Tree-Manipulating Systems and Church-Rosser Theorems

BARRY K. ROSEN

Harvard University, Cambridge, Massachusetts

ABSTRACT. Subtree replacement systems form a broad class of tree-manipulating systems. Systems with the "Church-Rosser property" are appropriate for evaluation or translation processes: the end result of a complete sequence of applications of the rules does not depend on the order in which the rules were applied. Theoretical and practical advantages of such flexibility are sketched. Values or meanings for trees can be defined by simple mathematical systems and then computed by the cheapest available algorithm, however intricate that algorithm may be.

We derive sufficient conditions for the Church-Rosser property and discuss their applications to recursive definitions, to the lambda calculus, and to parallel programming. Only the first application is treated in detail. We extend McCarthy's recursive calculus by allowing a choice between call-by-value and call-by-name. We show that recursively defined functions are single-valued despite the nondeterminism of the evaluation algorithm. We also show that these functions solve their defining equations in a "canonical" manner.

KEY WORDS AND PHRASES: nondeterministic algorithm, normal form, tree, operator-operand structure, recursive definition, conditional expression, call-by-value, call-by-name, recursion theorem, lambda calculus, Church-Rosser theorem

CR CATEGORIES: 5.20, 5.21, 5.23, 5.24

1. *Introduction*

Many tree-manipulating devices have been considered in logic, linguistics, and automata theory. These devices are ordinarily defined sui generis, rather than as special cases of a broad category that includes them all. This paper studies "subtree replacement systems," a broad class of tree-manipulating systems. The use of this framework is illustrated by general theorems analogous to the Church-Rosser theorem [4, Ch. 4] and by applications of these theorems. The theory is developed in Sections 2–6 and applied in Sections 7–9. More detailed versions of the proofs are presented in [16], where directions for further research are also discussed.

Some common mathematical terminology is reviewed in Section 2. In Section 3 we define "general replacement systems" and the "Church-Rosser property" for such systems. In a Church-Rosser system, whenever one applies transformation

A preliminary version of this paper was presented at the Second Annual ACM Symposium on Theory of Computing, May, 1970.

The research was supported in part by the National Science Foundation under Grant No. GS-1934 and in part by the National Aeronautics and Space Administration under Grant No. NGR 22-007-176.

Author's present address: IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598.

rules to an object R (and then to the resulting object, and so on) until no further rules are applicable, the final result does *not* depend on which of several applicable rules was chosen at each stage. This "single-valuedness" permits the rules to define an evaluation or translation process unambiguously. In a Church-Rosser system we can seek a strategy for applying the rules in a sequence that saves time or space. The results of Section 3 are not restricted to systems that manipulate trees. They fit any situation where changes in the data are made in discrete steps.

The basic notations and facts from the nonnumerical arithmetic of trees are presented in Section 4. We define subtree replacement systems and establish sufficient conditions for the Church-Rosser property in Sections 5 and 6. A wide variety of general replacement systems with trees as data are subtree replacement systems. Because the theorems deal with abstract properties of sets of rules rather than specific assumptions about the forms of the rules, they apply to many of these systems. In addition to the systems treated here, the theory can be applied to systems of combinatory logic (where a special case is implicit in Sanchis [17, Sec. 2]) and to several kinds of tree transducer that may be used to model syntax-directed compilers. The transducer applications are discussed in [16, Ch. 5].

The first application is in the study of recursive definitions. McCarthy [11] explained such definitions in terms of a nondeterministic algorithm for calculating values of recursively defined functions. According to this *algorithmic* explanation, the function defined by

$$f(x) := \textbf{if } x = 0 \textbf{ then } 1 \textbf{ else } x \times f(x - 1) \tag{1}$$

is the set of all ordered pairs $\langle \xi, \eta \rangle$ of numbers such that $f(\xi)$ can be evaluated to η by some sequence of applications of the formal rules. In Section 7 we extend the McCarthy calculus by allowing an option like the choice between call-by-value and call-by-name in ALGOL 60 [15, Sec. 4.7.3]. It is shown that recursive definitions specify single-valued partial functions despite the nondeterminism of the evaluation algorithm.

Recursive definitions may also be explained *semantically* as implicit definitions. Just as $Y^2 + Y - 6 = 0$ implicity defines the set of numbers $\{2, -3\}$, the recursive definition (1) implicitly defines a set of partial functions. Is the algorithmically defined function one of them? Which one? Morris [14, Ch. 3, Th. 2] conjectured that the algorithm specifies a solution which is extended by every other solution. (Strictly speaking, we must introduce a new datum ∞ and work with total functions, but this approximate wording is accurate enough for now.) In Section 8 we verify this conjecture. When restricted to definitions that call their parameters by value, our result is closely related to Kleene's "first recursion theorem" [8, Sec. 66, Th. 66].

In Section 9 we sketch two more applications: a proof of the classical Church-Rosser theorem and a strategy for verifying the "single-valuedness" of functions defined by algorithms or computer systems that permit asynchronous parallel processing.

2. *Terminology*

Standard mathematical notations are used as much as possible. We sometimes abbreviate "for all x in $\mathbf{N}$" by $(\forall x \in \mathbf{N})$, "there is a y less than z such that" by

$(\exists y < z)$, and so on. The abbreviation $\exists!$ is used for "there is exactly one" or "there is a unique"; this symbol is less widely known than $\forall$ and $\exists$.

For any sets X and Y, 2^X is the set of all subsets of X, and $X \times Y$ is the set of all *ordered pairs* $\langle x, y\rangle$ such that $x \in X$ and $y \in Y$. A *map* or *function* $F : X \to Y$ is any subset of $X \times Y$ such that, for each $x \in X$, there is at most one $y \in Y$ with $\langle x, y\rangle \in F$. The *domain* dom F is $\{x \in X \mid (\exists y \in Y)(\langle x, y\rangle \in F)\}$. If dom $F = X$, then F is *total*; otherwise, F is *partial*. For each $x \in$ dom F, the unique $y \in Y$ such that $\langle x, y\rangle \in F$ is denoted $F(x)$ or Fx, depending on which is more readable in each context. For any $y \in Y$ we also write

$$F^{-1}(y) = \{x \in \operatorname{dom} F \mid Fx = y\}$$

and, for any subset B of Y,

$$F^{-1}(B) = \{x \in \operatorname{dom} F \mid Fx \in B\}.$$

A total function $F : X \to Y$ is *injective* iff

$$(\forall x, x' \in X)(Fx = Fx' \text{ implies } x = x')$$

and is *surjective* iff

$$(\forall y \in Y)(\exists x \in X)(Fx = y).$$

A map that is both injective and surjective is *bijective*.

Given functions $F : X \to Y$ and $G : Y \to Z$, we set

$$G \circ F = \{\langle x, z\rangle \mid (\exists y \in Y)(\langle x, y\rangle \in F \;\&\; \langle y, z\rangle \in G)\},$$

so that $G \circ F : X \to Z$.

Any set $\rightarrow$ of ordered pairs is a *relation*. We write $x \rightarrow y$ rather than $\langle x, y\rangle \in \rightarrow$, but a symbol like $\rightarrow$ is still being used as the name of a set. Equations like $(\rightarrow) = (\rightarrow_1 \cup \rightarrow_2)$ are well formed and mean exactly what they say about sets of ordered pairs. Round brackets are used liberally to make such equations readable. When $(\rightarrow) \subseteq \mathbf{B} \times \mathbf{B}$ we say that $\rightarrow$ is a relation on $\mathbf{B}$.

The *composite* of relations $\rightarrow_1$ and $\rightarrow_2$ is defined by

$$(\rightarrow_1 \rightarrow_2) = \{\langle x, z\rangle \mid (\exists y)(x \rightarrow_1 y \;\&\; y \rightarrow_2 z)\}.$$

If F and G are functions, then they are also relations and we have $(FG) = G \circ F$.

The set $\{0,1,2,\cdots\}$ of *nonnegative* integers is denoted by $\mathbf{N}$. The letters i, j, k, J, K are often used as variables ranging over $\mathbf{N}$.

3. *General Replacement Systems*

Let $\mathbf{B}$ be a set of "objects" and let there be some means of "replacing" one object by another. We write $R \rightarrow S$ and pronounce this as "R can become S." For example, $\mathbf{B}$ might be the set of all possible arithmetic expressions, and $R \rightarrow S$ might mean that the expression R can be transformed to the expression S by performing one of the innermost indicated operations.

If R_0 can become R_1, if R_1 can become R_2, $\cdots$, and if R_{K-1} can become R_K, let us say that R_0 "can evolve" to R_K. When R_0 can evolve to R_K and R_K cannot become anything new, then R_K is to be considered the "value" or "meaning" of R_0. To avoid unwanted connotations, we use the term "normal form" instead.

Definition 3.1. Let $\mathbf{B}$ be any set, and let $\rightarrow$ be any binary relation on $\mathbf{B}$. Then $\mathfrak{B} = (\mathbf{B}, \rightarrow)$ is a *general replacement system* (GRS). A member T of $\mathbf{B}$ is *irreplaceable* iff

$$(\forall S \in \mathbf{B})(T \rightarrow S \text{ implies } T = S). \qquad (1)$$

Let $R,T \in \mathbf{B}$. Then T is a *normal form* for R iff

$$R \rightarrow^* T \ \&\ T \text{ is irreplaceable,} \qquad (2)$$

where $\rightarrow^*$ is the reflexive transitive closure of $\rightarrow$.

If a GRS is to be implemented on a computer that permits asynchronous parallel processing, then uniqueness of normal forms is obviously important. Even if a strictly deterministic sequencing mechanism is to decide which transformations are to occur, there is still reason to seek uniqueness. Suppose one has a sequencing mechanism $\mathfrak{M}$ and then is given a very different and complex mechanism $\mathfrak{M}'$ that purports to find the same normal forms more cheaply. Consider three ways that $\mathfrak{M}'$ might err in seeking the normal form defined by $\mathfrak{M}$ for an object R:

(a) $\mathfrak{M}'$ finds a normal form S' for R although $\mathfrak{M}$ finds no normal form.

(b) $\mathfrak{M}'$ finds a normal form S' for R although $\mathfrak{M}$ finds a normal form S with $S \neq S'$.

(c) $\mathfrak{M}'$ finds no normal form for R although $\mathfrak{M}$ finds a normal form S.

Error (a) is more properly considered an improvement: $\mathfrak{M}'$ gives output where the original mechanism would enter an endless computation. Error (b) cannot occur if normal forms are unique in the underlying GRS. Uniqueness of normal forms provides no assurance that error (c) does not occur, but the insights gained while proving uniqueness should assist in the analysis of this problem.

Definition 3.2. A GRS $\mathfrak{B} = (\mathbf{B}, \rightarrow)$ is *Church-Rosser* iff

$$(\forall R,S,S' \in \mathbf{B})[(R \rightarrow^* S \ \&\ R \rightarrow^* S') \text{ implies } (\exists T \in \mathbf{B})(S \rightarrow^* T \ \&\ S' \rightarrow^* T)].$$

Normal forms are unique in Church-Rosser systems. The name comes from the work of Church and Rosser [3] on this property for a particular GRS. There are several easy but useful theorems telling how to infer the Church-Rosser property for a complex system from various properties of its parts.

Definition 3.3. Let $\mathfrak{B}_1 = (\mathbf{B}, \rightarrow_1)$ and $\mathfrak{B}_2 = (\mathbf{B}, \rightarrow_2)$ be GRS's. Then $\mathfrak{B}_1$ *commutes* with $\mathfrak{B}_2$ iff

$$(\forall R,S_1,S_2 \in \mathbf{B})[(R \rightarrow_1^* S_1 \ \&\ R \rightarrow_2^* S_2) \text{ implies } (\exists T \in \mathbf{B})(S_1 \rightarrow_2^* T \ \&\ S_2 \rightarrow_1^* T)].$$

It is convenient to diagram the Church-Rosser property and commutativity, as in Figure 1. In general, a diagram asserts that, whenever given objects stand in the relations indicated by the filled circles and the arrows joining them, then further objects indicated by the open circles exist, so as to make the relations indicated by the other arrows true also.

LEMMA 3.4. *Let $\mathfrak{B} = (\mathbf{B}, \rightarrow)$ be a GRS. If there exists a binary relation $\rightarrow_\perp$ on $\mathbf{B}$ such that*

$$(\rightarrow_\perp^*) = (\rightarrow^*), \textit{ and} \qquad (1)$$

$$(\forall R,S,S' \in \mathbf{B})[(R \rightarrow_\perp S \ \&\ R \rightarrow_\perp S') \textit{ implies } (\exists T \in \mathbf{B})(S \rightarrow_\perp T \ \&\ S' \rightarrow_\perp T)], \qquad (2)$$

then $\mathfrak{B}$ is Church-Rosser.

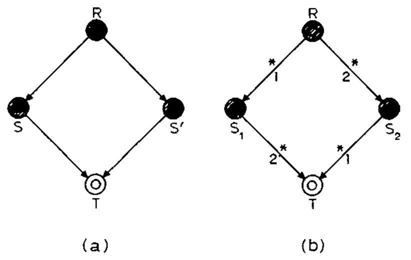

FIG. 1. Diagrams for the Church-Rosser property (a) and commutativity (b)

THEOREM 3.5. Commutative Union Theorem. *Let* $\{\mathfrak{B}_a \mid a \in A\}$ *be a family of Church-Rosser GRS's with* $\mathfrak{B}_a = (\mathbf{B}, \to_a)$ *for each* $a \in A$. *Let* $\mathfrak{B}_a$ *commute with* $\mathfrak{B}_b$ *for all* $a,b \in A$ *with* $a \neq b$. *Then the* union $\mathfrak{B} = (\mathbf{B}, \to)$ *is Church-Rosser, where*

$$(\mathbf{B}, \to) = (\mathbf{B}, \bigcup_{a \in A} \to_a).$$

PROOF. Let $\to_\perp$ be $\bigcup_{a\in A} \to_a{}^*$ and apply Lemma 3.4. ∎

The usefulness of this theorem is enhanced by the following sufficient condition for commutativity, which will be proven by diagrammatic reasoning.

