ALGEBRAIC THEORIES AND TREE REWRITING SYSTEMS

by

Mitchell Wand

Computer Science Department

Indiana University

Bloomington, Indiana 47401

TECHNICAL REPORT NO. 66

ALGEBRAIC THEORIES AND TREE REWRITING SYSTEMS

MITCHELL WAND

JULY, 1977

REVISED: JUNE, 1978

Research reported herein was supported in part by the National Science Foundation under grant number MCS75-06678 A01

Algebraic Theories and Tree Rewriting Systems

Mitchell Wand*

Abstract: We show how operational semantics may be obtained for algebraic definitions of data types. In an algebraic definition, the axioms generate an equivalence relation on expressions to be evaluated. Under an easily satisfied Church-Rosser condition, the operational semantics consists of a tree rewriting system which rewrites an expression into a normal form which is a representative of its equivalence class. Recent results in tree rewriting systems can be used to make the operational semantics deterministic.

Key Words: Algebra semantics, initial algebra semantics, operational semantics, tree rewriting systems, Church-Rosser property, algebraic theories, data types.

CR Categories: 5.24, 5.21.

Research reported herein was supported in part by the National Science Foundation under grant number MCS75-06678 A01

*Author's address: Computer Science Department, Indiana University, Bloomington, Indiana 47401

1. Introduction

It is well-known [5] that there is a connection between algebraic semantics [2, 4, 12] and tree-rewriting systems [7, 10]. The purpose of this note is to make that connection mathematically precise.

We adopt the point of view of [2], that "abstract" data types are algebraic theories, presented as a quotient of a free theory i.e., by generators and relations.(*) One specifies a computation in a data type by a morphism in the free theory; the morphism is then mapped by the quotient map to its value. This gives a denotational semantics for any computation.

To get an operational semantics, one must deal with representatives of the equivalence classes created by the relations. Given an arbitrary morphism in the free theory, one must determine (by computation) the representative of its equivalence class. If representatives are chosen wisely (e.g. the class of computations which evaluate to "2" ought to be represented by "2"), then a computational procedure for computing representatives will serve nicely as an operational semantics. This is a case of the equivalence problem [1], except that the equivalence relations are unchanging and infinite. Examples of this kind of computation are given in [5].

(*) Alternatively, one may take the view that the generators and relations present an algebra which is a quotient of an initial algebra [4].

In this note we show how to build such an operational semantics as a tree rewriting system. Under fairly weak conditions, the tree rewriting system will have the Church-Rosser property, and we can choose the normal forms (where they exist) as representatives. This allows us to exploit the growing knowledge about such systems [7,10] to obtain additional insight.

Section 2 gives the required definitions. Section 3 presents the main result: given a set of relations Δ, we construct a tree rewriting system SM_Δ and show that two morphisms t,t' are congruent modulo Δ iff one can be transformed into the other via SM_Δ. In Section 4 we provide conditions under which the tree rewriting system has the Church-Rosser property, and show that under these conditions, representatives may be computed by a deterministic rewriting strategy. Section 5 gives some concluding remarks.

2. Preliminaries

We present algebraic preliminaries here. Definitions for tree rewriting systems are deferred to Section 4. The reader is referred to [6, 8] for tutorials on the concepts of categories and algebraic theory.

If C is a category, $C(a,b)$ denotes the set of arrows or morphisms from object a to object b. If $f \in C(a,b)$ and $g \in C(b,c)$, their composition, a member of $C(a,c)$, is denoted gf. If $f \in C(a,b)$ then $dom(f) = a$ and $cod(f) = b$. Sets will denote the category whose objects are sets and whose morphisms are the usual set-theoretic functions.

Let S be a set whose elements are called sorts. An S-sorted operator alphabet Ω is a map $\Omega\, K \to S^* \times S$ for some set K.(1) If $s \in K$, and $\Omega s = (w,a)$, we say w is the domain of s and a is the codomain of S. If S has only one element, and $w = a^n$ (where $S = \{a\}$), we say s is n-ary; Ω is then a ranked alphabet. When no ambiguity results, we will write Ω for K and write "$s \in \Omega$". We use the categorical notation and write $\Omega(w,a)$ for $\{s \in K \mid \Omega s = (w,a)\}$. If we write $\Delta \subseteq \Omega$ if Δ is the restriction of Ω to K' for some $K' \subseteq K$.

