nabla Q}z_2, xR;\breve{T}\cdot S;\breve{Q}y}\quad (1'_b)}{\underbrace{z_2\,0'\,u_i \star v_i, x\overline{R}u_i, x\overline{S}v_i, y\overline{T\nabla Q}u_i \star v_i, y\overline{T\nabla Q}z_2, xR;\breve{T}\cdot S;\breve{Q}y}_{\Sigma_3}\quad \underbrace{z_2\,0'\,u_i \star v_i, u_i \star v_i\,1'\,z_2}_{\Sigma_4}}$$

Regarding branch Σ_4, we have

$$\frac{z_2\,0'\,u_i \star v_i, u_i \star v_i\,1'\,z_2}{z_2\,0'\,u_i \star v_i, z_2\,1'\,u_i \star v_i}\ (Sym)$$

The last sequence is clearly fundamental, and thus the branch is closed. Regarding branch Σ_3, we proceed as follows.

$$\cfrac{\cfrac{z_2\,0'\,u_i \star v_i, x\overline{R}u_i, x\overline{S}v_i, y\overline{T\nabla Q}u_i \star v_i, y\overline{T\nabla Q}z_2, xR;\breve{T}\cdot S;\breve{Q}y\ (N\nabla)}{x\overline{R}u_i, x\overline{S}v_i, u_i \star v_i\,0'\,r_j \star s_j, y\overline{T}r_j, y\overline{Q}s_j, xR;\breve{T}\cdot S;\breve{Q}y}\quad (P\cdot)}{\underbrace{x\overline{R}u_i, u \star v_i\,0'\,r_j \star s_j, y\overline{T}r_j, xR;\breve{T}y}_{\Sigma_{5,i,j}}\quad \underbrace{x\overline{S}v_i, u_i \star v_i\,0'\,r_j \star s_j, y\overline{Q}s_j, xS;\breve{Q}y}_{\Sigma_{6,i,j}}}$$

Since none of the sequences $\Sigma_{5,i,j}$ or $\Sigma_{6,i,j}$ are closed, we will derive a closed tree for each sequence. For the sequences $\Sigma_{5,i,j}$ we have:

$$\dfrac{x\overline{R}u_i, u_i \star v_i 0' r_j \star s_j, y\overline{T}r_j, xR;\breve{T}y}{\dfrac{x\overline{R}u_i, u_i 0' r_j, v_i 0' s_j, y\overline{T}r_j, xR;\breve{T}y}{x\overline{R}u_i, xRu_i \qquad \dfrac{u_i 0' r_j, y\overline{T}r_j, u_i\breve{T}y}{\dfrac{u_i 0' r_j, y\overline{T}r_j, yTu_i}{y\overline{T}u_i, yTu_i \quad u_i 0' r_j, u_i 1' r_j}\;(1'_b)}\;(P^{\smile})}\;(P;)}\;(N1')$$

Finally, for the sequences $\Sigma_{6,i,j}$ we have:

$$\dfrac{x\overline{S}v_i, u_i \star v_i 0' r_j \star s_j, y\overline{Q}s_j, xS;\breve{Q}y}{\dfrac{x\overline{S}v_i, u_i 0' r_j, v_i 0' s_j, y\overline{Q}s_j, xS;\breve{Q}y}{x\overline{S}v_i, xSv_i \qquad \dfrac{v_i 0' s, y\overline{Q}s, v_i\breve{Q}y}{\dfrac{v_i 0' s, y\overline{Q}s, yQv_i}{y\overline{Q}v_i, yQv_i \quad v_i 0' s_j, v_i 1' s_j}\;(1'_b)}\;(P^{\smile})}\;(P;)}\;(N1')$$

Let us now prove the other inclusion, namely that

$$R;\breve{T} \cdot S;\breve{Q} \leq (R\nabla S)\,;(T\nabla Q)^{\smile} .$$

$$\dfrac{x\overline{R;\breve{T} \cdot S;\breve{Q}} + (R\nabla S)\,;(T\nabla Q)^{\smile}y}{\dfrac{x\overline{R;\breve{T} \cdot S;\breve{Q}}y, x(R\nabla S)\,;(T\nabla Q)^{\smile}y}{\dfrac{x\overline{R;\breve{T}}y, x\overline{S;\breve{Q}}y, x(R\nabla S)\,;(T\nabla Q)^{\smile}y}{\underbrace{x\overline{R}u_1, u_1\overline{\breve{T}}y, x\overline{S;\breve{Q}}y, x(R\nabla S)\,;(T\nabla Q)^{\smile}y}_{\Theta_1} \quad \underbrace{x\overline{R}u_2, u_2\overline{\breve{T}}y, x\overline{S;\breve{Q}}y, x(R\nabla S)\,;(T\nabla Q)^{\smile}y}_{\Theta_2}}\;(N;)}\;(N\cdot)}\;(P+)$$

In the sequence Θ_1, $u_1 \in$ *UreVar* and $u_2 \in$ *CompVar*, We will proceed the derivation with Θ_1, since the same steps can be applied indistinctly to Θ_2.

$$\dfrac{x\overline{R}u_1, u_1\overline{\breve{T}}y, x\overline{S;\breve{Q}}y, x(R\nabla S);(T\nabla Q)^{\smile}y}{\underbrace{x\overline{R}u_1, u_1\overline{\breve{T}}y, x\overline{S}v_1, v_1\overline{\breve{Q}}y, x(R\nabla S);(T\nabla Q)^{\smile}y}_{\Theta_3} \quad \underbrace{x\overline{R}u_1, u_1\overline{\breve{T}}y, x\overline{S}v_2, v_2\overline{\breve{Q}}y, x(R\nabla S);(T\nabla Q)^{\smile}y}_{\Theta_4}}\;(N;)$$

In sequences Θ_3 and Θ_4, $v_1 \in$ *UreVar* and $v_2 \in$ *CompVar*. We will proceed the derivation with Θ_3, although the same steps apply to sequence Θ_4.

$$\dfrac{\dfrac{\dfrac{x\overline{R}u_1, u_1\overline{\breve{T}}y, x\overline{S}v_1, v_1\overline{\breve{Q}}y, x(R\nabla S)\,;(T\nabla Q)^{\smile}y \ (N^{\smile})}{x\overline{R}u_1, y\overline{T}u_1, x\overline{S}v_1, v_1\overline{\breve{Q}}y, x(R\nabla S)\,;(T\nabla Q)^{\smile}y \ (N^{\smile})}}{x\overline{R}u_1, y\overline{T}u_1, x\overline{S}v_1, y\overline{Q}v_1, x(R\nabla S)\,;(T\nabla Q)^{\smile}y}}{\underbrace{x\overline{R}u_1, x\overline{S}v_1, x(R\nabla S)u_1\star v_1}_{\Theta_5} \quad \underbrace{y\overline{T}u_1, y\overline{Q}v_1, u_1\star v_1(T\nabla Q)^{\smile}y}_{\Theta_6}} \ (P;)$$

Since both sequences Θ_5 and Θ_6 are not fundamental, we will proceed with the derivation. For sequence Θ_5 we have:

$$\dfrac{x\overline{R}u_1, x\overline{S}v_1, xR\nabla Su_1\star v_1}{u_1\star v_1 1' u_1\star v_1 \quad x\overline{R}u_1, xRu_1 \quad x\overline{S}v_1, xSv_1} \ (P\nabla)$$

The last sequences are all fundamental.

Finally, for sequence Θ_6 we have:

$$\dfrac{\dfrac{y\overline{T}u_1, y\overline{Q}v_1, u_1\star v_1(T\nabla Q)^{\smile}y \ (P^{\smile})}{y\overline{T}u_1, y\overline{Q}v_1, yT\nabla Qu_1\star v_1}}{u_1\star v_1 1' u_1\star v_1 \quad y\overline{T}u_1, yTu_1 \quad y\overline{Q}v_1, yQv_1} \ (P\nabla)$$

6.10 A Relational Proof System for Intuitionistic Logic

In this section we prove interpretability of intuitionistic logic in the fork logic *FL'* and extend the proof system *FLC* to a relational proof system for intuitionistic logic.

6.10.1 *Intuitionistic Logic*

The syntax and semantics of the intuitionistic logic (*Int*) are defined as follows.

Definition 6.41 The alphabet of *Int* is given by:

(1) an infinite countable set of propositional variables, that will be denoted by *PropVar*,
(2) the set of propositional connectives $\{\neg, \vee, \wedge, \rightarrow\}$, and
(3) the set of auxiliary symbols { "(", ",", ")" }.

Definition 6.42 The set of intuitionistic formulas (denoted by *IntFor*) is the smallest set satisfying

(1) $PropVar \subseteq IntFor$,
(2) If $\alpha, \beta \in IntFor$, then $\{(\neg\alpha), (\alpha \vee \beta), (\alpha \wedge \beta), (\alpha \rightarrow \beta)\} \subseteq IntFor$.

Definition 6.43 An intuitionistic model is a triple $\langle W, R, m \rangle$ in which

(1) $W \neq \emptyset$,
(2) $R \subseteq W \times W$ is a reflexive and transitive relation,
(3) $m : PropVar \rightarrow \mathcal{P}(W)$ satisfies the heredity condition given by:

$$\text{If } wRw' \text{ and } w \in m(p), \text{ then } w' \in m(p) \; .$$

Definition 6.44 Let $\mathfrak{I} = \langle W, R, m \rangle$ be an intuitionistic model. A formula α is satisfied in a world $w \in W$ (denoted by $\mathfrak{I}, w \models_{Int} \alpha$) if the following conditions are satisfied:

$\alpha = p_i \in PropVar$:

$$\mathfrak{I}, w \models_{Int} p_i \quad \text{iff} \quad w \in m(p_i) \; .$$

$\alpha = \neg\beta$:

$$\mathfrak{I}, w \models_{Int} \neg\beta \quad \text{iff} \quad (\forall w' \in W)\,(wRw' \Rightarrow \mathfrak{I}, w' \not\models_{Int} \beta) \; .$$

$\alpha = \beta \vee \gamma$:

$$\mathfrak{I}, w \models_{Int} \beta \vee \gamma \quad \text{iff} \quad \mathfrak{I}, w \models_{Int} \beta \text{ or } \mathfrak{I}, w \models_{Int} \gamma \; .$$

$\alpha = \beta \wedge \gamma$:

$$\mathfrak{I}, w \models_{Int} \beta \wedge \gamma \quad \text{iff} \quad \mathfrak{I}, w \models_{Int} \beta \text{ and } \mathfrak{I}, w \models_{Int} \gamma \; .$$

$\alpha = \beta \rightarrow \gamma$:

$$\mathfrak{I}, w \models_{Int} \beta \rightarrow \gamma \quad \text{iff}$$
$$(\forall w' \in W)\,(wRw' \text{ and } \mathfrak{I}, w' \models_{Int} \beta \text{ implies } \mathfrak{I}, w' \models_{Int} \gamma) \; .$$

Definition 6.45 A formula $\alpha \in IntFor$ is *true in an intuitionistic model* $\mathfrak{I} = \langle W, R, m \rangle$ (denoted by $\mathfrak{I} \models_{Int} \alpha$) if, for all $w \in W$, $\mathfrak{I}, w \models_{Int} \alpha$.

Definition 6.46 A formula is *Int-valid* if it is valid in all intuitionistic models.

6.10.2 *Interpretability of Intuitionistic Logic in FL′*

In this section we will present a mapping $T_I : \mathit{IntFor} \longrightarrow \mathit{RelDes}$ that will allow us to interpret the logic *Int* in the logic *FL′*.

Definition 6.47 Let us have a fork language with one constant symbol R interpreted as an accessibility relation from intuitionistic models. Let us define the recursive mapping $T_I : \mathit{IntFor} \longrightarrow \mathit{RelDes}$ as follows:

(1) $T_I(p_i) = R_i$ with $p_i \in \mathit{PropVar}$ and $R_i \in \mathit{RelVar}$.
(2) $T_I(\neg\alpha) = \overline{R;T_I(\alpha)}$.
(3) $T_I(\alpha \wedge \beta) = T_I(\alpha)\cdot T_I(\beta)$.
(4) $T_I(\alpha \vee \beta) = T_I(\alpha)+T_I(\beta)$.
(5) $T_I(\alpha \rightarrow \beta) = \overline{R;\left(T_I(\alpha)\cdot\overline{T_I(\beta)}\right)}$.

Since the accessibility relation in intuitionistic models satisfies conditions of reflexivity, transitivity and heredity, we will define abstract relational counterparts of these conditions, as follows:

(C1)	: $1'_{\mathsf{U}} \leq R$,	(reflexivity)
(C2)	: $R;R \leq R$,	(transitivity)
(C3)	: $\overline{(R_i\cdot R)\,;\overline{R_i}} = 1$,	(heredity)
(C4)	: $R_i \leq {}_{\mathsf{U}}1$ for all i,	(R_i has urelements in its domain)
(C5)	: $R_i;1 = R_i$ for all i,	(R_i is right-ideal)
(C6)	: $R \leq {}_{\mathsf{U}}1;1_{\mathsf{U}}$.	(R is defined in the set of urelements)

Lemma 6.4 *Let $\mathfrak{I} = \langle W, R', m \rangle$ be an intuitionistic model. Then, there exists $\mathcal{F} = \langle \mathfrak{A}, m' \rangle \in \mathsf{SPFM}$ constructed from $\mathfrak{I}$ satisfying conditions (C1)–(C6) such that for all $w \in W$ and for all $\varphi \in \mathit{IntFor}$*

$$\mathfrak{I}, w \models_{\mathit{Int}} \varphi \qquad \Longleftrightarrow \qquad w \in \mathsf{dom}\left(m'\left(T_I(\varphi)\right)\right).$$

Proof Define $\mathfrak{A}$ as the FullPFAU with set of urelements W, let $m'(R) = R'$, and for each $R_i \in \mathit{RelVar}$ define $m'(R_i) = \{\, \langle x, y\rangle : x \in m(p_i)\,\}$. Conditions (C1)–(C6) hold in $\mathcal{F}$ because of the way $m'(R)$ and $m'(R_i)$ are defined.

The remaining part of the proof proceeds by induction on the structure of the formula φ.

$\varphi = p_i \in PropVar$:

$$\begin{aligned}\mathfrak{I}, w \models_{Int} p_i &\text{ iff } w \in m(p_i)\\ &\text{ iff } w \in \mathsf{dom}\,(m'(R_i))\\ &\text{ iff } w \in \mathsf{dom}\,(m'\,(T_I\,(p_i)))\,.\end{aligned}$$

$\varphi = \neg\alpha$:

$$\begin{aligned}\mathfrak{I}, w \models_{Int} \neg\alpha &\text{ iff } (\forall w' \in W)\,(wR'w' \Rightarrow \mathfrak{I}, w' \not\models_{Int} \alpha)\\ &\text{ iff } (\forall w' \in W)\,(wR'w' \Rightarrow w' \notin \mathsf{dom}\,(m'\,(T_I(\alpha))))\\ &\text{ iff } (\nexists w' \in W)\,(wR'w' \wedge w' \in \mathsf{dom}\,(m'\,(T_I(\alpha))))\\ &\text{ iff } (\nexists w' \in A)\,(\langle w, w'\rangle \in m'(R) \wedge w' \in \mathsf{dom}\,(m'\,(T_I(\alpha))))\\ &\text{ iff } w \in \mathsf{dom}\left(\overline{m'(R);m'\,(T_I(\alpha))}\right)\\ &\text{ iff } w \in \mathsf{dom}\left(m'\left(\overline{R;T_I(\alpha)}\right)\right)\\ &\text{ iff } w \in \mathsf{dom}\,(m'\,(T_I(\neg\alpha)))\,.\end{aligned}$$

$\varphi = \alpha \vee \beta$:

$$\begin{aligned}\mathfrak{I}, w \models_{Int} \alpha \vee \beta &\text{ iff } \mathfrak{I}, w \models_{Int} \alpha \text{ or } \mathfrak{I}, w \models_{Int} \beta\\ &\text{ iff } w \in \mathsf{dom}\,(m'\,(T_I\,(\alpha))) \text{ or } w \in \mathsf{dom}\,(m'\,(T_I\,(\beta)))\\ &\text{ iff } w \in \mathsf{dom}\,(m'\,(T_I\,(\alpha)) \cup m'\,(T_I\,(\beta)))\\ &\text{ iff } w \in \mathsf{dom}\,(m'\,(T_I\,(\alpha \vee \beta)))\,.\end{aligned}$$

$\varphi = \alpha \wedge \beta$: Proceeding along the same lines as we did with $\vee$,

$$\mathfrak{I}, w \models_{Int} \alpha \wedge \beta \text{ iff } w \in \mathsf{dom}\,(m'\,(T_I\,(\alpha \wedge \beta)))\ .$$

$\varphi = \alpha \rightarrow \beta$:

$$\begin{aligned}&\mathfrak{I}, w \models_{Int} \alpha \rightarrow \beta\\ &\text{iff } (\forall w' \in W)\,(wR'w' \wedge \mathfrak{I}, w' \models_{Int} \beta \ \Rightarrow\ \mathfrak{I}, w' \models_{Int} \gamma)\\ &\text{iff } (\nexists w' \in W)\,(wR'w' \wedge \mathfrak{I}, w' \models_{Int} \beta \wedge \mathfrak{I}, w' \not\models_{Int} \gamma)\\ &\text{iff } (\nexists w' \in W)\,(wR'w' \wedge w' \in \mathsf{dom}\,(m'\,(T_I\,(\beta))) \wedge\\ &\quad w' \notin \mathsf{dom}\,(m'\,(T_I\,(\gamma))))\\ &\text{iff } (\nexists w' \in A)\,(\langle w, w'\rangle \in m'\,(R) \wedge\\ &\quad w' \in \mathsf{dom}\,(m'\,(T_I\,(\beta))) \wedge w' \notin \mathsf{dom}\,(m'\,(T_I\,(\gamma))))\end{aligned}$$

$$\text{iff } w \in \mathsf{dom}\left(\overline{m'(R)\,;\left(m'(T_I(\alpha))\cdot\overline{m'(T_I(\beta))}\right)}\right)$$

$$\text{iff } w \in \mathsf{dom}\left(m'\left(\overline{R;\left(T_I(\alpha)\cdot\overline{T_I(\beta)}\right)}\right)\right)$$

$$\text{iff } w \in \mathsf{dom}\left(m'\left(T_I(\alpha \to \beta)\right)\right).$$

□

Lemma 6.5 *Let $\mathcal{F} = \langle \mathfrak{A}, m\rangle \in \mathsf{SPFM}$ satisfying conditions (C1)–(C6). Then, there exists an intuitionistic model $\mathfrak{I} = \langle W, R', m'\rangle$ constructed from $\mathcal{F}$ such that for all $w \in W$ and for all $\varphi \in IntFor$*

$$w \in \mathsf{dom}\left(m\left(T_I(\varphi)\right)\right) \qquad \Longleftrightarrow \qquad \mathfrak{I}, w \models_{Int} \varphi.$$

Proof Let us define $W = Urel_{\mathfrak{A}}$, $R' = m(R)$, and for all $p_i \in PropVar$ define $m'(p_i) = \mathsf{dom}\left(m\left(R_i\right)\right)$. Notice that by conditions (C1)–(C6), R' is a reflexive and transitive relation on W, and the heredity condition is satisfied by m'. The remaining part of the proof proceeds by induction on the structure of the formula φ.

$\varphi = p_i$:

$$\begin{aligned} w \in \mathsf{dom}\left(m\left(T_I(p_i)\right)\right) &\text{ iff } w \in \mathsf{dom}\left(m\left(R_i\right)\right)\\ &\text{ iff } w \in m'(p_i)\\ &\text{ iff } \mathfrak{I}, w \models_{Int} p_i. \end{aligned}$$

$\varphi = \neg\alpha$:

$$\begin{aligned} & w \in \mathsf{dom}\left(m\left(T_I(\neg\alpha)\right)\right)\\ &\quad \text{iff } w \in \mathsf{dom}\left(m\left(\overline{R;\overline{T_I(\alpha)}}\right)\right)\\ &\quad \text{iff } w \in \mathsf{dom}\left(\overline{R';\overline{m\left(T_I(\alpha)\right)}}\right)\\ &\quad \text{iff } (\nexists w' \in A)\left(wR'w' \wedge w' \in \mathsf{dom}\left(m\left(T_I(\alpha)\right)\right)\right)\\ &\quad \text{iff } (\nexists w' \in W)\left(wR'w' \wedge w' \in \mathsf{dom}\left(m\left(T_I(\alpha)\right)\right)\right)\\ &\quad \text{iff } (\forall w' \in W)\left(wR'w' \;\Rightarrow\; w' \notin \mathsf{dom}\left(m\left(T_I(\alpha)\right)\right)\right) \end{aligned}$$

$$\begin{aligned} &\text{iff } (\forall w' \in W)\left(wR'w' \;\Rightarrow\; \mathfrak{I}, w' \not\models_{Int} \alpha\right)\\ &\text{iff } \mathfrak{I}, w \models_{Int} \neg\alpha. \end{aligned}$$

$\varphi = \alpha \vee \beta$:

$$\begin{aligned}
& w \in \mathsf{dom}\,(m\,(T_I(\alpha \vee \beta))) \\
& \quad \text{iff } w \in \mathsf{dom}\,(m\,(T_I(\alpha) + T_I(\beta))) \\
& \quad \text{iff } w \in \mathsf{dom}\,(m\,(T_I(\alpha)) \cup m\,(T_I(\beta))) \\
& \quad \text{iff } w \in \mathsf{dom}\,(m\,(T_I(\alpha))) \text{ or } w \in \mathsf{dom}\,(m\,(T_I(\beta))) \\
& \quad \text{iff } \mathfrak{I}, w \models_{Int} \alpha \text{ or } \mathfrak{I}, w \models_{Int} \beta \\
& \quad \text{iff } \mathfrak{I}, w \models_{Int} \alpha \vee \beta.
\end{aligned}$$

$\varphi = \alpha \wedge \beta$:

$$\begin{aligned}
& w \in \mathsf{dom}\,(m\,(T_I(\alpha \wedge \beta))) \\
& \quad \text{iff } w \in \mathsf{dom}\,(m\,(T_I(\alpha) \cdot T_I(\beta))) \\
& \quad \text{iff } w \in \mathsf{dom}\,(m\,(T_I(\alpha)) \cap m\,(T_I(\beta))) \\
& \quad \text{iff } w \in \mathsf{dom}\,(m\,(T_I(\alpha))) \text{ and } w \in \mathsf{dom}\,(m\,(T_I(\beta))) \\
& \quad \text{iff } \mathfrak{I}, w \models_{Int} \alpha \text{ and } \mathfrak{I}, w \models_{Int} \beta \\
& \quad \text{iff } \mathfrak{I}, w \models_{Int} \alpha \wedge \beta.
\end{aligned}$$

$\varphi = \alpha \rightarrow \beta$:

$$\begin{aligned}
& w \in \mathsf{dom}\,(m\,(T_I(\alpha \rightarrow \beta))) \\
& \quad \text{iff } w \in \mathsf{dom}\left(m\left(\overline{R;\left(T_I(\alpha) \cdot \overline{T_I(\beta)}\right)}\right)\right) \\
& \quad \text{iff } w \in \mathsf{dom}\left(\overline{R';\left(m\,(T_I(\alpha)) \cdot \overline{m\,(T_I(\beta))}\right)}\right) \\
& \quad \text{iff } (\nexists w' \in W)(w R' w' \wedge w' \in \mathsf{dom}\,(m(T_I(\alpha))) \\
& \qquad \wedge\, w' \notin \mathsf{dom}\,(m(T_I(\beta)))) \\
& \quad \text{iff } (\forall w' \in W)(w R' w' \wedge w' \in \mathsf{dom}\,(m(T_I(\alpha))) \\
& \qquad \Rightarrow\ w' \in \mathsf{dom}\,(m(T_I(\beta)))) \\
& \quad \text{iff } (\forall w' \in W)\,(w R' w' \wedge \mathfrak{I}, w' \models_{Int} \alpha\ \Rightarrow\ \mathfrak{I}, w' \models_{Int} \beta) \\
& \quad \text{iff } \mathfrak{I}, w \models_{Int} \alpha \rightarrow \beta.
\end{aligned}$$

□

Let us denote by $\mathcal{K}$ the class of those fork models $\langle \mathfrak{A}, m \rangle \in \mathsf{SPFM}$ where conditions (C1)–(C6) hold. In the remaining part of the chapter we will denote by $FL^{\mathcal{K}}$ the fork logic induced by the class of fork models $\mathcal{K}$ in the following way

$$\models_{FL^{\mathcal{K}}} xTy \qquad \Longleftrightarrow \qquad \forall \mathcal{M} \in \mathcal{K}\,(\mathcal{M} \models_{FL'} xTy)\ .$$

Theorem 6.14 *Let $\psi \in IntFor$. Then, given individual variables $x \in UreVar$ and $y \in CompVar$,*

$$\models_{Int} \psi \qquad \Longleftrightarrow \qquad \models_{FL^{\mathcal{K}}} xT_I(\psi)y\ .$$

Proof Let us prove the contrapositive. If $\not\models_{Int} \psi$, then there exists an intuitionistic model $\mathfrak{I} = \langle W, R, m \rangle$ and $w \in W$ such that $\mathfrak{I}, w \not\models_{Int} \psi$. Then, by Lemma 6.4 there exists a fork model $\mathcal{F} = \langle \mathfrak{A}, m' \rangle \in \mathcal{K}$ such that $w \notin \mathsf{dom}\,(m'\,(T_I\,(\psi)))$. Let ν be a valuation satisfying $\nu(x) = w$, then $\mathcal{F}, \nu \not\models_{FL^{\mathcal{K}}} xT_I(\psi)y$, and thus $\not\models_{FL^{\mathcal{K}}} xT_I(\psi)y$.

If $\not\models_{FL^{\mathcal{K}}} xT_I(\psi)y$, then there exists a fork model $\mathcal{F} = \langle \mathfrak{A}, m \rangle \in \mathcal{K}$ and a valuation ν such that $\langle \nu(x), \nu(y) \rangle \notin m\,(T_I(\psi))$. Thus, $\nu(x) \notin \mathsf{dom}\,(m\,(T_I(\psi)))$. By Lemma 6.5 there exists an intuitionistic model $\mathfrak{I} = \langle A, R', m' \rangle$ such that $\mathfrak{I}, \nu(x) \not\models_{Int} \psi$, and thus $\not\models_{Int} \psi$. □

6.10.3 *A Fork Logic Calculus for Intuitionistic Logic*

In this subsection we will present a calculus for intuitionistic logic based on the calculus *FLC*. The calculus will be obtained by adding specific rules and modifying the notion of fundamental sequence in the calculus *FLC*. The calculus will be denoted by *Int-FLC*.

Throughout this subsection we assume we are working with a fork language $\mathcal{L}(R)$ with only one constant symbol as in Def. 6.47.

Definition 6.48 A sequence of fork formulas Γ is *Int-fundamental* if any of the following conditions are true:

(1) Γ is fundamental according to Def. 6.36, or
(2) the fork formula $xRx \in \Gamma$ for some $x \in UreVar$.

Condition (2) reflects the property that the intuitionistic accessibility relation R is reflexive on the set of urelements.

We define the intuitionistic calculus *Int-FLC* by adding the following specific rules to those of *FLC*.

$$\frac{\Gamma, xRy, \Delta}{\Gamma, xRz, xRy, \Delta \quad \Gamma, zRy, xRy, \Delta} \quad (TranR)$$

$$\frac{\Gamma, xR_iy, \Delta}{\Gamma, zRx, xR_iy, \Delta \quad \Gamma, zR_iy, xR_iy, \Delta}\ (H)$$

$$\frac{\Gamma, xR_iy, \Delta}{\Gamma, xR_iy, xR_iz, \Delta}\ (RI)$$

$$\frac{\Gamma}{\Gamma, xRy}\ (RUr) \qquad\qquad \frac{\Gamma}{\Gamma, xR_iy}\ (VarUr)$$

In rules $(TranR)$ and (H), $z \in UreVar$ is arbitrary. In rule (RI), $z \in IndTerm$ is arbitrary. In rule (RUr) either x or y belong to $IndTerm \setminus UreVar$, and in rule $(VarUr)$, $x \in IndTerm \setminus UreVar$. The admissibility of rule $(TranR)$ is equivalent to the transitivity of the relation R. The admissibility of rule (H) is equivalent to the validity of the heredity condition. The admissibility of rule (RI) is equivalent to relational variables being interpreted as right-ideal relations. The admissibility of rule (RUr) is equivalent to R being defined only on urelements. Finally, the admissibility of rule $(VarUr)$ is equivalent to variables having urelements in their domain.

Notice that the last comments imply the soundness of the calculus *Int-FLC*.

Theorem 6.15 *The calculus Int-FLC is sound with respect to the logic* $FL^{\mathcal{K}}$*, i.e., given* $tQt' \in ForkFor$

$$\vdash_{Int\text{-}FLC} tQt' \qquad \Rightarrow \qquad \models_{FL^{\mathcal{K}}} tQt' \ .$$

Definition 6.49 A proof tree T of a sequence of formulas Γ is *Int-saturated* if in all open branches B, the following conditions are satisfied.

(1) Conditions (1) through (19) from Def. 6.39,
(2) If $xRy \in B$, then, for each $z \in UreVar$, either $xRz \in B$ or $zRy \in B$ by an application of rule $(TranR)$.
(3) If $xR_iy \in B$ $(R_i \in RelVar)$, then, for each $z \in UreVar$, either $zRx \in B$ or $zR_iy \in B$ by an application of rule (H).
(4) If $xR_iy \in B$ $(R_i \in RelVar)$, then, for all $z \in IndTerm$, $xR_iz \in B$ by an application of rule (RI).
(5) For all $x, y \in IndTerm$ such that $x \in IndTerm \setminus UreVar$ or $y \in IndTerm \setminus UreVar$, $xRy \in B$ by an application of rule (RUr).
(6) For all $x \in IndTerm \setminus UreVar$ and $y \in IndTerm$, $xR_iy \in B$ by an application of rule $(VarUr)$.

Theorem 6.16 *The calculus Int-FLC is complete with respect to the logic $FL^{\mathcal{K}}$, i.e., given $tQt' \in$ ForkFor*

$$\models_{FL^{\mathcal{K}}} tQt' \qquad \Rightarrow \qquad \vdash_{Int\text{-}FLC} tQt' \ .$$

Proof The proof will follow the lines of the proof of Thm. 6.13, and therefore the reader will be directed there for some parts.

Assume tQt' is not provable in *Int-FLC*. Then no proof tree exists that provides a proof for tQt'. In particular, no *Int*-saturated tree with root tQt' provides a proof. Therefore, if T is an *Int*-saturated tree, there must exist an infinite branch B in T.

Let $\equiv$ be the binary relation on *IndTerm* defined by

$$x \equiv y \qquad \Longleftrightarrow \qquad x1\text{'}y \notin B.$$

The proof that $\equiv$ is an equivalence relation is the same as in Thm. 6.13.

Let $\mathfrak{A}$ be the FullPFAU with set of urelements $\{\,|x| : x \in \mathit{UreVar}\,\}$ and pairing function $\star$ defined by $|x|\star|y| = |x\star y|$. Proving that $\star$ is well defined and injective is done as in Thm. 6.13.