LEMMA 3.6. Commutativity Lemma. *Let* $\mathfrak{B}_1 = (\mathbf{B}, \to_1)$ *and* $\mathfrak{B}_2 = (\mathbf{B}, \to_2)$ *be GRS's. Let* $\to_1{}^=$ *be the reflexive closure of* $\to_1$. *If*

$$(\forall R,S_1,S_2 \in \mathbf{B})\,[(R \to_1 S_1 \;\&\; R \to_2 S_2) \textit{ implies } (\exists T \in \mathbf{B})(S_1 \to_2{}^* T \;\&\; S_2 \to_1{}^= T)], \quad (1)$$

then $\mathfrak{B}_1$ *commutes with* $\mathfrak{B}_2$.

PROOF. We assume (1) and prove commutativity by two inductions. For each $K \in \mathbf{N}$, we prove that

$$(\forall R,S_1,S_2 \in \mathbf{B})[(R \to_1 S_1 \;\&\; R \to_2{}^K S_2) \text{ implies } (\exists T \in \mathbf{B})(S_1 \to_2{}^* T \;\&\; S_2 \to_1{}^= T)]. \quad (2)$$

When $K = 0$ we let T be S_1. To pass from K to $K + 1$, suppose (2) holds for K and that $R, S_1, S_2 \in \mathbf{B}$ have $R \to_1 S_1$ and $R \to_2{}^{K+1} S_2$. Let $P \in \mathbf{B}$ with $R \to_2{}^K P \to_2 S_2$. This situation is diagrammed by the filled circles and arrows between them at the top of Figure 2. We call this the *working diagram*. The procedure for adding to this diagram (with the help of whatever we are assuming or have already proven) will now be explained.

The induction hypothesis is expressed as boxed a in Figure 2. (For obvious reasons, we will write (Da) rather than boxed a.) Suppose (Da) is cut on a mimeograph *stencil*, this stencil is laid over the working diagram so that each filled circle of (Da) is over a circle in the working diagram, and each arrow joining the filled circles in (Da) is over a similarly labeled and directed arrow in the working diagram. (The circles R, S_1, P in the working diagram are used.) Now *print through* the stencil, forming the circle Q and the arrows from S_1 to Q and from P to Q. This step corresponds to a valid inference in first-order logic.

Next we note that (1) implies the corresponding statement with $R \to_1{}^= S_1$ in place of $R \to_1 S_1$. We express this trivial extension of (1) as (Db) in Figure 2. Consider (Db) as a stencil and lay it over the working diagram so that each filled circle in (Db) is over a circle in the working diagram, and each arrow joining filled circles

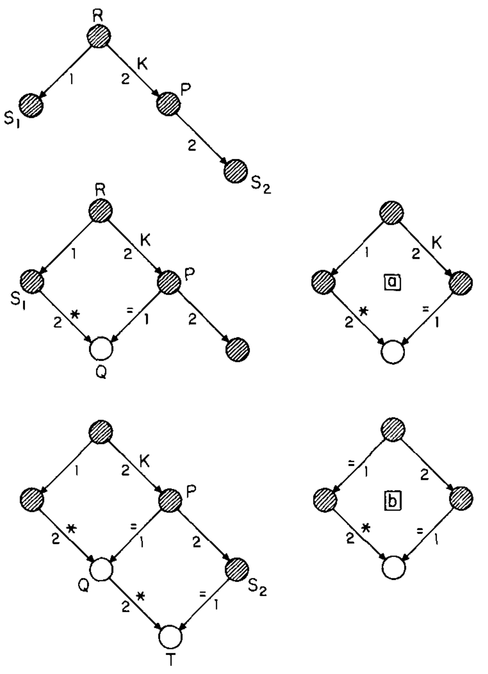

FIG. 2 Adding to a diagram with the help of two stencils

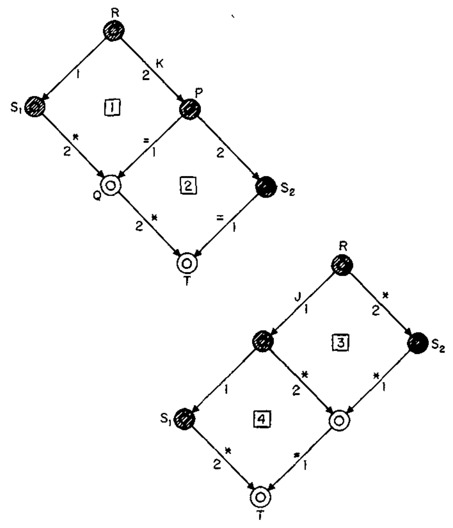

FIG. 3. Induction steps in the proof of the Commutativity Lemma (Lemma 3.6)

in (Db) is over a similarly labeled and directed arrow in the working diagram. (The circles P, Q, S_2 in the working diagram are used.) Now print through the stencil, forming the circle T and the arrows from Q to T and from S_2 to T. This also corresponds to a valid inference in first-order logic.

In adding open circles and arrows to the working diagram, we have built up a diagram that says

$$(\forall R,S_1,S_2,P \in \mathbf{B})[(R \to_1 S_1 \,\&\, R \to_2^K P \to_2 S_2) \text{ implies } (\exists Q, T \in \mathbf{B})(S_1 \to_2^* Q \to_2^* T \,\&\, P \to_1^= Q \,\&\, S_2 \to_1^= T)], \quad (3)$$

and the sequence of steps could be mechanically transformed into a deduction of (3) from the premises (Da) and (Db) alone. The diagrammatic manipulations in Figure 2 determine a proof of (3).

A more concise way to expound the diagrammatic proof of (3) is shown at the top of Figure 3. Here we display only the final state of the working diagram, together with numbers to indicate the order in which portions of the diagram appeared during the process of choosing stencils, laying them on the working diagram, and printing through. To *verify* a diagram like this is to check that it could indeed be built up from the filled circles and arrows between them alone, using only available stencils.

Having deduced (3) from the induction hypothesis (2), we note that (3) implies a statement similar to (2) but with $K + 1$ in place of K. We have passed from K to $K + 1$, and the inductive proof that every $K \in \mathbf{N}$ satisfies (2) is complete.

We now prove commutativity by proving that each $J \in \mathbf{N}$ satisfies

$$(\forall R,S_1,S_2 \in \mathbf{B})[(R \to_1^J S_1 \,\&\, R \to_2^* S_2) \text{ implies } (\exists T \in \mathbf{B})(S_1 \to_2^* T \,\&\, S_2 \to_1^* T)]. \quad (4)$$

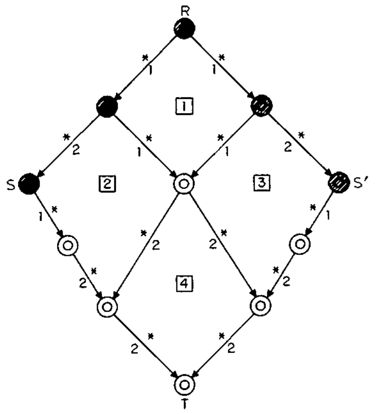

FIG 4. Diagram for the proof of Theorem 3.8

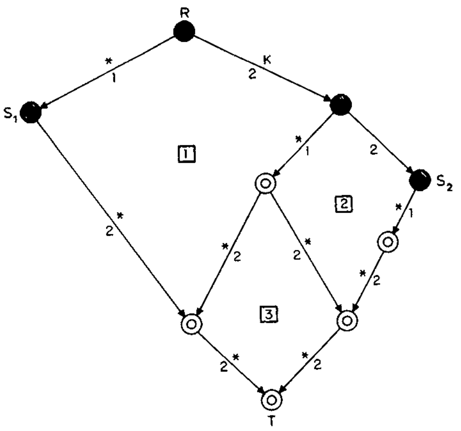

FIG. 5. Induction step in the proof of Lemma 3.9

When $J = 0$ we may let T be S_2. To pass from J to $J + 1$, we verify the diagram at the bottom of Figure 3. First (D3) is filled in by the induction hypothesis and then (D4) is filled in by the fact that (2) holds for all $K \in \mathbf{N}$. ▮

Theorem 3.5 and Lemma 3.6 were discovered by Hindley [5, Ch. 1, Th. 1.2, Lem. 1.3] and independently by the author.

There is a condition weaker than commutativity which implies that the union of two Church-Rosser GRS's is Church-Rosser.

Definition 3.7. Let $\mathfrak{B}_1 = (\mathbf{B}, \rightarrow_1)$ and $\mathfrak{B}_2 = (\mathbf{B}, \rightarrow_2)$ be GRS's. Then $\mathfrak{B}_1$ *requests* $\mathfrak{B}_2$ iff

$$(\forall R,S_1,S_2 \in \mathbf{B})[(R \rightarrow_1^* S_1 \;\&\; R \rightarrow_2^* S_2) \text{ implies } (\exists T \in \mathbf{B})(S_1 \rightarrow_2^* T \;\&\; S_2(\rightarrow_1^* \rightarrow_2^*)T)].$$

THEOREM 3.8. *Let* $\mathfrak{B}_1 = (\mathbf{B}, \rightarrow_1)$ *and* $\mathfrak{B}_2 = (\mathbf{B}, \rightarrow_2)$ *be Church-Rosser GRS's and let* $\mathfrak{B}_1$ *request* $\mathfrak{B}_2$. *Then the union* $\mathfrak{B} = (\mathbf{B}, \rightarrow)$, *where* $(\mathbf{B}, \rightarrow) = (\mathbf{B}, \rightarrow_1 \cup \rightarrow_2)$, *is Church-Rosser.*

PROOF. We will use Lemma 3.4. Let $\rightarrow_\perp$ be the composite $(\rightarrow_1^* \rightarrow_2^*)$, so that $(\rightarrow_\perp^*) = (\rightarrow^*)$ and verification of Figure 4 proves (2) of Lemma 3.4. ▮

Unfortunately, there seems to be no analogue of Lemma 3.6 for "requests." Only the star in $R \rightarrow_2^* S_2$ can be removed.

LEMMA 3.9. *Let* $\mathfrak{B}_1 = (\mathbf{B}, \rightarrow_1)$ *and* $\mathfrak{B}_2 = (\mathbf{B}, \rightarrow_2)$ *be GRS's, and* $\mathfrak{B}_2$ *be Church-Rosser. If*

$$(\forall R,S_1,S_2 \in \mathbf{B})[(R \rightarrow_1^* S_1 \;\&\; R \rightarrow_2 S_2) \textit{ implies } (\exists T \in \mathbf{B})(S_1 \rightarrow_2^* T \;\&\; S_2(\rightarrow_1^* \rightarrow_2^*)T)], \quad (1)$$

then $\mathfrak{B}_1$ *requests* $\mathfrak{B}_2$.

PROOF. We assume (1) and prove that $\mathfrak{B}_1$ requests $\mathfrak{B}_2$ by proving that each $K \in \mathbf{N}$ satisfies

$$(\forall R,S_1,S_2 \in \mathbf{B})[(R \rightarrow_1^* S_1 \;\&\; R \rightarrow_2^K S_2) \text{ implies } (\exists T \in \mathbf{B})(S_1 \rightarrow_2^* T \;\&\; S_2(\rightarrow_1^* \rightarrow_2^*)T)]. \quad (2)$$

We use induction on K. When $K = 0$, let T be S_1. We assume (2) holds for K and pass to $K + 1$ by verifying Figure 5. ▮

4. *Trees and Substitution*

This section describes some elementary concepts in the mathematics of trees. It corresponds to arithmetic and basic algebra in numerical mathematics. The system of arithmetic and the basic algebra for trees sketched here are mostly taken from the work of Brainerd [2, Sec. 2]. We have added convenient notations for string manipulation and many mnemonics. We have also extended some operations on nodes in trees to sets of "independent" nodes.

For any set A, the set of all strings of members of A is A^*. The members of A need not be "symbols" because strings are defined to be finite sequences. In particular, $\mathbf{N}^*$ is the set of all strings of nonnegative integers. We name strings in A^* by listing (within round brackets and separated by commas) the names for the members of A that occur in them. The null string is $(\ )$, the string consisting of just thirty-seven is (37), and the string consisting of three followed by seven is $(3,7)$. This is already the systematic way to write argument strings for functions of several variables, and it avoids all ambiguity in naming strings of strings.

The *length* of a string w is denoted $|w|$. The set of all strings of length J in A^* is A^J. In particular, $A^0 = \{(\ )\}$. For any string w and any $K \in \mathbf{N}$, the string $K{:}w$ consists of the first K entries in w, in the same order as in w. For $|w| \leq K$ this is w itself. The positions in w are numbered $0, 1, \cdots, |w| - 1$ and we write

$$w = (w_0, \cdots, w_{|w|-1}) = (w_0, \cdots, w_{-1}).$$

When the value of J is obvious or irrelevant, we write -1 rather than $J - 1$, as here. The word "last" is a good pronunciation for -1 here.

Concatenation is indicated by a dot:

$$(x_0, \cdots, x_{-1}) \cdot (y_0, \cdots, y_{-1}) = (x_0, \cdots, x_{-1}, y_0, \cdots, y_{-1}).$$

Now we define the *father* and *left brother* functions and the *ancestor* and *independence* relations on $\mathbf{N}^*$.

Definition 4.1. Let $m,n \in \mathbf{N}^*$. Define the following:

$$\mathrm{Fa}(n) = (|n| - 1) : n \qquad (\text{for } n \neq (\ )) \qquad (\text{father}), \tag{1}$$

$$\mathrm{LBr}(n) = \mathrm{Fa}(n) \cdot (n_{-1} - 1) \qquad (\text{for } n \neq (\ );\, n_{-1} \neq 0) \qquad (\text{left brother}), \tag{2}$$

$$m \text{ anc } n \text{ iff } |m| : n = m \qquad (\text{ancestor}), \tag{3}$$

$$m \perp n \text{ iff NOT}(m \text{ anc } n \text{ or } n \text{ anc } m) \qquad (\text{independence}). \tag{4}$$

Definition 4.2. A *tree domain* is any finite nonempty subset D of $\mathbf{N}^*$ closed under the father and left brother functions:

$$\mathrm{Fa}[D] \subseteq D \ \&\ \mathrm{LBr}[D] \subseteq D.$$

Trees are structures in which members of tree domains are labeled with "symbols" chosen from a set V.