An S-sorted algebraic theory (or just theory) is a category T whose objects are the elements of S^* and in which multiplication in S^* coincides with the categorical product. If T is a theory,

---

(1) S^* denotes the free monoid generated by S; we write $|w|$ for the length of w and Λ for the null string.

and $f_i \in T(u,w_i)$ (for i = 1,..,n), then the product morphism in $T(u, w_1...w_n)$ is denoted $[f_1,..,f_n]$. We write e_i for the projection morphisms. This includes the case where the w_i are arbitrary objects (not just sorts); thus, if $f \in T(w,a)$, we write $[1_w,f]$ for the evident morphism in $T(w,wa)$. We write 1 for 1_w where no confusion results. A theory functor is a product-preserving functor between theories. A subtheory of an S-sorted theory T is a subcategory T' of T which has the same objects of T and which is closed under the tupling operation [...] of T.

If Ω is an S-sorted operator alphabet, we may construct the free theory F_Ω by the usual methods; if $s \in \Omega$, then $s \in F_\Omega(\mathrm{dom}(s), \mathrm{cod}(s))$. $F_\Omega(w,a)$ consists of trees whose nodes are elements of the operator alphabet, whose leaves are either constant operators (i.e. whose domain is Λ) or projection operators with domain w, whose root has codomain a, and in which domains and codomains match appropriately throughout the tree. Composition in F_Ω is substitution of trees for leaves labelled by projection operators.

For example, let $S=\{a,i\}$, K={val,alt} $\Omega=\{(\underline{\mathrm{val}},(ai,i)),$ $(\underline{\mathrm{alt}},(aii,a))\}$. Then

$$\underline{\mathrm{alt}}[e_1,\underline{\mathrm{val}}[e_1,e_2],\underline{\mathrm{val}}[e_1,e_3]] \in F_\Omega(aii,a)$$
$$[e_1,\underline{\mathrm{val}}[e_1,e_3],\underline{\mathrm{val}}[e_1,e_2]] \in F_\Omega(aii,aii)$$

and the composition of these two morphisms is

$$\underline{alt}[e_1,\underline{val}[e_1,\underline{val}[e_1,e_3]],\underline{val}[e_1,\underline{val}[e_1,e_2]]] \in F_\Omega(aii,a)$$

If T is an S-sorted theory, so is T^2, where $T^2(u,v) = \{(f,g) \mid f,g \in T(u,v)\}$, with composition given by $(f,g)(f',g') = (ff',gg')$. An <u>equation</u> on T is an element of $T^2(w,a)$ for some $a \in S$. A <u>congruence</u> on T is a subtheory R of T^2 such that for each $u,v \in S^*$, $R(u,v)$ is an equivalence relation on $T(u,v)$. If R is a congruence on T, then we can form the quotient theory T/R via $T/R(u,v) = T(u,v)/R(u,v)$. T/R is also an S-sorted theory; it is the coequalizer of the evident diagram $R \rightarrow T^2 \rightrightarrows T$.

If Δ is a set of equations on T we can construct the smallest Δ-containing congruence on T as the set of theorems of a formal system E_Δ. The formal objects of E_Δ are the morphisms of T^2. We write $(f,f'):u \rightarrow v$ for $(f,f') \in T^2(u,v)$, and $\vdash(f,f'):u \rightarrow v$ if (f,f') is provable in E_Δ. The axioms and rules of E_Δ are as follows:

Axioms: If $(f,f'):w \rightarrow a \in \Delta$, then $\vdash(f,f'):w \rightarrow a$ $E\Delta$