Let us define, for $R' \in \mathit{RelVar} \cup \{\,R\,\}$,

$$\langle |t_1|, |t_2|\rangle \in m(R') \qquad \Longleftrightarrow \qquad t_1 R' t_2 \notin B.$$

That m is well-defined is proved as in Thm. 6.13.

The relation $m(R)$ is reflexive, for if there exists $x \in \mathit{UreVar}$ such that $\langle |x|, |x|\rangle \notin m(R)$, then $xRx \in B$. Then B would be a closed branch, which is a contradiction.

The relation $m(R)$ is transitive, for if there are $x_1, x_2, x_3 \in \mathit{UreVar}$ such that $\langle |x_1|, |x_2|\rangle \in m(R)$, $\langle |x_2|, |x_3|\rangle \in m(R)$ and $\langle |x_1|, |x_3|\rangle \notin m(R)$, then $x_1 R x_2 \notin B$, $x_2 R x_3 \notin B$ and $x_1 R x_3 \in B$. Then, applying rule (*TranR*) either $x_1 R x_2 \in B$ or $x_2 R x_3 \in B$, which is a contradiction.

The heredity condition holds, for if there are $x_1, x_2 \in \mathit{UreVar}$ and $t \in \mathit{IndTerm}$ such that $\langle |x_1|, |t|\rangle \in m(R_i)$ ($R_i \in \mathit{RelVar}$), $\langle |x_1|, |x_2|\rangle \in m(R)$ and $\langle |x_2|, |t|\rangle \notin m(R_i)$, then $x_1 R_i t \notin B$, $x_1 R x_2 \notin B$ and $x_2 R_i t \in B$. Applying rule (H) either $x_1 R_i t \in B$ or $x_1 R x_2 \in B$, which is a contradiction.

In a similar way we show that relational variables are interpreted as right-ideal relations.

$m(R) \subseteq \mathit{UreVar} \times \mathit{UreVar}$, for if there is $t \in \mathit{IndTerm} \setminus \mathit{UreVar}$ such that $|t| \in \mathsf{dom}\,(m(R))$ or $|t| \in \mathsf{ran}\,(m(R))$, then applying rule (*RUr*) we arrive at a contradiction.

For all $R_i \in \mathit{RelVar}$, $\mathsf{dom}(m(R_i)) \subseteq \mathit{UreVar}$, for if there are $t \in \mathit{IndTerm} \setminus \mathit{UreVar}$ and $t' \in \mathit{IndTerm}$ such that $\langle |t|, |t'| \rangle \in m(R_i)$, then $tR_it' \notin B$. Applying rule (VarUr) we arrive at a contradiction.

Therefore the structure $\langle \mathfrak{A}, m \rangle$ belongs to SPFM and satisfies (C1)–(C6). The remaining part of the proof is similar to the respective part of the proof of Thm. 6.13. □

From Thm. 6.16 the corollary below immediately follows.

Corollary 6.2 *Given a formula $\varphi \in \mathit{IntFor}$, $x \in \mathit{UreVar}$, and $y \in \mathit{CompVar}$, we have*

$$\models_{\mathit{Int}} \varphi \qquad \Longleftrightarrow \qquad \vdash_{\mathit{Int\text{-}FLC}} xT_I(\varphi)y \ .$$

Proof From Thm. 6.14, for any formula $\varphi \in \mathit{IntFor}$, $x \in \mathit{UreVar}$, and $y \in \mathit{CompVar}$,

$$\models_{\mathit{Int}} \varphi \qquad \Longleftrightarrow \qquad \models_{\mathit{FL}^{\mathcal{K}}} xT_I(\varphi)y \ . \tag{6.2}$$

From Thms. 6.15 and 6.16, we then obtain

$$\models_{\mathit{FL}^{\mathcal{K}}} xT_I(\varphi)y \qquad \Longleftrightarrow \qquad \vdash_{\mathit{Int\text{-}FLC}} xT_I(\varphi)y \ . \tag{6.3}$$

Joining (6.2) and (6.3), we then obtain

$$\models_{\mathit{Int}} \varphi \qquad \Longleftrightarrow \qquad \vdash_{\mathit{Int\text{-}FLC}} xT_I(\varphi)y \ .$$

□

6.10.3.1 *Example*

In order to see how the calculus works, let us consider a proof of the intuitionistic tautology $\neg\neg\neg\alpha \rightarrow \neg\alpha$. According to Cor. 6.2, it suffices to prove that $\vdash_{\mathit{Int\text{-}FLC}} xT_I(\neg\neg\neg\alpha \rightarrow \neg\alpha)y$. In order to keep an economic notation we will not apply the mapping T_I entirely from the beginning, but by parts according to our needs. Partial applications of mapping T_I will be evidenced using a rule denoted by (T_I).

$$\dfrac{\dfrac{xT_I(\neg\neg\neg\alpha \rightarrow \neg\alpha)y}{x\overline{R;\left(T_I(\neg\neg\neg\alpha)\cdot\overline{\overline{T_I(\neg\alpha)}}\right)}y}\ (T_I)}{\underbrace{x\overline{R}z_1, z_1\overline{T_I(\neg\neg\neg\alpha)\cdot\overline{\overline{T_I(\neg\alpha)}}}y}_{\Lambda_1} \quad \underbrace{x\overline{R}z_2, z_2\overline{T_I(\neg\neg\neg\alpha)\cdot\overline{\overline{T_I(\neg\alpha)}}}y}_{\Lambda_2}}\ (N;)$$

In sequences Λ_1 and Λ_2, $z_1 \in \mathit{UreVar}$ and $z_2 \in \mathit{CompVar}$.
Regarding sequence Λ_2, we have

$$\frac{x\overline{R}z_2, z_2\overline{T_I(\neg\neg\neg\alpha)\cdot\overline{\overline{T_I(\neg\alpha)}}}y}{x\overline{R}z_2, z_2\overline{T_I(\neg\neg\neg\alpha)\cdot\overline{\overline{T_I(\neg\alpha)}}}y, xRz_2}\ (RUr)$$

The last sequence is fundamental, and thus the branch is closed.
Regarding sequence Λ_1, we have

$$\begin{array}{c}
x\overline{R}z_1, z_1\overline{T_I(\neg\neg\neg\alpha)\cdot\overline{\overline{T_I(\neg\alpha)}}}y\ (N\cdot) \\ \hline
x\overline{R}z_1, z_1\overline{T_I(\neg\neg\neg\alpha)}y, z_1\overline{\overline{\overline{T_I(\neg\alpha)}}}y\ (N^-) \\ \hline
x\overline{R}z_1, z_1\overline{T_I(\neg\neg\neg\alpha)}y, z_1T_I(\neg\alpha)y\ (T_I) \\ \hline
x\overline{R}z_1, z_1\overline{\overline{\overline{R;T_I(\neg\neg\alpha)}}}y, z_1T_I(\neg\alpha)y\ (N^-) \\ \hline
x\overline{R}z_1, z_1R;T_I(\neg\neg\alpha)y, z_1T_I(\neg\alpha)y\ (T_I) \\ \hline
x\overline{R}z_1, z_1R;T_I(\neg\neg\alpha)y, z_1\overline{R;T_I(\alpha)}y\ (N;) \\ \hline
\underbrace{x\overline{R}z_1, z_1R;T_I(\neg\neg\alpha)y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y}_{\Lambda_3} \quad \underbrace{x\overline{R}z_1, z_1R;T_I(\neg\neg\alpha)y, z_1\overline{R}t_2, t_2\overline{T_I(\alpha)}y}_{\Lambda_4}
\end{array}$$

In sequences Λ_3 and Λ_4, $t_1 \in \mathit{UreVar}$ and $t_2 \in \mathit{CompVar}$.
Regarding sequence Λ_3 we have:

$$\begin{array}{c}
x\overline{R}z_1, z_1R;T_I(\neg\neg\alpha)y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y\ (P;) \\ \hline
\underbrace{x\overline{R}z_1, z_1Rt_1, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y}_{\Lambda_5} \quad \underbrace{x\overline{R}z_1, t_1T_I(\neg\neg\alpha)y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y}_{\Lambda_6}
\end{array}$$

Since the fork formulas z_1Rt_1 and $z_1\overline{R}t_1$ occur in Λ_5, this branch is closed.
Regarding sequence Λ_6 we have

$$\begin{array}{c}
x\overline{R}z_1, t_1T_I(\neg\neg\alpha)y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y\ (T_I) \\ \hline
x\overline{R}z_1, t_1\overline{R;T_I(\neg\alpha)}y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y\ (N;) \\ \hline
\underbrace{x\overline{R}z_1, t_1\overline{R}v_1, v_1\overline{T_I(\neg\alpha)}y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y}_{\Lambda_7} \quad \underbrace{x\overline{R}z_1, t_1\overline{R}v_2, v_2\overline{T_I(\neg\alpha)}y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y}_{\Lambda_8}
\end{array}$$

In sequences Λ_7 and Λ_8, $v_1 \in \mathit{UreVar}$ and $v_2 \in \mathit{CompVar}$.
Regarding sequence Λ_7 we have:

$$\begin{array}{c}
x\overline{R}z_1, t_1\overline{R}v_1, v_1\overline{T_I(\neg\alpha)}y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y\ (T_I) \\ \hline
x\overline{R}z_1, t_1\overline{R}v_1, v_1\overline{\overline{R;T_I(\alpha)}}y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y\ (N^-) \\ \hline
x\overline{R}z_1, t_1\overline{R}v_1, v_1R;T_I(\alpha)y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y\ (P;) \\ \hline
\underbrace{x\overline{R}z_1, t_1\overline{R}v_1, v_1Rv_1, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y}_{\Lambda_9} \quad \underbrace{x\overline{R}z_1, t_1\overline{R}v_1, v_1T_I(\alpha)y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y}_{\Lambda_{10}}
\end{array}$$

Since the fork formula $v_1 R v_1$ occurs in Λ_9, this branch is closed.
Regarding sequence Λ_{10} we have (denoting by T the term $T_I(\alpha)$):

$$\frac{x\overline{R}z_1, t_1\overline{R}v_1, v_1Ty, z_1\overline{R}t_1, t_1\overline{T}y \ (H)}{\underbrace{x\overline{R}z_1, t_1\overline{R}v_1, t_1Rv_1, v_1Ty, z_1\overline{R}t_1, t_1\overline{T}y}_{\Lambda_{11}} \quad \underbrace{x\overline{R}z_1, t_1\overline{R}v_1, t_1Ty, v_1Ty, z_1\overline{R}t_1, t_1\overline{T}y}_{\Lambda_{12}}}$$

Since the fork formulas $t_1\overline{R}v_1$ and t_1Rv_1 occur in Λ_{11}, the branch is closed. In a similar way, since the fork formulas $t_1T_I(\alpha)y$ and $t_1\overline{T_I(\alpha)}y$ occur in Λ_{12}, this branch is also closed.

Regarding sequence Λ_8, we have:

$$\frac{x\overline{R}z_1, t_1\overline{R}v_2, v_2\overline{T_I(\neg\alpha)}y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y \ (RUr)}{\underbrace{x\overline{R}z_1, t_1\overline{R}v_2, v_2\overline{T_I(\neg\alpha)}y, z_1\overline{R}t_1, t_1\overline{T_I(\alpha)}y, t_1Rv_2}_{\Lambda_{13}}}$$

Since the fork formulas $t_1\overline{R}v_2$ and t_1Rv_2 occur in Λ_{13}, the branch is closed.

Finally, regarding sequence Λ_4 we have:

$$\frac{x\overline{R}z_1, z_1R;T_I(\neg\neg\alpha)y, z_1\overline{R}t_2, t_2\overline{T_I(\alpha)}y \ (RUr)}{\underbrace{x\overline{R}z_1, z_1R;T_I(\neg\neg\alpha)y, z_1\overline{R}t_2, t_2\overline{T_I(\alpha)}y, z_1Rt_2}_{\Lambda_{14}}}$$

Since $z_1\overline{R}t_2$ and z_1Rt_2 occur in Λ_{14}, the branch is closed.

6.11 A Relational Proof System for Minimal Intuitionistic Logic

Minimal intuitionistic logic J was introduced by Johansson in 1936 [I. Johansson (1936)]. It differs from intuitionistic logic in that the axiom $\neg\alpha \rightarrow (\alpha \rightarrow \beta)$ is deleted. In [M. Fitting (1969)], Fitting introduced a Kripke-style semantics for the logic J. A Kripke model for J is a system $\mathfrak{M} = \langle W, R, Q, m \rangle$ where W is a nonempty set, R is a reflexive and transitive relation on W, $Q \subseteq W$ is a R-closed subset of W (that is, if $w \in Q$ and $\langle w, w' \rangle \in R$, then $w' \in Q$), and m is a meaning function which is defined as being for the Kripke semantics of the intuitionistic logic *Int* with the exception of the evaluation of negations:

$$\mathfrak{M}, w \models_J \neg\alpha \qquad \text{iff}$$
$$\text{for all } w', \text{ if } \langle w, w' \rangle \in R, \text{ then } M, w' \not\models_J \alpha \text{ or } w' \in Q\ .$$

Q is to be thought of as the set of those states of information which are inconsistent. The notion of truth of a formula in a model and validity are the same as for *Int*. It is known that a formula α is valid in J iff α is true in every finite model of J with antisymmetric relation R.

Interpretability of J in fork logic FL' is established by a translation T_J of formulas of J into relational terms. It coincides with translation T_I except for the translation of negated formulas:

$$T_J(\neg\alpha) = \overline{R;(T_J(\alpha)\cdot\overline{Q})}$$

where R and Q are relational constants interpreted as the accessibility relation from models of J and the right ideal relation that is a counterpart of the set Q from these models.

The relational proof system for J (that we will denote by *J-FLC*) consists of all the rules of the proof system of fork logic, the specific rules for *Int* and the following specific rules:

$$\frac{\Gamma, xQy, \Delta}{\Gamma, zQy, \Delta, xQy \quad \Gamma, zRx, \Delta, xQy} \ (Q1)$$

$$\frac{\Gamma, xQy, \Delta}{\Gamma, xQz, \Delta, xQy} \ (Q2)$$

$$\frac{\Gamma}{\Gamma, xQy} \ (Q3)$$

In rule $(Q1)$ $z \in \mathit{UreVar}$, in rule $(Q2)$ $z \in \mathit{IndTerm}$, and in rule $(Q3)$, $x \in \mathit{IndTerm} \setminus \mathit{UreVar}$ and $y \in \mathit{IndTerm}$.

Rule $(Q1)$ is admissible iff Q is R-closed, rule $(Q2)$ is admissible iff Q is a right-ideal relation, and rule $(Q3)$ is admissible iff Q has only urelements in its domain.

Notice that the abstract fork-algebraic equations

(C7) : $Q = Q;1$,
(C8) : $\overline{(R\cdot Q)\,;\overline{Q}} = 1$,

(C9) : $Q \leq {\cup}1$,

state that Q is an R-closed, right-ideal relation whose domain is made of urelements.

Let $\mathcal{L}(R,Q)$ be a fork language with two constant symbols. Let $\mathcal{K}'$ be the class of those fork models $\langle \mathfrak{A}, m \rangle$ for the language $\mathcal{L}(R,Q)$ satisfying conditions (C1)–(C9). Then $\mathcal{K}'$ induces a fork logic $FL^{\mathcal{K}'}$ as follows:

$$\models_{FL^{\mathcal{K}'}} xTy \qquad \Longleftrightarrow \qquad \forall \mathcal{M} \in \mathcal{K}' \, (\mathcal{M} \models_{FL'} xTy) \ .$$

From the admissibility of the specific rules ($Q1$)–($Q3$) we obtain the following theorem on the soundness of the calculus J-FLC.

Theorem 6.17 *The calculus J-FLC is sound with respect to the logic $FL^{\mathcal{K}'}$, i.e.,*

$$\vdash_{J\text{-}FLC} xTy \qquad \Rightarrow \qquad \models_{FL^{\mathcal{K}'}} xTy \ .$$

Definition 6.50 A proof tree T of a sequence of formulas Γ is J-saturated if in all open branches B, the following conditions are satisfied.

(1) T is *Int*-saturated,
(2) If $xQy \in B$, then, for all $z \in$ *UreVar*, either $zQy \in B$ or $zRx \in B$ applying rule ($Q1$),
(3) If $xQy \in B$, then, for all $z \in$ *IndTerm*, $xQz \in B$ applying rule ($Q2$),
(4) For all $x \in$ *IndTerm*$\setminus$*UreVar* and $y \in$ *IndTerm*, $xQy \in B$ applying rule ($Q3$).

Theorem 6.18 *The calculus J-Int is complete with respect to the logic $FL^{\mathcal{K}'}$, i.e.,*

$$\models_{FL^{\mathcal{K}'}} xSy \qquad \Rightarrow \qquad \vdash_{J\text{-}FLC} xSy \ .$$

Proof The proof will follow the lines of the proof of Thm. 6.13, and therefore the reader will be directed there for some parts.

Assume $tSt' \in$ *ForkFor* is valid in $FL^{\mathcal{K}'}$ but is not provable in J-FLC. Then no proof tree exists that provides a proof for tSt'. In particular, no J-saturated tree with root tSt' provides a proof. Therefore, if T is a J-saturated tree, there must exist an infinite branch B in T.

Let $\equiv$ be the binary relation on *IndTerm* defined by

$$x \equiv y \qquad \Longleftrightarrow \qquad x1'y \notin B \ .$$

The proof that $\equiv$ is an equivalence relation is as in Thm. 6.13.

Let $\mathfrak{A}$ be the FullPFAU with set of urelements $\{\, |x| : x \in \mathit{UreVar}\,\}$ and pairing function $\star$ defined by $|x|\star|y| = |x{\star}y|$. Proving that $\star$ is well defined and injective is done as in Thm. 6.13.

Let us define, for $R' \in \mathit{RelVar} \cup \{\, R, Q \,\}$,

$$\langle |t_1|, |t_2| \rangle \in m(R') \qquad \Longleftrightarrow \qquad t_1 R' t_2 \notin B \; .$$

That m is well-defined is proved as in Thm. 6.13.

That R is reflexive, transitive and that the heredity condition holds are all proved as in Thm. 6.13.

Assume that $\langle |w|, |w'| \rangle \in m(R)$, $\langle |w|, |x| \rangle \in m(Q)$ and $\langle |w'|, |x| \rangle \notin m(Q)$. Then, $wRw' \notin B$, $wQx \notin B$, and $w'Qx \in B$. Applying rule $(Q1)$ we immediately arrive at a contradiction, and thus $m(Q)$ is $m(R)$-closed.

Assume $\langle |x|, |y| \rangle \in m(Q)$, but $\langle |x|, |t| \rangle \notin m(Q)$ for some $t \in \mathit{IndTerm}$. Then, $xQt \in B$. Since the tree T is J saturated, applying rule $(Q2)$ we arrive at a contradiction, and thus $m(Q)$ is right-ideal.

In a similar way we show that relational variables are interpreted as right-ideal relations.

If there are $x \in \mathit{IndTerm} \setminus \mathit{UreVar}$ and $y \in \mathit{IndTerm}$ such that $\langle |x|, |y| \rangle \in m(R)$, then $xRy \notin B$. Applying rule $(Q3)$, $xRy \in B$, which is a contradiction.

Therefore the structure $\langle \mathfrak{A}, m \rangle$ belongs to SPFM and satisfies (C1)–(C9).

Let ν be the valuation defined by $\nu(x) = |x|$, for $x \in \mathit{IndVar}$. In Thm. 6.13 it is shown by induction that $V_\nu(t) = |t|$ for all $t \in \mathit{IndTerm}$.

The remaining part of the proof is as in Thm. 6.13. □

In order to be able to reason in the calculus *J-FLC* for proving minimal intuitionistic properties we still need to show the interpretability of the logic J in the logic $FL^{\mathcal{K}'}$.

Lemma 6.6 *Let $\mathfrak{J} = \langle W, R', Q', m \rangle$ be a minimal intuitionistic model. Then there exists a fork model $\mathcal{F} = \langle \mathfrak{A}, m' \rangle \in$ SPFM constructed from $\mathfrak{J}$ satisfying conditions (C1)–(C9) such that for all $w \in W$ and for all $\varphi \in \mathit{IntFor}$*

$$\mathfrak{J}, w \models_J \varphi \qquad \Longleftrightarrow \qquad w \in \mathsf{dom}\,(m'\,(T_J(\varphi))) \; .$$

Proof Let $\mathfrak{A}$ be the FullPFAU with set of urelements W. Define $m'(R) = R'$. Let $m'(Q) = \{\, \langle x, y \rangle : x \in Q' \,\}$, and for each $R_i \in \mathit{RelVar}$ define

$m'(R_i) = \{ \langle x, y \rangle : x \in m(p_i) \}$. Conditions (C1)–(C9) hold because of how m' is defined.

The remaining part of the proof proceeds by induction on the structure of the formula φ and follows the lines of the proof of Lemma 6.4 except for the case of negation, where the semantics differ. For the negation we proceed as follows:

$$\begin{aligned}
&\mathfrak{J}, w \models_J \neg\alpha \\
&\quad\text{iff } (\forall w' \in W)\,(wR'w' \;\Rightarrow\; \mathfrak{J}, w' \not\models_J \alpha \vee w' \in Q') \\
&\quad\text{iff } (\forall w' \in W)\,(\langle w, w' \rangle \in m'(R) \;\Rightarrow \\
&\qquad w' \notin \mathsf{dom}\,(m'\,(T_J(\alpha))) \vee w' \in \mathsf{dom}\,(m'(Q))) \\
&\quad\text{iff } (\nexists w' \in W)\,(\langle w, w' \rangle \in m'(R) \wedge \\
&\qquad w' \in \mathsf{dom}\,(m'\,(T_J(\alpha))) \wedge w' \notin \mathsf{dom}\,(m'(Q))) \\
&\quad\text{iff } (\nexists w' \in A)\,(\langle w, w' \rangle \in m'(R) \wedge \\
&\qquad w' \in \mathsf{dom}\,(m'\,(T_J(\alpha))) \wedge w' \notin \mathsf{dom}\,(m'(Q))) \\
&\quad\text{iff } w \in \mathsf{dom}\left(\overline{m'(R); \left(m'\,(T_J(\alpha)) \cdot \overline{m'(Q)}\right)}\right) \\
&\quad\text{iff } w \in \mathsf{dom}\left(m'\left(\overline{R; \left(T_J(\alpha) \cdot \overline{Q}\right)}\right)\right) \\
&\quad\text{iff } w \in \mathsf{dom}\,(m'\,(T_J(\neg\alpha)))\,.
\end{aligned}$$

□

Lemma 6.7 *Let $\mathcal{F} = \langle \mathfrak{A}, m \rangle \in \mathsf{SPFM}$, satisfying conditions (C1)–(C9). Then, there exists a minimal intuitionistic model $\mathfrak{J} = \langle W, R', Q', m' \rangle$ constructed from $\mathcal{F}$ such that for all $w \in Urel_{\mathfrak{A}}$ and for all $\varphi \in IntFor$*

$$w \in \mathsf{dom}\,(m\,(T_J(\varphi))) \qquad \Longleftrightarrow \qquad \mathfrak{J}, w \models_J \varphi\,.$$

Proof Let us define $W = Urel_{\mathfrak{A}}$, $R' = m(R)$, $Q' = \mathsf{dom}\,(m(Q))$, and for all $p_i \in PropVar$ define $m'(p_i) = \mathsf{dom}\,(m\,(R_i))$. Notice that by conditions (C1)–(C9), R' is a reflexive and transitive relation on W, the heredity condition is satisfied by m' and Q' is an R-closed right-ideal relation.

The remaining part of the proof proceeds by induction on the structure of the formula φ and follows the lines of the proof of Lemma 6.5, except for the case of the negation, where the semantics differ. For the negation we proceed as follows:

$$\begin{aligned}
&w \in \mathsf{dom}\,(m\,(T_J(\neg\alpha)))\\
&\quad \text{iff } w \in \mathsf{dom}\left(m\left(\overline{R;\,(T_J(\alpha)\cdot\overline{Q})}\right)\right)\\
&\quad \text{iff } w \in \mathsf{dom}\left(\overline{m(R);\,\left(m\,(T_J(\alpha))\cdot\overline{m(Q)}\right)}\right)\\
&\quad \text{iff } (\nexists w' \in A)((\langle w, w'\rangle \in m(R)\\
&\qquad \wedge\, w' \in \mathsf{dom}\,(m(T_J(\alpha))) \wedge w' \notin \mathsf{dom}\,(m(Q)))\\
&\quad \text{iff } (\nexists w' \in W)\,(w R' w' \wedge w' \in \mathsf{dom}\,(m\,(T_J(\alpha))) \wedge w' \notin Q')\\
&\quad \text{iff } (\forall w' \in W)\,(w R' w' \;\Rightarrow\; w' \notin \mathsf{dom}\,(m\,(T_J(\alpha))) \vee w' \in Q')\\
&\quad \text{iff } (\forall w' \in W)\,(w R' w' \;\Rightarrow\; \mathfrak{J}, w' \nvDash_J \alpha \vee w' \in Q')\\
&\quad \text{iff } \mathfrak{J}, w \models_J \neg\alpha.
\end{aligned}$$

□

Theorem 6.19 *Let $\psi \in$ IntFor. Then, given $x \in$ UreVar and $y \in$ IndVar,*

$$\models_J \psi \qquad \Longleftrightarrow \qquad \models_{FL^{\mathcal{K}'}} xT_J(\psi)y\ .$$

Proof Let us prove the contrapositive. If $\nvDash_J \psi$, then a minimal intuitionistic model $\mathfrak{J} = \langle W, R', Q', m\rangle$ exists and $w \in W$ such that $\mathfrak{J}, w \nvDash \psi$. Then, by Lemma 6.6 there exists $\mathcal{F} = \langle \mathfrak{A}, m'\rangle \in \mathcal{K}'$ such that $w \notin \mathsf{dom}\,(m'\,(T_J\,(\psi)))$. Let ν be a valuation satisfying $\nu(x) = w$, then $\mathcal{F}, \nu \nvDash_{FL^{\mathcal{K}'}} xT_J(\psi)y$, and thus $\nvDash_{FL^{\mathcal{K}'}} xT_J(\psi)y$.

If $\nvDash_{FL^{\mathcal{K}'}} xT_J(\psi)y$, then there exists $\mathcal{F} = \langle \mathfrak{A}, m\rangle \in \mathcal{K}'$ and a valuation ν such that $\langle \nu(x), \nu(y)\rangle \notin m\,(T_J(\psi))$. Thus, $\nu(x) \notin \mathsf{dom}\,(m\,(T_J(\psi)))$. By Lemma 6.7 a minimal intuitionistic model $\mathfrak{J}$ exists such that $\mathfrak{J}, \nu(x) \nvDash_J \psi$, and thus $\nvDash_J \psi$. □

From Thms. 6.17, 6.18 and 6.19 the corollary below follows immediately.

Corollary 6.3 *Given $\varphi \in$ IntFor, $x \in$ UreVar and $y \in$ CompVar,*

$$\models_J \varphi \qquad \Longleftrightarrow \qquad \vdash_{J\text{-}FLC} xT_J(\varphi)y\ .$$

Proof From Thm. 6.19, for any formula $\varphi \in$ *IntFor*

$$\models_J \varphi \qquad \Longleftrightarrow \qquad \models_{FL^{\mathfrak{K}'}} xT_J(\varphi)y\ . \tag{6.4}$$

From Thms. 6.17 and 6.18, we then obtain

$$\models_{FL^{\aleph'}} xT_J(\varphi)y \qquad \Longleftrightarrow \qquad \vdash_{J\text{-}FLC} xT_J(\varphi)y \ . \tag{6.5}$$

From (6.4) and (6.5),

$$\models_J \varphi \qquad \Longleftrightarrow \qquad \vdash_{J\text{-}FLC} xT_J(\varphi)y \ .$$

□

6.12 Relational Reasoning in Intermediate Logics

Intermediate logics are the logics whose valid formulas include all the formulas that are valid in intuitionistic logic but not necessarily all the tautologies of classical logic. In that sense these logics are between intuitionistic and classical logic. For many intermediate logics a Kripke semantics is known. Below we give examples of conditions that the accessibility relation is supposed to satisfy in Kripke models of some intermediate logics.

(I1) $\exists x \forall y\, (xRy)$
(I2) $\forall x \exists y\, (xRy \wedge \forall z\, (yRz \rightarrow y = z))$
(I3) $\forall x \forall y \exists z\, (zRx \wedge zRy \wedge \forall t\, (tRx \wedge tRy \rightarrow tRz))$
(I4) $\forall x \forall y \forall z\, (xRy \wedge xRz \rightarrow yRz \vee zRy \vee \forall t\, (yRt \rightarrow zRt))$
(I5) $\forall x \forall y \forall z\, (xRy \wedge xRz \rightarrow \exists t\, (yRt \wedge zRt))$
(I6) $\exists x \forall y\, (y \neq x \rightarrow xRy) \wedge \forall x \forall z \forall t\, (xRz \wedge xRt \rightarrow \neg zRt)$

The translation from formulas of intermediate logics into relational terms is the same as for formulas of intuitionistic logic. There are three methods of developing relational means of reasoning for intermediate logics within the framework of fork logic.

6.12.1 *Method 1*

We define a specific rule or a fundamental sequence for every condition on the accessibility relation in the underlying Kripke models of a given logic. The relational proof system for the logic consists in all the rules and fundamental sequences from the proof system of fork logic together with those new specific rules and/or fundamental sequences. For example, the rule corresponding to condition (I4) is the following:

$$\frac{\Gamma, \overbrace{yRz, zRy, zRt}^{\Phi}, \Delta}{\Gamma, xRy, \Phi, \Delta \quad \Gamma, xRz, \Phi, \Delta \quad \Gamma, yRt, \Phi, \Delta}\ (R4)$$

where x is a variable.

Proposition 6.1 *Rule ($R4$) is admissible in fork logic iff in every fork model the relation R satisfies condition ($I4$).*

Proof $\Rightarrow$) Notice that condition (I4) is equivalent to the following:

$$\forall x \forall y \forall z \forall t\,(xRy \wedge xRz \wedge yRt \rightarrow yRz \vee zRy \vee zRt)\ .$$

Assume that rule ($R4$) is admissible and suppose that in some fork model $\langle \mathfrak{A}, m \rangle$ condition (I4) is not satisfied. Hence, for some valuation ν in this model we have $\langle \nu(x), \nu(y)\rangle \in m(R)$, $\langle \nu(x), \nu(z)\rangle \in m(R)$, $\langle \nu(y), \nu(t)\rangle \in m(R)$, $\langle \nu(y), \nu(z)\rangle \notin m(R)$, $\langle \nu(z), \nu(y)\rangle \notin m(R)$, and $\langle \nu(z), \nu(t)\rangle \notin m(R)$. Consider an instance of rule ($R4$) with $\Gamma = x\overline{R}y, x\overline{R}z, y\overline{R}t$, and with empty Δ. Then, all the lower sequences of the rule are valid, so the upper sequence must be valid as well. But in the above model none of the formulas of the upper sequence are true under valuation ν, a contradiction.
$\Leftarrow$) It is clear that this implication also holds. □

6.12.2 *Method 2*

We use the following deduction theorem for fork algebras with urelements.