Definition 4.3. Let V be any set and let

$$V_* = \{R \mid R : \mathbf{N}^* \to V \text{ (partial) } \&\ \mathrm{dom}\, R \text{ is a tree domain}\}.$$

Definition 4.4. Let $R,S \in V_*$; $n \in \mathrm{dom}\, R$.

$$R/n = \{\langle p, x\rangle \mid \langle n \cdot p, x\rangle \in R\} \qquad (\textit{subtree} \text{ of } R \text{ at } n), \tag{1}$$

$$R(n \leftarrow S) = \{\langle m, x\rangle \mid \langle m, x\rangle \in R \mathbin{\&} \text{NOT } (n \text{ anc } m)\} \cup \{\langle n\cdot p, y\rangle \mid \langle p, y\rangle \in S\} \quad (\textit{replace } n \text{ by } S \text{ in } R). \tag{2}$$

Definition 4.5. Let $M,N \subseteq \mathbf{N}^*$.

$$M \perp N \text{ iff } (\forall m \in M)(\forall n \in N)(m \perp n). \tag{1}$$

The proofs of the following lemmas are trivial calculations.

LEMMA 4.6. *Let* $m,n \in \mathbf{N}^*$; $M,N \subseteq \mathbf{N}^*$.

$$m \textit{ anc } n \textit{ iff } (\exists p \in \mathbf{N}^*)(m\cdot p = n), \tag{1}$$

$$M\cdot\mathbf{N}^* = \{n \in \mathbf{N}^* \mid (\exists m \in M)(m \textit{ anc } n)\}, \tag{2}$$

$$(\textit{anc anc}) \subseteq (\textit{anc}) \qquad (\textit{transitivity}), \tag{3}$$

$$M \perp N \textit{ iff } M\cdot\mathbf{N}^* \cap N\cdot\mathbf{N}^* = \varnothing. \tag{4}$$

LEMMA 4.7. *Let* $R,S,T \in V_*$; $m,n \in \textit{dom } R$; $p \in \textit{dom } S$; $m \perp n$.

$$R(n \leftarrow S)/(n\cdot p) = S/p \qquad (\textit{embedding}), \tag{1}$$

$$R(n \leftarrow S(p \leftarrow T)) = R(n \leftarrow S)(n\cdot p \leftarrow T) \qquad (\textit{associativity}), \tag{2}$$

$$R(n \leftarrow S)/m = R/m \qquad (\textit{persistence}), \tag{3}$$

$$R(n \leftarrow S)(m \leftarrow T) = R(m \leftarrow T)(n \leftarrow S) \qquad (\textit{commutativity}). \tag{4}$$

For $M \subseteq \textit{dom } R$ *an independent set, we may define*

$$R(M \leftarrow S) = R(m_0 \leftarrow S) \cdots (m_{-1} \leftarrow S)$$

(*just* R *if* $M = \varnothing$), *where* $(m_0, \cdots, m_{-1})$ *is any list of the members of* M; *the order is irrelevant.* (5)

Any set of leaves *in dom* R *is independent,*
where n *is a leaf iff dom* $R \cap Fa^{-1}(n) = \varnothing$. (6)

Definition 4.8. Let $m,n,p \in \mathbf{N}^*$. The *quotient* of n by m is defined by $n/m = p$ iff $m\cdot p = n$.

For example, $(3,4,5,6)/(3,4) = (5,6)$ because $(3,4)\cdot(5,6) = (3,4,5,6)$. This notation is taken from [2, Sec. 2] without change.

LEMMA 4.9. *Let* $R,S \in V_*$; $m,n \in \textit{dom } R$; m *anc* n.

$$R/n = (R/m)/(n/m) \qquad (\textit{cancellation}), \tag{1}$$

$$R(n \leftarrow S)/m = (R/m)(n/m \leftarrow S) \qquad (\textit{distributivity}). \tag{2}$$

Lemmas 4.6, 4.7, and 4.9 are used constantly but rarely cited, just as commutativity and distributivity for numbers are used constantly but rarely cited in real analysis.

The natural algebraic structure of V_* leads to a convenient nomenclature. Each $a \in V$ defines an operation $\bar{a} : (V_*)^* \rightarrow V_*$. The tree $\bar{a}(R,S)$ is shown in Figure 6 for a specific choice of $(R,S) \in (V_*)^2$. Here we indicate the value of a function F at an argument w by Fw. The values of trees will be indicated similarly: Rm, not $R(m)$.

Definition 4.10. Let $a \in V$; $K \in \mathbf{N}$; $(R_0, \cdots, R_{-1}) \in (V_*)^K$. Then

$$\bar{a}(R_0, \cdots, R_{-1}) = \{\langle (\,), a\rangle\} \cup \bigcup_{k<K} \{\langle (k)\cdot m, y\rangle \mid \langle m, y\rangle \in R_k\}.$$

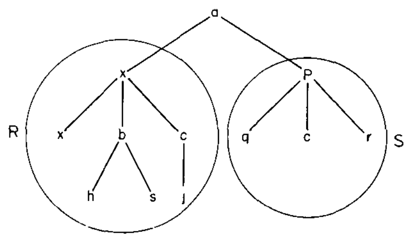

FIG. 6. The operation $\bar{a}$ is applied to (R,S)

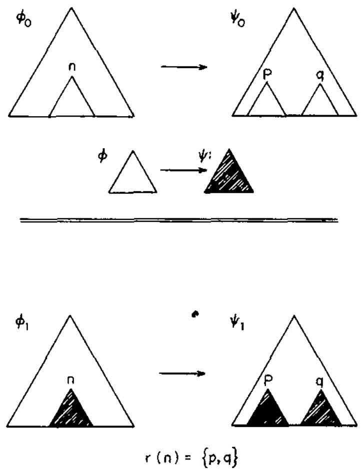

FIG. 7. The rule $\phi \to \psi$ is applied at n in ϕ_0 and at each of the residues of n in ψ_0 to form another rule $\phi_1 \to \psi_1$

Applying this definition to R in Figure 6, we have

$$R = \bar{x}(\bar{x}(\ ), \bar{b}(\bar{h}(\ ), \bar{s}(\ )), \bar{c}(\bar{\jmath}(\ ))).$$

We will henceforth omit the names of null argument strings when using this definition. When confusion between $a \in V$ and $\bar{a} : (V_*)^* \to V_*$ is unlikely, we will even omit the overbars, so that R has two abbreviated names:

$$R = \bar{x}(\bar{x},\bar{b}(\bar{h},\bar{s}),\ \bar{c}(\bar{\jmath})) = x(x,b(h,s),c(j)).$$

This amounts to the "pseudoterm" notation [20, Sec. 1].

In many applications only the *ranked* trees are of interest. There is a *rank function* $\rho : V \to \mathbf{N}$ and the symbols are thought of as "operators." Each operator x takes $\rho(x)$ operands and so a node labeled x in a tree should have exactly $\rho(x)$ sons. The set of all ranked trees with labels in V is denoted $V_\#$ and defined as follows:

Definition 4.11. Let $\rho : V \to \mathbf{N}$ and set

$$V_\# = V_\#(\rho) = \{R \in V_* \mid (\forall n \in \operatorname{dom} R)(\mid \operatorname{dom} R \cap \mathrm{Fa}^{-1}(n) \mid = \rho(Rn))\}.$$

5. *Subtree Replacement Systems*

If ϕ and ψ are trees, then the *rule* $\phi \to \psi$ may be applied to a tree R by matching a subtree of R against ϕ and then replacing that subtree by ψ.

Definition 5.1. A *subtree replacement system* (*SRS*) is any 4-tuple $\mathfrak{S} = (V, \mathbf{F}, \twoheadrightarrow, \mathbf{R})$, where

$$\mathbf{F} \subseteq V_* \qquad (\mathbf{F} \text{ is a } \textit{forest}), \tag{1}$$

$$\mathbf{R} \subseteq V_* \times V_* \qquad (\text{write } \phi \to \psi \text{ for } \langle\phi, \psi\rangle \in \mathbf{R}), \tag{2}$$

$$(\forall R \in \mathbf{F})(\forall S \in V_*)(R \twoheadrightarrow S \text{ iff } (\exists \phi \to \psi \in \mathbf{R})(\exists n \in \operatorname{dom} R)[R/n = \phi \;\&\; S = R(n \leftarrow \psi)]). \tag{3}$$

$$(\twoheadrightarrow) \subseteq \mathbf{F} \times \mathbf{F}. \tag{4}$$

This definition generalizes the notion of $\rightarrow$ in Brainerd's "regular systems" [2, Def. 3.2]. The crucial difference is that we do not require that $\mathbf{R}$ be finite. The use of infinite sets of rules will permit us to describe many complex tree-manipulating systems within the SRS framework. The complexity of these systems will be broken down into two stages: specification of an infinite set of rules by some finite means, and then the use of rules in this set to evaluate trees. The same letter will often be used for an SRS $(V, \mathbf{F}, \rightarrow, \mathbf{R})$ and for the corresponding GRS $(\mathbf{F}, \rightarrow)$.

Definition 5.2. An SRS $\mathfrak{S} = (V, \mathbf{F}, \rightarrow, \mathbf{R})$ is *unequivocal* iff $\mathbf{R}$ is a partial function: No two rules have the same left side.

The more complicated concept of "closed" SRS's is illustrated in Figure 7. The little rule $\phi \rightarrow \psi$ is applicable at a node n inside the left half of the big rule $\phi_0 \rightarrow \psi_0$. More concisely: $\phi_0/n = \phi$, where $n \in \operatorname{dom} \phi_0$ and $n \neq (\,)$. We suppose that n has "residues" p and q where this same subtree appears after ϕ_0 has been replaced by $\psi_0 : \psi_0/p = \psi_0/q = \phi_0/n$. Replacing ϕ by ψ at n in dom ϕ_0 and at each residue of n in dom ψ_0 leads to a pair of trees $\phi_1 \rightarrow \psi_1$; we could say that the little rule $\phi \rightarrow \psi$ has been applied to the big rule $\phi_0 \rightarrow \psi_0$. In a closed system, $\phi_1 \rightarrow \psi_1$ is also in the set of rules.

The formal definition is conveniently divided into two parts. A "residue map" for $\phi_0 \rightarrow \psi_0$ assigns to each node $n \in \operatorname{dom} \phi_0$ a set $r(n)$ of "residues" in dom ψ_0. Each node n may have any number (including zero) of residues, but each residue p of n must have $\psi_0/p = \phi_0/n$. We also require that independent nodes have independent residues: if $p \perp q$ in dom ϕ_0, then $r(p) \perp r(q)$, where the independence relation between sets of nodes is as defined in (1) of Def. 4.5. (This additional requirement is motivated more by technical considerations in the proof of the theorem we are approaching than by the basic intuitive idea.)

Definition 5.3. Let $\mathfrak{S} = (V, \mathbf{F}, \rightarrow, \mathbf{R})$ be an SRS; $\phi_0 \rightarrow \psi_0 \in \mathbf{R}$. A function r defined on dom ϕ_0 is a *residue map* for this rule iff

$$(\forall n \in \operatorname{dom} \phi_0)[r(n) \subseteq \operatorname{dom} \psi_0 \;\&\; (\forall m \in r(n))(\psi_0/m = \phi_0/n)] \quad \text{and} \qquad (1)$$

$$(\forall p, q \in \operatorname{dom} \phi_0)(p \perp q \text{ implies } r(p) \perp r(q)). \qquad (2)$$

Now suppose that a residue map $r[\phi_0, \psi_0]$ has been assigned to each rule and choose any specific rule $\phi_0 \rightarrow \psi_0$. Let r_0 be $r[\phi_0, \psi_0]$ and let $n \in \operatorname{dom} \phi_0$. In (1) above, we have $m \perp m'$ or $m = m'$ whenever $m, m' \in r_0(n)$, so $r_0(n)$ is an independent set of nodes. If some rule $\phi \rightarrow \psi$ has $\phi_0/n = \phi$, then we can form the pair of trees $\phi_1 \rightarrow \psi_1$ by applying $\phi \rightarrow \psi$ at n in ϕ_0 and at each residue of n in ψ_0. In a closed system $\phi_1 \rightarrow \psi_1$ must also be one of the rules. We will also require that nodes independent of n in ϕ_0 be unaffected by all this: If $p \perp n$ in dom ϕ_0, then p has the same residues in ψ_1 as in ψ_0:

$$r[\phi_1, \psi_1](p) = r[\phi_0, \psi_0](p).$$

(This additional requirement is motivated more by technical considerations in the proof of the theorem we are approaching than by the basic intuitive idea.)

Definition 5.4. Let $\mathfrak{S} = (V, \mathbf{F}, \rightarrow \mathbf{R})$ be an SRS. Then $\mathfrak{S}$ is *closed* iff it is possible to assign a residue map $r[\phi_0, \psi_0]$ to each rule $\phi_0 \rightarrow \psi_0 \in \mathbf{R}$ in such a way that the following holds: Whenever $\phi_0 \rightarrow \psi_0$, $\phi \rightarrow \psi \in \mathbf{R}$ and $n \in \operatorname{dom} \phi_0$ with $n \neq (\,)$ and $\phi_0/n = \phi$, then

$$\phi_1 \rightarrow \psi_1 \in \mathbf{R} \qquad (1)$$

and

$$(\forall p \in \operatorname{dom} \phi_0)[p \perp n \text{ implies } r[\phi_1, \psi_1](p) = r[\phi_0, \psi_0](p)], \tag{2}$$

where

$$\phi_1 = \phi_0(n \leftarrow \psi) \;\&\; \psi_1 = \psi_0(r[\phi_0, \psi_0](n) \leftarrow \psi). \tag{3}$$

In the next section we show that any system whose rules can be specified in a certain natural way is closed, so that tedious verifications of the conditions in Definitions 5.3 and 5.4 do not have to be carried out in every concrete situation.

That an unequivocal closed SRS should be Church-Rosser is quite plausible, but neither induction on lengths of derivations nor induction on sizes of trees is appropriate. Induction on the sizes of certain sets of nodes does work. The size of $N \subseteq \mathbf{N}^*$ is its cardinality $|N|$. Given an SRS $\mathfrak{C} = (V, \mathbf{F}, \rightarrow, \mathbf{R})$, we define a relation $\rightarrow_N$ on $\mathbf{F}$ for every $N \subseteq \mathbf{N}^*$.