For any $f \in T(u,v)$, $\vdash(f,f):u \rightarrow v$ ER

Rules:

$$\frac{(f,g):u \rightarrow v}{(g,f):u \rightarrow v}\ ES \qquad \frac{(f,g):u \rightarrow v \quad (g,h):u \rightarrow v}{(f,h):u \rightarrow v}\ ET$$

$$\frac{(g,g):w \rightarrow y \quad (f,f'):v \rightarrow w \quad (h,h):u \rightarrow v}{(gfh,gf'h):u \rightarrow y}\ EC$$

$$\frac{(f_1,f'_1):u \rightarrow a_1,\ldots,(f_n,f'_n):u \rightarrow a_n}{([f_1,\ldots,f_n],[f'_1,\ldots,f'_n]):u \rightarrow a_1..a_n}\ EP$$

Let $E_\Delta(u,v) = \{(f,f') \mid \vdash(f,f'):u\to v\}$. Axiom scheme EΔ ensures that every equation in Δ is in E_Δ ; rules ER, ES, and ET ensure that each $E_\Delta(u,v)$ is an equivalence relation; rule EC closes E_Δ to a subcategory of T^2 , and rule EP closes E_Δ under the product operation of T^2 . Hence E_Δ , with composition inherited from T^2 , is the smallest congruence on T containing Δ .

A theory may be presented by (Ω,Δ) where Ω is an operator alphabet (the <u>generators</u>) and Δ is a set of equations. (Ω,Δ) presents the theory T where $T(u,v) = F_\Omega(u,v)/\ E_\Delta(u,v)$. The functor $F:T_\Omega \to T$ sending each morphism to its equivalence class is a full theory functor.

Thus an operational semantics for the theory presented by (Ω,Δ) is an algorithm which, given a morphism f in F_Ω , finds a representative f' of the E_Δ-equivalence class of f , i.e. an f' such that

(1) f' is a representative

(2) $E_\Delta \vdash (f,f')$.

E_Δ is not a convenient deduction system in which to work. Our first concern, therefore, is to find computationally convenient deduction systems equivalent to E_Δ .

3. Equivalence of Deduction Systems

For this section, let T be an arbitrary S-sorted theory, and Δ be a set of equations on T. Elements of S will be denoted by $a,b,..,$ and elements of S^* by $u,v,w,..$.

Definition: Let sm_Δ, the operator alphabet of single Δ-moves, be defined by:

$$sm_\Delta(w,a) = \{(f[1,gh],\ f[1,g'h]) \mid \text{for some } w' \in S^*, b \in S, h \in T(w,w'),\ g,g' \in T(w',b),\ f \in T(wb,a), \text{ and } (g,g') \in \Delta\}$$

This definition is motivated by the ideas of Rosen [10]. To explain the connection, consider the case in which T is a free theory, and think of $sm_\Delta(w,a)$ as a general replacement system on $T(w,a)$. Then a typical element of $sm_\Delta(w,a)$ is shown in Figure 3.1. An occurrence of gh , attached to the tree f by e_n $(n = |w|+1)$, is replaced by $g'h$. Thus, $sm_\Delta(w,a)$ corresponds to the subtree replacement system on $T(w,a)$ given by the rules $\{(gh,g'h) \mid (g,g') \in \Delta,\ dom(h) = w\}$ (Rosen's component $\underline{R}$ [10, Def. 5.1]. If Δ is a set of rule-schemata [10, Def. 6.1], then this set of ordered pairs is the associated set of instances. Thus $sm_\Delta(w,a)$ allows rewriting of any left-hand-side of Δ occurring anywhere in a tree in $T(w,a)$.

The correspondence fails to be exact, however in two ways:

(1) e_n may occur in f more than once. In that case,

multiple copies of gh are rewritten in one step. Clearly, any single copy of gh can be rewritten by a suitable choice of f, and any multiple rewriting can be simulated by several steps of the corresponding subtree replacement system. Thus this difference is not crucial unless one is counting steps [7, Sec. 4].

2. $sm_\Delta(w,a)$ makes sense even if Δ is not a set of rule-schemata in Rosen's sense; indeed the results of this section hold even when T is not a free theory. Conversely, Rosen's notion of the domain of a parameter is more general than ours.