Theorem 6.20 *Let γ and γ' be fork algebra terms. Then,*

$$\gamma = 1 \models_{\mathsf{AFAU}} \gamma' = 1 \qquad \Longleftrightarrow \qquad \models_{\mathsf{AFAU}} 1;\overline{\gamma};1\ +\ \gamma' = 1\ .$$

Proof $\Rightarrow$) If $\gamma = 1 \models_{\mathsf{AFAU}} \gamma' = 1$, then, since $\mathsf{SAFAU} \subseteq \mathsf{AFAU}$,

$$\gamma = 1 \models_{\mathsf{SAFAU}} \gamma = 1\ .$$

Let $\mathfrak{A} \in \mathsf{SAFAU}$ and $m : \mathit{RelConst} \cup \mathit{RelVar} \rightarrow A$ be arbitrary. If $\mathfrak{A} \models \gamma = 1$, then by hypothesis $\mathfrak{A} \models \gamma' = 1$. Then

$$\mathfrak{A} \models 1;\overline{\gamma};1\ +\ \gamma' = 1\ .$$

If $\mathfrak{A} \not\models \gamma = 1$, then $\overline{\gamma} \neq 0$. Then, since $\mathfrak{A}$ is simple (and thus satisfies formula (2.1)), $\mathfrak{A} \models 1;\overline{\gamma};1 = 1$. Thus,

$$\mathfrak{A} \models 1;\overline{\gamma};1 \ + \ \gamma' = 1 \ .$$

Then, $\models_{\mathsf{SAFAU}} 1;\overline{\gamma};1 \ + \ \gamma' = 1$. By Thm. 6.2, we then have

$$\models_{\mathsf{AFAU}} 1;\overline{\gamma};1 \ + \ \gamma' = 1 \ .$$

$\Leftarrow$) Let $\mathfrak{A} \in \mathsf{AFAU}$ and $m : \mathit{RelConst} \cup \mathit{RelVar} \to A$ be arbitrary. By hypothesis $\mathfrak{A} \models 1;\overline{\gamma};1 \ + \ \gamma' = 1$. Notice that if $\mathfrak{A} \models \gamma = 1$ then $\mathfrak{A} \models \overline{\gamma} = 0$, and therefore $\mathfrak{A} \models 1;\overline{\gamma};1 = 0$. Thus, $\models_{\mathsf{AFAU}} 1;\overline{\gamma};1 \ + \ \gamma' = 1$ implies $\gamma = 1 \models_{\mathsf{AFAU}} \gamma' = 1$. □

The proof of the following corollary follows the same steps as the proof above.

Corollary 6.4 *Let γ and γ' be fork algebra terms. Then*

$$\gamma = 1 \models_{\mathcal{K}} \gamma' = \mathsf{U}1 \qquad \Longleftrightarrow \qquad \models_{\mathcal{K}} \mathsf{U}1;\overline{\gamma};1 \ + \ 1'_{\mathsf{U}};\gamma' = \mathsf{U}1 \ .$$

Let $L(\Gamma)$ be an intuitionistic logic, where $\Gamma = \{\gamma_1, \dots, \gamma_k\}$ is a finite set of first-order sentences imposing conditions on the accessibility relation.

Let $\mathcal{K}_\Gamma$ be the class of those fork models satisfying the set of equations $\{ T_{\nabla,\langle\rangle}(\gamma) = 1 : \gamma \in \Gamma \}$ plus conditions (C1)–(C6), and let FL_Γ be the fork logic induced by the class of fork models $\mathcal{K}_\Gamma$.

Lemma 6.8 *Let $\mathfrak{I} = \langle W, R', m \rangle$ be a model for the intuitionistic logic $L(\Gamma)$. Then a fork model $\mathcal{F} = \langle \mathfrak{A}, m' \rangle$ exists for the fork logic FL_Γ, constructed from $\mathfrak{I}$, such that for all $w \in W$ and for all $\varphi \in \mathit{IntFor}$*

$$\mathfrak{I}, w \models_{L(\Gamma)} \varphi \qquad \Longleftrightarrow \qquad w \in \mathsf{dom}\,(m'\,(T_I(\varphi))) \ .$$

Proof Define $\mathfrak{A}$ as the FullPFAU with set of urelements W, let $m'(R) = R'$, and for each $R_i \in \mathit{RelVar}$ define $m'(R_i) = \{ \langle x, y \rangle : x \in m(p_i) \}$. The equations in the set $\{ T_{\nabla,\langle\rangle}(\gamma) = 1 : \gamma \in \Gamma \}$ hold due to the way R' and R_i were defined (cf. Ch. 5).

The remaining part of the proof proceeds by induction on the structure of the formula φ, and is as in Lemma 6.4. □

Lemma 6.9 *Let $\mathcal{F} = \langle \mathfrak{A}, m \rangle \in \mathcal{K}_\Gamma$. Then there exists an intuitionistic model $\mathfrak{I} = \langle W, R', m' \rangle$ for the logic $L(\Gamma)$ constructed from $\mathcal{F}$ such that for* *all $w \in W$ and for all $\varphi \in IntFor$*

$$w \in \mathsf{dom}\,(m\,(T_I(\varphi))) \qquad \Longleftrightarrow \qquad \mathfrak{I}, w \models_{L(\Gamma)} \varphi \ .$$

Proof Let us define $W = Urel_{\mathfrak{A}}$, $R' = m(R)$, and for all $p_i \in PropVar$ define $m'(p_i) = \mathsf{dom}\,(m\,(R_i))$. Notice that the validity of the set of equations $\{\, T_{\nabla,\langle\rangle}(\gamma) = 1 : \gamma \in \Gamma \,\}$ in $\mathcal{F}$ implies the validity of the sentences Γ in $\mathfrak{I}$. The remaining part of the proof proceeds by induction on the structure of the formula φ as in Lemma 6.5. $\Box$

Theorem 6.21 *Let $\psi \in IntFor$. Then, given $x \in UreVar$ and $y \in CompVar$,*

$$\models_{L(\Gamma)} \psi \qquad \Longleftrightarrow \qquad \models_{FL_\Gamma} xT_I(\psi)y \ .$$

Proof Let us prove the contrapositive. If $\not\models_{Int} \psi$, then an intuitionistic model $\mathfrak{I} = \langle\, W, R, m \,\rangle$ for $L(\Gamma)$ and $w \in W$ exist such that $\mathfrak{I}, w \not\models_{Int} \psi$. Then, by Lemma 6.8 there exists a fork model $\mathcal{F} = \langle \mathfrak{A}, m' \,\rangle \in \mathcal{K}_\Gamma$ such that $w \notin \mathsf{dom}\,(m'\,(T_I\,(\psi)))$. Let ν be a valuation satisfying $\nu(x) = w$, then $\mathcal{F}, \nu \not\models_{FL_\Gamma} xT_I(\psi)y$, and thus $\not\models_{FL_\Gamma} xT_I(\psi)y$.

If $\not\models_{FL_\Gamma} xT_I(\psi)y$, then a fork model $\mathcal{F} = \langle \mathfrak{A}, m \,\rangle \in \mathcal{K}_\Gamma$ and a valuation ν exist such that $\langle \nu(x), \nu(y) \rangle \notin m\,(T_I(\psi))$. Thus, $\nu(x) \notin \mathsf{dom}\,(m\,(T_I(\psi)))$. By Lemma 6.9 there exists an intuitionistic model $\mathfrak{I} = \langle\, A, R', m' \,\rangle$ such that $\mathfrak{I}, \nu(x) \not\models_{L(\Gamma)} \psi$, and thus $\not\models_{L(\Gamma)} \psi$. $\Box$

Definition 6.51 We define the calculus $Int\text{-}FLC_\Gamma$ by the condition

$$\begin{aligned} &\vdash_{Int\text{-}FLC_\Gamma} tQt' \\ &\quad\Longleftrightarrow \quad \vdash_{Int\text{-}FLC} t \ \mathsf{u}1;\overline{T_{\nabla,\langle\rangle}(\gamma_1)\cdot \ldots \cdot T_{\nabla,\langle\rangle}(\gamma_k)};1 \ + \ 1'\mathsf{u};Q \ t' \ . \end{aligned}$$

Theorem 6.22 *Given $x \in UreVar$ and $y \in CompVar$,*

$$\models_{FL_\Gamma} xQy \qquad \Longleftrightarrow \qquad \vdash_{Int\text{-}FLC_\Gamma} xQy \ .$$

Proof

$$\begin{aligned} & \vdash_{Int\text{-}FLC_\Gamma} xQy \\ \Longleftrightarrow \ & \{\text{by Def. 6.51}\} \\ & \vdash_{Int\text{-}FLC} x \ \mathsf{u}1;\overline{T_{\nabla,\langle\rangle}(\gamma_1)\cdot \ldots \cdot T_{\nabla,\langle\rangle}(\gamma_k)};1 \ + \ 1'\mathsf{u};Q \ y \\ \Longleftrightarrow \ & \{\text{by Thms. 6.15 and 6.16}\} \\ & \models_{FL^{\mathcal{K}}} x \ \mathsf{u}1;\overline{T_{\nabla,\langle\rangle}(\gamma_1)\cdot \ldots \cdot T_{\nabla,\langle\rangle}(\gamma_k)};1 \ + \ 1'\mathsf{u};Q \ y \end{aligned}$$

$$
\begin{array}{ll}
\iff & \{\text{by Def. } FL^{\mathcal{K}}\} \\
& \forall \mathcal{M} \in \mathcal{K}, \mathcal{M} \models_{FL'} x \ \cup 1 ; \overline{T_{\nabla,\langle\rangle}(\gamma_1) \cdot \ldots \cdot T_{\nabla,\langle\rangle}(\gamma_k) ; 1} + 1'_{\cup} ; Q \ y \\
\iff & \{\text{by Def. } FL'\} \\
& \models_{\mathcal{K}} \cup 1 ; \overline{T_{\nabla,\langle\rangle}(\gamma_1) \cdot \ldots \cdot T_{\nabla,\langle\rangle}(\gamma_k) ; 1} + 1'_{\cup} ; Q = \cup 1 \\
\iff & \{\text{by Cor. } 6.4\} \\
& \{ T_{\nabla,\langle\rangle}(\gamma) = 1 : \gamma \in \Gamma \} \models_{\mathcal{K}} Q = \cup 1 \\
\iff & \{\text{by Def. } \mathcal{K}_\Gamma\} \\
& \models_{\mathcal{K}_\Gamma} Q = \cup 1 \\
\iff & \{\text{by Def. } FL_\Gamma\} \\
& \models_{FL_\Gamma} xQy. \qquad \square
\end{array}
$$

Let us assume we are working in an intermediate logic with Kripke semantics, whose accessibility relation is constrained by a finite set of sentences Γ. In order to establish the validity of a formula α, it is enough to verify that α holds in every Kripke model in which all the sentences in Γ hold.

An alternative procedure using the calculus *Int-FLC*$_\Gamma$ is the following:

(1) Translate the set of sentencesΓ to a set of relation designations.
(2) Translate formula α to a relation designation.
(3) Apply the deduction theorem (Cor. 6.4) in order to obtain a single relation designation R.
(4) Use the proof system *Int-FLC*$_\Gamma$ in order to prove the formula xRy.

Joining Thms. 6.21 and 6.22, we obtain the following corollary.

Corollary 6.5 *Let $\psi \in$ IntFor. Then, given individual variables $x \in$ UreVar and $y \in$ CompVar,*

$$\models_{L(\Gamma)} \psi \qquad \iff \qquad \vdash_{\textit{Int-FLC}_\Gamma} xT_I(\psi)y \ .$$

Proof By Thm. 6.21,

$$\models_{L(\Gamma)} \psi \qquad \iff \qquad \models_{FL_\Gamma} xT_I(\psi)y \ . \tag{6.6}$$

By Thm. 6.22,

$$\models_{FL_\Gamma} xT_I(\psi)y \qquad \iff \qquad \vdash_{\textit{Int-FLC}_\Gamma} xT_I(\psi)y \ . \tag{6.7}$$

Finally, by (6.6) and (6.7),

$$\models_{L(\Gamma)} \psi \qquad \iff \qquad \vdash_{\textit{Int-FLC}_\Gamma} xT_I(\psi)y \ .$$

$\square$

6.12.3 *Method 3*

Translate constraints from Γ and the formula α to be proved, into relational designations. Verify whether the term obtained from α is derivable from the terms obtained from the members of Γ. In order to test this derivability we apply equational means of reasoning within the theory of fork algebras as in the first part of the chapter (see also [M. Frias et al. (1997)c]).

This page is intentionally left blank

Chapter 7

A Calculus for Program Construction

7.1 Introduction

The most problematic part of the process of software development is maintenance. Usually, even after a system is considered to be finished by the team of developers, a long time goes by before the software is fully operational. This is due in part to the fact that most programmers tend to make mistakes when programming. When run for the first time, their programs usually do not terminate, or return unexpected results. Even when a program is accepted by the team of developers as a fine working program, usually the performance needs to be improved. Improving the performance results in code modification, and a new need for testing the program. Opposed to the previous idea is the notion of *formal program construction.* In a formal setting, we can reason about logical or algebraic properties of programs that are difficult or impossible to express in an informal setting. At the heart of formal program construction is the ability to calculate programs in much the same way as a mathematician solves a set of equations or proves a theorem constructively. As a consequence, a formally constructed program is correct by construction with respect to its specifications, and its derivation is a proof of its correctness.

A particular class of formalisms for program construction are those based on formal calculi. These formalisms have a logical basis. Specifications are formulas, and a certain subset of those formulas is considered to have an algorithmic meaning and is thus interpreted as programs from functional, logic, or imperative programming languages. Derivation rules resemble inference rules from logical frameworks.

Fork algebras arose in computer science when looking for a calculus for program construction based on binary relations. Programs are to be thought of as the relation they establish between input and output data. Functional calculi have been extensively used for program construction [J. Backus (1978); R. Bird (1986); R. Bird et al. (1993); R. Burstall et al. (1977); J. Jeuring (1994); L. Meertens (1987)], but their specification language is not declarative enough. Specifications are partial recursive functions (and thus programs in functional languages), that are optimized along the derivation process. Unfortunately, finding the functional specifications is not always easy, and a gap wider than is desirable is left between the original problem and its specification. Relations present some advantages over functions. Relations allow some operations, such as the converse and complement, that are not even defined in functional frameworks. These operations make relations more expressive than functions, and thus relational frameworks allow for more declarative specifications. Relations have been used in program construction for some time. In [R. C. Backhouse et al. (1993); R. C. Backhouse et al. (1991); R. Bird et al. (1997); H. Doornbos et al. (1997)], relations are introduced using a categorical approach, and used for defining a calculus for program construction. In [R. Berghammer et al. (1997)] and the references therein, the relational calculus is used for the construction of graph algorithms. In [B. Möller (1991); B. Möller (1993)], a framework for program construction based on relations (not necessarily binary ones) is presented, with applications in the derivation of graph and pointer algorithms. Other applications of binary relations in computer science are reported in the book [C. Brink et al. (1997)].

In this chapter a calculus for program construction based on fork algebras and generic algorithms is presented. The equational calculus of fork algebras has been used in program construction for some time [G. A. Baum et al. (1996); M. Frias et al. (1994); M. Frias et al. (1993); M. Frias et al. (1996); M. Frias et al. (1998); M. Frias et al. (1997)a; A. Haeberer et al. (1993)b; A. Haeberer et al. (1991)]. Here, the formalism adopted is the first-order theory of fork algebras. First-order formulas over relations are used in order to describe design strategies (such as case analysis, trivialization, divide-and-conquer, backtracking, etc.). Generic specifications using parameters describe a class of problems rather than a single problem. When the parameters satisfy enough properties, then it is possible to find a generic algorithm (also containing parameters) which solves the whole class of problems. The methodology to be presented here allows us to derive generic algorithms following some design strategies, starting from generic specifications. Examples will be presented on how to derive generic algorithms from generic specifications, following the presented design strategies.

The chapter is organized as follows. In Section 7.2 the notion of *filter* is introduced and the relationship among filters, sets and guarded commands is analyzed. In Section 7.3 the relational implication is presented and some of its useful properties for derivation of recursive programs are stated. In Section 7.4 the usefulness of the expressiveness and representability results with respect to the process of program construction is discussed. In Section 7.5 the methodology for program construction is described. In Section 7.6 several examples of program derivations are presented. Finally, in Section 7.8 the approach presented here is compared with previous approaches.

7.2 Filters and Sets

Filters are partial identities, i.e., relations F satisfying the condition $F \leq 1'$. The reason why they are called *filters* is because they can be used as strainers, filtering the information that reaches the input of a relation. For example, if F is a filter and R is an arbitrary relation, $F;R$ restricts the input of R to F. There is a clear relationship between filters from algebras of binary relations and sets. A filter F univocally characterizes a set, namely, the set $\{ x : xFx \}$. Also, given a set S, it univocally characterizes a filter, namely, the binary relation $\{ \langle x, x \rangle : x \in S \}$. We will denote the filter associated to a set S by $1'_S$. Given a filter F, by $\neg F$ we denote the term $\overline{F} \cdot 1'$. Notice that if $F = 1'_S$ for some set S, then $\neg F = 1'_{\overline{S}}$, the filter associated to the complement of the set S. This is justified by properties 1 and 2 in Thm. 7.1.

Theorem 7.1 *The following properties of filters are valid in all relation algebras:*

(1) If F is a filter, then $F + \neg F = 1'$,
(2) If F is a filter, then $F \cdot \neg F = 0$,
(3) $\neg Dom(R);1 = \overline{R;1}$,
(4) If F is functional, then

$$F;\neg Dom(R);1 = (Dom(F) \cdot \neg Dom(F;R));1 .$$

Proof

1.

$$\begin{aligned} F+\neg F &= F+\left(\overline{F}\cdot 1'\right) && \text{(by Def. } \neg F\text{)}\\ &= (F\cdot 1') + \left(\overline{F}\cdot 1'\right) && \text{(by } F \text{ filter)}\\ &= \left(F+\overline{F}\right)\cdot 1' && \text{(by BA)}\\ &= 1\cdot 1' && \text{(by BA)}\\ &= 1'. && \text{(by BA)} \end{aligned}$$

2.

$$\begin{aligned} F\cdot\neg F &= F\cdot\left(\overline{F}\cdot 1'\right) && \text{(by Def. } \neg F\text{)}\\ &= \left(F\cdot\overline{F}\right)\cdot 1' && \text{(by BA)}\\ &= 0. && \text{(by BA)} \end{aligned}$$

3. Let us show that

$$\neg Dom\,(R)\,;1\ \cdot\ R;1 = 0, \quad \text{and} \quad \neg Dom\,(R)\,;1\ +\ R;1 = 1\ .$$

$$\begin{aligned} &\neg Dom\,(R)\,;1\ \cdot\ R;1\\ &\quad = \neg Dom\,(R)\,;1\ \cdot\ Dom\,(R)\,;R;1 && \text{(by Thm. 2.3.11)}\\ &\quad = Dom\,(R)\,;\,(\neg Dom\,(R)\,;1\ \cdot\ R;1) && \text{(by Thm. 2.3.22)}\\ &\quad = (Dom\,(R)\,;\neg Dom\,(R)\,;1)\cdot(Dom\,(R)\,;R;1) && \text{(by Thm. 2.3.17)}\\ &\quad = (Dom\,(R)\cdot\neg Dom\,(R))\,;1\ \cdot\ Dom\,(R)\,;R;1 && \text{(by Thm. 2.3.7)}\\ &\quad = 0;1\ \cdot\ Dom\,(R)\,;R;1 && \text{(by Thm. 7.1.2)}\\ &\quad = 0\ \cdot\ Dom\,(R)\,;R;1 && \text{(by Thm. 2.3.1)}\\ &\quad = 0. && \text{(by BA)} \end{aligned}$$

$$\begin{aligned} &\neg Dom\,(R)\,;1\ +\ R;1 && (7.1)\\ &\qquad = \neg Dom\,(R)\,;1\ +\ Dom\,(R)\,;1 && \text{(by Thm. 2.3.14)}\\ &\qquad = (\neg Dom\,(R)+Dom\,(R))\,;1 && \text{(by Ax. 2)}\\ &\qquad = 1';1 && \text{(by Thm. 7.1.1)}\\ &\qquad = 1. && \text{(by Ax. 5)} \end{aligned}$$

4.

$$\begin{aligned} F;\neg Dom(R);1 &= F;\overline{R;1} && \text{(by 3)}\\ &= Dom(F);\overline{F;R;1} && \text{(by Thm. 2.3.19)}\\ &= Dom(F);\neg Dom(F;R);1 && \text{(by 3)}\\ &= (Dom(F)\cdot\neg Dom(F;R));1. && \text{(by Thm. 2.3.7)} \end{aligned}$$

□

Filters are used in program construction to model guards in if-then-else–like constructs, or guards from case-like constructs. Let us consider the following example.

```
Function $IsZero(n$ : Nat) : Boolean
Begin
    If $n = 0$ Then
       $\leftarrow$ true
    Else
       $\leftarrow$ false
    End If
End.
```

This function can be represented relationally by the following equation:

$$IsZero = 1'_0;\mathsf{C}_{\mathbf{true}} + 1'_{\succ 0};\mathsf{C}_{\mathbf{false}}$$

where:

(1) $1'_0$ is the filter $\{\langle 0,0\rangle\}$,
(2) $1'_{\succ 0}$ is the filter $\{\langle x,x\rangle : x > 0\}$,
(3) $\mathsf{C}_{\mathbf{true}}$ is the constant relation $\{\langle x, true\rangle : x \in U\}$,
(4) $\mathsf{C}_{\mathbf{false}}$ is the constant relation $\{\langle x, false\rangle : x \in U\}$.

7.3 The Relational Implication

In this section two operations on binary relations called *right residual* and *relational implication* respectively are defined. As we will see in Section 7.6, the relational implication is closely related to the specification of problems in fork algebras. We define the right residual of relations R and S (denoted

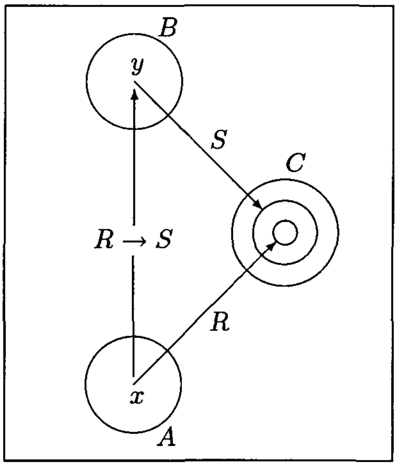

Fig. 7.1 The relational implication.

by $R\backslash S$) in terms of the relational operators previously defined, by

$$R\backslash S = \overline{\breve{R};\overline{S}}\,. \tag{7.2}$$

The set theoretical definition of the right residual is given by the following formula

$$R\backslash S = \{\langle x, y\rangle : \forall z\,(zRx \Rightarrow zSy)\}\ .$$

The abstract definition of the relational implication of relations R and S, is given by the equality

$$R \rightarrow S = \overline{R;\overline{\breve{S}}}, \tag{7.3}$$

while its set theoretical definition (see Fig. 7.1 for a graphical interpretation) is given by

$$R \rightarrow S = \{\langle x, y\rangle : \forall z\,(xRz \Rightarrow ySz)\}\ .$$

Figure 7.1 shows that a pair $\langle x, y\rangle$ is related via $R \rightarrow S$ whenever the range of x through R (the smallest circle) is contained in the range of y through S (the medium-sized circle). From (7.2) and (7.3) it is immediate that the relational implication is definable in terms of the right residual.

More explicitly, the following is true in every relation algebra:

$$R \rightharpoondown S = \breve{R} \backslash \breve{S}.$$

Since the relation algebraic characterization of the relational implication is non algorithmic (complement does not have nice properties when applied over a composition), some properties are presented next that will be very useful in the derivation of algorithms from relational specifications.

Besides some simple properties, such as

$$(P+Q) \rightharpoondown R = (P \rightharpoondown R) \cdot (Q \rightharpoondown R) \tag{7.4}$$

and

$$P \rightharpoondown (Q \cdot R) = (P \rightharpoondown Q) \cdot (P \rightharpoondown R), \tag{7.5}$$

(which follow directly from the definition), some more elaborated properties that lead to recursive relational expressions for computing the relational implication are presented.

Lemma 7.1 *For any relations P and Q,*

$$\neg Dom\,(P)\,;(P \rightharpoondown Q) = \neg Dom\,(P)\,;1\ .$$

Proof The inequality $\leq$ is trivial, due to the fact that $P \rightharpoondown Q \ \leq\ 1$. For the inequality $\geq$ we have

$$\begin{aligned}
&\neg Dom\,(P); (P \rightharpoondown Q) && (7.6)\\
&\qquad = \neg Dom\,(P)\,;\overline{P;\overline{\breve{Q}}} && \text{(by (7.3))}\\
&\qquad = \neg Dom\,(P)\,;\overline{Dom\,(P)\,;P;\overline{\breve{Q}}} && \text{(by Thm. 2.3.11)}\\
&\qquad \geq \neg Dom\,(P)\,;\overline{Dom\,(P)\,;1} && \text{(by monotonicity and BA)}\\
&\qquad = \neg Dom\,(P)\,;\neg Dom\,(P)\,;1 && \text{(by Thm. 7.1.3)}\\
&\qquad = \neg Dom\,(P)\,;1. && \text{(by Thm. 2.3.7)}
\end{aligned}$$

□

Lemma 7.2 *For any relations P, Q and R,*

$$\begin{aligned} &Dom(P+Q)\,;((P+Q)\rightarrow R)\\ &\quad= (Dom(P)\cdot Dom(Q));((P\rightarrow R)\cdot(Q\rightarrow R))\\ &\qquad+ (Dom(P)\cdot\neg Dom(Q));(P\rightarrow R)\\ &\qquad\qquad+ (\neg Dom(P)\cdot Dom(Q));(Q\rightarrow R)\ . \end{aligned}$$

Proof By Thm. 2.3.12,

$$\begin{aligned} &Dom(P+Q)\,;((P+Q)\rightarrow R)\\ &\quad= Dom(P)\,;((P+Q)\rightarrow R)\\ &\qquad+ Dom(Q)\,;((P+Q)\rightarrow R)\ . \end{aligned} \tag{7.7}$$

Let us now concentrate on the term

$$Dom(P)\,;((P+Q)\rightarrow R)\ .$$

$$\begin{aligned} & Dom(P)\,;((P+Q)\rightarrow R)\\ = \quad & \{\text{by } (7.4)\}\\ & Dom(P)\,;((P\rightarrow R)\cdot(Q\rightarrow R))\\ = \quad & \{\text{by Thm. } 7.1.1\}\\ & Dom(P)\,;((P\rightarrow R)\cdot((Dom(Q)+\neg Dom(Q))\,;(Q\rightarrow R)))\\ = \quad & \{\text{by Ax. 2 and BA}\}\\ & Dom(P)\,;((P\rightarrow R)\cdot(Dom(Q)\,;(Q\rightarrow R)))\\ & \quad + Dom(P)\,;((P\rightarrow R)\cdot(\neg Dom(Q)\,;(Q\rightarrow R)))\\ = \quad & \{\text{by Thm. 2.3.22 and Lemma 7.1}\}\\ & Dom(P)\,;Dom(Q)\,;((P\rightarrow R)\cdot(Q\rightarrow R))\\ & \quad + Dom(P)\,;\neg Dom(Q)\,;(P\rightarrow R)\\ = \quad & \{\text{by Thm. } 2.3.7\}\\ & (Dom(P)\cdot Dom(Q));((P\rightarrow R)\cdot(Q\rightarrow R))\\ & \quad + (Dom(P)\cdot\neg Dom(Q));(P\rightarrow R). \end{aligned}$$

Thus,

$$\begin{aligned} &Dom(P)\,;((P\rightarrow R)\cdot(Q\rightarrow R))\\ &\quad= (Dom(P)\cdot Dom(Q));((P\rightarrow R)\cdot(Q\rightarrow R))\\ &\qquad+ (Dom(P)\cdot\neg Dom(Q));(P\rightarrow R)\ . \end{aligned} \tag{7.8}$$

Reasoning along the same lines allows us to prove that

$$\begin{aligned} &Dom\,(Q)\,;\,((P+Q) \rightarrow R) \\ &\qquad = (Dom\,(P) \cdot Dom\,(Q));\,((P \rightarrow R)\cdot(Q \rightarrow R)) \\ &\qquad\qquad + (\neg Dom\,(P) \cdot Dom\,(Q));(Q \rightarrow R)\ . \end{aligned} \tag{7.9}$$

The lemma finally follows from (7.7), (7.8) and (7.9). □

Lemma 7.3 *If A is a functional relation, then*

$$Dom\,(A)\,;\,(A \rightarrow P) = A;\breve{P}\ .$$

Proof

$$\begin{aligned} Dom\,(A)\,;\,(A \rightarrow P) &= Dom\,(A)\,;\overline{A;\overline{\breve{P}}} && \text{(by (7.3))} \\ &= A;\overline{\overline{\breve{P}}} && \text{(by Thm. 2.3.19)} \\ &= A;\breve{P}. && \text{(by BA)} \end{aligned}$$

□

Lemma 7.4 *If B is a functional relation, then*

$$Dom\,(B)\,;\,(B;P\ \rightarrow\ Q) = B;\,(P \rightarrow Q)\ .$$

Proof

$$\begin{aligned} Dom\,(B)\,;\,(B;P\ \rightarrow\ Q) &= Dom\,(B)\,;\overline{B;P;\overline{\breve{Q}}} && \text{(by (7.3))} \\ &= B;\overline{P;\overline{\breve{Q}}} && \text{(by Thm. 2.3.19)} \\ &= B;\,(P \rightarrow Q)\,. && \text{(by (7.3))} \end{aligned}$$

□

Lemma 7.5 *Let A and B be functional relations. Let*

$$P = A\, +\, B;P \quad \textit{and} \quad T = P \rightarrow Q\ .$$

Moreover, let us assume that

$$Dom\,(P) = Dom\,(A) + Dom\,(B) \qquad \textit{and} \qquad Dom\,(A) \cdot Dom\,(B) = 0\ .$$

Then,

$$Dom\,(P)\,;T = A;\breve{Q} + B;T\ .$$

Proof The lemma follows from Lemmas. 7.2, 7.3 and 7.4. □

In a similar way to Lemma 7.5, we obtain a proof for Lemma 7.6 below.

Lemma 7.6 *Let A, B and C be functional relations. Let*

$$P = A + B;P + C;P \quad \textit{and} \quad T = P \rightarrow Q\ .$$

Moreover, let us suppose that $Dom\,(A)$, $Dom\,(B)$ and $Dom\,(C)$ are pairwise disjoint. Then,

$$Dom\,(P)\,;T = A;\breve{Q} + B;T + C;T\ .$$

From Lemmas 7.5 and 7.6 we see that the recursiveness of the relation P allows us to obtain a recursive specification for the relation T.