Definition 5.5. Let $\mathfrak{C} = (V, \mathbf{F}, \rightarrow, \mathbf{R})$ be an SRS. For any $N \subseteq \mathbf{N}^*$, let $(n_0, \cdots, n_{|N|-1})$ be a listing of the members of N and define $\rightarrow_N$ to be the set of all $\langle R, S\rangle$ in $\mathbf{F} \times V_*$ such that

N is an independent set of nodes in dom R, (1)

$$(\exists \phi, \psi \in (V_*)^{|N|}) [(\forall i < |N|)(R/n_i = \phi_i \;\&\; \phi_i \rightarrow \psi_i \in \mathbf{R}) \\ \& S = R(n_0 \leftarrow \psi_0) \cdots (n_{-1} \leftarrow \psi_{-1})]. \tag{2}$$

Now define

$$(\rightarrow_\perp) = \bigcup_{N \subseteq \mathbf{N}^*} (\rightarrow_N). \tag{3}$$

Suppose that N is indeed an independent finite set of nodes and that $R \rightarrow_N S$. For any $i < |N|$, we can show that

$$R(n_0 \leftarrow \psi_0) \cdots (n_{i-1} \leftarrow \psi_{i-1})/n_i = R/n_i$$

by i applications of persistence ((3) of Lemma 4.7), so that

$$R(n_0 \leftarrow \psi_0) \cdots (n_{i-1} \leftarrow \psi_{i-1}) \rightarrow R(n_0 \leftarrow \psi_0) \cdots (n_i \leftarrow \psi_i)$$

by application of $\phi_i \rightarrow \psi_i$ at n_i. Therefore

$$(\rightarrow_N) \subseteq (\rightarrow^{|N|}).$$

THEOREM 5.6. Main Theorem. *Any unequivocal closed SRS is Church-Rosser.*

PROOF. Let $\mathfrak{C} = (V, \mathbf{F}, \rightarrow, \mathbf{R})$ be an unequivocal closed SRS. We will use Lemma 3.4 to show that $(\mathbf{F}, \rightarrow)$ is Church-Rosser. Let $\rightarrow_\perp$ be as in Definition 5.5. Since $(\rightarrow_N) \subseteq (\rightarrow^{|N|})$ whenever N is an independent finite subset of $\mathbf{N}^*$ and $\rightarrow_N = \varnothing$ otherwise, the union $\rightarrow_\perp$ in (3) of Def. 5.5 is contained in $\rightarrow^*$. Therefore $\rightarrow_\perp$ is indeed a relation on $\mathbf{F}$.

We need (1) of Lemma 3.4: $(\rightarrow_\perp^*) = (\rightarrow^*)$. We already have $(\rightarrow_\perp) \subseteq (\rightarrow^*)$, so it will suffice to show that $(\rightarrow) \subseteq (\rightarrow_\perp)$. If $R \rightarrow S$, then some $n \in \operatorname{dom} R$ has $R \rightarrow_{\{n\}} S$ and therefore $R \rightarrow_\perp S$, as desired.

We verify (2) of Lemma 3.4 by proving the following statement inductively for all $K \in \mathbf{N}$:

$$\begin{aligned}&(\forall R,S,S' \in \mathbf{F})(\forall N,N' \subseteq \operatorname{dom} R)\\ &[(R \twoheadrightarrow_N S \;\&\; R \twoheadrightarrow_{N'} S' \;\&\; |N \cup N'| = K) \text{ implies}\\ &(\exists P \subseteq \operatorname{dom} S)(\exists P' \subseteq \operatorname{dom} S')(\exists T \in \mathbf{F})\\ &(S \twoheadrightarrow_P T \;\&\; S' \twoheadrightarrow_{P'} T \;\&\; P \cup P' \subseteq (N \cup N')\cdot\mathbf{N}^*)].\end{aligned} \qquad (1)$$

Let $K \in \mathbf{N}$ and let (1) hold for all $k < K$. Let R, S, S', N, N' be as in (1), with $|N \cup N'| = K$. Let rules $\phi_i \to \psi_i$ be applied at nodes n_i in N to form S, while rules $\phi_j' \to \psi_j'$ are applied at nodes n_j' in N' to form S'. We show how to choose P, P', T.

Case 1. $(N \perp N')$. Let $P = N'$; $P' = N$;

$$T = R(n_0 \leftarrow \psi_0) \cdots (n_{-1} \leftarrow \psi_{-1})(n_0' \leftarrow \psi_0') \cdots (n_{-1}' \leftarrow \psi_{-1}').$$

Then P, P', T have all the desired properties by persistence ((3) of Lemma 4.7) and commutativity ((4) of Lemma 4.7).

Case 2. (NOT $N \perp N'$). We may assume some member of N is an ancestor of some member of N'. After permuting the listings for N and N' as needed, we may assume that some $J > 0$ has n_0 anc n_j' for all $j < J$ and NOT$(n_0$ anc $n_j')$ for all $j \geq J$. Let N_0 be $N - \{n_0\}$ and N_0' be $N' - \{n_j' \mid j < J\}$. Then $n_0 \perp (N_0 \cup N_0')$ by independence of N_0', transitivity ((3) of Lemma 4.6), and (4) of Def. 4.1 and Def. 4.5.

Case 2.1. $(n_0 = n_j'$ for some $j < J$; choose one). Let R_0 be $R(n_0 \leftarrow \psi_0)$. Since $\mathfrak{C}$ is unequivocal, $R_0 = R(n_j' \leftarrow \psi_j')$ also. We have

$$R_0 \twoheadrightarrow_{N_0} S \;\&\; R_0 \twoheadrightarrow_{N' - \{n_j'\}} S' \;\&\; |N_0 \cup N' - \{n_j'\}| = K - 1.$$

By the induction hypothesis there are P, P', T with $S \twoheadrightarrow_P T$ and $S' \twoheadrightarrow_{P'} T$ and

$$P \cup P' \subseteq (N_0 \cup N' - \{n_j'\})\cdot\mathbf{N}^* \subseteq (N \cup N')\cdot\mathbf{N}^*.$$

Case 2.2. $(n_0 \neq n_j'$, all $j < J)$. We wish to apply the induction hypothesis to $N_0 \cup N_0'$, so we construct trees R_0, S_0, S_0' that have $R_0 \twoheadrightarrow_{N_0} S_0$ and $R_0 \twoheadrightarrow_{N_0'} S_0'$. The construction hinges on a rule $\bar{\phi} \to \bar{\psi}$. Let $r[\phi_0, \psi_0]$ be the residue map for $\phi_0 \to \psi_0$ in the definition of closed systems (Def. 5.4). Since $\{n_j'/n_0 \mid j < J\}$ is an independent subset of dom ϕ_0, the sets

$$M_j = r[\phi_0, \psi_0](n_j'/n_0) \qquad \text{for all } j < J$$

are independent subsets of dom ψ_0 with $M_j \perp M_k$ whenever $j \neq k$. The union $M = \bigcup_{j<J} M_j$ is therefore an independent subset of dom ψ_0.

Define a pair of trees $\bar{\phi} \to \bar{\psi}$ by

$$\begin{aligned}\bar{\phi} &= \phi_0(n_0'/n_0 \leftarrow \psi_0') \cdots (n_{J-1}'/n_0 \leftarrow \psi_{J-1}'),\\ \bar{\psi} &= \psi_0(M_0 \leftarrow \psi_0') \cdots (M_{J-1} \leftarrow \psi_{J-1}').\end{aligned} \qquad (2)$$

By induction on J, we can show that $\bar{\phi} \to \bar{\psi} \in \mathbf{R}$.

Figure 8 will be verified next. Let

$$R_0 = R(n_0 \leftarrow \bar{\psi}), \qquad S_0 = S(n_0 \leftarrow \bar{\psi}), \qquad S_0' = S'(n_0 \leftarrow \bar{\psi}). \qquad (3)$$

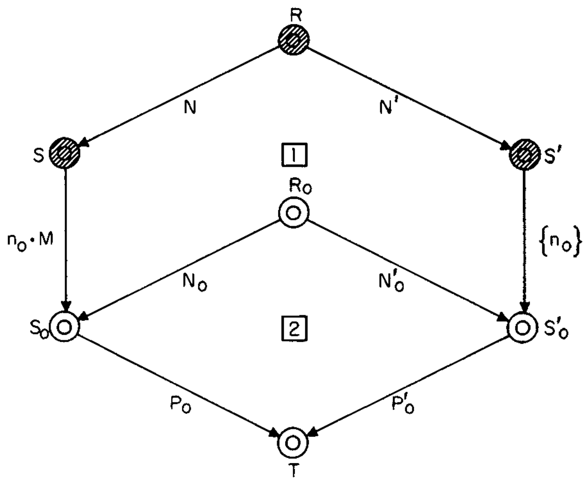

FIG 8. Case 2.2 in the proof of the Main Theorem (Theorem 5.6)

By J applications of distributivity ((2) of Lemma 4.9) to (2) above,

$$\begin{aligned}\bar{\phi} &= (R/n_0)(n_0'/n_0 \leftarrow \psi_0') \cdots (n_{J-1}'/n_0 \leftarrow \psi_{J-1}') \\ &= R(n_0' \leftarrow \psi_0') \cdots (n_{J-1}' \leftarrow \psi_{J-1}')/n_0 .\end{aligned}$$

By $n_0 \perp N_0'$ and $|N_0'|$ applications of persistence ((3) of Lemma 4.7), this becomes

$$\begin{aligned}\bar{\phi} &= R(n_0' \leftarrow \psi_0') \cdots (n_{J-1}' \leftarrow \psi_{J-1}') \cdots (n_{-1}' \leftarrow \psi_{-1}')/n_0 \\ &= S'/n_0 .\end{aligned}$$

By $\bar{\phi} = S'/n_0$, $\bar{\phi} \to \bar{\psi} \in \mathbf{R}$, and $S_0' = S'(n_0 \leftarrow \bar{\psi})$ in (3), we have

$$S' \rightarrow_{\{n_0\}} S_0'. \tag{4}$$

Because M is an independent subset of dom ψ_0 and $\psi_0 = S/n_0$, $n_0 \cdot M$ is an independent subset of dom S. We will show that

$$S \rightarrow_{n_0 \cdot M} S_0 , \tag{5}$$

where $\phi_j' \to \psi_j'$ is applied at $n_0 \cdot p$ whenever $j < J$ and $p \in M_j$. Suppose $j < J$ and $p \in M_j$. By $|N| - 1$ applications of persistence ((3) of Lemma 4.7) and then one application of embedding ((1) of Lemma 4.7), we have

$$\begin{aligned}S/(n_0 \cdot p) &= R(n_0 \leftarrow \psi_0) \cdots (n_{-1} \leftarrow \psi_{-1})/(n_0 \cdot p) \\ &= R(n_0 \leftarrow \psi_0)/(n_0 \cdot p) \\ &= \psi_0/p.\end{aligned}$$

By (1) of Def. 5.3 (the definition of residue maps), we have

$$\psi_0/p = \phi_0/(n_j'/n_0) = \phi_j',$$

so that $S/(n_0 \cdot p) = \phi_j'$. By $|M|$ applications of associativity ((2) of Lemma 4.7) in (3) and (2),

$$\begin{aligned}S_0 &= S(n_0 \leftarrow \psi_0(M_0 \leftarrow \psi_0') \cdots (M_{J-1} \leftarrow \psi_{J-1}')) \\ &= S(n_0 \leftarrow \psi_0)(n_0 \cdot M_0 \leftarrow \psi_0') \cdots (n_0 \cdot M_{J-1} \leftarrow \psi_{J-1}') \\ &= S(n_0 \cdot M_0 \leftarrow \psi_0') \cdots (n_0 \cdot M_{J-1} \leftarrow \psi_{J-1}'),\end{aligned}$$

since $S(n_0 \leftarrow \psi_0) = S$. This completes the proof of (5).

We now claim that

$$R_0 \twoheadrightarrow_{N_0} S_0 \;\&\; R_0 \twoheadrightarrow_{N_0'} S_0'. \tag{6}$$

First we must show that $R_0 \in \mathbf{F}$. By applying $\phi_j' \rightarrow \psi_j'$ at n_j' for each $j < J$ and then applying $\bar{\phi} \rightarrow \bar{\psi}$ at n_0, we get

$$R \twoheadrightarrow_{\perp} R(n_0' \leftarrow \psi_0') \cdots (n_{J-1}' \leftarrow \psi_{J-1}') \twoheadrightarrow R(n_0 \leftarrow \bar{\psi}) = R_0 .$$

Therefore $R_0 \in \mathbf{F}$.

We can show that $R_0 \twoheadrightarrow_{N_0} S_0$ by applying $\phi_i \rightarrow \psi_i$ at n_i for each i with $0 < i < |N|$. This proves the first half of (6). Similar reasoning with n_J', $\cdots$, n_{-1}' in place of n_1, $\cdots$, n_{-1} leads to the second half of (6).

By (4), (5), and (6), the upper portion (D1) of Figure 8 is now complete. Because $|N_0 \cup N_0'| = K - 1 - J < K$, the induction hypothesis yields P_0, P_0', T with $S_0 \twoheadrightarrow_{P_0} T$; $S_0' \twoheadrightarrow_{P_0'} T$; $P_0 \cup P_0' \subseteq (N_0 \cup N_0')\cdot\mathbf{N}^*$. This completes the lower portion (D2) of Figure 8.

Let $P = n_0\cdot M \cup P_0$ and $P' = \{n_0\} \cup P_0'$, so that P and P' are independent subsets of dom S and dom S', respectively. By $S_0 \twoheadrightarrow_{P_0} T$ and (5), we have $S \twoheadrightarrow_P T$. By $S_0' \twoheadrightarrow_{P_0'} T$ and (4), we have $S' \twoheadrightarrow_{P'} T$. Finally,

$$P \cup P' \subseteq n_0\cdot\mathbf{N}^* \cup (N_0 \cup N_0')\cdot\mathbf{N}^* \subseteq (N \cup N')\cdot\mathbf{N}^*. \quad \blacksquare$$

As we have remarked, some of the restrictions in the definitions of residue maps and of closed SRS's are based less on the intuitive ideas than on the need to show that $\bar{\phi} \rightarrow \bar{\psi} \in \mathbf{R}$ in the proof of the Main Theorem above. It would be helpful if the definitions could be weakened so that unequivocal closed SRS's would still be Church-Rosser but more systems would be unequivocal and closed. The following modification of the closure concept is too weak to support a Church-Rosser theorem, but it does help verify commutativity, as in Lemma 10.3 of [16].