We next construct a formal system SM_Δ. The formal objects of SM_Δ are the morphisms of T^2, and the axioms and rules are as follows:

Axioms: If $(f,f') \in sm_\Delta(w,a)$, then $\vdash (f,f'):w \to a$ SMΔ

If $f \in T(w,a)$, then $\vdash (f,f):w \to a$ MR

Rules: $$\frac{(f,g):w \to a}{(g,f):w \to a}\ \text{MS} \qquad \frac{(f,g):w \to a \quad (g,h):w \to a}{(f,h):w \to a}\ \text{MT}$$

$$\frac{(f_1,g_1):w \to a_1 \ ,\ldots,\ (f_n,g_n):w \to a_n}{([f_1,\ldots,f_n],[g_1,\ldots,g_n]):w \to a_1\ldots a_n}\ \text{MP}$$

Let $SM_\Delta(u,v) = \{(f,f') \mid SM_\Delta \vdash (f,f'):u \to v\}$. The main theorem of this section is that for all $u,v \in S^*$, $SM_\Delta(u,v) = E_\Delta(u,v)$. The proof proceeds by a series of lemmas.

Lemma 3.1. $SM_\Delta \vdash (f,f'):u \to v$ iff for all i, $1 \le i \le |v|$, $SM_\Delta \vdash (e_i f, e_i f')$.

Proof: If $|v| = 1$, then the result is trivial. If $|v| > 1$, (f,f') can be deduced only by an application of rule MP. Therefore $(e_i f, e_i f')$ must already have been deduced. ∎

Lemma 3.2. If $a \in S$ and $SM_\Delta \vdash (f,f'):w \to a$, then there is a derivation of (f,f') in SM_Δ which does not use rule MP.

Proof: If MP is used in the derivation of (f,f') and n (the number of hypotheses of MP) is 1, then it may be eliminated, since the hypothesis of the rule is the same as its conclusion. If MP is used in the derivation of (f,f') and $n>1$, then it must be the last step in the derivation, since no rule of SM_Δ is applicable to its conclusion. But this is impossible, since $|a| = 1$. ∎

Lemma 3.3. If $SM_\Delta \vdash (f,f'):w \to a$, $k \in T(u,w)$ $h \in T(w,v)$, and $g \in T(av,b)$, then $SM_\Delta \vdash (g[f,h]k, g[f',h]k):u \to b$.

Proof: By induction on the proof of (f,f') in SM_Δ . If (f,f') is an axiom in SM_Δ , so is $(g[f,h]k, g[f',h]k)$, by easy calculation. If the last step in the proof of (f,f') is

$$\frac{(f,f'') \qquad (f'',f')}{(f,f')} \quad \text{MT}$$

then, by the induction hypothesis, $SM_\Delta \vdash (g[f,h]k, g[f'',h]k):w \to b$

and $SM_\Delta \vdash (g[f'',h]k, g[f',h]k):w \to b$. Hence, by MT, $SM_\Delta \vdash (g[f,h], g[f',h])$.

MS is similar; occurrences of MP are trivial, since $|cod(f)| = 1$. ∎

Theorem 3.1 $E_\Delta = SM_\Delta$

Proof (i) $SM_\Delta \subseteq E_\Delta$: Every axiom of SM_Δ is provable in E_Δ, and every rule in SM_Δ is a rule in E_Δ. Therefore $SM_\Delta \subseteq E_\Delta$.

(ii) $E_\Delta \subseteq SM_\Delta$. The proof is by induction on proof in E_Δ; we show inter alia how to convert a proof in E_Δ into a proof in SM_Δ. The base step is that every axiom of E_Δ is an axiom of SM_Δ. We now have one case in the induction step for each rule of E_Δ.

(ES): If $SM_\Delta \vdash (f,g):u \to v$, then for each i, $1 \leq i \leq |v|$, $SM_\Delta \vdash (e_i f, e_i g):u \to v_i$ (where v_i is the i-th letter of v). Hence by MS, $SM_\Delta \vdash (e_i g, e_i f)$. Then by MP, $SM_\Delta \vdash (g,f)$.