Lemma 7.7 *For every relation R,*

$$\breve{P};(P \rightarrow R) \leq \breve{R}\ .$$

Proof

$$\begin{aligned}
\breve{P};(P \rightarrow R) \leq \breve{R} &\iff (\breve{P};(P \rightarrow R))\cdot\overline{\breve{R}} = 0 && \text{(BA)}\\
&\iff (P;\overline{\breve{R}})\ \cdot\ (P \rightarrow R) = 0 && \text{(by Ax. 7)}\\
&\iff P;\overline{\breve{R}}\ \cdot\ \overline{P;\overline{\breve{R}}} = 0 && \text{(by (7.3))}\\
&\iff 0 = 0. && \text{(BA)}
\end{aligned}$$

□

Lemma 7.8 *Let R be an antisymmetric relation (i.e., $R\cdot\breve{R} \leq 1'$), then, the relation $P\cdot(P \rightarrow R)$ is functional for all relation P.*

Proof In order to show that $P\cdot(P \rightarrow R)$ is functional, we will prove that

$$(P\cdot(P \rightarrow R))^{\smile};(P\cdot(P \rightarrow R)) \leq 1'\ .$$

$$
\begin{aligned}
&(P\cdot(P\to R))^{\smile};(P\cdot(P\to R)) \\
&= \left(\breve{P}\cdot(P\to R)^{\smile}\right);(P\cdot(P\to R)) && \text{(by Thm. 2.3.5)} \\
&\leq \left(\breve{P};(P\cdot(P\to R))\right)\cdot((P\to R)^{\smile};(P\cdot(P\to R))) && \text{(by Thm. 2.3.16)} \\
&\leq \left(\breve{P};(P\to R)\right)\cdot((P\to R)^{\smile};P) && \text{(by monotonicity)} \\
&= \left(\breve{P};(P\to R)\right)\cdot\left(\breve{P};(P\to R)\right)^{\smile} && \text{(by Ax. 6)} \\
&\leq \breve{R}\cdot\breve{\breve{R}} && \text{(by Lemma 7.7)} \\
&= R\cdot\breve{R} && \text{(by BA and Ax. 4)} \\
&\leq 1\text{'}. && \text{(by Hyp.)}
\end{aligned}
$$

□

7.4 Representability and Expressiveness in Program Construction

As a consequence of Thm. 4.3, the first-order theories of AFA and PFA are the same, and thus a natural semantics can be attributed to first-order formulas over abstract relations in terms of binary relations. This is a very important property in a calculus for program construction. The equivalence between the first-order theories of PFA and AFA guarantees that any first-order property valid for proper fork algebras can be proved syntactically from the axioms describing abstract fork algebras. This has a direct application in program construction. Let us consider an intermediate step in a derivation of an algorithm from a relational specification S_0. The derivation has a shape

$$S_0, S_1, \ldots, S_k,$$

where for all i, $1 \leq i \leq k$, S_i is obtained from $S_0, \ldots, S_{i-1}$ by means of the derivation rules. If S_k is still not the algorithm we are looking for, then further steps must be performed. If resorting to thinking about binary relations shows that a valid first-order property allows S_k to evolve to a new expression E (which is closer to the intended algorithm), then the representation theorem guarantees that a syntactic proof $S_k, S_{k+1}, \ldots, E$ exists, allowing us to reach the formula E from S_k. This shows that the heuristics arising from considering *concrete* binary relations can be employed throughout the process of program derivation using the rules and axioms of the calculus for *abstract* fork algebras. Another important property stems from the fact that only a finite number of axioms are necessary for describing the class of abstract fork algebras. Thus, the syntactic proofs mentioned above can be more easily performed with the assistance of a computer system.

Regarding the expressiveness of fork algebras, it was proved in Ch. 4.2 that first-order theories can be interpreted as equational theories in fork algebras. Theorem 5.6 shows that a wide class of problems (at least those that can be described in first-order logic) can be specified in the equational calculus of fork algebras. Moreover, the abstract relational specification can be obtained algorithmically from the first-order specification by using the mapping $T_{\nabla,\sigma}$.

7.5 A Methodology for Program Construction

In this section the outlines of a methodology for program construction based on fork algebras using generic algorithms (program schemes) are presented. As will be shown in the next section, there is a useful relationship between the structure or form of a generic relational specification and a generic algorithm (a set of parameterized 'algorithmic' equations) to compute this specification.

The starting point of the methodology is a formal specification of the problem to be solved. In this case, first-order logic with equality will be used as the specification language, because it is a simple formal language that is taught in most computer science courses. Along the description of the methodology, an example will be outlined that will be thoroughly discussed in Section 7.6. For the examples, the notion of *generator* will be required. Intuitively, generators retrieve the components (members) of elements from structured types. Examples of generators that will be used in Section 7.6 include retrieving the elements of a list, retrieving the elements of a tree, retrieving all the sublists of a list, etc.

Description of the example problem:

> Let $\mathsf{S}(\beta)$ be a structured type, and let $G \subseteq \mathsf{S}(\beta) \times \beta$ be a generator. Select those generated elements that satisfy
>
> (1) a condition c_1, and
>
> (2) a condition c_2 with respect to all the generated ele-

ments that satisfy the condition c_1.

In the previous description S could be a type functor [R. Bird et al. (1997)]. The sample problem P can be specified by the first-order formula:

$$P(x,y) \iff G(x,y) \wedge c_1(y) \ \wedge \ \forall z\,(G(x,z) \wedge c_1(z) \ \Rightarrow \ c_2(z,y)) \ . \quad (7.10)$$

Notice how close formula (7.10) is to the natural language description of problem P, thus giving a totally declarative specification.

Once a first-order specification of a problem is given, a relational specification must be obtained. In order to obtain this specification, we can proceed in one of the following two ways. Applying Thm. 5.6, from a first-order specification φ and using the mapping $T_{\nabla,\sigma}$ we will obtain a relational term $T_{\nabla,\sigma}(\varphi)$ that captures the meaning of problem P. Unfortunately, the term resulting from applying the mapping $T_{\nabla,\sigma}$ is not always very adequate with respect to the process of program derivation. The second method (the one to be used here), consists of reducing the first-order formula φ into an equation e_φ using *the set-theoretical definition of the relational operators*. Notice that, given a formula, there are many ways in which this reduction can be done. For the example it is possible to proceed as follows.

Define P', C_1, C_2 and G' as new binary relations. Intuitively, the binary relation P' will stand for the predicate P in the sense that

$$xP'y \qquad \iff \qquad P(x,y),$$

C_1 will stand for the unary predicate c_1 in the sense that

$$xC_1x \qquad \iff \qquad c_1(x),$$

C_2 will stand for c_2 in the sense that

$$xC_2y \qquad \iff \qquad c_2(x,y),$$

and G' stands for G in the sense described by the formula

$$xG'y \qquad \iff \qquad G(x,y) \ .$$

Notice that C_1 is a filter, and this will in general be the method used for representing unary predicates.

From the previous definitions, it is easy to check that

$$P' = G';C_1 \ \cdot \ (G';C_1 \ \rightarrow \ C_2) \ .$$

Once a generic relational specification is obtained (generic in the sense that no assumption is made about the relations C_1 or C_2), we will choose a design strategy that will guide the process of deriving an algorithm from this specification. In programming in general, examples of design strategies include case analysis, trivialization, divide-and-conquer, backtracking, and many more [A. V. Aho et al. (1983); R. Bird et al. (1997); H. Partsch (1990); D. Smith (1985); D. Smith (1987)]. In the methodology presented here, the first-order language of fork algebras will be used to express such design strategies (recall that from Thm. 4.3 and the discussion following it, formulas from the first-order theory of fork algebras have a standard semantics in terms of concrete binary relations). A trivial example of a design strategy is case analysis (C_A). A problem is said to be solved using this strategy if the domain of the problem can be partitioned, let us say, in k parts $D_1, \ldots, D_k$, and we find k algorithms $A_1, \ldots, A_k$ such that A_i solves the given problem when its domain is restricted to the part D_i. This can be more simply and formally stated by the following formula over relations:

$$C\_A(R, R_1, \ldots, R_k) \iff \bigwedge_{1 \le i < j \le k} Dom\,(R_i) \cdot Dom\,(R_j) = 0 \;\wedge\; R = \sum_{i=1}^{k} R_i\,. \quad (7.11)$$

Formula (7.11) is to be read as follows:

> 'Problem R is solved by case-analysis using problems $R_1, \ldots, R_k$'.

Notice that (7.11) provides the means to solve problem R, that is, given an input a for R there is to find R_i such that $a \in Dom\,(R_i)$ and then compute $R_i(a)$.

The strategy of trivialization ($Triv$) is a particular instance of case analysis where one of the subproblems is assumed to be easy to solve. Easy subproblems are, for example, those whose solution does not depend on the original problem (non-recursive parts), or for which a solution is at hand. We then have

$$Triv(R, R_0, R_1, \ldots, R_k) \iff C\_A(R, R_0, R_1, \ldots, R_k) \;\wedge\; Easy(R_0). \quad (7.12)$$

The relations $R_0, R_1, \ldots, R_k$ are usually determined by properties of the problem domain. In general, domains allow for 'natural' partitions (empty and nonempty lists, trees of height 1 or greater, etc.). In [H. Partsch (1990), pp. 201–202], these partitions are obtained by introducing tautologies. Hence, the following heuristic can be used in order to determine relations $R_0, R_1, \ldots, R_k$.

Heuristic 7.1 Let D be a domain (type) characterized by the partial identity 1'_D. Assume there are identities $1\text{'}_0, 1\text{'}_1, \ldots, 1\text{'}_k$ such that $1\text{'}_D = 1\text{'}_0 + 1\text{'}_1 + \cdots + 1\text{'}_k$. In order to find the problem (relation) R_i, $1 \leq i \leq k$, define $R_i = 1\text{'}_i ; R$, provided $Easy(1\text{'}_0 ; R)$ holds.

Formula (7.12) is to be read as follows:

'Problem R is solved by trivialization using relations $R_0, R_1, \ldots, R_k$ with easy R_0'.

Formula (7.12) provides a means to solve problem R, namely, using case-analysis in the case of inputs outside the domain of R_0, and the simple problem R_0 otherwise. The difference between trivialization and case-analysis is that in the latter we may need to derive solutions for all subproblems, while in the former, problem R_0 presents a definite improvement in the derivation process.

The strategy of recomposition (*Recomp*) is defined by the formula

$$Recomp(R, Split, Q_1, \ldots, Q_k, Join) \iff R = Split ; (Q_1 \otimes \cdots \otimes Q_k) ; Join, \quad (7.13)$$

where the relations *Split* and *Join* stand for programs so that the first one effectively decomposes the data, and the latter combines the results of $Q_1, \ldots, Q_k$ in order to provide a solution for R.

By joining the strategies of recomposition (cf. (7.13)) and trivialization, we obtain the following formalization of divide-and-conquer:

$$D\&C(R, R_0, Split, Q_1, \ldots, Q_k, Join) \iff \exists Q\, (Triv(R, R_0, Q) \ \wedge \ Recomp(Q, Split, Q_1, \ldots, Q_k, Join)),$$

where the variable R may appear inside some of the terms $Q_1, \ldots, Q_k$, but does not affect the term R_0.

Generally, relations $Q_1, \ldots, Q_k$ will be either 1' or the relation R itself. This is supported for example by Smith [D. Smith (1985)] in his schema of divide-and-conquer algorithms. How to find the parameters for the *D&C* strategy is then explained by the following heuristic.

Heuristic 7.2 Among the relations $Q_1, \ldots, Q_k$, those that will take the value R are obtained by folding of the definition of R. After no more foldings are possible, the remaining relations are to be set to 1'. If we are heading correctly towards a divide-and-conquer algorithm, at this point we should be dealing with a term that looks approximately like

$$S_0 ; \left(\begin{array}{c} S_1 ; R ; J_1 \\ \otimes \\ \vdots \\ \otimes \\ S_i ; R ; J_i \\ \otimes \\ T_{i+1} \\ \otimes \\ \vdots \\ \otimes \\ T_k \end{array} \right) ; J_0,$$

where R does not occur in any of $T_{i+1}, \ldots, T_k$. The last step is rewriting T_m $(i < m \leq k)$ as $S_m ; J_m$, with S_m the 'Split' part and J_m the 'Join' part. Finally, let $\mathit{Split} := S_0 ; (S_1 \otimes \cdots \otimes S_k)$ and $\mathit{Join} := (J_1 \otimes \cdots \otimes J_k) ; J_0$.

There are many interesting problems in computer science for which efficient solutions are achieved using divide-and-conquer — for example, searching, sorting, matrix multiplication, Fourier's transform, etc. There are also many problems whose solution is closely related to the strategy of divide-and-conquer but that cannot be put into the schema. Let us consider the following problem *Subtree* as an example:

> Given a tree t and a node n, *Subtree* retrieves the subtree of t whose root is the node n.

If we blindly apply the schema of divide-and-conquer, making two recursive calls for treating the left and right subtrees, one of these calls will always be undefined — namely, that call that treats the subtree that does not contain the node n. Thus, before making the recursive call it must be decided what the parts of the original datum for which recursively calling

Subtree will yield an output are. In this particular example, it suffices to check which subtree of t contains the node n. These kind of problems suggest the definition of a generalized version of the strategy of divide-and-conquer. The strategy is defined as follows:

$$\begin{aligned}
&\mathit{GenD\&C}\,(R, R_0, D_1, \mathit{Split}_1, A_1, \mathit{Join}_1, \ldots, D_k, \mathit{Split}_k, A_k, \mathit{Join}_k) \iff \\
&\quad \bigwedge_{1\le i\le k} D_i \le 1' \ \wedge\ \mathit{Triv}\,(R, R_0, D_1;R, \ldots, D_k;R)\ \wedge \\
&\qquad \bigwedge_{1\le i\le k} \exists Q_{i_1}, \ldots, Q_{i_{j_i}} (A_i = Q_{i_1} \otimes \cdots \otimes Q_{i_{j_i}}\ \wedge \\
&\qquad\qquad \mathit{Recomp}(D_i;R, \mathit{Split}_i, Q_{i_1}, \ldots, Q_{i_{j_i}}, \mathit{Join}_i))\ . \qquad (7.14)
\end{aligned}$$

Notice that when k equals 1, *GenD&C* becomes *D&C*. Once the identities D_i $(1 \le i \le k)$ are identified, heuristic 7.2 can be applied to find the remaining parameters. In order to find the identities D_i, we use the following heuristic.

Heuristic 7.3 Assume the specification contains some subexpression with shape $(G;R_1 \rightarrow R_2)$ for relations G, R_1 and R_2. Notice that this is a reasonable assumption because relational implications naturally appear from first-order specifications containing universal quantifiers. Assume also that $G = f_1;G_1 + \cdots + f_k;G_k$, where f_i is a functional relation for all i, $1 \le i \le k$ (G could be for instance a generator). Let us see the case for $k = 2$. Then, we reason as follows:

$$\begin{aligned}
& G;R_1 \rightarrow R_2 \\
= & \ \{\text{by Def. } G\} \\
& (f_1;G_1 + f_2;G_2)\,;R_1 \rightarrow R_2 \\
= & \ \{\text{by Ax. 2 and (7.4)}\} \\
& (f_1;G_1;R_1 \rightarrow R_2) \cdot (f_2;G_2;R_1 \rightarrow R_2) \\
= & \ \{\text{by Lemmas 7.1 and 7.4}\} \\
& (f_1;(G_1;R_1 \rightarrow R_2) + \neg \mathit{Dom}\,(f_1)\,;1) \\
& \quad \cdot (f_2;(G_2;R_1 \rightarrow R_2) + \neg \mathit{Dom}\,(f_2)\,;1)\,.
\end{aligned}$$

Distributing $+$ over $\cdot$, we obtain the terms:

(1) $f_1;(G_1;R_1 \rightarrow R_2) \cdot f_2;(G_2;R_1 \rightarrow R_2)$,
(2) $f_1;(G_1;R_1 \rightarrow R_2) \cdot \neg \mathit{Dom}\,(f_2)\,;1$,
(3) $\neg \mathit{Dom}\,(f_1)\,;1 \cdot f_2;(G_2;R_1 \rightarrow R_2)$, and
(4) $\neg \mathit{Dom}\,(f_1)\,;1 \cdot \neg \mathit{Dom}\,(f_2)\,;1$.

The domains of these relations are contained, respectively, in the filters

$$\begin{array}{ll} Dom\,(f_1)\cdot Dom\,(f_2)\,, & Dom\,(f_1)\cdot\neg Dom\,(f_2)\,, \\ \neg Dom\,(f_1)\cdot Dom\,(f_2)\,, & \neg Dom\,(f_1)\cdot\neg Dom\,(f_2)\,. \end{array}$$

Among these domains (which, notice, are all disjoint), the nonempty ones become the identities D_i.

Each strategy comes with an associated explanation about how to construct a program solving the original problem. It is easy to see how the previously given strategies induce the structure of the programs. For example, it is clear from the definition of case analysis that whenever $C\_A(R, R_0, R_1)$ holds, we can infer that $R = R_0 + R_1$, thus giving a program (equation) of the desired shape solving the problem R. In the same way, when $D\&C(R, R_0, Split, Q_1, \ldots, Q_k, Join)$ holds, we have a program with shape

$$R = R_0\ +\ Split;(Q_1 \otimes \cdots \otimes Q_k);Join$$

solving R. Also, when (7.14) holds, we have a program with shape

$$R = R_0\ +\ \sum_{i=1}^{k} D_i;Split_i;\ \left(Q_{i_1} \otimes \cdots \otimes Q_{i_{j_i}}\right);Join_i. \tag{7.15}$$

The filters D_i in (7.15) play the role of guards in case-like constructs.

Notice that strategies are in general formulas of the form

$$Strat(R, X_1, \ldots, X_n) \quad \Longleftrightarrow \quad Strat\_Definition(R, X_1, \ldots, X_n), \tag{7.16}$$

where *Strat* is a $(n+1)$-ary predicate symbol (name and parameters of the strategy), and *Strat_Definition* is a formula on the relational variables R, $X_1, \ldots, X_n$, involving previously defined strategies. Deriving a generic algorithm GA for solving a generic problem GP whose generic relational specification is $GS(P_1, \ldots, P_k)$ using a strategy defined as in (7.16), consists of finding relational terms $T_1(P_1, \ldots, P_k), \ldots, T_n(P_1, \ldots, P_k)$ such that

$$\begin{array}{l} Theory(D_1), \ldots, Theory(D_m), GS(P_1, \ldots, P_k) \vdash_{\mathsf{AFA}} \\ \qquad Strat\_Definition(GP, T_1(P_1, \ldots, P_k), \ldots, T_n(P_1, \ldots, P_k)). \end{array} \tag{7.17}$$

In (7.17), $Theory(D_1), \ldots, Theory(D_m)$ are relational specifications of the domains of the problem [R. Berghammer (1991); R. Berghammer et al. (1993)c; R. Berghammer et al. (1993)b], and the symbol $\vdash_{\mathsf{AFA}}$ denotes first-order logic entailment under the theory of AFA.

Notice that the algorithms characterized by the strategies are as a matter of fact fork algebraic equations. Thus, in order to find terms $T_1(P_1,\ldots,P_k),\ldots,T_n(P_1,\ldots,P_k)$ we will resort to equational reasonings using the axioms of fork algebras, plus those equations describing the domains $D_1,\ldots,D_m$. The general strategy we will use for deriving recursive algorithms will be *Unfolding/Folding* [J. Darlington (1975)].

The terms $T_1(P_1,\ldots,P_k),\ldots,T_n(P_1,\ldots,P_k)$ required in (7.17), and found as described in the previous paragraph, are either algorithms if they are built with algorithmic combinators, or can be considered as relational specifications of simpler problems.

In Section 7.6 we will show that for the sample problem, if we define

$$\begin{array}{lcl} D_{R,1} & := & Dom(R_1)\cdot Dom(F_1;P')\,, \\ D_{R,\neg 1} & := & Dom(R_1)\cdot\neg Dom(F_1;P')\,,\ \text{and} \\ D_{\neg R,1} & := & \neg Dom(R_1)\cdot Dom(F_1;P')\,, \end{array}$$

we can deduce

$$\begin{array}{l} [P' = G';C_1\cdot(G';C_1 \rightarrow C_2) \\ \qquad\qquad \wedge\ G' = 1'_b;R_0 + 1'_{\neg b};R_1 + 1'_{\neg b};F_1;G' \\ \qquad\qquad\quad \wedge\ \breve{C}_2\cdot C_2 \leq 1' \ \wedge\ Easy(1'_b;P')] \\ \qquad \Rightarrow\ GenD\&C(P', 1'_b;P', D_{R,1}, R_1\nabla F_1, 1'\otimes P', Join_1, \\ \qquad\qquad\qquad D_{R,\neg 1}, R, 1', C_1\cdot C_2, D_{\neg R,1}, F_1, P', 1'), \end{array} \quad (7.18)$$

where R_1 and F_1 are functional relations, $1'_b$ and $1'_{\neg b}$ are filters that produce a partition of the domain $1'_{S(\beta)}$, and

$$Join_1 := \Big((C_1\cdot C_2 \otimes \breve{C}_2) + (C_1;\breve{C}_2 \otimes 1')\Big)\,;\breve{2}\ .$$

According to the strategy *GenD&C*, we obtain the algorithm

$$\begin{array}{rcl} P' & = & 1'_b;P' \ +\ D_{R,1};\begin{array}{c}R_1\\ \nabla\\ F_1\end{array};\begin{array}{c}1'\\ \otimes\\ P'\end{array};Join_1 \\ & & \qquad +\ D_{R,\neg 1};R;(C_1\cdot C_2) \ +\ D_{\neg R,1};F_1;P'\ . \end{array}$$

In our formalism a strategy associates to each generic specification $GS(P_1,\ldots,P_k)$ a generic algorithm $GA(T_1,\ldots,T_n)$. If we instantiate the parameters $P_1,\ldots,P_k$ occurring in specification GS with relation designations $R_1,\ldots,R_k$, and derive algorithms $A_1,\ldots,A_n$ (using this same methodology) for terms $T_1(R_1,\ldots,R_k),\ldots,$ $T_n(R_1,\ldots,R_k)$, we obtain a particular algorithm, namely $A := GA(A_1,\ldots,A_n)$ now for the less generic problem $S := GS(R_1,\ldots,R_k)$. Let us see how this is reflected in the problem chosen as an example. If in the sample problem we take*

(1) $\beta := Nat$ and $\mathsf{S}(\beta) := List(Nat)$,
(2) $G' = 1'_{L^1};Hd + 1'_{L^{\succ 1}};Hd + 1'_{L^{\succ 1}};Tl;G'$ (i.e., $R_0 = Hd$, $R_1 = Hd$ and $F_1 = Tl$),
(3) $C_1 := 1'$, and
(4) $C_2 := \preceq$, the standard ordering between natural numbers,

then P' specifies the problem of finding the minimum element in a list. Notice that $Join_1$, under this instantiation, specifies the problem of finding the minimum between two natural numbers. Since it is not in an algorithmic form, this same methodology can be applied to derive an algorithm for this problem. Once this is done, (7.18) can be optimized to obtain a divide-and-conquer algorithm for finding the minimum element of a list.

In Fig. 7.2 we give a graphical description of the methodology we propose.

7.6 Examples

In this section we will present several generic problems for which we will proceed as follows.

(1) We will specify the problems in first-order logic, in a totally declarative manner.
(2) We will specify the problems with fork-algebraic equations obtained from the first-order specifications.

*We will denote by Hd and Tl the functions that, given a list $[e:l]$, retrieve the element e and the list l, respectively. By *Cons* we denote the function that, given an element e and a list l, produces as output the list $[e:l]$. By $1'_{L^k}$ we denote the filter over lists of length k, and by $1'_{L^{\succ k}}$ the filter over lists of length greater than k. By $\preceq$ we will denote the standard ordering between natural and integer numbers.

$$\begin{tikzcd}
GS(P_1,\dots,P_k) \arrow[r, "Strat"] \arrow[d, "\begin{array}{c}P_i := R_i\\ 1\le i\le k\end{array}"'] & GA(T_1(P_1,\dots,P_k),\dots,T_n(P_1,\dots,P_k)) \arrow[d, "\begin{array}{c}\text{derive } A_i\\ \text{to compute}\\ T_i(R_1,\dots,R_k)\end{array}"] \\
S & A := GA(A_1,\dots,A_n) \arrow[l, "\text{computes}"]
\end{tikzcd}$$

Fig. 7.2 Methodology for program construction.

(3) We will derive generic algorithms for these generic problems using the presented design strategies.

(4) We will solve some specific problems by using the generic algorithms.

The derivations we will present may seem too long and detailed, but they are essential in order to show that smooth syntactical derivations of algorithms are possible by using the methodology we propose. Also, these derivations are themselves constructive proofs of correctness of generic programs with respect to generic specifications. In this sense, the amount of effort invested in making (or understanding) complex derivations is largely compensated for by their usefulness in designing concrete programs.

7.6.1 *First Example*

The first example will be the problem used in the previous section as an example. The problem was informally specified by the sentence:

> Let $\mathsf{S}(\beta)$ be a structured type, and let $G \subseteq \mathsf{S}(\beta) \times \beta$ be a generator. Select those generated elements that satisfy
>
> (1) a condition c_1, and
>
> (2) a condition c_2 with respect to all the generated elements that satisfy the condition c_1.

Problem P can be specified by the first-order formula:

$$P(x,y) \iff G(x,y) \wedge c_1(y) \;\wedge\; \forall z\,(G(x,z) \wedge c_1(z) \;\Rightarrow\; c_2(z,y)) \;. \qquad (7.19)$$

Recalling the set-theoretical definition of the relational operators, we can transform (7.19) into the following abstract relational specification:

$$P = G;C_1 \cdot (G;C_1 \rightarrow C_2),$$

where our only assumption about C_1 and C_2 is that C_1 is a filter.

It is at this point when the form of the generator becomes important. Let us assume that $1'_{\mathsf{S}(\beta)} = 1'_b + 1'_{\neg b}$, where $1'_b$ represents the '*base*' part of the type $\mathsf{S}(\beta)$. Suitable forms for generators are, for instance:

$$G_1 = 1'_b;R_0 + 1'_{\neg b};R_1 + 1'_{\neg b};F_1;G_1, \tag{7.20}$$

$$G_2 = 1'_b;R_0 + 1'_{\neg b};R_1 + 1'_{\neg b};F_1;G_2 + 1'_{\neg b};F_2;G_2. \tag{7.21}$$

In (7.20) and (7.21) we assume that R_0 and R_1 are fixed relations with R_1 functional, and that $F_1 \subseteq \mathsf{S}(\beta) \times \mathsf{S}(\beta)$ and $F_2 \subseteq \mathsf{S}(\beta) \times \mathsf{S}(\beta)$ are functions that decompose data. Since G_2 is more general than G_1 (take F_2 to be F_1 in (7.21)), we will work with G_2, and therefore the specification becomes

$$P = G_2;C_1 \cdot (G_2;C_1 \rightarrow C_2)\ . \tag{7.22}$$

In order to simplify the presentation of the derivation, we will make the following assumptions:

$$Ass_1 : Dom(F_1) = Dom(F_2),$$
$$Ass_2 : Dom(R_1) \leq Dom(F_1).$$

Notice that in the derivation below, these assumptions can be easily dropped and the changes will be minor, resulting only in longer formulas but no technical complications.

We will derive a generalized divide-and-conquer solution for the problem. Thus, we must find relations R_0, D_1, $Split_1$, A_1, $Join_1, \ldots$, D_k, $Split_k$, A_k and $Join_k$ such that the formula

$$\bigwedge_{1 \leq i \leq k} D_i \leq 1' \ \wedge\ Triv(R, R_0, D_1;R, \ldots, D_k;R)\ \wedge$$
$$\bigwedge_{1 \leq i \leq k} \exists Q_{i_1}, \ldots, Q_{i_{j_i}} (A_i = Q_{i_1} \otimes \cdots \otimes Q_{i_{j_i}} \ \wedge$$
$$Recomp(D_i;R, Split_i, Q_{i_1}, \ldots, Q_{i_{j_i}}, Join_i))$$

holds.