Definition 5.7. Let $\mathfrak{C} = (V, \mathbf{F}, \twoheadrightarrow, \mathbf{R})$ be an SRS; $\mathbf{R}_1, \mathbf{R}_2 \subseteq \mathbf{R}$; $\phi_0 \rightarrow \psi_0 \in \mathbf{R}_1$; $n \in \operatorname{dom} \phi_0$; $n \neq (\;)$; $M \subseteq \operatorname{dom} \psi_0$ with M independent. Then $\phi_0 \rightarrow \psi_0$ is *pseudoclosed* at (n, M) with respect to $(\mathbf{R}, \mathbf{R}_2)$ iff, whenever $\phi \rightarrow \psi \in \mathbf{R}_2$ and $\phi_0/n = \phi$, there is some $\bar{\psi} \in V_*$ such that

$$\phi_0(n \leftarrow \psi) \rightarrow \psi_0(M \leftarrow \bar{\psi}) \in \mathbf{R}_1, \tag{1}$$

$$(\forall p \in M)(\psi_0/p \rightarrow \bar{\psi} \in \mathbf{R}_2). \tag{2}$$

LEMMA 5.8. *Let* $\mathfrak{C} = (V, \mathbf{F}, \twoheadrightarrow, \mathbf{R})$ *be an SRS*; $\mathbf{R}_1, \mathbf{R}_2 \subseteq \mathbf{R}$; $R, S, S' \in \mathbf{F}$; $m, m' \in \operatorname{dom} R$. *Suppose* $R \twoheadrightarrow_{\{m\}} S$; $R \twoheadrightarrow_{\{m'\}} S'$; m *anc* $m' \neq m$; $R/m \rightarrow S/m \in \mathbf{R}_1$; $R/m' \rightarrow S'/m' \in \mathbf{R}_2$. *Suppose* $M \subseteq$ *dom* (S/m), *independent, with* $R/m \rightarrow S/m$ *pseudoclosed at* $(m'/m, M)$ *with respect to* $(\mathbf{R}_1, \mathbf{R}_2)$. *Then there is a* $T \in \mathbf{F}$ *such that*

$$S \twoheadrightarrow^{|M|} T \qquad (\textit{using rules in } \mathbf{R}_2),$$

$$S' \twoheadrightarrow T \qquad (\textit{using a rule in } \mathbf{R}_1).$$

PROOF. Let $\phi_0 \rightarrow \psi_0$ be $R/m \rightarrow S/m$; n be m'/m; $\phi \rightarrow \psi$ be $R/m' \rightarrow S'/m'$. Let $\bar{\psi}$ be as in Def. 5.7 and set $\phi_1 = \phi_0(n \leftarrow \psi)$; $\psi_1 = \psi_0(M \leftarrow \bar{\psi})$, so that $\phi_1 \rightarrow \psi_1 \in \mathbf{R}_1$ by (1) of Def. 5.7. Let T be $R(m \leftarrow \psi_1)$. Then

$$S'/m = (R/m)(m'/m \leftarrow \psi) = \phi_0(n \leftarrow \psi) = \phi_1,$$

$$T = R(m' \leftarrow \psi)(m \leftarrow \psi_1) = S'(m \leftarrow \psi_1).$$

So $S' \rightarrow T$ using a rule in $\mathbf{R}_1$. On the other hand, suppose $p \in M$, so that $\psi_0/p \rightarrow \bar{\psi} \in \mathbf{R}_2$ by (2) of Def. 5.2. Then

$$S/(m\cdot p) = (S/m)/p = \psi_0/p,$$
$$T = R(m \leftarrow \psi_0(M \leftarrow \bar{\psi})) = S(m\cdot M \leftarrow \bar{\psi}).$$

So $S \rightarrow^{|M|} T$ using rules in $\mathbf{R}_2$. ▌

6. *Parameters and Rule-Schemata*

We present a flexible general method for specifying sets of rules that can generate important infinite sets from finite specifications, and that lets us verify properties of the sets of rules by inspecting these specifications.

For the rest of this section, we suppose that sets V and U are given, and that to each $u \in U$ there has been assigned a subset D_u of V_*. Members of U will be called *parameters*; each parameter u has *domain* D_u. (In some examples $U \cap V = \varnothing$, but in others $U \subseteq V$.)

Definition 6.1. A *template* is any $R \in (V \cup U)_*$ such that $R^{-1}(U)$ is a set of leaves in R. A *rule-schema* is any pair $R \rightarrow S$ of templates such that

$$(\forall u \in U)(|R^{-1}(u)| \leq 1), \tag{1}$$

$$(\forall u \in U)(S^{-1}(u) \neq \varnothing \text{ implies } R^{-1}(u) \neq \varnothing). \tag{2}$$

To form an "instance" of a rule schema, we must substitute trees for parameters. This is the familiar idea of substituting "expressions" for "variables" in algebra and logic.

Definition 6.2. Let $R \rightarrow S$ be a rule-schema. If $(u_0, \cdots, u_{K-1})$ with $K \in \mathbf{N}$ is a list of the members of $\{u \in U \mid R^{-1}(u) \neq \varnothing\}$ and $T_k \in D_{u_k}$ for each $k < K$, then $\phi \rightarrow \psi$ is an *instance of* $R \rightarrow S$, where

$$\begin{aligned} \phi &= R(R^{-1}(u_0) \leftarrow T_0) \cdots (R^{-1}(u_{-1}) \leftarrow T_{-1}), \\ \psi &= S(S^{-1}(u_0) \leftarrow T_0) \cdots (S^{-1}(u_{-1}) \leftarrow T_{-1}). \end{aligned} \tag{1}$$

If L is a set of rule-schemata, then

$$\mathbf{R}_L = \{\phi \rightarrow \psi \in V_* \times V_* \mid (\exists R \rightarrow S \in L)\ (\phi \rightarrow \psi \text{ is an instance of } R \rightarrow S)\}. \tag{2}$$

Given an instance $\phi \rightarrow \psi$ of a rule-schema $R \rightarrow S$, there is a natural way to define a residue map for $\phi \rightarrow \psi$. Let $n \in \operatorname{dom} \phi$. There is at most one node $m \in R^{-1}(U)$ such that m anc n because $R^{-1}(U)$ is an independent set of nodes. If there is such a node m, let $u = Rm$ and let $T = \phi/m$ so that $T \in D_u$ in Definition 6.1. For each node p in S with $Sp = u$, we have $\psi/p = T$ because T was substituted for u at p also. Therefore

$$\psi/(p\cdot(n/m)) = T/(n/m) = (\phi/m)/(n/m) = \phi/n$$

by $\psi/p = T$ and two applications of cancellation ((1) of Lemma 4.9). Thus $p\cdot(n/m)$ has the basic property of residue nodes: A copy of ϕ/n appears at $p\cdot(n/m)$ in ψ. All such $p\cdot(n/m)$ will be considered residues of n.

LEMMA 6.3. *Let $\phi \rightarrow \psi$ be an instance of a rule-schema $R \rightarrow S$. Define a map from dom ϕ into the set of subsets of $\mathbf{N}^*$ by*

$$r(n) = \{p\cdot(n/m) \mid m \in R^{-1}(U) \;\&\; m \text{ anc } n \;\&\; Sp = Rm\}.$$

Then r is a residue map for $\phi \rightarrow \psi$.

PROOF. Condition (1) in Def. 5.3 was verified above. We must verify (2) of Def. 5.3, namely, independent nodes have independent residues.

It will suffice to prove that

$$(\forall n, n' \in \text{dom } \phi)(\forall q' \in r(n'))(q \text{ anc } q' \text{ implies } n \text{ anc } n').$$

Suppose $q \in r(n)$; $q' \in r(n')$; q anc q'. For some m anc n and some m' anc n' we have $m, m' \in R^{-1}(U)$. For some $p \in S^{-1}(Rm)$ and some $p' \in S^{-1}(Rm')$ we have $q = p\cdot(n/m)$ and $q' = p'\cdot(n'/m')$. By q anc q' we have $p\cdot\mathbf{N}^* \cap p'\cdot\mathbf{N}^* \neq \varnothing$, while $p \perp p'$ or $p = p'$ because p and p' are leaves in S. Therefore $p = p'$. By $p \in S^{-1}(Rm) \cap S^{-1}(Rm')$, we have $Rm = Rm'$. In the definition of rule-schemata we have $m = m'$ by (1) of Def. 6.1, so that q anc q' now implies that n anc n', as desired. ▌

Replacing parts of an instance $\phi \rightarrow \psi$ of a schema will often lead to another instance of the same schema. If we replace only parts of ϕ in which we substituted for parameters, if these replacements do not result in substitutions from outside the domains of parameters, and if the appropriate replacements are made at residues in ψ also, then the result $\tilde{\phi} \rightarrow \tilde{\psi}$ should still be an instance of the schema.

LEMMA 6.4. *Let a rule $\phi \rightarrow \psi$ be an instance of a rule schema $R \rightarrow S$. Let N be an independent subset of dom ϕ and let $Q[n] \in V_*$ for each $n \in N$. For any listing $(n_0, \cdots, n_{K-1})$ of the members of N, set*

$$\begin{aligned} \tilde{\phi} &= \phi(n_0 \leftarrow Q[n_0]) \cdots (n_{-1} \leftarrow Q[n_{-1}]), \\ \tilde{\psi} &= \psi(r(n_0) \leftarrow Q[n_0]) \cdots (r(n_{-1}) \leftarrow Q[n_{-1}]), \end{aligned} \tag{1}$$

where r is the residue map from Lemma 6.3. For each $m \in R^{-1}(U)$, let $(N_0^m, \cdots, N_{K_m-1}^m)$ be a listing of the members of N descended from m. Suppose

$$(\forall k < K)(\exists m \in R^{-1}(U))(m \text{ anc } n_k), \tag{2}$$

$$(\forall m \in R^{-1}(U))[(\phi/m)(N_0^m/m \leftarrow Q[N_0^m]) \cdots (N_{-1}^m/m \leftarrow Q[N_{-1}^m]) \in D_{Rm}]. \tag{3}$$

Then $\tilde{\phi} \rightarrow \tilde{\psi}$ is an instance of $R \rightarrow S$.

PROOF. Let $(\mu_0, \cdots, \mu_{J-1})$ be any listing of the members of $R^{-1}(U)$, and let $u_j = R\mu_j$ for each $j < J$. Write N_k^j rather than $N_k^{\mu_j}$ for N_k^m when $m = \mu_j$. For each $j < J$, set

$$\bar{T}_j = (\phi/\mu_j)(N_0^j/\mu_j \leftarrow Q[N_0^j]) \cdots (N_{-1}^j/\mu_j \leftarrow Q[N_{-1}^j]),$$

so that $\bar{T}_j \in D_{u_j}$ by (3). We therefore have an instance $\bar{\phi} \rightarrow \bar{\psi}$ of $R \rightarrow S$, where

$$\begin{aligned} \bar{\phi} &= R(\mu_0 \leftarrow \bar{T}_0) \cdots (\mu_{-1} \leftarrow \bar{T}_{-1}), \\ \bar{\psi} &= S(S^{-1}(u_0) \leftarrow \bar{T}_0) \cdots (S^{-1}(u_{-1}) \leftarrow \bar{T}_{-1}). \end{aligned} \tag{4}$$

Now note that (2) establishes a bijection between indices $i < K$ and pairs of indices (j, k) with $j < J$ and $k < K_{m_j}$ such that $n_i = N_k^j$. The residue map from Lemma 6.3 evaluates to $r(n_i) = S^{-1}(u_j)\cdot(N_k^j/\mu_j)$. We can use associativity ((2) of Lemma 4.7) and commutativity ((4) of Lemma 4.7) to calculate from (1) and (4) that $\tilde{\phi} = \bar{\phi}$ and $\tilde{\psi} = \bar{\psi}$. Therefore $\tilde{\phi} \rightarrow \tilde{\psi}$ is an instance of $R \rightarrow S$. ▌

THEOREM 6.5. Rule-Schemata Theorem. *Let L be a set of rule-schemata and let*

$\mathfrak{S}$ be an SRS of the form $\mathfrak{S} = (V, \mathbf{F}, \twoheadrightarrow, \mathbf{R}_L)$. Suppose that each $\phi_0 \to \psi_0 \in \mathbf{R}_L$ has

$$(\exists! R \to S \in L)(\phi_0 \to \psi_0 \text{ is an instance of } R \to S), \qquad (1)$$

$$(\forall n \in dom\ \phi_0)(\forall \phi \to \psi \in \mathbf{R}_L)$$

$$[(n \neq (\ ) \ \&\ \phi_0/n = \phi) \text{ implies } (\exists m \in R^{-1}(U))(m \text{ anc } n \ \&\ (\phi_0/m)(n/m \leftarrow \psi) \in D_{Rm})], \qquad (2)$$

where R is the left half of the rule-schema $R \to S$ with $\phi_0 \to \psi_0$ as an instance. Then $\mathfrak{S}$ is closed.

Proof. Since each rule is an instance of a unique rule-schema, residue maps may be assigned to the rules by Lemma 6.3. We verify the properties of closed systems in Definition 5.4 by calculations based on Lemma 6.4 (in the case $K = 1$). ▌

Although only a very special case of Lemma 6.4 has been used so far, the full generality is needed to establish additional results in the application areas.

Hindley (personal communication) has found conditions that imply the Church-Rosser property for systems of combinatory logic. When applied to rule-schemata in the special format used in combinatory logic, these conditions insure that the logical system is unequivocal and closed when expressed as an SRS; the proof that this is so is a specialization of the proof of the Rule Schemata Theorem (Theorem 6.5). The Church-Rosser property then follows from the Main Theorem (Theorem 5.6).

Hindley's conditions were inspired by Sanchis' proof of the Church-Rosser property for one specific combinatory system [17, Sec. 2, Lem. 1]. Implicit in Sanchis' proof are broad outlines of the proofs of Theorems 6.5 and 5.6, as restricted to this one set of rule-schemata and the set of rules it generates. The simplicity of this particular system made it appropriate to ignore all details and to carry out the proofs on an intuitive level.

7. *An Algorithmic Explanation of Recursion*

The factorial function may be "defined" by writing

(a) $f(x) := \textbf{if } x = 0 \textbf{ then } 1 \textbf{ else } x \times f(x - 1)$.

The circularity of this definition renders it useless without some further explanation. Following Morris [14, Ch. 3], we recognize two sorts of explanations for recursive definitions: algorithmic and semantic. The Main Theorem (Theorem 5.6) leads to satisfactory and intimately related solutions for two basic problems posed by algorithmic and semantic explanations. We treat the "functionality" problem in this section and the "validity" problem in the next section.

First we review McCarthy's concept of recursive definitions [11]. A recursive definition is a set of equations of the form

$$f(a_0, \cdots, a_{k-1}) := e,$$

where f is a "function letter" used to name the function we are defining, and e is any expression built up from function letters, names of given functions that have already been defined, the **if** ⋯ **then** ⋯ **else** ⋯ construction, variables chosen from among $a_0, \cdots, a_{k-1}$, and constants. In [11] there can be only one equation beginning with f for each function letter f, and the arguments $a_0, \cdots, a_{k-1}$ must all be variables. These restrictions will be slightly relaxed below.