(ET): Similar to ES

(EP): Rule EP is identical to rule MP; no change necessary

(EC): We must show that if $SM_\Delta \vdash (f,f'):v \to w$, $g \in T(w,y)$, $h \in T(u,v)$, then $SM_\Delta \vdash (gfh, gf'h):u \to y$. By Lemma 3.1, we may assume without loss of generality that $|y| = 1$.

If $|cod(f)| = 1$, then the result follows by Lemma 3.3. If $|w| = n>1$, then by Lemma 3.1 we have $SM_\Delta \vdash (e_i f, e_i f')$ for $1 \le i \le n$. Now construct morphisms $q_j \in T(v,w)$ for each $j, (0 \le j \le n)$, so that for each i $(1 \le i \le n)$,

$$e_i q_j = \begin{cases} e_i f & i<j \\ e_i f' & i \ge j \end{cases}$$

Thus the q_j have the property that exactly one component of the product changes from $e_i f$ to $e_i f'$ between consecutive q_j's. Hence Lemma 3.3 applies, and for all j, $0 \le j < n$, $SM_\Delta \vdash (gq_j h, gq_{j+1} h)$. By repeated applications of MT, $SM_\Delta \vdash (gq_0 h, gq_n h)$. But $q_0 = f$ and $q_n = f'$. So $SM_\Delta \vdash (gfh, gf'h)$. This completes the induction step. ∎

Note that by Lemma 3.2, $SM_\Delta(w,a)$ is the reflexive, symmetric, transitive closure of $sm_\Delta(w,a)$. Thus $(f,f') \in E_\Delta(w,a)$ iff there exist $f_0,..,f_n \in T(w,a)$ such that $f = f_0$, $f_n = f'$, and for each i, either (f_i, f_{i+1}) or (f_{i+1}, f_i) belongs to $sm_\Delta(w,a)$.

In sm_Δ only one node of the tree (plus copies) is rewritten at each step. In dealing with tree rewriting systems, it is useful to allow multiple, parallel rewriting at each step. For this we define the operator alphabet of Δ-<u>moves</u> and a formal system M_Δ :

Definition: The operator alphabet m_Δ of Δ-moves is defined by $m_\Delta(w,a) = \{(f[1,k], f[1,k']) \mid$ for some $u,v \in S^*$, $k,k' \in T(w,u)$, $f \in T(wu,a)$, and for each i, $1 \le i \le |u|$, there exist g,g' and h s.t. $e_i k = gh$, $e_i k' = g'h$, and $(g,g') \in \Delta\}$

Thus instead of a single rewriting (as in SM_Δ), we have parallel rewritings in each component of k.

The formal system M_Δ is defined in the same way as SM_Δ, except that the axiom scheme SM_Δ is replaced by:

If $(f,f') \in m_\Delta(w,a)$, then $\vdash (f,f'):w \to a$ $M\Delta$

Theorem 3.2: $M_\Delta = SM_\Delta = E_\Delta$

Proof: Every axiom in SM_Δ is an axiom in M_Δ; every axiom in M_Δ is provable (via rule MT only) in SM_Δ. The rest is the previous theorem. ∎

Corollary 3.1: For any $w \in S^*$, $a \in S$, $E_\Delta(w,a)$ is the reflexive, symmetric, transitive closure of $m_\Delta(w,a)$ or of $sm_\Delta(w,a)$.

Proof: Theorem 3.2 and Lemma 3.2. ∎

As a result, each $m_\Delta(w,a)$ may be treated as a separate equivalence problem, without worrying about substitution instances or any other morphism set.

4. The Church-Rosser Property

In the previous section we showed how $E_\Delta(w,a)$ could be characterized as the reflexive, symmetric, transitive closure of a set of instances. This characterization does not yield a natural choice of representatives. If, however, we can avoid the symmetric closure, then a natural choice appears: we choose as representatives those morphisms on which no further rewrites may be made. Such a plan is possible if m_Δ (or sm_Δ) has the Church-Rosser property. We first show why this simplification is possible, and then sketch conditions under which m_Δ (or sm_Δ) has the Church-Rosser property.