Since $1'_{S(\beta)} = 1'_b + 1'_{\neg b}$, we will begin with a Trivialization step using Heuristic 7.1. If $Easy(1'_b;P)$ holds, we must concentrate on the relation $1'_{\neg b};P$. Thus, we will now derive an expression for the relation $1'_{\neg b};P$ of the form required in (7.14).

$$\begin{aligned}
& 1'_{\neg b};P \\
= & \{\text{ by (7.22) }\} \\
& 1'_{\neg b};(G_2;C_1 \cdot (G_2;C_1 \rightarrow C_2)) \\
= & \{\text{ Unfolding the definition of the generator } G_2 \} \\
& (R_1 + F_1;G_2 + F_2;G_2);C_1 \\
& \quad \cdot ((R_1 + F_1;G_2 + F_2;G_2);C_1 \rightarrow C_2) \\
= & \{\text{ Applying Ax. 2 several times }\} \\
& (R_1;C_1 + F_1;G_2;C_1 + F_2;G_2;C_1) \\
& \quad \cdot ((R_1;C_1 + F_1;G_2;C_1 + F_2;G_2;C_1) \rightarrow C_2) \\
= & \{\text{ by (7.4) }\} \\
& (R_1;C_1 + F_1;G_2;C_1 + F_2;G_2;C_1) \\
& \quad \cdot (R_1;C_1 \rightarrow C_2) \cdot (F_1;G_2;C_1 \rightarrow C_2) \cdot (F_2;G_2;C_1 \rightarrow C_2) \\
= & \{\text{ Distributing } + \text{ over } \cdot \} \\
& (R_1;C_1) \cdot (R_1;C_1 \rightarrow C_2) \cdot (F_1;G_2;C_1 \rightarrow C_2) \cdot (F_2;G_2;C_1 \rightarrow C_2) \\
& + (F_1;G_2;C_1) \cdot (R_1;C_1 \rightarrow C_2) \cdot (F_1;G_2;C_1 \rightarrow C_2) \cdot (F_2;G_2;C_1 \rightarrow C_2) \\
& + (F_2;G_2;C_1) \cdot (R_1;C_1 \rightarrow C_2) \cdot (F_1;G_2;C_1 \rightarrow C_2) \cdot (F_2;G_2;C_1 \rightarrow C_2).
\end{aligned}$$

In order to shorten notation we will denote the term $G_2;C_1 \rightarrow C_2$ by X. Let us consider the term

$$(R_1;C_1) \cdot (R_1;C_1 \rightarrow C_2) \cdot (F_1;G_2;C_1 \rightarrow C_2) \cdot (F_2;G_2;C_1 \rightarrow C_2) \ . \quad (7.23)$$

$$\begin{aligned}
& (R_1;C_1) \cdot (R_1;C_1 \rightarrow C_2) \cdot (F_1;G_2;C_1 \rightarrow C_2) \cdot (F_2;G_2;C_1 \rightarrow C_2) \\
= \quad & \{\text{ by Lemmas 7.3 and 7.4 using } Ass_1 \text{ and } Ass_2 \} \\
& (R_1;C_1) \cdot (R_1;C_1;\breve{C}_2) \cdot (F_1;X) \cdot (F_2;X) \\
= \quad & \{\text{ by Thm. 2.3.17 }\} \\
& (R_1;(C_1 \cdot C_1;\breve{C}_2)) \cdot (F_1;X) \cdot (F_2;X) \\
= \quad & \{\text{ by Thm. 2.3.22 }\} \\
& (R_1;(C_1 \cdot C_2)) \cdot (F_1;X) \cdot (F_2;X) \\
\leq \quad & \{\text{ because by (7.22) } G_2;C_1 \geq P, \text{ and monotonicity }\}
\end{aligned}$$

$$
\begin{aligned}
& (R_1;(C_1\cdot C_2))\cdot(F_1;(P \rightarrow C_2))\cdot(F_2;(P \rightarrow C_2)) \\
= \; & \{\text{if } C_2 \text{ is antisymmetric, and Lemmas 7.1, 7.3, 7.8}\} \\
& (R_1;(C_1\cdot C_2))\cdot(F_1;(P;\breve{C}_2 + \neg Dom(P);1)) \\
& \quad \cdot(F_2;(P;\breve{C}_2 + \neg Dom(P);1)) \\
= \; & \{\text{by Ax. 2 and BA}\} \\
& (R_1;(C_1\cdot C_2))\cdot(F_1;P;\breve{C}_2)\cdot(F_2;P;\breve{C}_2) \\
& + (R_1;(C_1\cdot C_2))\cdot(F_1;P;\breve{C}_2)\cdot(F_2;\neg Dom(P);1) \\
& + (R_1;(C_1\cdot C_2))\cdot(F_1;\neg Dom(P);1)\cdot(F_2;P;\breve{C}_2) \\
& + (R_1;(C_1\cdot C_2))\cdot(F_1;\neg Dom(P);1)\cdot(F_2;\neg Dom(P);1).
\end{aligned}
$$

Let us consider now the term

$$
\begin{aligned}
& (F_1;G_2;C_1)\cdot(R_1;C_1 \rightarrow C_2) \\
& \qquad \cdot(F_1;G_2;C_1 \rightarrow C_2)\cdot(F_2;G_2;C_1 \rightarrow C_2)\;. \qquad (7.24)
\end{aligned}
$$

$$
\begin{aligned}
& (F_1;G_2;C_1)\cdot(R_1;C_1 \rightarrow C_2)\cdot(F_1;G_2;C_1 \rightarrow C_2)\cdot(F_2;G_2;C_1 \rightarrow C_2) \\
= & \{\text{by Lemmas 7.3 and 7.4 using } Ass_1 \text{ and } Ass_2\} \\
& (F_1;G_2;C_1)\cdot(R_1;C_1;\breve{C}_2 + \neg Dom(R_1;C_1);1)\cdot(F_1;X)\cdot(F_2;X) \\
= & \{\text{by BA and Thm. 2.3.17}\} \\
& (F_1;((G_2;C_1)\cdot X))\cdot(R_1;C_1;\breve{C}_2 + \neg Dom(R_1;C_1);1)\cdot(F_2;X) \\
= & \{\text{folding } P\} \\
& (F_1;P)\cdot(R_1;C_1;\breve{C}_2 + \neg Dom(R_1;C_1);1)\cdot(F_2;X) \\
\leq & \{\text{because by (7.22) } G_2;C_1 \geq P\text{, and monotonicity}\} \\
& (F_1;P)\cdot(R_1;C_1;\breve{C}_2 + \neg Dom(R_1;C_1);1)\cdot(F_2;(P \rightarrow C_2)) \\
= & \{\text{if } C_2 \text{ is antisymmetric, and Lemmas 7.1, 7.3, 7.8}\} \\
& (F_1;P)\cdot(R_1;C_1;\breve{C}_2 + \neg Dom(R_1;C_1);1)\cdot(F_2;(P;\breve{C}_2 + \neg Dom(P);1)) \\
= & \{\text{by BA}\} \\
& (R_1;C_1;\breve{C}_2)\cdot(F_1;P)\cdot(F_2;P;\breve{C}_2) \\
& + (R_1;C_1;\breve{C}_2)\cdot(F_1;P)\cdot(F_2;\neg Dom(P);1) \\
& + (\neg Dom(R_1;C_1);1)\cdot(F_1;P)\cdot(F_2;P;\breve{C}_2) \\
& + (\neg Dom(R_1;C_1);1)\cdot(F_1;P)\cdot(F_2;\neg Dom(P);1).
\end{aligned}
$$

Finally, if we consider the term

$$(F_2;G_2;C_1) \cdot (R_1;C_1 \to C_2) \\ \cdot (F_1;G_2;C_1 \to C_2) \cdot (F_2;G_2;C_1 \to C_2), \quad (7.25)$$

a derivation along the lines of the previous one proves that:

$$\begin{aligned}(F_2;G_2;C_1) \cdot (R_1;C_1 \to C_2) \cdot (F_1;G_2;C_1 \to C_2) \cdot (F_2;G_2;C_1 \to C_2) \\ \leq\ & (R_1;C_1;\breve{C}_2)\cdot(F_1;P;\breve{C}_2)\cdot(F_2;P) \\ +\ & (R_1;C_1;\breve{C}_2)\cdot(F_1;\neg \mathit{Dom}(P);1)\cdot(F_2;P) \\ +\ & (\neg \mathit{Dom}(R_1;C_1);1)\cdot(F_1;P;\breve{C}_2)\cdot(F_2;P) \\ +\ & (\neg \mathit{Dom}(R_1;C_1);1)\cdot(F_1;\neg \mathit{Dom}(P);1)\cdot(F_2;P).\end{aligned}$$

Joining the derivations performed from terms (7.23), (7.24) and (7.25), we obtain:

$$\begin{aligned}1'_{\neg b};P \ \leq\ & (R_1;(C_1\cdot C_2))\cdot(F_1;P;\breve{C}_2)\cdot(F_2;P;\breve{C}_2) \\ +\ & (R_1;(C_1\cdot C_2))\cdot(F_1;P;\breve{C}_2)\cdot(F_2;\neg \mathit{Dom}(P);1) \\ +\ & (R_1;(C_1\cdot C_2))\cdot(F_1;\neg \mathit{Dom}(P);1)\cdot(F_2;P;\breve{C}_2) \\ +\ & (R_1;(C_1\cdot C_2))\cdot(F_1;\neg \mathit{Dom}(P);1)\cdot(F_2;\neg \mathit{Dom}(P);1) \\ +\ & (R_1;C_1;\breve{C}_2)\cdot(F_1;P)\cdot(F_2;P;\breve{C}_2) \\ +\ & (R_1;C_1;\breve{C}_2)\cdot(F_1;P)\cdot(F_2;\neg \mathit{Dom}(P);1) \\ +\ & (\neg \mathit{Dom}(R_1;C_1);1)\cdot(F_1;P)\cdot(F_2;P;\breve{C}_2) \\ +\ & (\neg \mathit{Dom}(R_1;C_1);1)\cdot(F_1;P)\cdot(F_2;\neg \mathit{Dom}(P);1) \\ +\ & (R_1;C_1;\breve{C}_2)\cdot(F_1;P;\breve{C}_2)\cdot(F_2;P) \\ +\ & (R_1;C_1;\breve{C}_2)\cdot(F_1;\neg \mathit{Dom}(P);1)\cdot(F_2;P) \\ +\ & (\neg \mathit{Dom}(R_1;C_1);1)\cdot(F_1;P;\breve{C}_2)\cdot(F_2;P) \\ +\ & (\neg \mathit{Dom}(R_1;C_1);1)\cdot(F_1;\neg \mathit{Dom}(P);1)\cdot(F_2;P).\end{aligned}$$

Let us define the filters

$$\begin{aligned}D_{R,1,2} &:= \mathit{Dom}(R_1)\cdot \mathit{Dom}(F_1;P)\cdot \mathit{Dom}(F_2;P), \\ D_{R,1,\neg 2} &:= \mathit{Dom}(R_1)\cdot \mathit{Dom}(F_1;P)\cdot \neg \mathit{Dom}(F_2;P), \\ D_{R,\neg 1,2} &:= \mathit{Dom}(R_1)\cdot \neg \mathit{Dom}(F_1;P)\cdot \mathit{Dom}(F_2;P), \\ D_{R,\neg 1,\neg 2} &:= \mathit{Dom}(R_1)\cdot \neg \mathit{Dom}(F_1;P)\cdot \neg \mathit{Dom}(F_2;P), \\ D_{\neg R,1,2} &:= \neg \mathit{Dom}(R_1;C_1)\cdot \mathit{Dom}(F_1;P)\cdot \mathit{Dom}(F_2;P), \\ D_{\neg R,1,\neg 2} &:= \neg \mathit{Dom}(R_1;C_1)\cdot \mathit{Dom}(F_1;P)\cdot \neg \mathit{Dom}(F_2;P), \\ D_{\neg R,\neg 1,2} &:= \neg \mathit{Dom}(R_1;C_1)\cdot \neg \mathit{Dom}(F_1;P)\cdot \mathit{Dom}(F_2;P).\end{aligned}$$

Since by Thm. 7.1.4 $F_i;\neg Dom(P);1 = (Dom(F_i)\cdot\neg Dom(F_i;P));1$ $(i=1,2)$, using the previously defined filters and Thm. 2.3.22,

$$\begin{aligned}
1'_{\neg b};P \leq\; & D_{R,1,2};\left(R_1;(C_1\cdot C_2)\;\cdot\;F_1;P;\breve{C}_2\;\cdot\;F_2;P;\breve{C}_2\right)\\
&+D_{R,1,2};\left(R_1;C_1;\breve{C}_2\;\cdot\;F_1;P\;\cdot\;F_2;P;\breve{C}_2\right)\\
&+D_{R,1,2};\left(R_1;C_1;\breve{C}_2\;\cdot\;F_1;P;\breve{C}_2\;\cdot\;F_2;P\right)\\
&+D_{R,1,\neg 2};\left(R_1;(C_1\cdot C_2)\;\cdot\;F_1;P;\breve{C}_2\right)\\
&+D_{R,1,\neg 2};\left(R_1;C_1;\breve{C}_2\;\cdot\;F_1;P\right)\\
&+D_{R,\neg 1,2};\left(R_1;(C_1\cdot C_2)\;\cdot\;F_2;P;\breve{C}_2\right)\\
&+D_{R,\neg 1,2};\left(R_1;C_1;\breve{C}_2\;\cdot\;F_2;P\right)\\
&+D_{R,\neg 1,\neg 2};R_1;(C_1\cdot C_2)\\
&+D_{\neg R,1,2};\left(F_1;P\;\cdot\;F_2;P;\breve{C}_2\right)\\
&+D_{\neg R,1,2};\left(F_1;P;\breve{C}_2\;\cdot\;F_2;P\right)\\
&+D_{\neg R,1,\neg 2};F_1;P\;+\;D_{\neg R,\neg 1,2};F_2;P.
\end{aligned}$$

Applying Thm. 3.2.1 several times,

$$\begin{aligned}
1'_{\neg b};P \leq\; & \begin{pmatrix} R_1;(C_1\cdot C_2)\\ \nabla\\ \begin{pmatrix} F_1;P;\breve{C}_2\\ \nabla\\ F_2;P;\breve{C}_2\end{pmatrix};\breve{\mathbb{2}}\end{pmatrix};\breve{\mathbb{2}}\\
&+\begin{pmatrix} R_1;C_1;\breve{C}_2\\ \nabla\\ \begin{pmatrix} F_1;P\\ \nabla\\ F_2;P;\breve{C}_2\end{pmatrix};\breve{\mathbb{2}}\end{pmatrix};\breve{\mathbb{2}}+\begin{pmatrix} R_1;C_1;\breve{C}_2\\ \nabla\\ \begin{pmatrix} F_1;P;\breve{C}_2\\ \nabla\\ F_2;P\end{pmatrix};\breve{\mathbb{2}}\end{pmatrix};\breve{\mathbb{2}}\\
&+D_{R,1,\neg 2};\begin{pmatrix} R_1;(C_1\cdot C_2)\\ \nabla\\ F_1;P;\breve{C}_2\end{pmatrix};\breve{\mathbb{2}}+D_{R,1,\neg 2};\begin{pmatrix} R_1;C_1;\breve{C}_2\\ \nabla\\ F_1;P\end{pmatrix};\breve{\mathbb{2}}\\
&+D_{R,\neg 1,2};\begin{pmatrix} R_1;(C_1\cdot C_2)\\ \nabla\\ F_2;P;\breve{C}_2\end{pmatrix};\breve{\mathbb{2}}+D_{R,\neg 1,2};\begin{pmatrix} R_1;C_1;\breve{C}_2\\ \nabla\\ F_2;P\end{pmatrix};\breve{\mathbb{2}}\\
&+D_{\neg R,1,2};\begin{pmatrix} F_1;P\\ \nabla\\ F_2;P;\breve{C}_2\end{pmatrix};\breve{\mathbb{2}}+D_{\neg R,1,2};\begin{pmatrix} F_1;P;\breve{C}_2\\ \nabla\\ F_2;P\end{pmatrix};\breve{\mathbb{2}}\\
&+D_{R,\neg 1,\neg 2};R_1;(C_1\cdot C_2)+D_{\neg R,1,\neg 2};F_1;P+D_{\neg R,\neg 1,2};F_2;P\,. \qquad (7.26)
\end{aligned}$$

Applying Thm. 3.2.9 and Ax. 2 in (7.26),

$$
\begin{aligned}
1'_{\neg b};P \leq{}& \\
&\begin{pmatrix} R_1 \\ \nabla \\ F_1 \\ \nabla \\ F_2 \end{pmatrix} ; \begin{pmatrix} 1' \\ \otimes \\ P \\ \otimes \\ P \end{pmatrix} ; \left(\begin{array}{c} C_1 \cdot C_2 \\ \otimes \\ \begin{pmatrix} \breve{C}_2 \\ \otimes \\ \breve{C}_2 \end{pmatrix} \end{array} + \begin{array}{c} C_1 ; \breve{C}_2 \\ \otimes \\ \begin{pmatrix} 1' \\ \otimes \\ \breve{C}_2 \end{pmatrix} \end{array} + \begin{array}{c} C_1 ; \breve{C}_2 \\ \otimes \\ \begin{pmatrix} \breve{C}_2 \\ \otimes \\ 1' \end{pmatrix} \end{array} \right) ; \begin{pmatrix} 1' \\ \otimes \\ \breve{\mathbb{2}} \end{pmatrix} ; \breve{\mathbb{2}} \\
&+ D_{R,1,\neg 2} ; \begin{array}{c} R_1 \\ \nabla \\ F_1 \end{array} ; \begin{array}{c} 1' \\ \nabla \\ P \end{array} ; \begin{pmatrix} C_1 \cdot C_2 & & C_1 ; \breve{C}_2 \\ \otimes & + & \otimes \\ \breve{C}_2 & & 1' \end{pmatrix} ; \breve{\mathbb{2}} \\
&+ D_{R,\neg 1,2} ; \begin{array}{c} R_1 \\ \nabla \\ F_2 \end{array} ; \begin{array}{c} 1' \\ \nabla \\ P \end{array} ; \begin{pmatrix} C_1 \cdot C_2 & & C_1 ; \breve{C}_2 \\ \otimes & + & \otimes \\ \breve{C}_2 & & 1' \end{pmatrix} ; \breve{\mathbb{2}} \\
&+ D_{\neg R,1,2} ; \begin{array}{c} F_1 \\ \nabla \\ F_2 \end{array} ; \begin{array}{c} P \\ \nabla \\ P \end{array} ; \begin{pmatrix} 1' & & \breve{C}_2 \\ \otimes & + & \otimes \\ \breve{C}_2 & & 1' \end{pmatrix} ; \breve{\mathbb{2}} \\
&+ D_{R,\neg 1,\neg 2} ; R_1 ; (C_1 \cdot C_2) + D_{\neg R,1,\neg 2} ; F_1 ; P + D_{\neg R,\neg 1,2} ; F_2 ; P. \qquad (7.27)
\end{aligned}
$$

Let us define

$$
\begin{array}{lcllcl}
Split_1 & := & R_1 \nabla (F_1 \nabla F_2), & A_1 & := & 1' \otimes (P \otimes P), \\
Split_2 & := & R_1 \nabla F_1, & A_2 & := & 1' \otimes P, \\
Split_3 & := & R_1 \nabla F_2, & A_3 & := & 1' \otimes P, \\
Split_4 & := & F_1 \nabla F_2, & A_4 & := & P \otimes P, \\
Split_5 & := & R_1, & A_5 & := & 1', \\
Split_6 & := & F_1, & A_6 & := & P, \\
Split_7 & := & F_2, & A_7 & := & P.
\end{array}
$$

Let us also define

$$
\begin{array}{lcl}
Join_1 & := & \left(\begin{array}{c} C_1 \cdot C_2 \\ \otimes \\ \begin{pmatrix} \breve{C}_2 \\ \otimes \\ \breve{C}_2 \end{pmatrix} \end{array} + \begin{array}{c} C_1 ; \breve{C}_2 \\ \otimes \\ \begin{pmatrix} 1' \\ \otimes \\ \breve{C}_2 \end{pmatrix} \end{array} + \begin{array}{c} C_1 ; \breve{C}_2 \\ \otimes \\ \begin{pmatrix} \breve{C}_2 \\ \otimes \\ 1' \end{pmatrix} \end{array} \right) ; \begin{pmatrix} 1' \\ \otimes \\ \breve{\mathbb{2}} \end{pmatrix} ; \breve{\mathbb{2}}, \\
Join_2 & := & Join_3 := \begin{pmatrix} C_1 \cdot C_2 & & C_1 ; \breve{C}_2 \\ \otimes & + & \otimes \\ \breve{C}_2 & & 1' \end{pmatrix} ; \breve{\mathbb{2}}, \\
Join_4 & := & \begin{pmatrix} 1' & & \breve{C}_2 \\ \otimes & + & \otimes \\ \breve{C}_2 & & 1' \end{pmatrix} ; \breve{\mathbb{2}}, \\
Join_5 & := & C_1 \cdot C_2, \\
Join_6 & := & Join_7 := 1'.
\end{array}
$$

It is easy to check that if C_2 is antisymmetric, $Join_i$ $(1 \leq i \leq 7)$ as defined are functional relations. Thus, the term on the right hand side of (7.27) stands for a functional relation. From Lemma 7.8 and Thm. 2.3.18,

$$
\begin{aligned}
&1'_{\neg b};P = \\
&\begin{array}{c} R_1 \\ \nabla \\ \left(\begin{array}{c} F_1 \\ \nabla \\ F_2 \end{array}\right) \end{array} ; \begin{array}{c} 1' \\ \otimes \\ \left(\begin{array}{c} P \\ \otimes \\ P \end{array}\right) \end{array} ; \left(\begin{array}{c} C_1 \cdot C_2 \\ \otimes \\ \left(\begin{array}{c} \breve{C}_2 \\ \otimes \\ \breve{C}_2 \end{array}\right) \end{array} + \begin{array}{c} C_1 ; \breve{C}_2 \\ \otimes \\ \left(\begin{array}{c} 1' \\ \otimes \\ \breve{C}_2 \end{array}\right) \end{array} + \begin{array}{c} C_1 ; \breve{C}_2 \\ \otimes \\ \left(\begin{array}{c} \breve{C}_2 \\ \otimes \\ 1' \end{array}\right) \end{array} \right) ; \left(\begin{array}{c} 1' \\ \otimes \\ \breve{2} \end{array}\right) ; \breve{2} \\
&\qquad + D_{R,1,\neg 2} ; \begin{array}{c} R_1 \\ \nabla \\ F_1 \end{array} ; \begin{array}{c} 1' \\ \nabla \\ P \end{array} ; \left(\begin{array}{c} C_1 \cdot C_2 \\ \otimes \\ \breve{C}_2 \end{array} + \begin{array}{c} C_1 ; \breve{C}_2 \\ \otimes \\ 1' \end{array} \right) ; \breve{2} \\
&\qquad + D_{R,\neg 1,2} ; \begin{array}{c} R_1 \\ \nabla \\ F_2 \end{array} ; \begin{array}{c} 1' \\ \nabla \\ P \end{array} ; \left(\begin{array}{c} C_1 \cdot C_2 \\ \otimes \\ \breve{C}_2 \end{array} + \begin{array}{c} C_1 ; \breve{C}_2 \\ \otimes \\ 1' \end{array} \right) ; \breve{2} \\
&\qquad + D_{\neg R,1,2} ; \begin{array}{c} F_1 \\ \nabla \\ F_2 \end{array} ; \begin{array}{c} P \\ \nabla \\ P \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \breve{C}_2 \end{array} + \begin{array}{c} \breve{C}_2 \\ \otimes \\ 1' \end{array} \right) ; \breve{2} \\
&\qquad + D_{R,\neg 1,\neg 2} ; R_1 ; (C_1 \cdot C_2) + D_{\neg R,1,\neg 2} ; F_1 ; P + D_{\neg R,\neg 1,2} ; F_2 ; P.
\end{aligned}
$$

We then finally have

$$
\begin{aligned}
P = G_2;C_1 \cdot (G_2;C_1 \rightarrow C_2) \wedge \breve{C}_2 \cdot C_2 \leq 1' \wedge Easy(1'_b;P) \\
\Rightarrow GenD\&C(P, 1'_b;P, D_{R,1,2}, Split_1, A_1, Join_1, \ldots, \\
D_{\neg R,\neg 1,2}, Split_7, A_7, Join_7) \ . \qquad (7.28)
\end{aligned}
$$

If we use G_1 instead of G_2, and define

$$
\begin{array}{lll lll}
D_{R,1} & := & Dom(R_1) \cdot Dom(F_1;P), & Split_1 & := & R_1 \nabla F_1, \\
D_{R,\neg 1} & := & Dom(R_1) \cdot \neg Dom(F_1;P), & Split_2 & := & R_1, \\
D_{\neg R,1} & := & \neg Dom(R_1) \cdot Dom(F_1;P), & Split_3 & := & F_1,
\end{array}
$$

$$
\begin{array}{lll lll}
A_1 & := & 1' \otimes P, & Join_1 & := & ((C_1 \cdot C_2 \otimes \breve{C}_2) + (C_1;\breve{C}_2 \otimes 1'));\breve{2}, \\
A_2 & := & 1', & Join_2 & := & C_1 \cdot C_2, \\
A_3 & := & P, & Join_3 & := & 1',
\end{array}
$$

we can prove, under the assumption that C_2 is antisymmetric, that

$$
\begin{aligned}
P = G_1;C_1 \cdot (G_1;C_1 \rightarrow C_2) \wedge \breve{C}_2 \cdot C_2 \leq 1' \wedge Easy(1'_b;P) \\
\Rightarrow GenD\&C(P, 1'_b;P, D_{R,1}, Split_1, A_1, Join_1, \\
D_{R,\neg 1}, Split_2, A_2, Join_2, D_{\neg R,1}, Split_3, A_3, Join_3) \ . \qquad (7.29)
\end{aligned}
$$

7.6.1.1 *Finding the Minimum Element in a List*

Let the relation $Has \subseteq List(Nat) \times Nat$ be defined by the condition

$$Has = Hd + Tl; Has \ .$$

Has is a generator of type G_1. Let us consider the problem of finding the minimum element in a list. The problem can be specified in first-order logic by the formula

$$l\,Min\,x \iff l\,Has\,x \land \forall y\,(l\,Has\,y \Rightarrow x \preceq y) \ . \qquad (7.30)$$

The abstract relational specification, obtained from (7.30), is given by the equation

$$Min = Has \cdot (Has \rightarrow \preceq) \ .$$

Since the relation $\preceq$ is antisymmetric and $Easy\,(1'_{L^1}; Min)$ holds (the minimum of a list with just one element is that element), taking $C_1 := 1'$ and $C_2 := \preceq$ in (7.29) we have

$$\begin{aligned} &GenD\&C(Min, 1'_{L^1}; Hd, Dom\,(Hd) \cdot Dom\,(Tl; Min)\,, Hd \nabla Tl, \\ &\quad 1' \otimes Min, Join_1, Dom\,(Hd) \cdot \neg Dom\,(Tl; Min)\,, Hd, 1', \\ &\qquad 1'_{Nat}, \neg Dom\,(Hd) \cdot Dom\,(Tl; Min)\,, Tl, Min, 1'), \qquad (7.31) \end{aligned}$$

where $Join_1$ is defined by the condition

$$Join_1 = \left((1' \cdot \preceq \ \otimes \ \breve{\preceq}) + (1'; \breve{\preceq} \ \otimes \ 1')\right); \breve{2} \ .$$

Since $\preceq$ is reflexive, $1' \cdot \preceq \; = 1'_{Nat}$. Notice also that $\breve{\preceq} = \succeq$. We then have

$$Join_1 = ((1' \otimes \succeq) + (\succeq \otimes 1')); \breve{2},$$

which is a specification of the problem *min_num* that finds the minimum between two numbers.

Since $Dom\,(Hd) = 1'_{L^{\succ 0}}$ and $Dom\,(Tl; Min) = 1'_{L^{\succ 1}}$, it is clear that

$$Dom\,(Hd) \cdot \neg Dom\,(Tl; Min) = 1'_{L^1},$$

and

$$\neg Dom\,(Hd) \cdot Dom\,(Tl; Min) = 0 \ .$$

Recalling formulas (7.14) and (7.28), the part of the algorithm given by

$$(Dom\,(Hd) \cdot \neg Dom\,(Tl;Min))\,;Hd;1'\,;1'_{Nat}$$

is subsumed by the base case of the algorithm. Also, since the part of the algorithm given by the term

$$(\neg Dom\,(Hd) \cdot Dom\,(Tl;Min))\,;Tl;Min;1'$$

equals 0 (because $\neg Dom\,(Hd) \cdot Dom\,(Tl;Min) = 0$), (7.31) is equivalent to

$$GenD\&C(Min, 1'_{L^1};Hd, 1'_{L^{\succ 1}}, Hd \nabla Tl, 1' \otimes Min, Join_1)\ . \tag{7.32}$$

Therefore, from (7.32) and (7.14) the following algorithm (in fork algebraic form) is immediately at hand:

$$Min = 1'_{L^1};Hd + 1'_{L^{\succ 1}};\begin{array}{c} Hd \\ \nabla \\ Tl \end{array};\begin{array}{c} 1' \\ \otimes \\ Min \end{array};min\_num\ . \tag{7.33}$$

The algorithm presented in (7.33) corresponds to the following recursive function:

```
Function $Min(l$ : List(Nat)) : Nat
Begin
    If $Length(l) = 1$ Then
        $\leftarrow Hd(l)$
    Else
        $\leftarrow min\_num\,(Hd\,(l)\,, Min\,(Tl\,(l)))$
    End If
End.
```

7.6.1.2 *Finding the Minimum Common Ancestor*

In [G. A. Baum et al. (1996)] we presented as an example the derivation of an algorithm for finding the minimum common ancestor of a pair of nodes in a binary tree (See Fig. 7.3). The problem is informally specified as follows:

> Given a binary tree t and two nodes x and y, find that node a in t that is the closest ancestor of x and y.

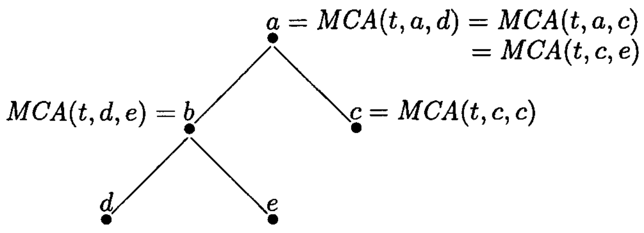

Fig. 7.3 Some computations of the relation *MCA* for a tree *t*.

Let us use a relation $HA \subseteq (Tree(\alpha) \times \alpha) \times \alpha$ (*HA* abbreviating *has ancestor*), which is meant to produce, given a tree t and a node x, the ancestors of x in t. A formal specification of a relation *MCA* capturing the problem in the language of the elementary theory of fork relations is given by

$$\begin{aligned} t\star(x\star y)MCA\,a \iff & t\star x HA\,a \ \wedge\ t\star y HA\,a \ \wedge \\ & \forall z\,((t\star x HA\,z \ \wedge\ t\star y HA\,z) \Rightarrow t\star a HA\,z)\,, \end{aligned}$$

and a specification of *HA* is given by the formula[†]:

$$t\star x HA\,a \iff \exists t'\,(t \sqsupseteq t' \ \wedge\ t'\,root\,a \ \wedge\ t'\,in\,x)\ .$$

In [G. A. Baum et al. (1996)] we gave the following abstract relational specification for the relation *HA*:

$$HA = \begin{array}{c} \sqsupseteq \\ \otimes \\ 1' \end{array} ; \left(\begin{array}{c} \pi\,;root \\ \nabla \\ \left(\begin{array}{c} in \\ \otimes \\ 1' \end{array} \right) ;\check{2} \end{array} \right) ;\pi,$$

[†]By $\sqsupseteq$ we denote the relation that relates a tree with its subtrees, by *root* the relation that relates a tree with its root node, and by *in* the relation that relates a tree with its elements. By $1'_{T^k}$ we denote the filter over tree of height k, by $1'_{T^{\succ k}}$ the filter over tree whose height is greater than k, and by $1'_{T^*}$ we denote the filter over the set of all trees.

and defined a relation $CA \subseteq (Tree(\alpha) \times (\alpha \times \alpha)) \times \alpha$ (characterizing the common ancestors of a pair of nodes in a tree) by

$$CA = ((1' \otimes \pi)\,;HA) \cdot ((1' \otimes \rho)\,;HA) \ .$$

In [G. A. Baum et al. (1996), p. 188] we derived the following recursive version of the relation CA:

$$
\begin{aligned}
CA = {} & \begin{array}{c} 1'_{T^1} \\ \otimes \\ 1' \end{array} ; \begin{array}{c} root \\ \otimes \\ \breve{2} \end{array} ;\breve{2} \\
+ {} & \begin{array}{c} 1'_{T^{\succ 1}} \\ \otimes \\ 1' \end{array} ; \left(Dom\left(\begin{array}{c} in \\ \otimes \\ \pi \end{array} ;\breve{2} \right) \cdot Dom\left(\begin{array}{c} in \\ \otimes \\ \rho \end{array} ;\breve{2} \right) \right) ;\pi;root \\
+ {} & \begin{array}{c} 1'_{T^{\succ 1}} \\ \otimes \\ 1' \end{array} ; \begin{array}{c} right \\ \otimes \\ 1' \end{array} ;CA \\
+ {} & \begin{array}{c} 1'_{T^{\succ 1}} \\ \otimes \\ 1' \end{array} ; \begin{array}{c} left \\ \otimes \\ 1' \end{array} ;CA \ .
\end{aligned}
$$

From the previously defined relations, we presented the following abstract relational specification of the relation MCA:

$$MCA = ((\pi \nabla CA) \cdot (CA \ \rightarrow \ HA))\,;\rho \ .$$

If we consider the relation $MCA^{\diamond}$ defined by

$$MCA^{\diamond} = (\pi \nabla CA) \cdot (CA \ \rightarrow \ HA) \, ,$$

we cannot apply the generic algorithm derived in Section 7.6.1 because the specification does not have the right pattern (compare with (7.22)).