Each equation

$$f(a_0, \cdots, a_{k-1}) := e$$

in a recursive definition generates a set $\mathbf{R}[f]$ of *rules* of the form

$$f(A_0, \cdots, A_{k-1}) \to E,$$

where each A_i is an expression that may be substituted for a_i, and E is the result of performing all these substitutions in e. For each given function g, we also have a set $\mathbf{R}[g]$ consisting of all rules

$$g(\xi_0, \cdots, \xi_{-1}) \to \eta,$$

where η is a constant representing the value of g for the constants $\xi_0, \cdots, \xi_{-1}$. For conditional expressions we also have the set $\mathbf{R}[C]$ of all rules

$$\textbf{if true then } E \textbf{ else } E' \to E,$$

$$\textbf{if false then } E \textbf{ else } E' \to E'.$$

Expressions are to be evaluated by matching subexpressions against the left sides of rules, then replacing these subexpressions by the right sides of the rules.

When an algorithmic explanation is nondeterministic as in [11], the "single-valuedness" of the relation between input and output needs to be shown. The general remarks motivating this *functionality problem* in Section 3 are applicable here. They can be fleshed out with examples where ingenious implementations of recursive definitions could save a great deal of time [12, Sec. 2.2] or space [19].

Now we can proceed with the formal development. For the rest of this section and the next one, let D be the *data space* of objects on which we compute. For each k in the set $\mathbf{P}$ of *positive integers* we consider any set G_k of *given functions*. Each $g \in G_k$ is a partial function $g : \mathrm{D}^k \to \mathrm{D}$, whose computations are taken for granted. In applications the given functions are the hardware capabilities, library subroutines, and so on, that the programmer can use as black boxes. The examples in this section will use $\mathrm{D} = \mathbf{N}$ and given functions from arithmetic.

In order to write recursive definitions of new functions in terms of old ones and each other, let F_k be an infinite set of *function letters* for each $k \in \mathbf{P}$. The sets F_k are to be disjoint from each other, from D, and from the sets G_j. Finally, let W and X be infinite sets that are disjoint from each other and from all the previous sets. For reasons that will be clear shortly, we say that W is the set of *call-by-name parameters* and X is the set of *call-by-value parameters.*

The given functions, function letters, parameters, and data combine to form a total vocabulary V with a *rank function* $\rho : V \to \mathbf{N}$ as follows:

$$V = \mathrm{D} \cup W \cup X \cup F \cup G$$

where

$$F = \bigcup_{k \in \mathbf{P}} F_k, \qquad G = \bigcup_{k \in \mathbf{P}} G_k,$$

$$\rho(f) = k \quad \text{for} \quad f \in F_k \cup G_k, \qquad \rho(a) = 0 \quad \text{for} \quad a \in \mathrm{D} \cup W \cup X.$$

The forest $V_\#$ defined by Def. 4.11 consists of all operator-operand structures for expressions constructed from this vocabulary. For example, the pair of trees in-

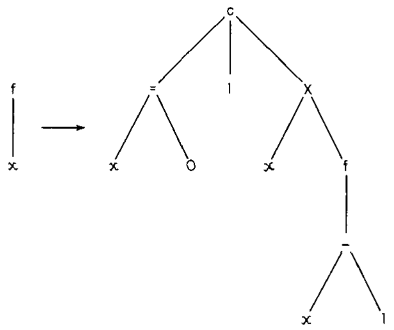

FIG. 9. Operator-operand structures in the definition of factorials

volved in defining factorials by (a) is shown in Figure 9, which assumes $C \in F_3$; $=, \times, \dot{-} \in G_2$; $f \in F_1$; $x \in X$; $\mathrm{D} = \mathbf{N}$. We will retain the overbars in the algebraic nomenclature (Def. 4.10) to prevent confusion between the *tree* $\overline{+}(\bar{4},\bar{6})$ with three nodes and the *value* $+(4,6) = 10$ of the function $+$ at the argument string $(4,6)$. Thus $+(4,6) = +(5,5)$ but $\overline{+}(\bar{4},\bar{6}) \neq \overline{+}(\bar{5},\bar{5})$.

The pair of trees in Figure 9 is to be a rule-schema in the sense of Definition 6.1. The set $U = W \cup X$ is to be the set of parameters. The domains of the parameters are specified as follows:

7.1. $(\forall u \in W)(D_u = V_\#)$ call-by-name, $(\forall x \in X)(D_x = \mathrm{D}_\#)$ call-by-value.

Call-by-name is used in the definition of the conditional operator needed for our definition of factorials. To compute conditionals, we use any two schemata of the form

(b) $$\bar{C}(\overline{\text{yes}}, \bar{u}, \bar{v}) \to \bar{u}, \qquad \bar{C}(\overline{\text{no}}, \bar{u}, \bar{v}) \to \bar{v},$$

where

(c) $$C \in F_3 \;\&\; u,v \in W \;\&\; \text{yes,no} \in \mathrm{D} \;\&\; u \neq v \;\&\; \text{yes} \neq \text{no}.$$

If we think of (b) as the definition of a "procedure" C and we think of $\bar{C}(\overline{\text{yes}},A,B)$ as a call on C with A as the "actual parameter" in place of the "formal parameter" u, then A and B should be passed *unevaluated*, as in ALGOL 60 call-by-name [15, Sec. 4.7.3.2]. If A has a value but B does not, then it is incorrect as well as inefficient to try to evaluate all subexpressions of $\bar{C}(\overline{\text{yes}},A,B)$ before evaluation of the whole expression. In Figure 9, on the other hand, the only way to evaluate $\bar{f}(A)$ is to let A evolve to $\bar{\xi}$ for some $\xi \in \mathrm{D}$ and then use the rule

$$\bar{f}(\bar{\xi}) \to \bar{C}(\overline{=}(\bar{\xi},\bar{0}),\bar{1},\overline{\times}(\bar{\xi},\bar{f}(\overline{\dot{-}}(\bar{\xi},\bar{1}))))$$

to enter the procedure f, as in ALGOL 60 call-by-value [15, Sec. 4.7.3.1].

It is easy to write a set of rule-schemata that does *not* define a function, such as $\{\bar{f}(\bar{x}) \to \bar{3}, \bar{f}(\bar{u}) \to \bar{4}\}$. We formalize conditions under which sets of schemata could reasonably be expected to define functions. The left half R of each schema $R \to S$ should be of the form $\bar{f}(\bar{a}_0, \cdots, \bar{a}_{k-1})$ with $f \in F_k$ and $a_i \in \mathrm{D} \cup W \cup X$ for all $i < k$. This is equivalent to saying that $R \in V_\#$ with $R(\ ) \in F$ and $\operatorname{dom} R \subseteq \mathbf{N}^0 \cup \mathbf{N}^1$. As in our schemata for conditionals, there may be two schemata $R \to S$ with the same root label $R(\ )$, but then there must be differences between constant arguments that will separate the instances of the two schemata.

Definition 7.2. A *recursive definition* (RD) is any set L of rule-schemata in $V_{\#} \times V_{\#}$ such that

$$(\forall R \rightarrow S \in L)(R(\ ) \in F \ \&\ \operatorname{dom} R \subseteq \mathbf{N}^1 \cup \mathbf{N}^0), \tag{1}$$

$$\begin{aligned}&(\forall R \rightarrow S, R' \rightarrow S' \in L)\\ &\quad([R \rightarrow S \neq R' \rightarrow S' \ \&\ R(\ ) = R'(\ )]\\ &\qquad\qquad \text{implies } (\exists j \in \mathbf{N})[R(j), R'(j) \in \mathrm{D} \ \&\ R(j) \neq R'(j)]).\end{aligned} \tag{2}$$

In defining rule-schemata in Section 6, we required that every parameter occurring in S should occur in R when $R \rightarrow S$ is a rule-schema ((2) of Def. 6.1). Therefore no two rules have the same left half in the set $\mathbf{R}_{\{R\rightarrow S\}}$ of instances of any schema $R \rightarrow S$. By (2), $\phi \neq \phi'$ whenever $\phi \rightarrow \psi$, $\phi' \rightarrow \psi'$ are instances of $R \rightarrow S$, $R' \rightarrow S' \in L$ and $R' \rightarrow S' \neq R \rightarrow S$. Therefore no two rules in $\mathbf{R}_L$ have the same left half.

Definition 7.3. Let L be an RD and define an SRS $\mathfrak{C}_L$ by

$$\mathbf{R}_{\mathrm{giv}} = \{\bar{g}(\bar{\xi}_0, \cdots, \bar{\xi}_{-1}) \rightarrow \bar{\eta} \mid g \in G \ \&\ g(\xi_0, \cdots, \xi_{-1}) = \eta\}, \ \text{and} \tag{1}$$

$$\mathfrak{C}_L = (V, V_{\#}, \twoheadrightarrow, \mathbf{R}_{\mathrm{giv}} \cup \mathbf{R}_L). \tag{2}$$

Definitions 7.2 and 7.3 extend the McCarthy formalism in two ways. First, they allow members of D as well as parameters to appear at argument positions in the left sides of schemata, as in the "proper programs" of E. K. Blum [1, Sec. 4]. This flexibility renders the original ad hoc treatment of conditionals [11, p. 42, Rule 1] unnecessary. Second, our definitions introduce an explicit choice between call-by-value with X and call-by-name with W. The original evaluation rules specified call-by-value for given functions [11, p. 42, Rule 2] and call-by-name for function letters [11, p. 42, Rule 3], although call-by-value seems intended in the examples.

For any RD L, the set of rules $\mathbf{R}_{\mathrm{giv}} \cup \mathbf{R}_L$ in $\mathfrak{C}_L$ may also be described as $\mathbf{R}_M$ for $M = \mathbf{R}_{\mathrm{giv}} \cup L$, because each $\phi \rightarrow \psi \in \mathbf{R}_{\mathrm{giv}}$ is a schema with no parameters and with itself as the only instance. The *SRS* $\mathfrak{C}_L$ does have the form $(V, \mathbf{F}, \twoheadrightarrow, \mathbf{R}_M)$ for M a set of schemata, so it is appropriate to use the Rule-Schemata Theorem (Theorem 6.5).

LEMMA 7.4. *Let L be an* RD. *Then $W \cup X$ is a set of parameters and* $\mathbf{R}_{\mathrm{giv}} \cup L$ *is a set of rule-schemata such that*

(1) $\mathbf{R}_{\mathrm{giv}} \cup \mathbf{R}_L = \mathbf{R}_M$ *where* $M = \mathbf{R}_{\mathrm{giv}} \cup L$;

(2) *each* $\phi_0 \rightarrow \psi_0 \in \mathbf{R}_{\mathrm{giv}} \cup \mathbf{R}_L$ *is an instance of just one* $R \rightarrow S \in \mathbf{R}_{\mathrm{giv}} \cup L$;

(3) *suppose* $\phi_0 \rightarrow \psi_0 \in \mathbf{R}_{\mathrm{giv}} \cup \mathbf{R}_L$ *is an instance of* $R \rightarrow S \in \mathbf{R}_{\mathrm{giv}} \cup L$; $n \in$ *dom* ϕ_0; $n \neq (\ )$. *Then some* $m \in R^{-1}(\mathrm{D} \cup W \cup X)$ *has* m *anc* n *and either* $\phi_0 m \in \mathrm{D}$ *or* $Rm \in W \ \&\ (\forall \psi \in V_{\#})[(\phi_0/m)(n/m \leftarrow \psi) \in D_{Rm}]$;

(4) $\mathfrak{C}_L$ is *Church-Rosser.*

PROOF. Conditions (1) and (2) are trivial.

Suppose $\phi_0 \rightarrow \psi_0 \in \mathbf{R}_{\mathrm{giv}} \cup \mathbf{R}_L$ is an instance of $R \rightarrow S \in L$. Suppose $n \in \operatorname{dom} \phi_0$ with $n \neq (\ )$, so that n begins with an integer n_0 such that $n_0 < \rho(\phi_0(\ ))$. Set $m = (n_0)$, so that $m \in \mathbf{N}^1$ and m anc n. In the definition of RD's, (1) of Def. 7.2 implies that $Rm \in \mathrm{D} \cup W \cup X$. If $Rm \in W$, then $D_{Rm} = V_{\#}$, so any $\psi \in V_{\#}$ has $(\phi_0/m)(n/m \leftarrow \psi) \in D_{Rm}$. Otherwise, $Rm \in \mathrm{D} \cup X$ and hence $\phi_0 m \in \mathrm{D}$. This proves (3).

We use the Main Theorem (Theorem 5.6) to prove (4). It is clear that $\mathfrak{C}_L$ is unequivocal. We must show that $\mathfrak{C}_L$ is closed. By (2) and the Rule-Schemata Theorem (Theorem 6.5), it will suffice to prove (2) of Theorem 6.5: Whenever

$\phi_0 \to \psi_0$, $\phi \to \psi \in \mathbf{R}_L \cup \mathbf{R}_{\mathrm{giv}}$, $\phi_0 \to \psi_0$ is an instance of $R \to S \in \mathbf{R}_{\mathrm{giv}} \cup L$, $n \in \operatorname{dom} \phi_0$, $n \neq (\ )$, and $\phi_0/n = \phi$, then some $m \in R^{-1}(W \cup X)$ has m anc n and $(\phi_0/m)(n/m \leftarrow \psi) \in D_{Rm}$. But this follows from (3) and $\phi(\ ) \notin$ D. ▌

The result $\mathrm{eval}_L R$ of *evaluating* a tree $R \in V_{\#}$ under an RD L will be defined as ξ whenever $\xi \in$ D and $R \twoheadrightarrow^* \bar{\xi}$ in D_L. It will be convenient to make evaluation a total function by letting a new object ∞ be $\mathrm{eval}_L R$ whenever R has no normal form in $\mathrm{D}_{\#}$. Although evaluation is not a bottom-up algorithm, it will be shown that replacing a subtree R/n of a tree R by a tree S that evaluates like R/n does not change the evaluation of R.