Let A be a set, $R \subseteq A \times A$, and let R^* be the reflexive transitive closure of R. We say R has the Church-Rosser property iff for every $x,y,z \in A$, if $(x,y) \in R^*$ and $(x,z) \in R^*$, there is a $t \in A$ such that $(y,t) \in R^*$ and $(z,t) \in R^*$. We say t is an R-normal form of x iff $(x,t) \in R^*$ and there is no u such that $(t,u) \in R$. If R has the Church-Rosser property, then every $x \in A$ has at most one R-normal form. Let $R\#$ denote the reflexive, symmetric, transitive closure of R.

Theorem 4.1: If R has the Church-Rosser property, $(x,y) \in R\#$ and y is R-normal, then $(x,y) \in R^*$.

Proof: It is well-known that if R is Church-Rosser and $(x,y) \in R\#$, then there is a z such that $(x,z) \in R^*$ and $(y,z) \in R^*$ (e.g. [7, Thm. 2.1.1]). If y is R-normal, then $z = y$. ∎

In combination with Corollary 3.1, this gives us the basic result connecting theories and rewriting systems:

Theorem 4.2: Let T be an S-sorted theory, let Δ be a set of equations on T, $w \in S^*$, and $a \in S$. If $m_\Delta(w,s)$ has the Church-Rosser property and $f \in T(w,a)$ is equivalent modulo E_Δ to some $m_\Delta(w,a)$-normal morphism f', then $(f,f') \in [m_\Delta(w,a)]^*$.

Proof: Corollary 3.1 and Theorem 4.1. ∎

Therefore, if $m_\Delta(w,a)$ has the Church-Rosser property, we can choose $m_\Delta(w,a)$-normal forms as representatives (when they exist), and compute the representatives of an arbitrary morphism $f \in T(w,a)$ by rewriting, using $m_\Delta(w,a)$. Morphisms without normal forms yield non-terminating calculations. This is the mathematical result which underlies the "direct implementation" of [5].

The difference between this and Corollary 3.1 is that the Church-Rosser property relieves the interpreter of the obligation to consider symmetric closure. In the absence of the Church-Rosser property, the interpreter, seeking to find an $m_\Delta(w,a)$-normal form, would be obliged to apply the rules from right to left as well as left to right.(*) The OBJ interpreter does just that; in order to avoid repetitions it keeps a table of all trees obtained during the current calculation [3]. By using theorem 4.2, we can obtain an algorithm as follows:

(*) Indeed, in the absence of the Church-Rosser property, $m_\Delta(w,a)$-normal forms may not even be representatives: an equivalence class may contain more than one normal form.

Algorithm 4.1:

Step 1. Initialize a queue with the input tree f.

Step 2. Remove a tree from the front of the queue. Apply all possible sm_Δ (or m_Δ) moves to it. If any of the resulting trees are normal, stop. Otherwise insert them at the rear of the queue and go to step 2.

Thus we trade generality for space. Luckily, if T is free theory, and the morphisms of T(w,a) are just trees, then under fairly weak conditions, each $m_\Delta(w,a)$ has the Church-Rosser property. The following definition and theorem are translations of [10].

Definition. Let Ω be an S-sorted operator alphabet, and let Δ be a set of equations on F_Ω. We say Δ has property R iff

(i) if $(f,f') \in \Delta$, then f' has no variable symbols not in f, and f is not a projection.

(ii) if $(f,f') \in \Delta$, then f has no repeated variables.

(iii) if (f,f'), $(g,g') \in \Delta$, and for some h and h', $fh = gh'$, then $f=g$ and $f'=g'$ (no common substitution instances) and (iv) if $(f,f') \in \Delta$ and $(k,k') \in \Delta$ and there exist h,p s.t. $fh = g[1,kp]$, then there exists an r s.t. $g=fr$. (non-overlapping).

Theorem 4.3 (Rosen[10]) if Δ has property R, then $sm_\Delta(w,a)$ has the Church-Rosser property for each $w \in S^*$ and $a \in S$.