In order to find an adequate specification for $MCA^{\diamond}$, we define relations HA' and CA' with types

$$HA' \subseteq (Tree(\alpha) \times \alpha) \times (Tree(\alpha) \times \alpha)$$

and

$$CA' \subseteq (Tree(\alpha) \times (Tree(\alpha) \times (\alpha \times \alpha))) \times (Tree(\alpha) \times \alpha)$$

as follows:

$$HA' = \pi \nabla HA,$$

and

$$\begin{aligned}
CA' = {} & \begin{pmatrix} 1'_{T^*} \\ \otimes \\ 1'_{T^1} \\ \otimes \\ 1' \end{pmatrix} ; \begin{pmatrix} 1' \\ \otimes \\ root \\ \otimes \\ \breve{2} \end{pmatrix} ; \breve{2} \\
+ {} & \begin{pmatrix} 1'_{T^*} \\ \otimes \\ 1'_{T^{\succ 1}} \\ \otimes \\ 1' \end{pmatrix} ; \underbrace{\begin{pmatrix} 1' \\ \otimes \\ \left(Dom \left(\begin{pmatrix} in \\ \otimes \\ \pi \end{pmatrix} ; \breve{2} \right) \cdot Dom \left(\begin{pmatrix} in \\ \otimes \\ \rho \end{pmatrix} ; \breve{2} \right) \right) \end{pmatrix} ; \pi ; root}_{R_1} \\
+ {} & \begin{pmatrix} 1'_{T^*} \\ \otimes \\ 1'_{T^{\succ 1}} \\ \otimes \\ 1' \end{pmatrix} ; \underbrace{\begin{pmatrix} 1' \\ \otimes \\ right \\ \otimes \\ 1' \end{pmatrix}}_{F_1} ; CA' \\
+ {} & \begin{pmatrix} 1'_{T^*} \\ \otimes \\ 1'_{T^{\succ 1}} \\ \otimes \\ 1' \end{pmatrix} ; \underbrace{\begin{pmatrix} 1' \\ \otimes \\ left \\ \otimes \\ 1' \end{pmatrix}}_{F_2} ; CA' .
\end{aligned}$$

Notice that the relation CA' differs from the relation CA in that it has an extra input (of type $Tree(\alpha)$) which is preserved and returned untouched as output.

It is easy to see that if we define MCA' using HA' and CA' by

$$MCA' = CA' \cdot \left(CA' \rightarrow HA'\right),$$

then $MCA^\diamond = (\pi \nabla 1') ; MCA'$, and thus,

$$MCA = (\pi \nabla 1') ; MCA' ; \rho. \tag{7.34}$$

Since CA' is a generator of type G_2 and Ass_1 and Ass_2 hold, we are in the right position to use the generic algorithm derived in Section 7.6.1 in order to find an algorithm computing the relation MCA'. Before applying the schema, notice that

$$Dom\left(MCA'\right) = \begin{array}{c} 1'_{T^*} \\ \otimes \\ \left(Dom\left(in \otimes \pi\right);\breve{2}\right) \cdot Dom\left(\left(in \otimes \rho\right);\breve{2}\right) \end{array},$$

i.e., MCA' is defined in an input $\langle t_1, \langle t_2, \langle x, y\rangle\rangle\rangle$ whenever x and y are nodes of t_2. As an elementary property of trees,

$$Dom\left(F_1;MCA'\right) \cdot Dom\left(F_2;MCA'\right) = 0,$$

i.e., nodes cannot appear both in the left and right subtrees. Also, notice that

$$\neg Dom\left(R_1\right) \cdot Dom\left(F_1;MCA'\right) = \neg Dom\left(R_1\right) \cdot Dom\left(F_2;MCA'\right) = 0,$$

i.e., if a node is not in a tree, then it is neither in the left nor in the right subtrees. Thus, from the previous reasoning, the following formula holds:

$$\begin{array}{l} GenD\&C(MCA', 1'_{base};MCA', D_{R,1,\neg 2}, R_1 \nabla F_1, 1' \otimes MCA', Join_1, \\ \qquad D_{R,\neg 1,2}, R_1 \nabla F_2, 1' \otimes MCA', Join_2, D_{R,\neg 1,\neg 2}, R_1, 1', 1'), \quad (7.35) \end{array}$$

where

$$1'_{base};MCA' = \left(\begin{array}{c} 1'_{T^*} \\ \otimes \\ T^1 \\ \otimes \\ \left(\begin{array}{c} 1' \\ \otimes \\ 1' \end{array}\right) \end{array}\right) ; \left(\begin{array}{c} 1' \\ \otimes \\ root \\ \otimes \\ \breve{2} \end{array}\right) ;\breve{2},$$

and

$$Join_1 = Join_2 = \left(\left(HA'^{\smile} \otimes 1'\right) + \left(1' \otimes HA'^{\smile}\right)\right);\breve{2}\ .$$

Notice that by Thm. 3.2.17, $Join_1$ and $Join_2$ can be rewritten as

$$Dom\left(\left(1' \otimes HA'\right);\breve{2}\right);\rho \ + \ Dom\left(\left(HA' \otimes 1'\right);\breve{2}\right);\pi\ .$$

We now have

$$
\begin{aligned}
& D_{R,1,\neg 2} ; \begin{array}{c} R_1 \\ \nabla \\ F_1 \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MCA' \end{array} ; Join_1 + D_{R,\neg 1,2} ; \begin{array}{c} R_1 \\ \nabla \\ F_2 \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MCA' \end{array} ; Join_2 \\
= & \{ \text{by Thm. 3.2.9} \} \\
& D_{R,1,\neg 2} ; \begin{array}{c} 1' \\ \nabla \\ F_1 \end{array} ; \begin{array}{c} R_1 \\ \otimes \\ 1' \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MCA' \end{array} ; Join_1 + D_{R,\neg 1,2} ; \begin{array}{c} 1' \\ \nabla \\ F_2 \end{array} ; \begin{array}{c} R_1 \\ \otimes \\ 1' \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MCA' \end{array} ; Join_2 \\
= & \{ \text{by } Join_1 = Join_2 \text{ and Ax. 2} \} \\
& \left(D_{R,1,\neg 2} ; \begin{array}{c} 1' \\ \nabla \\ F_1 \end{array} + D_{R,\neg 1,2} ; \begin{array}{c} 1' \\ \nabla \\ F_2 \end{array} \right) ; \begin{array}{c} R_1 \\ \otimes \\ 1' \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MCA' \end{array} ; Join_1 .
\end{aligned}
$$

From formula (7.34), the following program computes the relation MCA.

```
Function $MCA(t : \mathbf{Tree}(\alpha); x, y : \alpha) : \alpha$
Var
    $aux : \mathbf{Tree}(\alpha)$,
    $o : \alpha$.
Begin
    $\langle aux, o \rangle := MCA'(t, t, x, y)$,
    $\leftarrow o$
End.
```

A program to compute the relation MCA' is obtained from formula (7.35) and the derivation above.

```
Function $MCA'(t_1, t_2 : \mathbf{Tree}(\alpha); x, y : \alpha) : \mathbf{Tree}(\alpha) \times \alpha$
Var
    $e : \alpha$,
    $t', t'' : \mathbf{Tree}(\alpha)$.
Begin
    If $Heigth(t_2) = 1$ Then
      If $x = y = root(t_2)$ Then
        $\leftarrow \langle t_1, x \rangle$
    Else
      If $x$, $y$ occur in $left(t_2)$ or $x$, $y$ occur in $right(t_2)$ Then
        If $x$, $y$ occur in $left(t_2)$ Then
          $t' := left(t_2)$
        Else
          $t' := right(t_2)$
        End If
        $r := root(t_2)$,
        $\langle t'', e \rangle := MCA'(t_1, t', x, y)$,
        If $t_1 = t''$ and $r$ is an ancestor of $e$ in $t''$ Then
          $\leftarrow \langle t'', e \rangle$
        If $t_1 = t''$ and $e$ is an ancestor of $r$ in $t_1$ Then
          $\leftarrow \langle t_1, r \rangle$
      Else
        $\leftarrow \langle t_1, root(t_2) \rangle$
      End If
    End If
End.
```

Notice that in order to apply the strategy and obtain an algorithm it was necessary to perform an embedding when defining relations HA' and CA'. This embedding was motivated by the methodology, and thus the following heuristic arises.

Heuristic 7.4 In order to obtain a generator and a specification of the right form, it may be necessary to perform some embeddings.

The algorithm can be further optimized, but, as it stands now, it allows us to find a solution for computing the relation MCA. The experience we gained from comparing the previous derivation with the one given in [G. A. Baum et al. (1996)], is that once the generic algorithm is available, then finding the correct embedding takes only a short amount of time and an algorithm is easily obtained. In the derivation given in [G. A. Baum et al. (1996)], we carried out all the derivation of the generic algorithm, plus details that did not help at all in deriving that algorithm. Also, from a methodological point of view, the approach followed here seems much more appropriate.

7.6.2 *Second Example*

For this example we will need the following definition.

Definition 7.1 A list l is said to be a *contiguous sublist* of a list l' if there exist lists l_1 and l_2 such that $l' = l_1 \,\&\, l \,\&\, l_2$, where & denotes list concatenation.

Let us now consider the following generic problem.

> Given a list l, find the contiguous sublists l' of l satisfying (a) a condition c_1, and (b) l' is f-maximal with respect to all the contiguous sublists of l that satisfy the condition c_1 ($f : \mathbf{List(Int)} \to \mathbf{Int}$, functional).

If we assume that we already have a specification for the generator of contiguous sublists GCS, then the problem is specified by the following formula in the elementary theory of fork relations:

$$\begin{aligned} l P l' \quad &\Longleftrightarrow \\ &l\, GCS\, l' \wedge c_1(l') \wedge \forall l'' \left(l\, GCS\, l'' \wedge c_1(l'') \;\Rightarrow\; f(l') \succeq f(l'')\right) \;. \end{aligned} \tag{7.36}$$

From the first-order specification given in (7.36), we immediately obtain the following abstract relational specification:

$$P = GCS;C_1 \cdot \left(GCS;C_1 \;\rightarrow\; f;\succeq;\breve{f}\right) \;. \tag{7.37}$$

The generator of contiguous sublists GCS is specified by the following abstract relational equation

$$GCS = 1'_{L^1} + 1'_{L^{\succ 1}};STA + 1'_{L^{\succ 1}};Tl;GCS, \tag{7.38}$$

where the relation STA is specified by the recursive equation

$$STA = 1'_{L^1} + 1'_{L^{\succ 1}};\begin{array}{c} Hd \\ \nabla \\ \mathsf{C}_{Nil} \end{array};Cons + 1'_{L^{\succ 1}};\begin{array}{c} Hd \\ \nabla \\ Tl;STA \end{array};Cons \;. \tag{7.39}$$

Intuitively, relation STA generates the contiguous sublists that $STArt$ in the head of the list, and thus, the process of generating all the contiguous sublists can be divided between generating the contiguous sublists starting in the head, and generating the contiguous sublists located in the tail of the list.

Since for lists of length zero or one the problem is easy, in order to derive a divide-and-conquer algorithm for this problem it suffices to find relations $Split$, Q_1, Q_2 and $Join$ such that

$$Recomp\,(1'_{L^{\succ 1}};P, Split, Q_0, Q_1, Join)$$

holds.

$$
\begin{aligned}
& 1'_{L^{\succ 1}};P \\
= \; & \{\text{Unfolding the specification of } P \text{ given in (7.37)}\} \\
& 1'_{L^{\succ 1}};\left(GCS;C_1 \cdot \left(GCS;C_1 \rightarrow f;\succeq;\breve{f}\right)\right) \\
= \; & \{\text{Unfolding the definition of } GCS \text{ (cf. (7.38))}\} \\
& (1'_{L^{\succ 1}};STA + 1'_{L^{\succ 1}};Tl;GCS);C_1 \\
& \cdot \left((1'_{L^{\succ 1}};STA + 1'_{L^{\succ 1}};Tl;GCS);C_1 \rightarrow f;\succeq;\breve{f}\right) \\
= \; & \{\text{By Ax. 2}\} \\
& (1'_{L^{\succ 1}};STA;C_1 + 1'_{L^{\succ 1}};Tl;GCS;C_1) \\
& \cdot \left((1'_{L^{\succ 1}};STA;C_1 + 1'_{L^{\succ 1}};Tl;GCS;C_1) \rightarrow f;\succeq;\breve{f}\right) \\
= \; & \{\text{By (7.4) and elementary Boolean algebra}\} \\
& 1'_{L^{\succ 1}};STA;C_1 \cdot \left(STA;C_1 \rightarrow f;\succeq;\breve{f}\right) \cdot \left(Tl;GCS;C_1 \rightarrow f;\succeq;\breve{f}\right) \\
& + 1'_{L^{\succ 1}};Tl;GCS;C_1 \cdot \left(STA;C_1 \rightarrow f;\succeq;\breve{f}\right) \cdot \left(Tl;GCS;C_1 \rightarrow f;\succeq;\breve{f}\right).
\end{aligned}
$$

Thus,

$$
\begin{aligned}
& 1'_{L^{\succ 1}};P = \\
& \quad ((1'_{L^{\succ 1}};STA;C_1) \\
& \qquad \cdot (STA;C_1 \rightarrow f;\succeq;\breve{f}) \cdot (Tl;GCS;C_1 \rightarrow f;\succeq;\breve{f})) \\
& \quad + ((1'_{L^{\succ 1}};Tl;GCS;C_1) \\
& \qquad \cdot (STA;C_1 \rightarrow f;\succeq;\breve{f}) \cdot (Tl;GCS;C_1 \rightarrow f;\succeq;\breve{f}))\,. \qquad (7.40)
\end{aligned}
$$

Let us consider now the relation $MAXSTA$ defined by the equation

$$MAXSTA = STA;C_1 \cdot \left(STA;C_1 \rightarrow f;\succeq;\breve{f}\right)\,. \qquad (7.41)$$

In order to continue with the derivation we make the following assumptions:

$Ass_1 : 1'_{L^1};C_1 = 1'_{L^1}$,

$Ass_2 : Dom\,(f) = 1'_{L^*}$,

$Ass_3 : Cons;C_1 = (1' \otimes C_1);C_2;Cons$, for some $C_2 \leq 1'$,

$Ass_4 : (1'_{Int} \otimes 1');Cons;f = (g \otimes f);Add$, with $g \subseteq Int \times Int$ functional,

$$Ass_5 : \begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ;C_2 = Dom\left(\begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ;C_2\right);\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array}$$

$Ass_6 : \mathsf{C}_{Nil};f = \mathsf{C}_0$, (the empty list has f-value 0).

In Section 7.7 a complete derivation of the following divide-and-conquer algorithm for $MAXSTA$ is presented.

$$MAXSTA = 1\text{'}_{L^{1}} + 1\text{'}_{L^{\succ 1}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1\text{'} \\ \otimes \\ MAXSTA \end{array} ; Join_{MAXSTA}, \tag{7.42}$$

where the relation $Join_{MAXSTA}$ is defined by the following conditions

$$\begin{aligned} D_1 &:= Dom\,(f ; \preceq ; 1\text{'}_0)\,, \\ D_2 &:= Dom\,(f ; \succeq ; 1\text{'}_0)\,, \end{aligned}$$

and

$$Join_{MAXSTA} :=$$

$$C_2 ; \left(\begin{array}{c} 1\text{'} \\ \otimes \\ D_1 ; \complement_{Nil} \end{array} + \begin{array}{c} 1\text{'} \\ \otimes \\ D_2 \end{array} \right) ; Cons + \neg C_2 ; \begin{array}{c} 1\text{'} \\ \otimes \\ \complement_{Nil} \end{array} ; Cons\ .$$

Once we have a divide-and-conquer algorithm for computing the relation $MAXSTA$, we can continue with the derivation of a divide-and-conquer algorithm solving the original problem P.

Recalling (7.40), let us concentrate first on the relation E defined by

$$\begin{aligned} E := {} & (1\text{'}_{L^{\succ 1}} ; STA ; C_1) \\ & \cdot \left(STA ; C_1 \rightarrow f ; \succeq ; \breve{f} \right) \cdot \left(Tl ; GCS ; C_1 \rightarrow f ; \succeq ; \breve{f} \right) . \end{aligned}$$

$$\begin{aligned} E &= 1\text{'}_{L^{\succ 1}} ; MAXSTA \cdot \left(Tl ; GCS ; C_1 \rightarrow f ; \succeq ; \breve{f} \right) && \text{(by (7.41))} \\ &= 1\text{'}_{L^{\succ 1}} ; MAXSTA \cdot Tl ; \left(GCS ; C_1 \rightarrow f ; \succeq ; \breve{f} \right) && \text{(by Lemma 7.4)} \\ &= 1\text{'}_{L^{\succ 1}} ; MAXSTA \cdot Tl ; \overline{GCS ; C_1 ; \overline{\left(f ; \succeq ; \breve{f} \right)^{\smallsmile}}} && \text{(by (7.3))} \\ &= 1\text{'}_{L^{\succ 1}} ; MAXSTA \cdot Tl ; \overline{GCS ; C_1 ; \overline{f ; \breve{\succeq} ; \breve{f}}} && \text{(by Ax. 6)} \\ &= 1\text{'}_{L^{\succ 1}} ; MAXSTA \cdot Tl ; \overline{GCS ; C_1 ; f ; \overline{\preceq} ; \breve{f}} \\ & && \text{(by Thms. 2.3.19, 2.3.21 and } \breve{\succeq} = \preceq) \\ &= 1\text{'}_{L^{\succ 1}} ; MAXSTA \cdot Tl ; \overline{GCS ; C_1 ; f ; \succ ; \breve{f}}. && \text{(by } \overline{\preceq} = \succ) \end{aligned}$$

Then,

$$E = 1'_{L^{\succ 1}};MAXSTA \cdot Tl;\overline{GCS;C_1;f;\succ;\breve{f}}\,. \tag{7.43}$$

Since, by definition, the relation P produces as output those contiguous subsequences satisfying C_1 that are also f-maximal,

$$GCS;C_1;f;\succ \;=\; P;f;\succ,$$

and thus,

$$E = 1'_{L^{\succ 1}};MAXSTA \cdot Tl;\overline{P;f;\succ;\breve{f}}\,. \tag{7.44}$$

Notice that P returns f-maximal sequences, and therefore, even though there may be several f-maximal subsequences for a given sequence, their f-value must be the same. Thus, the relation $P;f$ is functional. This can be proved syntactically by showing that $(P;f)^{\smile};(P;f) \le 1'$, and is left as an exercise.

Resuming the derivation for the relation E, we have

$$\begin{aligned}
E &= 1'_{L^{\succ 1}};MAXSTA \cdot Tl;\overline{P;f;\succ;\breve{f}} && \text{(by (7.44))}\\
&= 1'_{L^{\succ 1}};MAXSTA \cdot Tl;P;f;\overline{\succ};\breve{f} && \text{(by Thms. 2.3.19 and 2.3.21)}\\
&= 1'_{L^{\succ 1}};MAXSTA \cdot Tl;P;f;\preceq;\breve{f} && \text{(by } \overline{\succ} = \preceq\text{)}\\
&= 1'_{L^{\succ 1}};(MAXSTA \cdot Tl;P;f;\preceq;\breve{f}) && \text{(by Thm. 2.3.22)}\\
&= 1'_{L^{\succ 1}};\begin{pmatrix} MAXSTA \\ \nabla \\ Tl;P;f;\preceq;\breve{f} \end{pmatrix};\breve{2} && \text{(by Thm. 3.2.1)}\\
&= 1'_{L^{\succ 1}};\begin{pmatrix} MAXSTA \\ \nabla \\ Tl;P \end{pmatrix};\begin{pmatrix} 1' \\ \otimes \\ f;\preceq;\breve{f} \end{pmatrix};\breve{2} && \text{(by Thm. 3.2.9)}\\
&= 1'_{L^{\succ 1}};\begin{pmatrix} MAXSTA \\ \nabla \\ Tl;P \end{pmatrix};Dom\begin{pmatrix} f & \\ \otimes & ;\breve{2} \\ f;\preceq & \end{pmatrix};\pi. && \text{(by Thm. 3.2.19)}
\end{aligned}$$

Let us concentrate now on the relation F defined by

$$\begin{aligned}
F := {} & 1'_{L^{\succ 1}};Tl;GCS;C_1 \\
& \cdot \left(STA;C_1 \to f;\succeq;\breve{f}\right) \cdot \left(Tl;GCS;C_1 \to f;\succeq;\breve{f}\right).
\end{aligned}$$

A derivation similar to the one used for relation E in order to arrive to (7.43) shows that

$$F = 1'_{L^{\succ 1}};Tl;P \cdot \overline{STA;C_1;f;\succ;\breve{f}}\ . \tag{7.45}$$

Since by definition the relation *MAXSTA* produces f-maximal subsequences that start in the head of the list given as input,

$$STA;C_1;f;\succ\ = MAXSTA;f;\succ\ . \tag{7.46}$$

Thus,

$$\begin{aligned} F &= 1'_{L^{\succ 1}};Tl;P \cdot \overline{STA;C_1;f;\succ;\breve{f}} && \text{(by (7.45))}\\ &= 1'_{L^{\succ 1}};Tl;P \cdot \overline{MAXSTA;f;\succ;\breve{f}}. && \text{(by (7.46))} \end{aligned}$$

We then deduce that

$$F = 1'_{L^{\succ 1}};Tl;P \cdot \overline{MAXSTA;f;\succ;\breve{f}}\ . \tag{7.47}$$

A similar proof to the one showing that the relation $P;f$ is functional, shows that $MAXSTA;f$ is also functional. Then,

$$\begin{aligned} F &= 1'_{L^{\succ 1}};Tl;P \cdot \overline{MAXSTA;f;\succ;\breve{f}} && \text{(by (7.47))}\\ &= 1'_{L^{\succ 1}};Tl;P \cdot MAXSTA;f;\overline{\succ};\breve{f} && \text{(by Thms. 2.3.19 and 2.3.21)}\\ &= 1'_{L^{\succ 1}};Tl;P \cdot MAXSTA;f;\preceq;\breve{f} && \text{(by } \overline{\succ} = \preceq\text{)}\\ &= 1'_{L^{\succ 1}};(Tl;P \cdot MAXSTA;f;\preceq;\breve{f}) && \text{(by Thm. 2.3.22)}\\ &= 1'_{L^{\succ 1}};\left(\begin{array}{c} MAXSTA;f;\preceq;\breve{f} \\ \nabla \\ Tl;P \end{array}\right);\breve{2} && \text{(by Thm. 3.2.1)}\\ &= 1'_{L^{\succ 1}};\left(\begin{array}{c} MAXSTA \\ \nabla \\ Tl;P \end{array}\right);\left(\begin{array}{c} f;\preceq;\breve{f} \\ \otimes \\ 1' \end{array}\right);\breve{2} && \text{(by Thm. 3.2.9)}\\ &= 1'_{L^{\succ 1}};\left(\begin{array}{c} MAXSTA \\ \nabla \\ Tl;P \end{array}\right);Dom\left(\begin{array}{c} f;\preceq \\ \otimes \\ f \end{array};\breve{2}\right);\rho. && \text{(by Thm. 3.2.19)} \end{aligned}$$

We have now arrived to the first algorithmic expression for computing the relation P. Since $1'_{L^1};P = 1'_{L^1}$ and by (7.40) we have

$$1'_{L^{\succ 1}};P = E + F,$$

then P equals

$$1'_{L^1} + 1'_{L^{\succ 1}};\begin{pmatrix} MAXSTA \\ \nabla \\ Tl;P \end{pmatrix};Dom\begin{pmatrix} f & \\ \otimes & ;\breve{2} \\ f;\preceq & \end{pmatrix};\pi$$
$$+\ 1'_{L^{\succ 1}};\begin{pmatrix} MAXSTA \\ \nabla \\ Tl;P \end{pmatrix};Dom\begin{pmatrix} f;\preceq & \\ \otimes & ;\breve{2} \\ f & \end{pmatrix};\rho\ .$$

Let us define the filters D_3 and D_4 by:

$$D_3 := Dom\left((f \otimes f;\preceq);\breve{2}\right),\quad D_4 := Dom\left((f;\preceq \otimes f);\breve{2}\right)\ .$$

Applying Ax. 2, we arrive at the equation

$$P = 1'_{L^1} + 1'_{L^{\succ 1}};\begin{matrix} MAXSTA \\ \nabla \\ Tl;P \end{matrix};(D_3;\pi + D_4;\rho)\ . \tag{7.48}$$

Even though (7.48) is algorithmic, the algorithm can be improved. Let us define the relation $MAXF$ by the equation

$$MAXF = 1'_{L^{\succ 1}};(MAXSTA \nabla Tl;P)\ .$$

Unfolding the definitions for relations $MAXSTA$ and P given in (7.42) and (7.48), and applying elementary properties of fork algebras,

$$MAXF = \underbrace{1'_{L^{\succ 1}};\begin{pmatrix} \begin{matrix} Hd \\ \nabla \\ Tl \end{matrix} ; \begin{matrix} 1' \\ \otimes \\ MAXSTA \end{matrix} ;Join_{MAXSTA} \\ \nabla \\ Tl;1'_{L^1} \end{pmatrix}}_{T_1}$$

$$+\ 1'_{L^{\succ 1}};\underbrace{\left(\begin{array}{c}\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};Join_{MAXSTA}\\ \nabla\\ Tl;\begin{array}{c}MAXSTA\\ \nabla\\ Tl;P\end{array};(D_3;\pi+D_4;\rho)\end{array}\right)}_{T_2}.$$

Let us analyze terms T_1 and T_2, one at a time. For term T_1, the subterm $Tl;1'_{L^1}$ equals $1'_{L^2};Tl$. This implies that the input for the whole term must be restricted to lists of length two. Thus,

$$T_1 = 1'_{L^2};\left(\begin{array}{c}\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};Join_{MAXSTA}\\ \nabla\\ Tl\end{array}\right). \tag{7.49}$$

We then proceed as follows:

$$\begin{aligned}
T_1 &= 1'_{L^2};\left(\begin{array}{c}\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};Join_{MAXSTA}\\ \nabla\\ Tl\end{array}\right) && \text{(by (7.49))}\\
&= 1'_{L^2};\left(\begin{array}{c}\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};Join_{MAXSTA}\\ \nabla\\ \left(\begin{array}{c}Hd\\ \nabla\\ Tl\end{array}\right);\rho\end{array}\right) && \text{(by Thm. 3.2.2)}\\
&= 1'_{L^2};\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\left(\begin{array}{c}\begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};Join_{MAXSTA}\\ \nabla\\ \rho\end{array}\right). && \text{(by Thm. 3.2.4)}
\end{aligned}$$

Since the lists that will reach the input of the relation *MAXSTA* are all of length one, and $1'_{L^1}; MAXSTA = 1'_{L^1}$, we then conclude

$$T_1 = 1'_{L^2} ; (Hd \nabla Tl) ; (Join_{MAXSTA} \nabla \rho) \ .$$

Regarding term T_2, a derivation similar to the one performed from T_1 allows us to prove that

$$T_2 \geq 1'_{L^{\succ 2}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MAXF \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \ ; Join_{MAXSTA} \\ \pi \\ \nabla \\ \rho ; (D_3;\pi + D_4;\rho) \end{array} \right) . \tag{7.50}$$

Notice that even though we do not arrive at an equality, the term on the right hand side of (7.50) has the same domain as term T_2, and therefore it is a refinement. Thus, the equation

$$\begin{aligned} MAXF = {} & 1'_{L^2} ; (Hd \nabla Tl) ; (Join_{MAXSTA} \nabla \rho) \\ & + 1'_{L^{\succ 2}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MAXF \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \ ; Join_{MAXSTA} \\ \pi \\ \nabla \\ \rho ; (D_3;\pi + D_4;\rho) \end{array} \right) \end{aligned} \tag{7.51}$$

is a divide-and-conquer algorithm computing a refinement of the relation *MAXF*. Equation (7.51) can be slightly optimized, by using properties of lists and Ax. 2, to the equation

$$\begin{aligned} MAXF = 1'_{L^{\succeq 2}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \Bigg(& \begin{array}{c} 1' \\ \otimes \\ 1'_{L^1} \end{array} ; \left(\begin{array}{c} Join_{MAXSTA} \\ \nabla \\ \rho \end{array} \right) \\ & + \begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succ 1}} \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MAXF \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \ ; Join_{MAXSTA} \\ \pi \\ \nabla \\ \rho ; (D_3;\pi + D_4;\rho) \end{array} \right) \Bigg) . \end{aligned} \tag{7.52}$$

Finally, the problem P is solved using the auxiliary algorithm $MAXF$ as indicated by the following equation:

$$P = 1'_{L^1} + 1'_{L^{\succ 1}} ; MAXF ; (D_3 ; \pi + D_4 ; \rho) \ . \tag{7.53}$$

Eq. (7.52) corresponds to the following program.

```
Function $MAXF(l : \mathbf{List(Int)}) : \mathbf{List(Int)} \times \mathbf{List(Int)}$
Var $h$ : Int, $t$, $o_1$, $o_2$, $l_1$, $l_2$ : List(Int),
Begin
    $h := Hd(l)$, $t := Tl(l)$,
    If $Length(t) = 1$ Then
        If $C_2(h,t)$ Then
            If $f(t) \leq 0$ Then $o_1 := [h]$
            Else $o_1 := l$
            End If
        Else
            $o1 := [h]$
        End If
        $\leftarrow \langle o_1, t \rangle$
    Else
        $\langle l_1, l_2 \rangle := MAXF(t)$,
        If $C_2(h, l_1)$ Then
            If $f(l_1) \leq 0$ Then $o_1 := [h]$
            Else $o_1 := [h : l_1]$
            End If
        Else
            $o_1 := [h]$
        End If
        If $f(l_1) \geq f(l_2)$ Then $o_2 := l_1$
        Else $o_2 := l_2$
        End If
        $\leftarrow \langle o_1, o_2 \rangle$,
    End If
End.
```

In the remaining part of this section we will derive algorithms for solving two problems whose specifications are instances of the generic problem P. The problems that we will use as examples are related to problems already studied in the literature. The first problem consists of finding the sublist with maximum sum. In [D. Smith (1987)] a divide-and-conquer algorithm for solving this problem is derived. The second problem consists of finding the longest plateau. An algorithm solving a weakened version of this problem – the input list is assumed to be sorted – is derived in [D. Gries (1981)] using the predicate transformer *wp* (weakest precondition).

7.6.2.1 *Finding the Contiguous Sublists of Maximum Sum*

The problem of finding a contiguous sublist of maximum sum was treated as a case study in [D. Smith (1987)]. There, a divide-and-conquer algorithm is derived for solving this specific problem. The problem is informally specified as follows.

> Given a list l having integer numbers as elements, find a contiguous sublist of l with maximum sum.

This problem has a clear specification in the elementary theory of fork relations given by the formula:

$$l\,MAXSUM\,l' \iff l\,GCS\,l' \wedge \forall l''\,(l\,GCS\,l'' \Rightarrow Sum(l') \succeq Sum(l'')), \quad (7.54)$$

where $Sum \subseteq List(Int) \times Int$ computes the sum of the elements of the list given as input. A relational specification for $MAXSUM$ is given by the following equation:

$$MAXSUM = GCS \cdot (GCS \rightarrow Sum;\succeq;Sum^{\smile}) \ .$$

If we take $C_1 := 1'$, then (7.54) has the same shape as (7.37). Let us check that assumptions Ass_1–Ass_6 hold.

Since $C_1 = 1'$, $1'_{L^1};C_1 = 1'_{L^1}$, and thus Ass_1 holds.

Since $Dom\,(Sum) = 1'_{L^*}$, Ass_2 also holds.

If we define $C_2 := 1'$,

$$Cons;C_1 = Cons;1' = Cons = \begin{matrix} 1' \\ \otimes \\ 1' \end{matrix} ;1';Cons = \begin{matrix} 1' \\ \otimes \\ C_1 \end{matrix} ;C_2;Cons,$$

and Ass_3 holds.

Since

$$\begin{array}{c} 1'_{Int} \\ \otimes \\ 1' \end{array} ; Cons ; Sum = \begin{array}{c} 1'_{Int} \\ \otimes \\ Sum \end{array} ; Add,$$

defining $g := 1'_{Int}$ we are done with Ass_4.