THEOREM 7.5. Evaluation Theorem. *Let* $\infty \notin$ D *and set* $\bar{\mathrm{D}} = \mathrm{D} \cup \{\infty\}$. *Let L be any RD. There is a unique surjection*

$$eval_L : V_{\#} \to \bar{\mathrm{D}} \qquad (total) \tag{1}$$

such that

$$(\forall R \in V_{\#})(\forall \xi \in \mathrm{D})(eval_L R = \xi \text{ iff } R \twoheadrightarrow^* \bar{\xi} \text{ in } \mathfrak{C}_L), \tag{2}$$

$$(\forall R,S \in V_{\#})(R \twoheadrightarrow^* S \text{ in } \mathfrak{C}_L \text{ implies } eval_L R = eval_L S), \text{ and} \tag{3}$$

$$(\forall R,S \in V_{\#})(\forall n \in dom\ R)$$
$$(eval_L S = eval_L R/n \text{ implies } eval_L R(n \leftarrow S) = eval_L R). \tag{4}$$

PROOF. If $R \twoheadrightarrow^* \bar{\xi}$, then $\bar{\xi}$ is a normal form for R in $\mathfrak{C}_L$. By the Church-Rosser result ((4) of Lemma 7.4), there is just one function eval_L satisfying (1) and (2). Since $\mathrm{eval}_L \bar{\xi} = \xi$ for each $\xi \in$ D and $\mathrm{eval}_L \bar{a} = \infty$ for each $a \in W \cup X$, this function is surjective. Now (3) follows from (2) and the Church-Rosser property. Finally, suppose R,S $\in V_{\#}$; $n \in \operatorname{dom} R$; $\mathrm{eval}_L S = \mathrm{eval}_L R/n$. We show that $\mathrm{eval}_L R(n \leftarrow S) = \mathrm{eval}_L R$.

Case 1 ($\mathrm{eval}_L S \neq \infty$). Let $\mathrm{eval}_L S = \xi = \mathrm{eval}_L R/n$. Then

$$R \twoheadrightarrow^* R(n \leftarrow \bar{\xi}) \text{ by } R/n \twoheadrightarrow^* \bar{\xi}, \qquad R(n \leftarrow S) \twoheadrightarrow^* R(n \leftarrow \bar{\xi}) \text{ by } S \twoheadrightarrow^* \bar{\xi}.$$

Therefore $\mathrm{eval}_L R(n \leftarrow S) = \mathrm{eval}_L R(n \leftarrow \bar{\xi}) = \mathrm{eval}_L R$ by (3).

Case 2 ($\mathrm{eval}_L S = \infty$). We may still have a finite value for $\mathrm{eval}_L R$.

Case 2.1 ($\mathrm{eval}_L R \neq \infty$). Let $K \in \mathbf{N}$; $\xi \in$ D; $(R_0, \cdots, R_K) \in (V_{\#})^{K+1}$ with

$$R = R_0 \twoheadrightarrow R_1 \twoheadrightarrow \cdots \twoheadrightarrow R_K = \bar{\xi}.$$

For each $k < K$, let a rule $\phi_k \to \psi_k$ be applied at a node m_k in R_k to form R_{k+1}. Let r_k be the residue map for $\phi_k \to \psi_k$ defined by Lemma 6.3. We will define a subset N_k of dom R_k for each $k \leq K$. Intuitively, suppose that n has been painted red, that a red node stays red when a rule is applied there, and that residues of red nodes are painted red whenever a rule is applied. Then N_k is the set of all red nodes in R_k. Formally,

$$N_0 = \{n\},$$
$$N_{k+1} = \{p \in N_k \mid \mathrm{NOT} m_k \text{ anc } p\} \cup (\{m_k\} \cap N_k)$$
$$\cup \{m_k \cdot q \mid (\exists p \in N_k)[m_k \text{ anc } p \neq m_k \ \&\ q \in r_k(p/m_k)]\}.$$

For each $k \leq K$, N_k is an independent subset of dom R_k such that

$$(\forall p \in N_k)(\mathrm{eval}_L R_k/p = \infty) \tag{5}$$

by (3) and the fact that $R/n \twoheadrightarrow^* R_k/p$ whenever $p \in N_k$.

For each $k \leq K$, set

$$T_k = R_k(N_k \leftarrow S). \tag{6}$$

For each $k < K$ we claim that

$$T_k \rightarrow^{=} T_{k+1}. \tag{7}$$

If $m_k \in N_k \cdot \mathbf{N}^*$, then $T_k = T_{k+1}$; so we may assume $m_k \notin N_k \cdot \mathbf{N}^*$. By part (3) of Lemma 7.4 and by Lemma 6.4,

$$\phi_k(N_k/m_k \leftarrow S) \rightarrow \psi_k(r_k(N_k/m_k) \leftarrow S) \in \mathbf{R}_{\text{giv}} \cup \mathbf{R}_L.$$

Direct calculations using the elementary properties of trees from Section 4 show that this rule is applicable to T_k at m_k and that T_{k+1} results.

By (6) and (7) we have $R(n \leftarrow S) = T_0 \rightarrow^* T_k = \xi(N_k \leftarrow S)$, while (5) implies that $N_K = \emptyset$. Therefore $\text{eval}_L R(n \leftarrow S) = \xi = \text{eval}_L R$.

Case 2.2 ($\text{eval}_L R = \infty$). Suppose $\text{eval}_L R(n \leftarrow S) \neq \infty$. Then $\text{eval}_L R(n \leftarrow S)$ $(n \leftarrow R/n) \neq \infty$ by the reasoning of Case 2.1. But $R(n \leftarrow S)$ $(n \leftarrow R/n) = R$. So this is absurd. Therefore $\text{eval}_L R(n \leftarrow S) = \infty = \text{eval}_L R$. ▌

If L is an RD and $f \in F$, then we can define a total function $Lf : \bar{\mathrm{D}}^{\rho(f)} \rightarrow \bar{\mathrm{D}}$ by the evaluation process.

Definition 7.6. Let L be an RD and let T_∞ be any member of $V_\#$ such that $\text{eval}_L T_\infty = \infty$. For each $\theta \in \bar{\mathrm{D}}$, set

$$\tilde{\theta} = \textbf{if } \theta \in \mathrm{D} \textbf{ then } \bar{\theta} \textbf{ else } T_\infty. \tag{1}$$

For each $f \in V$ and all $\theta \in \bar{\mathrm{D}}^{\rho(f)}$, set

$$Lf(\theta_0, \cdots, \theta_{-1}) = \text{eval}_L \bar{f}(\tilde{\theta}_0, \cdots, \tilde{\theta}_{-1}). \tag{2}$$

When $f \in \mathrm{D}$ we have dom $Lf = \{(\ )\}$, and the only value assumed is $Lf(\ ) = f$. When $f \in W \cup X$ we have $Lf(\ ) = \infty$. When $f \in G$, Lf is the natural extension of f from a partial function on $\mathrm{D}^{\rho(f)}$ to a total function on $\bar{\mathrm{D}}^{\rho(f)}$. When $f \in F$, the values of Lf depend on the choice of L and may be finite despite infinite arguments. For example, if L includes the schema for conditionals, then $LC(\text{yes},17,\infty) = 17$.

What have we really gained by adding ∞ to the data space and expressing the theory in terms of total functions? One advantage is simplicity. Suppose that $\Phi : \mathrm{D}^2 \rightarrow \mathrm{D}$ and $\Psi : \mathrm{D} \rightarrow \mathrm{D}$, with both functions partial. An expression such as $\Phi(\xi,\Psi(\eta))$ is quite awkward to work with if we have no assurance that $\eta \in \text{dom}\ \Psi$ and that $(\xi,\Psi(\eta)) \in \text{dom}\ \Phi$. Suppose $\eta \notin \text{dom}\ \Psi$ but $\Phi(37,-)$ is a constant function on D. Is it correct to write $\Phi(37,\Psi(\eta)) = \Phi(37,94)$? Kleene [8, Sec. 63] introduces special notions of equality and of composition of functions in order to handle such problems consistently. We only need the usual intuitive notions, yet we can make finer distinctions. If $\bar{\Phi} : \bar{\mathrm{D}}^2 \rightarrow \bar{\mathrm{D}}$ is a total function and $\bar{\Phi}(37,-)$ is constant on D, then $\bar{\Phi}(37,\infty) = \bar{\Phi}(37,94)$ will be true if $\bar{\Phi}(37,-)$ is actually constant on all of $\bar{\mathrm{D}}$ but will be false otherwise.

The fine distinctions permitted by the use of total functions also enrich the theory. In the Section 8 we prove a theorem that cannot be expressed in the language of [8]. The introduction of ∞ here is very much like the addition of ∞ to the number system in the study of infinite series: The mathematics becomes somewhat richer and simpler than before, but none of the real difficulties are magically removed.

We have defined RD's and have explained them algorithmically: For any RD L, the SRS $\mathfrak{C}_L$ provides a nondeterministic algorithm for evaluating expressions, and any function letter f defines a function $Lf : \bar{D}^{\rho(f)} \to \bar{D}$ that is single-valued because normal forms are unique in the Church-Rosser system $\mathfrak{C}_L$. The fact that $Lf \cap (D^{\rho(f)} \times D)$ is single-valued whenever $f \in F$ and L is an RD *without call-by-name* has been proven by E. K. Blum [1, Sec. 4, Th. 1], whose "proper programs" are essentially equivalent to this kind of RD. (The additional assumptions on the data space and given functions in [1] are irrelevant to the functionality problem.) Blum's proof depends on the restriction to call-by-value and does not cover our result.

8. *A Semantic Explanation of Recursion*

We return to the RD for factorials, but now we name the given function $\{\langle(\xi, \eta), \zeta\rangle \mid \xi, \eta, \zeta \in D \;\&\; [(\xi = \eta \;\&\; \zeta = \text{yes}) \text{ or } (\xi \neq \eta \;\&\; \zeta = \text{no})]\}$ by "EQ" rather than by "=" to prevent confusion between it and the relation of equality on $\bar{D}$.

The rule for $\bar{f}(\bar{6})$ may be thought of as an equation

$$\bar{f}(\bar{6}) = \bar{C}(\overline{\mathrm{EQ}}(\bar{6},\bar{0}),\bar{1},\bar{\times}(\bar{6},\bar{f}(\bar{\dot{-}}(\bar{6},\bar{1})))) \tag{1}$$

that may be true or false, depending on which operations are considered to be the meanings of the operators. We are concerned with whether $\bar{f}(\bar{6})$ and $\bar{C}(\cdots)$ have the same *value* in $\bar{D}$; we already know they are not the same tree! We therefore write (1) more explicitly as

$$\mathrm{val}_I\bar{f}(\bar{6}) = \mathrm{val}_I\bar{C}(\overline{\mathrm{EQ}}(\bar{6},\bar{0}),\bar{1},\bar{\times}(\bar{6},\bar{f}(\bar{\dot{-}}(\bar{6},\bar{1})))) \tag{2}$$

where I is an "interpretation" of the function letters that assigns an actual function $I_f : \bar{D}^{\rho(f)} \to \bar{D}$ (total) to each $f \in F$.

A *semantic explanation* for RD's is any scheme for assigning value maps val_I to interpretations I, so that an interpretation I can be said to *solve* an RD L if $\mathrm{val}_I\phi = \mathrm{val}_I\psi$ for each rule $\phi \to \psi \in \mathbf{R}_L$. Under the natural semantic explanation we will use, every RD L has solutions. Indeed, the assignment $I_f = Lf$ of functions to function letters defined by (2) of Def. 7.6 is a solution. This particular solution can also be characterized in a purely semantic way—it is the unique solution that agrees with every other solution wherever it has a finite value. We call this the "canonical" solution.

If we call members of $F \cup G$ "operators" and the members of $D \cup W \cup X$ "individual symbols," then $V_\#$ corresponds to a set of "terms" in logic. A logician would "interpret" operators f of rank k as operations $I_f : \bar{D}^k \to \bar{D}$ and individual symbols a as members $J_a \in \bar{D}$, in order to interpret terms as denoting members of D. It will be more convenient to treat "symbols" of rank 0 consistently. We will say that each individual symbol is interpreted as an operation $I_a : \bar{D}^0 \to \bar{D}$, so that $I_a = \{\langle(\,), \phi\rangle\}$ for some $\phi \in \bar{D}$. This will save some unnecessary case analyses.

Definition 8.1. An *interpretation* of V on $\bar{D}$ is any function I assigning to each $\alpha \in V$ a total function $I_\alpha : \bar{D}^{\rho(\alpha)} \to \bar{D}$.

Once an operation I_α has been assigned to each $\alpha \in V$ by an interpretation I, values in $\bar{D}$ can indeed be assigned to all the "terms" in $V_\#$ by straightforward bottom-up applications of operations. More precisely, the following lemma is proven by induction on the sizes of trees.

LEMMA 8.2. *Let I be an interpretation of V on $\bar{\mathrm{D}}$. There is a unique function*

$$val_I : V_\# \rightarrow \bar{\mathrm{D}}\ (total) \tag{1}$$

such that

$$(\forall \alpha \in V)(\forall T \in (V_\#)^{\rho(\alpha)})[val_I\bar{\alpha}(T_0, \cdots, T_{-1}) = I_\alpha(val_I T_0, \cdots, val_I T_{-1})]. \tag{2}$$

An interpretation I of V on $\bar{\mathrm{D}}$ *solves* a recursive definition L if it agrees with the meanings of the constants, parameters, and given functions and turns every instance of L into a true equation. A solution for L that agrees with all other solutions wherever it is finite is a *canonical* solution.

Definition 8.3. Let L be an RD. An interpretation I of V on $\bar{\mathrm{D}}$ is a *solution* for L iff

$$(\forall \xi \in \mathrm{D})(I_\xi = \{\langle(\ ), \xi\rangle\}), \tag{1}$$

$$(\forall a \in W \cup X)(I_a = \{\langle(\ ), \infty\rangle\}), \tag{2}$$

$$(\forall g \in G)(\forall \theta \in \bar{\mathrm{D}}^{\rho(g)})(I_g(\theta) = \mathbf{if}\ \theta \in \mathrm{dom}\, g\ \mathbf{then}\ g(\theta)\ \mathbf{else}\ \infty), \tag{3}$$

$$(\forall \phi \rightarrow \psi \in \mathbf{R}_L)(\mathrm{val}_I\phi = \mathrm{val}_I\psi). \tag{4}$$

A solution I for L is *canonical* iff every solution J for L satisfies

$$(\forall \alpha \in V)(\forall \theta \in \bar{\mathrm{D}}^{\rho(\alpha)})[I_\alpha(\theta) \neq \infty \text{ implies } J_\alpha(\theta) = I_\alpha(\theta)]. \tag{5}$$

THEOREM 8.4. Validity Theorem. *Let L be an RD and define an interpretation I of V on $\bar{\mathrm{D}}$ by setting $I_\alpha = L\alpha$ for each $\alpha \in V$. Then*

$$(\forall R \in V_\#)(val_I R = eval_L R), \tag{1}$$

$$I \textit{ is the } \text{unique } \textit{canonical solution for } L. \tag{2}$$

PROOF. We prove (1) by induction on the size $|R|$ of R. Suppose R is a tree such that $\mathrm{val}_I T = \mathrm{eval}_L T$ whenever $|T| < |R|$. We show $\mathrm{val}_I R = \mathrm{eval}_L R$. Let $\alpha = R(\ )$ and $k = \rho(\alpha)$, so that

$$R = \bar{\alpha}(T_0, \cdots, T_{-1}) \text{ with } (T_0, \cdots, T_{-1}) \in (V_\#)^k.$$

Let $\theta_i = \mathrm{val}_I T_i$ for each $i < k$. By Lemma 8.2 and Def. 7.6,

$$\mathrm{val}_I R = I_\alpha(\theta_0, \cdots, \theta_{-1}) = L\alpha(\theta_0, \cdots, \theta_{-1}) = \mathrm{eval}_L\bar{\alpha}(\tilde{\theta}_0, \cdots, \tilde{\theta}_{-1}). \tag{3}$$

By the induction hypothesis, $\theta_i = \mathrm{eval}_L T_i$ for each $i < k$. Therefore

$$\mathrm{eval}_L\tilde{\theta}_i = \mathbf{if}\ \theta_i \in \mathrm{D}\ \mathbf{then}\ \theta_i\ \mathbf{else}\ \infty = \theta_i = \mathrm{eval}_L T_i\,.$$

By k applications of part (4) of the Evaluation Theorem (Theorem 7.5), we may replace $\tilde{\theta}_0, \cdots, \tilde{\theta}_{-1}$ by $T_0, \cdots, T_{-1}$ in $\bar{\alpha}(\tilde{\theta}_0, \cdots, \tilde{\theta}_{-1})$ without changing the value of eval_L. Therefore (3) implies that

$$\mathrm{val}_I R = \mathrm{eval}_L\bar{\alpha}(T_0, \cdots, T_{-1}) = \mathrm{eval}_L R.$$

Now we must prove (2). By the definition of canonical solutions ((5) of Def. 8.3), there can be *at most one* canonical solution. Now we show that I is a canonical solution.

To show that I is a solution we must verify conditions (1)–(4) of Def. 8.3. The first three are trivial. We must show that $\mathrm{val}_I\phi = \mathrm{val}_I\psi$ for each $\phi \rightarrow \psi \in \mathbf{R}_L$ to verify (4) of Def. 8.3. To do this we apply (1) to the special case

$$(\forall\phi \rightarrow \psi \in \mathbf{R}_L)(\mathrm{eval}_L\phi = \mathrm{eval}_L\psi)$$

of part (3) in the Evaluation Theorem (Theorem 7.5). Therefore I is a solution for L.

Suppose J is also a solution. A straightforward induction shows that each $K \in \mathbf{N}$ has the property

$$(\forall R \in V_*)(\forall \xi \in \mathrm{D})(R \rightarrow^K \xi \text{ implies } \mathrm{val}_J R = \xi), \tag{4}$$

and this implies that I is canonical. ▌

Morris [14, Ch. 3, Th. 2] stated a conjecture that amounts to this theorem when RD's are defined formally in the manner of Section 7. He suggested that the theorem could be derived from a small extension of a theorem about the lambda calculus [14, Ch. 3, Th. 7]. Rather than formally relate the McCarthy calculus to the less intuitive lambda calculus, we proved the Validity Theorem directly.

Suppose that H, J are solutions for an RD L with canonical solution I. Let $f \in F$ and let $\Delta_f = \{\theta \in \bar{\mathrm{D}}^{\rho(f)} \mid Lf(\theta) \not\simeq \infty\}$. Then each $\theta \in \Delta_f$ has $I_f(\theta) \not\simeq \infty$ by the Validity Theorem and so $I_f(\theta) = H_f(\theta)$ and $I_f(\theta) = J_f(\theta)$ because I is canonical. Therefore $(\forall\theta \in \Delta_f)[H_f(\theta) = J_f(\theta)]$. This is an extension of McCarthy's principle of "recursion induction" [11, Sec. 8], which asserts only that $(\forall\theta \in \Delta_f \cap \mathrm{D}^{\rho(f)})[H_f(\theta) = J_f(\theta)]$.

The Validity Theorem is also related to Kleene's "first recursion theorem" [8, Sec. 66, Th. 26]: Any partial recursive functional $\mathfrak{F}$ has a unique minimal fixed point defined by formal calculations using the set of equations that specifies $\mathfrak{F}$. A functional here is a mapping from interpretations of V on $\bar{\mathrm{D}}$ to interpretations of V on $\bar{\mathrm{D}}$. Minimality is defined for a partial ordering $\leq$:

$$I \leq J \text{ iff } (\forall\alpha \in V)[I_\alpha \cap (\bar{\mathrm{D}}^{\rho(\alpha)} \times \mathrm{D}) \subseteq J_\alpha \cap (\bar{\mathrm{D}}^{\rho(\alpha)} \times \mathrm{D})].$$

To state the Validity Theorem in terms of functionals, we assign a functional $\mathfrak{F}_L$ to each RD L in the obvious way. It is easy to show that I is a fixed point of $\mathfrak{F}_L$ iff I is a solution for L, so that I is a minimal fixed point of $\mathfrak{F}_L$ iff I is a canonical solution for L. The Validity Theorem can therefore be restated as follows:

(*) Any functional $\mathfrak{F}$ such that $\mathfrak{F} = \mathfrak{F}_L$ for some RD L has a unique minimal fixed point I, and $I_f = Lf$ for each $f \in F$.

There are two differences between (*) and the "first recursion theorem." The first difference is that only numerals can be substituted for variables in the latter [8, Sec. 54], so that only call-by-value is allowed. There are no restrictions on call-by-name in (*). The second difference is that (*) does place syntactic restrictions on the equations that are not required in [8].

We do not consider the restriction to RD's in (*) to be very serious, since it is well known that RD's can define all the partial recursive functions in terms of successor and equality [11, Sec. 9] and that RD's are generally quite convenient for defining functions in terms of other functions [10, 11]. Under the restriction to integer data in [8], it appears likely that RD's suffice for defining all partial recursive functionals, but that has not been demonstrated.

9. *Other Applications*

In this section we remark on two other areas where the methods and results of Sections 2–6 are applicable: the lambda calculus and parallel programming.

The diagrammatic proof technique and nearly all the results from Sections 2–6 are used in [16, Ch. 4] to prove the classical Church-Rosser theorem. The full lambda calculus of Curry and Feys [4, Ch. 3] is expressed as a union of two GRS's: $\mathfrak{B}_\beta$ defined by beta reduction and $\mathfrak{B}_{\eta\delta}$ defined by eta and delta reduction. This union is Church-Rosser because $\{\mathfrak{B}_\beta, \mathfrak{B}_{\eta\delta}\}$ is a family of Church-Rosser systems that commute with each other. (This divide-and-conquer strategy was discovered by Hindley [5] and independently by the author.)

To prove that $\mathfrak{B}_\beta$ is Church-Rosser we introduce a new symbol σ and simulate a beta reduction $(\lambda xS)R >_\beta [R/x]S$ by applying a rule

$$\bar{\gamma}(\bar{\lambda}(\bar{x}, S), R) \rightarrow \bar{\sigma}(R, \bar{x}, S) \tag{1}$$

and then a set $\mathbf{R}_\sigma$ of rules such that $\bar{\sigma}(R,\bar{x},S) \rightarrow^* [R/x]S$. The set of all rules (1) is called $\mathbf{R}_\gamma$. (Essentially the same construction was suggested independently by Mitschke [13].) The proof that $\mathbf{R}_\gamma$ and $\mathbf{R}_\sigma$ correctly simulate $\mathfrak{B}_\beta$ is reasonably simple except for some extremely tedious verifications required when changes of bound variables are treated rigorously. The fact that $\mathbf{R}_\gamma$ and $\mathbf{R}_\sigma$ define Church-Rooser GRS's is then exploited to prove that $\mathfrak{B}_\beta$ is Church-Rosser. This method is more transparent than the classical approach of proving that a complex array of postulates implies the Church-Rosser property and is satisfied by $\mathfrak{B}_\beta$ [4, Ch. 4; also 5, 6, 18], but the new method of Martin-Löf [9] for proving that $\mathfrak{B}_\beta$ is Church-Rosser is far more elegant than any previous proof, including ours.[1]

The whole-part theorems from Section 3 may also be used in showing the "single-valuedness" of functions defined by algorithms or computer systems that permit asynchronous parallel processing. In the following discussion we use the notation of Karp and Miller [7]. Many parallel processing methods can be conveniently expressed as special cases of their parallel program schemata.

Let $\mathbf{B}$ be the set of all triples $\alpha = (c,q,\mu)$ for an interpreted schema, where c is an assignment of data to the memory cells, q is a state of the control, and μ is an assignment of input queues to operations. Each symbol σ in the alphabet Σ of the schema defines a relation on $\mathbf{B}$ by $\alpha \rightarrow_\sigma \beta$ iff $\alpha \cdot \sigma = \beta$ in [7, Def. 1.5]. One way to show that the final configuration of the system after a halting computation is determined by the initial configuration is to show that the GRS $(\mathbf{B}, \rightarrow) = (\mathbf{B}, \bigcup_{\sigma\in\Sigma} \rightarrow_\sigma)$ is Church-Rosser. (This is much weaker than "determinacy" as defined in [7, Def. 1.9).)

Since each $\rightarrow_\sigma$ is a partial function on $\mathbf{B}$, each $(\mathbf{B}, \rightarrow_\sigma)$ is a Church-Rosser system. The union process is associative and commutative, so there is a multitude of ways to analyze $(\mathbf{B}, \rightarrow)$ as a hierarchy of simpler systems, with the systems on each level being unions of systems on the level below. For each such analysis, Theorems 3.5 and 3.8 tell us that appropriately connected Church-Rosser parts form Church-Rosser wholes, while Lemmas 3.6 and 3.9 assist in showing that the parts are indeed appropriately connected.

We can therefore approach each specific interpreted schema with general tools

[1] *Note added in proof:* H. Barendregt has informed me that the general strategy used in [9] is based on lectures by W. W. Tait.

applicable to any union of general replacement systems. Perhaps we can also use the general tools to show that any interpreted schema satisfying certain conditions leads to a Church-Rosser system, where the conditions are reasonably easy to verify in many practical situations. No such results are known at present.

ACKNOWLEDGMENTS. I wish to thank P. J. Downey for suggesting an improvement in the definition of closed SRS's and Dr. H. Barendregt for calling my attention to [9]. Professor R. Hindley and Dr. G. Mitschke kindly supplied copies of their dissertations for comparison with earlier versions of this work. Professor Hindley also called my attention to [17] and suggested some improvements in the proof of the classical Church-Rosser theorem. Professors R. V. Book, F. D. Lewis, and W. A. Woods, Dr. H. R. Strong, and the referees all suggested many improvements in the exposition.

I am especially grateful to Professor Book, who supervised this research. His guidance and support have been beyond the power of words to acknowledge.

REFERENCES

1. BLUM, E. K. Towards a theory of semantics and compilers for programming languages. *J. Comput. Syst. Sci. 3* (1969), 248–275.
2. BRAINERD, W. S. Tree generating regular systems. *Inform. and Contr. 14* (1969), 217–231.
3. CHURCH, A., AND ROSSER, J B Some properties of conversion *Trans. Amer Math Soc. 39* (1936), 472–482.
4. CURRY, H. B., AND FEYS, R. *Combinatory Logic.* North-Holland Pub. Co., Amsterdam, 1958.
5. HINDLEY, R. The Church-Rosser property and a result in combinatory logic. Ph.D. Th., U. of Newcastle-upon-Tyne, England, 1964.
6. HINDLEY, R. An abstract form of the Church-Rosser theorem, Part I. *J. Symbolic Logic 34* (1969), 545–560.
7. KARP, R. M., AND MILLER, R. E. Parallel program schemata. *J. Comput. Syst. Sci. 3* (1969), 147–195.
8. KLEENE, S. C. *Introduction to Metamathematics.* Van Nostrand, New York, 1952.
9. MARTIN-LOF, P. A theory of types (unpublished manuscript, 1971).
10. MCCARTHY, J. Recursive functions of symbolic expressions and their computation by machine. *Comm. ACM 3*, 4 (Apr. 1960), 184–195.
11. MCCARTHY, J. Basis for a mathematical theory of computation In *Computer Programming and Formal Systems*, P. Braffort, and D. Hirschberg (Eds), North-Holland Pub. Co , 1963, pp. 33–70.
12. MINSKY, M. Form and content in computer science. *J. ACM 17*, 2 (Apr. 1970), 197–215.
13 MITSCHKE, G. Eine algebraische Behandlung von λ-*K*-Kalkul und Kombinatorischer Logik. Ph.D. Th., Rheinischen Friedrich-Wilhelms U., Bonn, Germany, 1970.
14. MORRIS, J. H. JR Lambda-calculus models of programming languages. Proj. MAC, Rep. MAC-TR-57, MIT, Cambridge, Mass., 1968.
15 NAUR, P. (Ed.) Revised report on the algorithmic language ALGOL 60 *Comm. ACM 6*, 1 (Jan. 1963), 1–17.
16. ROSEN, B. K. Subtree replacement systems. Tech Rep. 2–71, Ctr. for Res. in Computing Technology, Harvard U., Cambridge, Mass., 1971.
17. SANCHIS, L. E. Functionals defined by recursion. *Notre Dame J. Formal Logic 8* (1967), 161–174.
18. SCHROER, D. E. The Church-Rosser theorem. Ph.D. Th., Cornell U., Ithaca, N.Y., 1965.
19. STRONG, H. R. JR. Translating recursion equations into flowcharts Second Annual ACM Symp. on Theory of Computing, Northampton, Mass , 1970, 184–197.
20. THATCHER, J. W. Characterizing derivation trees of context-free grammars through a generalization of finite automata theory. *J. Comput. Syst. Sci. 1* (1967), 317–322.

RECEIVED DECEMBER 1970; REVISED DECEMBER 1971