Regarding Ass_5, notice that

$$\begin{array}{c} 1' \\ \otimes \\ STA ; C_1 \end{array} ; C_2 = \begin{array}{c} 1' \\ \otimes \\ STA ; 1' \end{array} ; 1' = \begin{array}{c} 1' \\ \otimes \\ STA \end{array} .$$

Notice also that since $Dom\,(MAXSTA) = 1'_{L^{\succeq 1}}$,

$$Dom \left(\begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ; C_2 \right) = Dom \left(\begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} \right) = \begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succeq 1}} \end{array} .$$

Then,

$$Dom \left(\begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ; C_2 \right) ; \begin{array}{c} 1' \\ \otimes \\ STA ; C_1 \end{array} = \begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succeq 1}} \end{array} ; \begin{array}{c} 1' \\ \otimes \\ STA \end{array} = \begin{array}{c} 1' \\ \otimes \\ STA \end{array} ,$$

and Ass_5 holds.

Finally, since $\mathsf{C}_{Nil} ; Sum = \mathsf{C}_0$, Ass_6 holds.

If we now instantiate the relational algorithms given in (7.52) and (7.53), we have

$$MAXSUM = 1'_{L^1} + 1'_{L^{\succ 1}} ; MAXF ; (D_3 ; \pi + D_4 ; \rho) , \tag{7.55}$$

where

$$D_3 = Dom \left(\begin{array}{c} Sum \\ \otimes \\ Sum ; \preceq \end{array} ; \breve{2} \right) \quad \text{and} \quad D_4 = Dom \left(\begin{array}{c} Sum ; \preceq \\ \otimes \\ Sum \end{array} ; \breve{2} \right) .$$

The relation $MAXF$ is defined by

$$MAXF = 1'_{L^{\succeq 2}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left[\begin{array}{c} 1' \\ \otimes \\ 1'_{L^1} \end{array} ; \begin{array}{c} Join_{MAXSTA} \\ \nabla \\ \rho \end{array} + \right.$$

$$\left. \begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succ 1}} \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MAXF \end{array} ; \left(\begin{array}{c} \begin{array}{c} 1' \\ \otimes \\ \pi \end{array} ; Join_{MAXSTA} \\ \nabla \\ \rho ; (D_3 ; \pi + D_4 ; \rho) \end{array} \right) \right], \quad (7.56)$$

where

$$Join_{MAXSTA} = \left(\begin{array}{c} 1' \\ \otimes \\ Dom\,(Sum ; \preceq ; 1'_0) ; \mathsf{C}_{Nil} \end{array} + \begin{array}{c} 1' \\ \otimes \\ Dom\,(Sum ; \succeq ; 1'_0) \end{array} \right) ; Cons.$$

Equations (7.55) and (7.56) correspond to the algorithms below.

```
**Function** $MAXSUM(l : \textbf{List(Int)}) : \textbf{List(Int)}$
**Var**
    $l_1, l_2$ : **List(Int)**
**Begin**
    **If** $Length(l) = 1$ **Then**
        $\leftarrow l$
    **Else**
        $\langle l_1, l_2 \rangle := MAXF(l)$
        **If** $Sum(l_1) \geq Sum(l_2)$ **Then**
            $\leftarrow l_1$
        **Else**
            $\leftarrow l_2$
        **End If**
    **End If**
**End.**
```

```
**Function** $MAXF(l : \mathbf{List(Int)}) : \mathbf{List(Int)} \times \mathbf{List(Int)}$
**Var** $h$ : **Int**, $t, o_1, o_2, l_1, l_2$ : **List(Int)**
**Begin**
    $h := Hd(l), t := Tl(l)$
    **If** $Length(t) = 1$ **Then**
      **If** $Sum(t) \leq 0$ **Then** $o_1 := [h]$
      **Else** $o_1 := l$
      **End If**
      $\leftarrow \langle o_1, t \rangle$
    **Else**
      $\langle l_1, l_2 \rangle := MAXF(t)$
      **If** $Sum(l_1) \leq 0$ **Then** $o_1 := [h]$
      **Else** $o_1 := [h : l_1]$
      **End If**
      **If** $Sum(l_1) \geq Sum(l_2)$ **Then** $o_2 := l_1$
      **Else** $o_2 := l_2$
      **End If**
      $\leftarrow \langle o_1, o_2 \rangle$
    **End If**
**End.**
```

7.6.2.2 *Finding the Longest Plateau*

The problem we will solve in this section is a strengthened version of a problem used as example in [D. Gries (1981)]. Therein, given a sorted list, it is given to find the longest *plateau*, i.e., the longest contiguous sublist of the input list whose elements are all the same. Here we will drop the assumption of the input list being sorted, and will derive an algorithm for finding the longest plateau in an arbitrary list of integers. The problem is informally specified by the following sentence.

Given a list l, find the longest plateau p in l.

A formal specification is given by the following formula in the elementary theory of fork relations:

$$\begin{aligned} l\,LPLATEAU\,p \iff\ & l\,GCS\,p \wedge Plateau(p) \\ & \wedge \forall p' \left(l\,GCS\,p' \wedge Plateau(p') \Rightarrow Length(p) \succeq Length(p') \right), \end{aligned} \quad (7.57)$$

where the unary predicate *Plateau* is defined by the formula

$$Plateau(p) \quad \Longleftrightarrow \quad \forall x \forall y \left(p\, Has\, x \wedge p\, Has\, y \;\Rightarrow\; x = y \right) \; .$$

A relational specification of the problem *LPLATEAU* (obtained from (7.57)) is given by the equation

$$\begin{aligned} LPLATEAU = {} & (GCS;Plateau) \\ & \cdot \; (GCS;Plateau \;\rightarrow\; Length;\succeq;Length^{\smile}) \, , \end{aligned} \tag{7.58}$$

where the relation *Plateau* is defined by the recursive equation

$$Plateau = 1'_{L\preceq 1} + 1'_{L\succ 1}; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1' \\ \otimes \\ Plateau \end{array} ;Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ;\breve{2} \right) ;Cons \; .$$

Notice that (7.58) has the same shape as (7.37). Let us check that assumptions Ass_1–Ass_6 hold.

Ass_1 is trivially true. Since *Length* is total on the set of lists, Ass_2 also holds. Let us define

$$C_2 = (1' \otimes 1'_{L^0}) + Dom\left((1' \otimes Hd);\breve{2}\right) \; .$$

Then,

$$\begin{aligned}
& Cons;C_1 \\
& \quad = Cons;Plateau \\
& \quad = Cons;\left(1'_{L\preceq 1} + 1'_{L\succ 1}; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1' \\ \otimes \\ Plateau \end{array} ;Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ;\breve{2} \right) ;Cons \right) \\
& \quad = Cons;1'_{L\preceq 1} + Cons;1'_{L\succ 1}; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1' \\ \otimes \\ Plateau \end{array} ;Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ;\breve{2} \right) ;Cons \\
& \quad = \begin{array}{c} 1' \\ \otimes \\ 1'_{L0} \end{array} ;Cons + \begin{array}{c} 1' \\ \otimes \\ 1'_{L\succeq 1} \end{array} ;Cons; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1' \\ \otimes \\ Plateau \end{array} ;Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ;\breve{2} \right) ;Cons \\
& \quad = \begin{array}{c} 1' \\ \otimes \\ Plateau;1'_{L0} \end{array} ;Cons + \begin{array}{c} 1' \\ \otimes \\ 1'_{L\succeq 1} \end{array} ; \begin{array}{c} 1' \\ \otimes \\ Plateau \end{array} ;Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ;\breve{2} \right) ;Cons \\
& \quad = \begin{array}{c} 1' \\ \otimes \\ Plateau;1'_{L0} \end{array} ;Cons + \begin{array}{c} 1' \\ \otimes \\ Plateau \end{array} ;Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ;\breve{2} \right) ;Cons
\end{aligned}$$

$$= \begin{array}{c} 1' \\ \otimes \\ Plateau \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ 1'_{L^0} \end{array} + Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ; \breve{2} \right) \right) ; Cons$$

$$= \begin{array}{c} 1' \\ \otimes \\ Plateau \end{array} ; C_2 ; Cons.$$

Thus, Ass_3 also holds.

Regarding Ass_4, notice that

$$\begin{array}{c} 1'_{Int} \\ \otimes \\ 1' \end{array} ; Cons ; Length = \begin{array}{c} 1'_{Int} ; \mathsf{C}_1 \\ \otimes \\ Length \end{array} ; Add,$$

and defining $g := 1'_{Int} ; \mathsf{C}_1$, Ass_4 holds.

In a similar way assumptions Ass_5 and Ass_6 are proved.

If we now instantiate the relational algorithms given in (7.52) and (7.53), we obtain

$$LPLATEAU = 1'_{L^1} + 1'_{L^{\succeq 1}} ; MAXF ; (D_3 ; \pi + D_4 ; \rho) , \qquad (7.59)$$

where

$$D_3 = Dom \left(\begin{array}{c} Length \\ \otimes \\ Length ; \preceq \end{array} ; \breve{2} \right) \quad \text{and} \quad D_4 = Dom \left(\begin{array}{c} Length ; \preceq \\ \otimes \\ Length \end{array} ; \breve{2} \right) .$$

The relation $MAXF$ is defined by

$$MAXF = 1'_{L^{\succeq 2}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left[\begin{array}{c} 1' \\ \otimes \\ 1'_{L^1} \end{array} ; \begin{array}{c} Join_{MAXSTA} \\ \nabla \\ \rho \end{array} + \right.$$

$$\left. \begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succ 1}} \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MAXF \end{array} ; \left(\begin{array}{c} \begin{array}{c} 1' \\ \otimes \\ \pi \end{array} ; Join_{MAXSTA} \\ \nabla \\ \rho ; (D_3 ; \pi + D_4 ; \rho) \end{array} \right) \right] ,$$

where

$$
\begin{aligned}
&Join_{MAXSTA} = \\
&C_2 ; \left(\begin{array}{c} 1' \\ \otimes \\ Dom\,(Length ; \preceq ; 1'_0)\,; \mathsf{C}_{Nil} \end{array} + \begin{array}{c} 1' \\ \otimes \\ Dom\,(Length ; \succeq ; 1'_0) \end{array} \right) ; Cons \\
&\qquad\qquad\qquad\qquad\qquad + \neg C_2 ; \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons\ .
\end{aligned}
$$

Notice that $Dom\,(Length ; \preceq ; 1'_0) = 1'_{L^0}$ and $Dom\,(Length ; \succeq ; 1'_0) = 1'_{L^{\succeq 0}}$. Then,

$$
\begin{aligned}
Join_{MAXSTA} &= C_2 ; \left(\begin{array}{c} 1' \\ \otimes \\ 1'_{L^0} \end{array} + \begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succeq 0}} \end{array} \right) ; Cons + \neg C_2 ; \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons \\
&= C_2 ; \begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succeq 0}} \end{array} ; Cons + \neg C_2 ; \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons.
\end{aligned}
$$

Since $C_2 = (1' \otimes 1'_{L^0}) + Dom\left((1' \otimes Hd)\,; \breve{2}\right)$, simple equational reasoning allows to deduce that

$$
\neg C_2 = (1' \otimes 1'_{L^{\succ 0}})\,; \neg Dom\left((1' \otimes Hd)\,; \breve{2}\right)\ .
$$

Then,

$$
\begin{aligned}
&\begin{array}{c} 1' \\ \otimes \\ 1'_{L^1} \end{array} ; Join_{MAXSTA} = \\
&\begin{array}{c} 1' \\ \otimes \\ 1'_{L^1} \end{array} ; \underbrace{\left(Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ; \breve{2} \right) + \neg Dom \left(\begin{array}{c} 1' \\ \otimes \\ Hd \end{array} ; \breve{2} \right) ; \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} \right) ; Cons}_{T} . \qquad (7.60)
\end{aligned}
$$

Notice also that

$$
\begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succ 0}} \end{array} ; Join_{MAXSTA} = \begin{array}{c} 1' \\ \otimes \\ 1'_{L^{\succ 0}} \end{array} ; T\ . \qquad (7.61)
$$

Thus,

$$\begin{array}{ll}
& 1'_{L\succeq 2};MAXF \\
= & \{\text{by (7.60) and (7.61)}\} \\
& \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left[\begin{array}{c} 1' \\ \otimes \\ 1'_{L^1} \end{array} ; \begin{array}{c} T \\ \nabla \\ \rho \end{array} + \begin{array}{c} 1' \\ \otimes \\ 1'_{L\succ 1} \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MAXF \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \;;T \\ \pi \\ \nabla \\ \rho;(D_3;\pi + D_4;\rho) \end{array} \right) \right] \\
= & \{\text{by Thm. 3.2.9 and Ax. 2}\} \\
& \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left[\begin{array}{c} 1' \\ \otimes \\ 1'_{L^1} \end{array} ; \begin{array}{c} 1' \\ \nabla \\ \rho \end{array} + \begin{array}{c} 1' \\ \otimes \\ 1'_{L\succ 1} \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MAXF \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \pi \\ \nabla \\ \rho;(D_3;\pi + D_4;\rho) \end{array} \right) \right] ; \begin{array}{c} T \\ \otimes \\ 1' \end{array} .
\end{array}$$

We then have

$$MAXF = 1'_{L\succeq 2}; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left[\begin{array}{c} 1' \\ \otimes \\ 1'_{L^1} \end{array} ; \begin{array}{c} 1' \\ \nabla \\ \rho \end{array} + \right.$$

$$\left. \begin{array}{c} 1' \\ \otimes \\ 1'_{L\succ 1} \end{array} ; \begin{array}{c} 1' \\ \otimes \\ MAXF \end{array} ; \left(\begin{array}{c} \left(\begin{array}{c} 1' \\ \otimes \\ \pi \end{array} \right) \\ \nabla \\ \rho;(D_3;\pi + D_4;\rho) \end{array} \right) \right] ; \begin{array}{c} T \\ \otimes \\ 1' \end{array} . \qquad (7.62)$$

Equations (7.59) and (7.62) correspond to the algorithms below.

```
**Function** $LPLATEAU(l : \textbf{List(Int)}) : \textbf{List(Int)}$
**Var**
    $l_1, l_2 : \textbf{List(Int)}$
**Begin**
    **If** $Length(l) = 1$ **Then** $\leftarrow l$
    **Else**
        $\langle l_1, l_2\rangle := MAXF(l)$
        **If** $Length(l_1) \geq Length(l_2)$ **Then** $\leftarrow l_1$
        **Else** $\leftarrow l_2$
        **End If**
    **End If**
**End.**
```

```
Function $MAXF(l : \mathbf{List(Int)}) : \mathbf{List(Int)} \times \mathbf{List(Int)}$
Var $h : \mathbf{Int}$, $t, o_1, o_2, l_1, l_2 : \mathbf{List(Int)}$
Begin
    $h := Hd(l), t := Tl(l)$
    If $Length(t) = 1$ Then
      $o_2 := t, l_1 := t$
    Else
      $\langle l_1, l_2 \rangle := MAXF(t)$
      If $Length(l_1) \geq Length(l_2)$ Then $o_2 := l_1$
      Else $o_2 := l_2$
      End If
    End If
    If $h = Hd(l_1)$ Then $o_1 := [h : l_1]$
    Else $o_1 := [h]$
    End If
    $\leftarrow \langle o_1, o_2 \rangle$
End.
```

7.7 A *D&C* Algorithm for *MAXSTA*

In this section we include the complete derivation of the algorithm for computing the relation *MAXSTA* given in (7.42). We start by deriving the base case of the algorithm, that is, when the input list has length 1.

$$
\begin{aligned}
&1'_{L^1};MAXSTA \\
&\quad = 1'_{L^1};\left(STA;C_1 \cdot \left(STA;C_1 \rightarrow f;\succeq;\breve{f}\right)\right) && \text{(by (7.41))} \\
&\quad = 1'_{L^1};STA;C_1 \cdot 1'_{L^1};\left(STA;C_1 \rightarrow f;\succeq;\breve{f}\right) && \text{(by Thm. 2.3.17)} \\
&\quad = 1'_{L^1};C_1 \cdot 1'_{L^1};\left(1'_{L^1};STA;C_1 \rightarrow f;\succeq;\breve{f}\right) && \text{(by Lemma 7.4)} \\
&\quad = 1'_{L^1};C_1 \cdot 1'_{L^1};\left(1'_{L^1};C_1 \rightarrow f;\succeq;\breve{f}\right) && \text{(by (7.39))} \\
&\quad = 1'_{L^1};C_1 \cdot 1'_{L^1};C_1;\left(f;\succeq;\breve{f}\right)^{\smile} && \text{(by Lemma 7.3)} \\
&\quad = 1'_{L^1} \cdot 1'_{L^1};\left(f;\succeq;\breve{f}\right)^{\smile} && \text{(by } Ass_1\text{)}
\end{aligned}
$$

$$\begin{aligned} &= 1'_{L^1} \cdot 1'_{L^1} ; \breve{\breve{f}} ; \breve{\succeq} ; \breve{f} && \text{(by Ax. 6)} \\ &= 1'_{L^1} \cdot 1'_{L^1} ; f ; \preceq ; \breve{f}. && \text{(by Ax. 4 and } \breve{\succeq} = \preceq) \end{aligned}$$

Notice also that

$$\begin{aligned} 1'_{L^1} ; f ; \preceq ; \breve{f} &\geq 1'_{L^1} ; f ; \breve{f} && (\preceq \geq 1'_{Int}) \\ &\geq 1'_{L^1} ; Dom\,(f) && \text{(by Def. 2.5)} \\ &= 1'_{L^1} ; 1'_{L^*} && \text{(by } Ass_2) \\ &= 1'_{L^1} \cdot 1'_{L^*} && \text{(by Thm. 2.3.7)} \\ &= 1'_{L^1}. && \text{(property of lists)} \end{aligned}$$

Thus, since $1'_{L^1} ; f ; \preceq ; \breve{f} \geq 1'_{L^1}$,

$$1'_{L^1} ; MAXSTA = 1'_{L^1} \ . \tag{7.63}$$

Once the base case has been derived, let us concentrate on the derivation of the recursive case. Since we are looking for a divide-and-conquer solution, we must find relations *Split*, Q_1, Q_2 and *Join* such that the predicate

$$Recomp\,(1'_{L^{\succ 1}} ; MAXSTA, Split, Q_1, Q_2, Join)$$

holds.

If we unfold the definition of *STA* in *MAXSTA*, and apply properties of the relational implication, we can prove that

$$\begin{aligned} &1'_{L^{\succ 1}} ; MAXSTA = \\ &\left[\left(1'_{L^{\succ 1}} ; \begin{matrix} Hd \\ \nabla \\ \mathsf{C}_{Nil} \end{matrix} ; Cons ; C_1 \right) \cdot \left(\begin{matrix} Hd \\ \nabla \\ \mathsf{C}_{Nil} \end{matrix} ; Cons ; C_1 \ \rightarrow \ f ; \succeq ; \breve{f} \right) \right. \\ &\qquad\qquad \left. \cdot \left(\begin{matrix} Hd \\ \nabla \\ Tl ; STA \end{matrix} ; Cons ; C_1 \ \rightarrow \ f ; \succeq ; \breve{f} \right) \right] \\ &+ \left[\left(1'_{L^{\succ 1}} ; \begin{matrix} Hd \\ \nabla \\ Tl ; STA \end{matrix} ; Cons ; C_1 \right) \cdot \left(\begin{matrix} Hd \\ \nabla \\ \mathsf{C}_{Nil} \end{matrix} ; Cons ; C_1 \ \rightarrow \ f ; \succeq ; \breve{f} \right) \right. \\ &\qquad\qquad \left. \cdot \left(\begin{matrix} Hd \\ \nabla \\ Tl ; STA \end{matrix} ; Cons ; C_1 \ \rightarrow \ f ; \succeq ; \breve{f} \right) \right] . \end{aligned}$$

We will call A the first term in the sum, and B the second one. Let us analyze term A first.

$$A = \left(1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ \mathrm{C}_{Nil} \end{array} ; Cons ; C_1 \right) \cdot \left(\begin{array}{c} Hd \\ \nabla \\ \mathrm{C}_{Nil} \end{array} ; Cons ; C_1 ; f ; \preceq ; \breve{f} \right)$$

$$\cdot \left(\begin{array}{c} Hd \\ \nabla \\ Tl;STA \end{array} ; Cons ; C_1 \rightarrow f ; \succeq ; \breve{f} \right) \qquad \text{(by Lemma 7.3)}$$

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ \mathrm{C}_{Nil} \end{array} ; Cons ; C_1 \cdot \left(\begin{array}{c} Hd \\ \nabla \\ Tl;STA \end{array} ; Cons ; C_1 \rightarrow f ; \succeq ; \breve{f} \right) \quad \text{(by Thm. 2.3.17)}$$

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ \mathrm{C}_{Nil} \end{array} ; Cons \cdot \left(\begin{array}{c} Hd \\ \nabla \\ Tl;STA;C_1 \end{array} ; C_2 ; Cons \rightarrow f ; \succeq ; \breve{f} \right)$$

(by Ass_1 and Ass_3)

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl;\mathrm{C}_{Nil} \end{array} ; Cons \cdot \left(\begin{array}{c} Hd \\ \nabla \\ Tl;STA;C_1 \end{array} ; C_2 ; Cons \rightarrow f ; \succeq ; \breve{f} \right)$$

(by Def. 2.5 and Thm. 2.3.14)

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1' \\ \otimes \\ \mathrm{C}_{Nil} \end{array} ; Cons \cdot \left(\begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; Cons \rightarrow f ; \succeq ; \breve{f} \right)$$

(by Thm. 3.2.9)

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \mathrm{C}_{Nil} \end{array} ; Cons \cdot \left(\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; Cons \rightarrow f ; \succeq ; \breve{f} \right) \right)$$

(by Thm. 2.3.17 and Lemma 7.4)

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \mathrm{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; Cons ; \overline{\left(f ; \succeq ; \breve{f} \right)}^{\smile}} \right) \qquad \text{(by (7.3))}$$

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \mathrm{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; Cons ; \overline{f ; \preceq ; \breve{f}}} \right)$$

(by Ax. 6 and $\breve{\preceq} = \succeq$)

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \mathrm{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; Cons ; f ; \overline{\preceq} ; \breve{f}} \right)$$

(by Thms. 2.3.19 and 2.3.21)

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \mathrm{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; Cons ; f ; \succ ; \breve{f}} \right) \qquad \text{(by } \overline{\preceq} = \succ \text{)}$$

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; \begin{array}{c} g \\ \otimes \\ f \end{array} ; Add ; \succ ; \breve{f}} \right) \qquad \text{(by } Ass_4\text{)}$$

$$= 1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; \begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ; Add ; \breve{f}} \right) .$$

(by property of *Add*)

Then, A equals

$$1'_{L\succ 1} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; \begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ; Add ; \breve{f}} \right) . \tag{7.64}$$

Notice that by monotonicity of the operators involved,

$$\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; \begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ; Add ; \breve{f} \leq \begin{array}{c} 1 \\ \otimes \\ 1 \end{array} ; Cons . \tag{7.65}$$

Let us define

$$A_1 = \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; \begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ; Add ; \breve{f}} .$$

Then,

$$A_1 = \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; \begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ; Add ; \breve{f} \cdot \begin{array}{c} 1 \\ \otimes \\ 1 \end{array} ; Cons}$$

(by Def. A_1 and (7.65))

$$= \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; \begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ; Add ; \breve{f} \cdot \begin{array}{c} 0'+1' \\ \otimes \\ 1 \end{array} ; Cons}$$

(by Def. 0' and BA)

$$= \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; C_2 ; \begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ; Add ; \breve{f} \cdot \left(\begin{array}{c} 0' \\ \otimes \\ 1 \end{array} + \begin{array}{c} 1' \\ \otimes \\ 1 \end{array} \right) ; Cons}$$

(by Thm. 3.2.11)

$$= \begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons \cdot \overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}\cdot\left(\begin{array}{c}0'\\ \otimes\\ 1\end{array};Cons+\begin{array}{c}1'\\ \otimes\\ 1\end{array};Cons\right)}$$

(by Ax. 2)

$$= \begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons \cdot \overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}\cdot\begin{array}{c}0'\\ \otimes\\ 1\end{array};Cons}$$
$$\cdot\ \overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}\cdot\begin{array}{c}1'\\ \otimes\\ 1\end{array};Cons}.$$

(by BA)

We then have

$$A_1 = \begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons \cdot \underbrace{\overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}\cdot\begin{array}{c}0'\\ \otimes\\ 1\end{array};Cons}}_{A_2}$$
$$\cdot\ \underbrace{\overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}\cdot\begin{array}{c}1'\\ \otimes\\ 1\end{array};Cons}}_{A_3}. \quad (7.66)$$

$(1'\otimes\mathsf{C}_{Nil});Cons\ \cdot A_2$

$$= \begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons \cdot \overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}\cdot\begin{array}{c}0'\\ \otimes\\ 1\end{array};Cons} \quad \text{(by Def. } A_2\text{)}$$

$$\geq (1'\otimes\mathsf{C}_{Nil});Cons\cdot\overline{(0'\otimes 1);Cons} \quad \text{(by BA)}$$

$$= \begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\cdot\overline{\begin{array}{c}0'\\ \otimes\\ 1\end{array}};Cons \quad \text{(by Thm. 2.3.21)}$$

$$= \begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\cdot\begin{array}{c}1'\\ \otimes\\ 1\end{array};Cons \quad \text{(by Thm. 3.2.20)}$$

$$= \left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot\begin{array}{c}1'\\ \otimes\\ 1\end{array}\right);Cons \quad \text{(by Thm. 2.3.20)}$$

$$= \begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons. \quad \text{(by Thm. 3.2.8)}$$

Thus,

$$(1' \otimes \mathsf{C}_{Nil})\,;Cons\ \cdot A_2 = (1' \otimes \mathsf{C}_{Nil})\,;Cons\ . \tag{7.67}$$

For the next derivation we will use the following valid property that follows from Ass_4:

$$\begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ;Add;\breve{f} \cdot \begin{array}{c} 1' \\ \otimes \\ 1 \end{array} = \begin{array}{c} 1' \\ \otimes \\ f;\succ;\breve{f} \end{array} . \tag{7.68}$$

$$\begin{aligned}
&\begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ;Cons \cdot A_3 \\
&= \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ;Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ;C_2; \begin{array}{c} g;\succeq \\ \otimes \\ f;\succ \end{array} ;Add;\breve{f} \cdot \begin{array}{c} 1' \\ \otimes \\ 1 \end{array} ;Cons} && \text{(by Def. } A_3\text{)} \\
&= \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ;Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ;C_2; \begin{array}{c} 1' \\ \otimes \\ f;\succ;\breve{f} \end{array} ;Cons} && \text{(by (7.68))} \\
&= \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ;Cons \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ;C_2; \begin{array}{c} 1' \\ \otimes \\ f;\succ;\breve{f} \end{array}} ;Cons && \text{(by Thm. 2.3.21)} \\
&= \left(\begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} \cdot \overline{\begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ;C_2; \begin{array}{c} 1' \\ \otimes \\ f;\succ;\breve{f} \end{array}} \right) ;Cons && \text{(by Thm. 2.3.20)} \\
&= \left(\begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} \cdot \overline{Dom\left(\begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ;C_2 \right) ; \begin{array}{c} 1' \\ \otimes \\ STA;C_1 \end{array} ; \begin{array}{c} 1' \\ \otimes \\ f;\succ;\breve{f} \end{array}} \right) ;Cons \\
&&& \text{(by } Ass_5\text{)} \\
&= \left(\begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} \cdot \overline{Dom\left(\begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ;C_2 \right) ; \begin{array}{c} 1' \\ \otimes \\ STA;C_1;f;\succ \end{array} ; \begin{array}{c} 1' \\ \otimes \\ \breve{f} \end{array}} \right) ;Cons. \\
&&& \text{(by Thm. 3.2.10)}
\end{aligned}$$

Since the relation *MAXSTA* produces f-maximal lists as output by definition, then

$$STA;C_1;f;\succ\, = MAXSTA;f;\succ\ . \tag{7.69}$$

If we denote by D the filter $Dom\,((1' \otimes MAXSTA);C_2)$, then the derivation continues as follows:

$$\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\cdot A_3$$

$$=\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot D;\overline{\begin{array}{c}1'\\ \otimes\\ MAXSTA;f;\succ;\breve{f}\end{array}}\right);Cons \qquad \text{(by (7.69))}$$

$$=\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot (D+\neg D);D;\overline{\begin{array}{c}1'\\ \otimes\\ MAXSTA;f;\succ\end{array};\begin{array}{c}1'\\ \otimes\\ \breve{f}\end{array}}\right);Cons \qquad \text{(by Thm. 7.1.1)}$$

$$=\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot D;D;\overline{\begin{array}{c}1'\\ \otimes\\ MAXSTA;f;\succ\end{array};\begin{array}{c}1'\\ \otimes\\ \breve{f}\end{array}}\right);Cons$$

$$+\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot \neg D;D;\overline{\begin{array}{c}1'\\ \otimes\\ MAXSTA;f;\succ\end{array};\begin{array}{c}1'\\ \otimes\\ \breve{f}\end{array}}\right);Cons \qquad \text{(by Ax. 2 and BA)}$$

$$=\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot D;D;\overline{\begin{array}{c}1'\\ \otimes\\ MAXSTA;f\end{array};\begin{array}{c}1'\\ \otimes\\ \succ\end{array};\begin{array}{c}1'\\ \otimes\\ \breve{f}\end{array}}\right);Cons$$

$$+\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot \neg D;1\right);Cons \qquad \text{(by Thm. 3.2.10 and Lemma 7.1)}$$

$$=\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot D;\begin{array}{c}1'\\ \otimes\\ MAXSTA;f\end{array};\overline{\begin{array}{c}1'\\ \otimes\\ \succ\end{array}};\begin{array}{c}1'\\ \otimes\\ \breve{f}\end{array}\right);Cons+\neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons$$
$$\text{(by Thms. 2.3.19, 2.3.21, and 2.3.8)}$$

$$=\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array}\cdot D;\begin{array}{c}1'\\ \otimes\\ MAXSTA;f\end{array};\begin{array}{c}1'\\ \otimes\\ \preceq\end{array};\begin{array}{c}1'\\ \otimes\\ \breve{f}\end{array}\right);Cons+\neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons$$
$$\text{(by Thm. 3.2.20 and } \overline{\succ}=\preceq\text{)}$$

$$=D;\left(\begin{array}{c}1'\\ \otimes\\ 1;1'_{L^0}\end{array}\cdot\begin{array}{c}1'\\ \otimes\\ MAXSTA;f;\preceq;\breve{f}\end{array}\right);Cons+\neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons$$
$$\text{(by Def. } \mathsf{C}_{Nil}\text{, and Thm. 3.2.10)}$$

$$=D;\begin{array}{c}1'\\ \otimes\\ MAXSTA;f;\preceq;\breve{f};1'_{L^0}\end{array};Cons+\neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons \qquad \text{(by Thm. 2.3.22)}$$

$$=D;\begin{array}{c}1'\\ \otimes\\ MAXSTA;f;\preceq;\breve{\mathsf{C}}_0;1'_{L^0}\end{array};Cons+\neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons \qquad \text{(by } Ass_6\text{)}$$

$$= D;\begin{array}{c}1'\\ \otimes\\ MAXSTA;f;\preceq;1'_0;1;1'_{L^0}\end{array};Cons + \neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons \qquad \text{(by Def. } \mathsf{C}_0\text{)}$$

$$= D;\begin{array}{c}1'\\ \otimes\\ MAXSTA;Dom\,(f;\preceq;1'_0)\,;\mathsf{C}_{Nil}\end{array};Cons + \neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons.$$
$$\text{(by Thm. 2.3.14 and Def. } \mathsf{C}_{Nil}\text{)}$$

Thus,

$$(1'\otimes\mathsf{C}_{Nil})\,;Cons\cdot A_3 =$$
$$D;\begin{array}{c}1'\\ \otimes\\ MAXSTA;Dom\,(f;\preceq;1'_0)\,;\mathsf{C}_{Nil}\end{array};Cons + \neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\,. \qquad (7.70)$$

Recalling the previous definitions and derivations,

$$A = 1'_{L\succ 1};\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\cdot\overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}}\right) \qquad \text{(by (7.64))}$$

$$= 1'_{L\succ 1};(Hd\nabla Tl)\,;A_1 \qquad \text{(by Def. } A_1\text{)}$$

$$= 1'_{L\succ 1};\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\cdot\overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}\cdot\begin{array}{c}0'\\ \otimes\\ 1\end{array};Cons}\right.$$
$$\left.\cdot\,\overline{\begin{array}{c}1'\\ \otimes\\ STA;C_1\end{array};C_2;\begin{array}{c}g;\succeq\\ \otimes\\ f;\succ\end{array};Add;\breve{f}\cdot\begin{array}{c}1'\\ \otimes\\ 1\end{array};Cons}\right) \qquad \text{(by (7.66))}$$

$$= 1'_{L\succ 1};\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\cdot A_2\cdot A_3\right) \qquad \text{(by Def. } A_2 \text{ and } A_3\text{)}$$

$$= 1'_{L\succ 1};\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\left(\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\cdot A_3\right) \qquad \text{(by (7.67))}$$

$$= 1'_{L\succ 1};\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\left(D;\begin{array}{c}1'\\ \otimes\\ MAXSTA;Dom\,(f;\preceq;1'_0)\,;\mathsf{C}_{Nil}\end{array};Cons\right.$$
$$\left.+\neg D;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\right) \qquad \text{(by (7.70))}$$

$$= 1'_{L^{\succ 1}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA ; Dom\,(f ; \preceq ; 1'_0) ; \mathsf{C}_{Nil} \end{array} ; Cons \right.$$
$$\left. + \neg D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA ; \mathsf{C}_{Nil} \end{array} ; Cons \right) . \quad \text{(by 2.3.14 and Def. } \mathsf{C}_{Nil}\text{)}$$

Then,

$$A = 1'_{L^{\succ 1}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \left(D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA ; Dom\,(f ; \preceq ; 1'_0) ; \mathsf{C}_{Nil} \end{array} ; Cons \right.$$
$$\left. + \neg D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA ; \mathsf{C}_{Nil} \end{array} ; Cons \right) . \tag{7.71}$$

In order to continue with the derivation of a divide-and-conquer algorithm for the relation *MAXSTA*, proceeding along the lines of the derivation for A, we deduce:

$$B = 1'_{L^{\succ 1}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA ; Dom\,(f ; \succeq ; 1'_0) \end{array} ; Cons . \tag{7.72}$$

Let us define the following partial identities:

$$D_1 := Dom\,(f ; \preceq ; 1'_0) ,$$
$$D_2 := Dom\,(f ; \succeq ; 1'_0) .$$

Joining the results for A (Eq. (7.71)) and B (Eq. (7.72)) to the derivation of the base case (Eq. (7.63)), we obtain

$$MAXSTA = 1'_{L^1} +$$
$$1'_{L^{\succ 1}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; D ; \left(\begin{array}{c} 1' \\ \otimes \\ MAXSTA ; D_1 ; \mathsf{C}_{Nil} \end{array} + \begin{array}{c} 1' \\ \otimes \\ MAXSTA ; D_2 \end{array} \right) ; Cons$$
$$+ 1'_{L^{\succ 1}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \neg D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ; \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons ,$$

which, by Ax. 2 and Thm. 3.2.10, yields the equation

$$
\begin{aligned}
MAXSTA = {}& \\
& 1'_{L^1} + 1'_{L^{\succ 1}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ; \left(\begin{array}{c} 1' \\ \otimes \\ D_1 ; \mathsf{C}_{Nil} \end{array} + \begin{array}{c} 1' \\ \otimes \\ D_2 \end{array} \right) ; Cons \\
& + 1'_{L^{\succ 1}} ; \begin{array}{c} Hd \\ \nabla \\ Tl \end{array} ; \neg D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ; \begin{array}{c} 1' \\ \otimes \\ \mathsf{C}_{Nil} \end{array} ; Cons.
\end{aligned}
\tag{7.73}
$$

Equation (7.73) provides a recursive algorithm for computing the relation *MAXSTA*, but according to our definitions it does not follow the pattern of divide-and-conquer algorithms. Recalling the definition of the filter D, it is easy to prove that

(1) $D; (1' \otimes MAXSTA) = (1' \otimes MAXSTA) ; C_2$,
(2) $\neg D; (1' \otimes MAXSTA) = (1' \otimes MAXSTA) ; \neg C_2$.

The proofs are as follows.

$$
\begin{aligned}
D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} &= D ; \begin{array}{c} 1' \\ \otimes \\ STA ; C_1 \cdot MAXSTA \end{array} && \text{(by (7.41))} \\
&= D ; \left(\begin{array}{c} 1' \\ \otimes \\ STA ; C_1 \end{array} \cdot \begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} \right) && \text{(by Thm. 3.2.8)} \\
&= \begin{array}{c} 1' \\ \otimes \\ STA ; C_1 \end{array} \cdot D ; \begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} && \text{(by Thm. 2.3.22)} \\
&= \begin{array}{c} 1' \\ \otimes \\ STA ; C_1 \end{array} \cdot \begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} ; C_2 && \text{(by } Ass_5\text{)} \\
&= \left(\begin{array}{c} 1' \\ \otimes \\ STA ; C_1 \end{array} \cdot \begin{array}{c} 1' \\ \otimes \\ MAXSTA \end{array} \right) ; C_2 && \text{(by Thm. 2.3.22)}
\end{aligned}
$$

$$= \underset{STA;C_1 \cdot MAXSTA}{\overset{1'}{\otimes}} ;C_2 \qquad \text{(by Thm. 3.2.8)}$$

$$= \underset{MAXSTA}{\overset{1'}{\otimes}} ;C_2. \qquad \text{(by (7.41))}$$

Regarding property (2), we will use the following property from Boolean algebra. If $a \cdot b = 0$, $a+b = c$, $a \cdot d = 0$, and $a+d = c$, then $b = d$. Notice now that using Ax. 2,

$$D; \underset{MAXSTA}{\overset{1'}{\otimes}} + \neg D; \underset{MAXSTA}{\overset{1'}{\otimes}} = \underset{MAXSTA}{\overset{1'}{\otimes}} .$$

Also,

$$D; \underset{MAXSTA}{\overset{1'}{\otimes}} + \underset{MAXSTA}{\overset{1'}{\otimes}} ;\neg C_2$$

$$= \underset{MAXSTA}{\overset{1'}{\otimes}} ;C_2 + \underset{MAXSTA}{\overset{1'}{\otimes}} ;\neg C_2 \qquad \text{(by (1))}$$

$$= \underset{MAXSTA}{\overset{1'}{\otimes}} ;(C_2 + \neg C_2) \qquad \text{(by Ax. 2)}$$

$$= \underset{MAXSTA}{\overset{1'}{\otimes}} . \qquad \text{(by Thm. 7.1.1 and Ax. 5)}$$

It is also clear that

$$D; \underset{MAXSTA}{\overset{1'}{\otimes}} \cdot \neg D; \underset{MAXSTA}{\overset{1'}{\otimes}} = 0 .$$

Finally,

$$
\begin{aligned}
D;\ \begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array} \cdot \begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};\neg C_2
&= \begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};C_2 \cdot \begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};\neg C_2 && \text{(by (1))}\\
&= 0.
\end{aligned}
$$

Using then the previously mentioned property about Boolean algebras we deduce that (1) holds. Then, using (1) and (2) in (7.73),

$$
\begin{aligned}
&MAXSTA =\\
&1'_{L^1} + 1'_{L^{\succ 1}};\begin{array}{c}Hd\\ \nabla\\ Tl\end{array};\begin{array}{c}1'\\ \otimes\\ MAXSTA\end{array};\left[C_2;\left(\begin{array}{c}1'\\ \otimes\\ D_1;\mathsf{C}_{Nil}\end{array} + \begin{array}{c}1'\\ \otimes\\ D_2\end{array}\right);Cons\right.\\
&\qquad\qquad\qquad\qquad\left. + \neg C_2;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons\right]. \qquad (7.74)
\end{aligned}
$$

If we define

$$
Join_{MAXSTA} := C_2;\left(\begin{array}{c}1'\\ \otimes\\ D_1;\mathsf{C}_{Nil}\end{array} + \begin{array}{c}1'\\ \otimes\\ D_2\end{array}\right);Cons + \neg C_2;\begin{array}{c}1'\\ \otimes\\ \mathsf{C}_{Nil}\end{array};Cons,
$$

then, by (7.74), the predicate

$$
D\&C\,(MAXSTA, 1'_{L^1}, 1'_{L^{\succ 1}};(Hd \nabla Tl), 1', MAXSTA, Join_{MAXSTA})
$$

holds.

7.8 Comparison with Previous Work

The notion of generic algorithm is not new. Already in 1985 the wide spectrum language CIP-L [F. L. Bauer et al. (1985)] allowed us to work with *program schemes*. Also, program design strategies were incorporated as *transformation rules*. The advantage of our calculus is its completeness, for there is no theorem showing that given any two program schemes P_1 and P_2 with the same semantics, it is possible to carry on the derivation

$$\begin{array}{c} P_1 \\ \updownarrow \\ \hline P_2 \end{array}$$

in CIP-L.

In [B. Möller (1991); M. Russling (1996)a; M. Russling (1996)b] a framework for program construction based on an *algebra of formal languages* is presented. In [M. Russling (1996)b] these algebras are used for deriving generic algorithms for the treatment of graphs. The operators from the algebras are defined in set-theory using variables over '*words*' besides variables over languages. From these set-theoretical definitions, the author derives some valid properties only involving variables that range over formal languages, but, opposed to the fork algebra case, there is no proof of whether these properties axiomatize the algebras or not. Also, variables ranging over words are used in proofs (something that is avoided in fork algebras by using only variables over relations), and reasoning in set-theory is carried on.

In [D. Smith (1985); D. Smith (1987)] some strategies are presented which aim to help in the design of divide-and-conquer algorithms. A program scheme describing an arbitrary divide-and-conquer algorithm is given, and the strategies are used for finding the adequate program pieces. Notice that since no complete calculus is given, then the author reasons in first-order logic using variables over individuals, something avoided in fork algebras. Also, the lack of such calculus makes the author find some of the missing parts using his intuition. This can be seen for example in the derivation of the MIN algorithm in [D. Smith (1985), pp. 45–46], where the *Split* operator and the subproblems *Id* and *MIN* are fixed by hand, and the *Join* part is derived. In our case, once the generator is fixed (something we believe equivalent to choosing the *Split* operator), the subproblems *Id* and *MIN* are found by unfolding/folding in fork algebras, and the *Join* is obtained in the process of satisfying the predicate $D\&C$.

In [R. Bird et al. (1997)] an approach similar to ours is presented. Relations are not introduced as elements in models of logical theories, but rather as arrows in categories known as *allegories* [P. Freyd et al. (1990)]. While in our framework the discussion in Section 7.4 shows that first-oder formulas over abstract relations can always be interpreted as formulas on binary relations, in [R. Bird et al. (1997), p. 95] a weaker result is used that guarantees this 'completeness' only for Horn sentences. Horn sentences are formulas of the form $e_0 \wedge \cdots \wedge e_{n-1} \Rightarrow e_n$, where $e_0, \ldots, e_n$ are identities between relational designations. Horn sentences are adequate when performing equational reasoning, but fall short in describing design strategies. Something that also distinguishes both frameworks is the background required for mastering the process of program construction. While in the categorical framework a fairly non-trivial amount of category theory is required, in the fork algebraic framework only first-order logic, equational reasoning and basic set-theory for understanding binary relations are required. Of course, this categorical background also has clear advantages, such as the use of functors in the description of problems, allowing us to derive type-generic algorithms in a very efficient way.

This page is intentionally left blank

Bibliography

Aho, A. V., Hopcroft, J. E. and Ullman, J. D., *Data Structures and Algorithms* (1983), (Addison-Wesley, Reading, MA).

Andréka, H. and Németi, I., *General algebraic logic: A perspective on 'What is logic'*, in D. M. Gabbay (Ed.), *What is a Logical System?*, Vol. 4 of Studies in Logic and Computation, pp. 393–443, (Oxford, Clarendon Press, 1994).

Andréka, H., Németi, I. and Sain, I., *Applying algebraic logic to logic*, in Proceedings of the Third International Conference on Algebraic Methodology and Software Technology (AMAST'93), University of Twente, Enschede, The Netherlands, 21–25 June 1993, (Springer-Verlag), pp. 5–26.

Andréka, H. and Sain, I., *Connections between algebraic logic and initial algebra semantics of CF languages*, in B. Dömölki and T. Gergely (Eds.), *Mathematical Logic in Computer Science (Proc. Coll. Salgótarján, 1978)*, Vol. 26 of Colloq. Math. Soc. J. Bolyai, Amsterdam, 1981, pp. 25–83.

Backhouse, R. C., de Bruin P. J., Malcolm, G., Voermans, E. and van der Woude, J. C. S. P., *Relational Catamorphisms*, in Proceedings of the IFIP TC2/WG2.1 Working Conference on Constructing Programs from Specifications, (Elsevier Science Publishers B. V.), pp. 287–318, 1991.

Backhouse, R. C. and Hoogendijk, P., *Elements of a Relational Theory of Datatypes*, Formal Program Development, IFIP TC2/WG 2.1 State-of-the-Art Report, LNCS 755, (Springer-Verlag), pp. 7–42, 1993.

Backus, J., *Can Programming be Liberated from the Von Neumann Style? A Functional Style and its Algebra of Programs*, Communications of the ACM vol. 21 (1978), pp. 613–641.

Bauer, F. L., Berghammer, R., Broy, M., Dosch, W., Geiselbrechtinger, F., Gnatz, R., Hangel, E., Hesse, W., Krieg–Brückner, B., Laut, A., Matzner, T., Möller, B., Nickl, F., Partsch, H., Pepper, P., Samelson, K., Wirsing, M. and Wössner, H., *The Wide Spectrum Language CIP–L* (1985), LNCS 183, Springer-Verlag.

Baum, G. A., Frias, M. F., Haeberer, A. M. and Martínez López, P. E., *From*

Specifications to Programs: A Fork–algebraic Approach to Bridge the Gap, in Proceedings of MFCS'96, LNCS 1113, (Springer-Verlag, 1996, pp. 180–191).

Berghammer, R., *Relational Specifications of Data Types and Programs*, Technical Report 9109, Fakultät für Informatik, Universität der Bundeswehr München, 1991.

Berghammer, R., Gritzner T.F., and Schmidt, G., *Prototyping Relational Specifications Using Higher-Order Objects*, in Heering, J., Meinke, K., Möller, B. and Nipkow, T. (Eds.), Higher-Order Algebra, Logic and Term Rewriting, 1st. International Workshop, HOA'93, LNCS 816, (Springer-Verlag, 1993), pp. 56–75.

Berghammer, R. and Schmidt, G., *Relational Specifications*, in C. Rauszer (Ed.), Algebraic Logic, Banach Center Publications vol. 28, Polish Academy of Sciences, 1993, pp. 167–190.

Berghammer, R. and von Karger, B., *Algorithms from Relational Specifications*, Chapter 9 of [C. Brink et al. (1997)].

Bird, R., *Transformational Programming and the Paragraph Problem*, Science of Computer Programming vol. 6, No. 2 (1986), (Elsevier Science Publishers B. V.), pp. 159–189.

Bird, R. and de Moor O., *List Partitions*, Formal Aspects of Computing vol. 5, No. 1 (1993), (Springer-Verlag), pp. 67–78.

Bird, R. and de Moor O., *Algebra of Programming* (1997), (Prentice Hall).

Birkhoff, G., *On the Structure of Abstract Algebras*, Proceedings of the Cambridge Phylosophical Society, vol. 31 (1935), pp. 433–454.

Birkhoff, G., *Subdirect Unions in Universal Algebra*, Bulletin of the American Mathematical Society, vol. 50 (1944), pp. 764–768.

Blackburn, P., de Rijke, M. and Venema, Y., *Modal Logic*, Cambridge Tracts in Theoretical Computer Science, vol. 53, 2001.

Blok, W., and Pigozzi, D., *Algebraizable Logics*, Memoirs of the American Mathematical Society, vol. 77, 1989.

Booch, G., Jacobson, I. and Rumbaugh, J., *The Unified Modeling Language User Guide*, The Addison-Wesley Object Technology Series, 1998.

Brink, C., Kahl, W. and Schmidt, G. (Eds.), *Relational Methods in Computer Science* (1997), (Springer Wien–New York).

Burstall, R. M. and Darlington, J., *A Transformation System for Developing Recursive Programs*, Journal of the ACM vol. 24, No. 1 (1977), pp. 44–67.

Buszkowski, W. and Orlowska, E., *Indiscernibility-Based Formalisation of Dependencies in Information Systems.* In Orlowska, E. (Ed.), Incomplete Information: Rough set analysis, (Physica Verlag, 1997).

Chin, L. H. and Tarski, A., *Distributive and Modular Laws in the Arithmetic of Relation Algebras*, University of California Publications in Mathematics (1951), (University of California), pp. 341–384.

Darlington, J., *Applications of Program Transformation to Program Synthesis*, in Proceedings of the International Symposium on Proving and Improving

Programs, Arc-et-Senans, France, July 1-3, 1975, pp. 133–144.

De Morgan, A., *On the Syllogism, and Other Logical Writings* (1966), (Yale University Press).

Demri, S., Orlowska, E. and Rewitzky, I., *Towards reasoning about Hoare relations*, Annals of Mathematics and Artificial Intelligence vol. 12 (1994), pp. 265–289.

Demri, S. and Orlowska, E., *Logical Analysis of Demonic Nondeterministic Programs*, Theoretical Computer Science vol. 166 (1–2) (1996).

Doornbos, H., van Gasteren, N. and Backhouse, R. C., *Programs and Datatypes*, Chapter 10 of [C. Brink et al. (1997)].

Enderton, H. E., *A Mathematical Introduction to Logic* (1972), Academic Press, Inc.

Fitting, M., *Intuitionistic logic, model theory and forcing* (1969), North-Holland, Amsterdam.

Freyd, P. J. and Ščedrov, A., *Categories, Allegories* (1990), Mathematical Library, vol. 39, North-Holland.

Frias, M. F. and Aguayo, N. G., *Natural Specifications vs. Abstract Specifications. A Relational Approach*, in Proceedings of SOFSEM '94, Milovy, Czech Republic, pp. 17–22, November 1994.

Frias, M. F., Aguayo N. G. and Novak B., *Development of Graph Algorithms with Fork Algebras*, in Proceedings of the XIX Latinamerican Conference on Informatics, 1993, pp. 529–554.

Frias, M. F., Baum, G. A. and Haeberer, A. M., *Adding Design Strategies to Fork Algebras*, in Proceedings of Perspectives of System Informatics, LNCS 1181, (Springer-Verlag, 1996), pp. 214–226.

Frias M.F., Baum G.A. and Haeberer A.M., *Fork Algebras in Algebra, Logic and Computer Science*, Fundamenta Informaticae vol. 32 (1997), pp. 1–25.

Frias M.F., Baum G.A. and Haeberer A.M., *Representability and Program Construction within Fork Algebras*, Logic Journal of the IGPL, Vol. 6, No. 2, (Oxford University Press, 1998), pp. 229–259.

Frias, M. F., Baum, G. A., Haeberer, A. M. and Veloso, P. A. S., *Fork Algebras are Representable*, Bulletin of the Section of Logic vol. 24, No. 2 (1995), University of Lódź, pp. 64–75.

Frias, M. F., Haeberer, A. M. and Veloso, P. A. S., *A Finite Axiomatization for Fork Algebras*, Logic Journal of the IGPL vol. 5, No. 3, (Oxford University Press, 1997) pp. 311–319.

Frias M.F. and Orlowska E., *Equational Reasoning in Non Classical Logics*, Journal of Applied Non Classical Logics, Vol. 8, No. 1-2 (1998), pp. 27–66.

Frias, M. F. and Orlowska E., *A Proof System for Fork Algebras and its Applications to Reasoning in Logics Based on Intuitionism*, Logique et Analyse vol. 150–151–152 (1997), pp. 239–284.

Gries D., *The Science of Programming* (1981), Texts and Monographs in Computer Science, (Springer-Verlag).

Gries, D., *Equational logic as a tool*, LNCS 936, (Springer-Verlag, 1995), pp. 1–17.

Gries, D. and Schneider, F. B., *A Logical Approach to Discrete Mathematics*, (Springer-Verlag, 1993).

Gyuris, V., *A Short Proof for Representability of Fork Algebras*, Logic Journal of the IGPL vol. 3, No. 5 (1995), (Oxford University Press), pp. 791–796.

Haeberer, A. M., Baum, G. A. and Schmidt G., *Dealing with Non-Constructive Specifications Involving Quantifiers*, MCC 4/93, Departamento de Informática, PUC-Rio, May 1993.

Haeberer, A. M., Baum, G. A. and Schmidt G., *On the Smooth Calculation of Relational Recursive Expressions out of First-Order Non-Constructive Specifications Involving Quantifiers*, in Proceedings of the International Conference on Formal Methods in Programming and Their Applications, LNCS 735, (Springer-Verlag, 1993), pp. 281–298.

Haeberer, A. M. and Veloso, P. A. S., *Partial Relations for Program Derivation: Adequacy, Inevitability and Expressiveness*, in Constructing Programs from Specifications – Proceedings of the IFIP TC2 Working Conference on Constructing Programs from Specifications, (North-Holland, 1991), pp. 319–371.

Halmos, P., *Algebraic Logic* (1962), Chelsea.

Barwise, J., (Ed.), *Handbook of Mathematical Logic* (1977), North-Holland.

Harel, D., *Dynamic Logic*, Handbook of Philosophical Logic, Vol. II, (D. Reidel Publishing Company, 1984), pp. 497–604.

Harel, D., Kozen, D. and Tiuryn, J., *Dynamic Logic* (2000), (MIT Press).

Henkin, L., Monk, D. and Tarski, A., *Cylindric Algebras, Part I* (1971), Studies in Logic and the Foundations of Mathematics, vol. 64, (North-Holland).

Henkin, L., Monk, D. and Tarski, A., *Cylindric Algebras, Part II* (1985), Studies in Logic and the Foundations of Mathematics, vol. 115, (North-Holland).

Herment, M. and Orlowska, E., *Handling information logics in a graphical proof editor*, Computational Intelligence vol. 11, No. 2, (1995), pp. 297–322.

Jeuring, J. T., *The Derivation of On-Line Algorithms with an Application to Finding Palindromes*, Algorithmica vol. 11, No. 2 (1994), pp. 146–184.

Johansson, I., *Der minimalkalkuel, ein reduzierter intuitionistischer Formalismus*, Compositio Mathematica vol. 4 (1936), pp. 119–136.

Jónsson, B., and Tarski, A., *Boolean Algebras with Operators, PART II*, American Journal of Mathematics vol. 74 (1952), pp. 127–162.

Kripke, S., *Semantical analysis of modal logic I*, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik vol. 9, (1963), pp. 67–96.

Kripke, S., *Semantical analysis of intuitionistic logic.* In: Crossley, J. N. and Dummett, M. A. (Eds.) Formal Systems and Recursive Functions, North Holland, 1965.

Löwenheim, L., *Uber Möglichkeiten im Relativkalkul*, Mathematische Annalen vol. 76 (1915), pp. 447–470.

Lyndon, R., *The Representation of Relational Algebras*, Annals of Mathematics (Series 2) vol. 51 (1950), pp. 707–729.

Lyndon, R., *The Representation of Relational Algebras, Part II*, Annals of Mathematics (Series 2) vol. 63 (1956), pp. 294–307.

MacCaull, W., *Relational Proof System for Linear and Other Substructural Logics*, Logic Journal of the IGPL vol. 5, No. 5 (1997), pp. 673–697.

Maddux, R. D., *Some Sufficient Conditions for the Representability of Relation Algebras*, Algebra Universalis vol. 8 (1978), pp. 162–172.

Maddux, R. D., *A Sequent Calculus for Relation Algebras*, Annals of Pure and Applied Logic vol. 25 (1983), pp. 73–101.

Maddux, R. D., *Introductory Course on Relation Algebras, Finite-Dimensional Cylindric Algebras and their Interconnections*, Colloquia Mathematica Societatis János Bolyai vol. 54, Algebraic Logic, Budapest, Hungary, 1988, North-Holland, pp. 361–392.

Maddux, R. D., *Finitary Algebraic Logic*, Zeitschr. f. math. Logik und Grundlagen d. Math. vol. 35 (1989), pp. 321–332.

Maddux, R. D., *The Origin of Relation Algebras in the Development and Axiomatization of the Calculus of Relations*, Studia Logica vol. L 3/4 (1991), pp. 421–455.

McKenzie, R. N. W., *The Representation of Integral Relation Algebras*, Michigan Mathematical Journal vol. 17 (1970), pp. 279–287.

Meertens, L., *Algorithmics - Toward Programming as a Mathematical Activity*, in De Bakker, J. W., Hazewinkel, M. and Lenstra, J. K., (Eds.), Mathematics and Computer Science, CWI Monographs 1, North–Holland, pp. 3–42, 1987.

Mikulás, S., Sain, I. and Simon, A., *Complexity of the Equational Theory of Relational Algebras with Projection Elements.* Bulletin of the Section of Logic vol. 21, No. 3 (1992), University of Lódź, pp. 103–111.

Möller, B., *Relations as a Program Development Calculus*, in Constructing Programs from Specifications – Proceedings of the IFIP TC2 Working Conference on Constructing Programs from Specifications, (North-Holland, 1991), pp. 373–397.

Möller, B., *Derivation of Graph and Pointer Algorithms*, Formal Program Development, IFIP TC2/WG 2.1 State-of-the-Art Report, LNCS 755, (Springer-Verlag, 1993), pp. 123–160.

Monk, J. D., *On Representable Relation Algebras*, Michigan Mathematical Journal, vol. 11, 1964, pp. 207–210.

Németi, I., *Algebraizations of quantifier logics*, Studia Logica vol. 50 (1991), pp. 485–569.

Orlowska, E., *Relational interpretation of modal logics.* In: Andreka, H., Monk, D. and Nemeti, I. (Eds.), Algebraic Logic. Colloquia Mathematica Societatis Janos Bolyai vol. 54, (North-Holland, 1988), pp. 443–471.

Orlowska, E., *Relational proof system for relevant logics*, Journal of Symbolic Logic vol. 57 (1992), pp. 1425–1440.

Orlowska, E., *Relational semantics for nonclassical logics: Formulas are relations.* In: Wolenski, J. (Ed.), Philosophical Logic in Poland, (Kluwer, 1994), pp. 167–186.

Orlowska, E., *Relational proof systems for modal logics.* In: Wansing, H. (Ed.), Proof Theory of Modal Logics, (Kluwer, 1996), pp. 55–77.

Partsch, H. A., *Specification and Transformation of Programs. A Formal Approach to Software Development* (1990), Texts and Monographs in Computer Science, (Springer-Verlag).

Peirce, Ch. S., *Collected Papers* (1933), (Harvard University Press, Cambridge).

Rasiowa, H. and Sikorski, R., *The Mathematics of Metamathematics* (1963), (Polish Science Publishers, Warsaw).

Russling, M., *Deriving a Class of Layer-Oriented Graph Algorithms*, Science of Computer Programming vol. 26 (1996), Elsevier Science Publishers B. V., pp. 117–132.

Russling, M., *Deriving General Schemes for Classes of Graph Algorithms*, AMNS 13, Augsburg, 1996.

Sain, I. and Németi, I., *Fork Algebras in Usual as well as in Non-well-founded Set Theories*, Studia Logica Library (a special volume dedicated to the memory of H.Rasiowa), to appear; extended abstract appeared in Bulletin of the Section of Logic, University of Lódź, Vol. 24, N. 3-4, 1995, pp. 158–168, pp. 182–192.

Schmidt, G. and Ströhlein, T., *Relations and Graphs* (1993), EATCS Monographs in Theoretical Computer Science, (Springer-Verlag).

Schröder, E. F. W. K., *Vorlesungen über die Algebra der Logik (exacte Logik)* (1895) vol. 3, "Algebra und Logik der Relative", part I, Leipzig.

Smith, D. R., *Top-Down Synthesis of Divide-and-Conquer Algorithms*, Artificial Intelligence vol. 27 (1985), pp. 43–96.

Smith, D. R., *Applications of a Strategy for Designing Divide-and-Conquer Algorithms*, Science of Computer Programming vol. 8 (1987), Elsevier Science Publishers B. V., pp. 213–229.

Tarski, A., *On the Calculus of Relations*, Journal of Symbolic Logic vol. 6 (1941), pp. 73–89.

Tarski, A., *Some metalogical results concerning the calculus of relations*, Journal of Symbolic Logic vol. 18 (1953), pp. 188–189.

Tarski, A., *Contributions to the Theory of Models, III*, Koninklijkle Nederlandsle Akademie van Wetenschappen. Proceedings. Series A. Mathematical Sciences (Indagationes Mathematicae, 18) vol. 58 (1955), pp. 56–64.

Tarski, A. and Givant, S., *A Formalization of Set Theory without Variables* (1987), American Mathematical Society Colloquium Publications, vol. 41, American Mathematical Society.

Veloso, P. A. S. and Haeberer, A. M., *A Finitary Relational Algebra for Classical First-Order Logic*, Bulletin of the Section of Logic vol. 20, No. 2 (1991), University of Lódź, pp. 52–62.

Veloso, P. A. S. and Haeberer, A. M., *On Fork Algebras and Program Derivation*, MCC 32/93, Departamento de Informática, PUC-Rio, December 1993.

Veloso, P. A. S., Haeberer, A. M. and Baum G. A., *On Formal Program Construction within an Extended Calculus for Binary Relations*, MCC 19/92, Departamento de Informática, PUC-Rio, May 1992.

Veloso, P. A. S., Haeberer, A. M., and Frias, M. F., *Fork Algebras as Algebras of Logic*, in Abstracts of the Logic Colloquium '94, July, 1994, p. 127. Also in Bulletin of Symbolic Logic vol. 1, No. 2 (1995), pp. 265–266.

This page is intentionally left blank

Index

Advances in Logic – Vol. 2

Fork Algebras in Algebra, Logic and Computer Science

Fork algebras are a formalism based on the relational calculus, with interesting algebraic and metalogical properties. Their representability is especially appealing in computer science, since it allows a closer relationship between their language and models. This book gives a careful account of the results and presents some applications of Fork algebras in computer science, particularly in system specification and program construction. Many applications of Fork algebras in formal methods are foreseen, and the book covers all the essentials in order to provide the reader with a better understanding.

World Scientific
www.worldscientific.com
4899 hc

ISBN 981-02-4876-8
9 789810 248765