Proof: Conditions (i) and (ii) assert that Δ is a set of

rule-schemata [10, Def. 6.1]. Condition (iii) implies condition (1) of [10, Thm. 6.5] and also implies that the set of instances of Δ is unequivocal. Condition (iv) is our transcription of condition (2) of [10, Thm. 6.5]. So the result follows by Theorems 6.5 and 5.6 of [10]. ∎

Corollary 4.1. If Δ has property R, then $m_\Delta(w,a)$ has the Church-Rosser property for each $w \in S^*$ and $a \in S$. ∎

In fact, most sets of axioms, such as those arising from metacircular interpreters [9], have property R. We may combine these theorems to get:

Corollary 4.2. Let Ω be an S-sorted operator alphabet, and let Δ be a set of equations with property R. If $f \in F_\Omega(w,a)$ is equivalent modulo E_Δ to some $m_\Delta(w,a)$ - normal morphism f', then $(f,f') \in [m_\Delta(w,a)]^*$. ∎

Thus, if Δ has property R, then Algorithm 4.1 may be used to compute representatives for the theory presented by (Ω,Δ). In fact, we can state a stronger result.

Definition Let Ω be an S-sorted operator alphabet and let Δ be a set of equations. A move $(f,f') \in m_\Delta(w,a)$ is outermost if each rewrite site has no proper ancestor which is a rewrite site; it is parallel outermost if every outermost rewrite site is rewritten.

Lemma 4.1 If Δ has property R, for each $f \in F_\Omega(w,a)$ there exists at most one parallel outermost move $(f,f') \in m_\Delta(w,a)$.

Proof By condition (iv). ∎

Let omr(f) denote this unique f' , if it exists.

Theorem 4.4 (O'Donnell [7]) Let Ω be an S-sorted operator alphabet, and let Δ be a set of equations with property R . If $f \in F_\Omega(w,a)$ is equivalent modulo E_Δ to some $m_\Delta(w,a)$-normal morphism f' , then f' is obtainable from f by some sequence of parallel outermost moves.

Proof If Δ has property R , then Δ is outer [7, Def. 5.2.2]. Hence Theorems 5.2.2 and 3.4.1 of [7] apply. ∎

By Lemma 4.1, this gives a deterministic rewriting strategy which is complete:

Algorithm 4.2

```
x:=f;    (the input morphism)
while ~ normal(x) do
    x:=omr(x);
output x
```

Note that the theorems of substance occurred in Section 3. Having reduced E_Δ to the tree rewriting systems M_Δ or SM_Δ , in this section we needed only to apply known results on tree rewriting systems.

5. Concluding Remarks

We have shown how the equivalence problem for E_Δ may be reduced, in many useful cases, to a deterministic transitive closure problem which may be solved using Algorithm 4.2. We believe that the reductions of Section 3 may be independently useful for proof-theoretic analyses of more complicated algebraic systems.

It is also worth noting that from the point of view of operational semantics, there is little difference between single-sorted theories and many-sorted theories. Given an S-sorted presentation (Ω,Δ), one can obtain an 1-sorted presentation by identifying each object $w\epsilon S^*$ with its length. Rewriting in the two theories proceeds in exactly the same fashion. The only difference is that there will be some trees which are legal in the 1-sorted theory, but are not morphisms in the S-sorted theory (they give each operator the right number of arguments, but some of the types are wrong). Rewriting on these trees proceeds until the type error is discovered, at which time rewriting becomes blocked. The net effect is that "compile-time" errors in the S-sorted theory turn into "run-time" errors in the 1-sorted theory.

References

1. Galler, B. A. and Fischer, M. J. An improved equivalence algorithm. Comm. ACM 7 (1964), pp. 301-303.

2. Goguen, J. A. Correctness and equivalence of data types. Mathematical Systems Theory, (Udine, 1975) (G. Marchesihi and S. K. Mitter, eds.) Lecture Notes in Economics and Mathematical Systems, Vol. 131, Springer, 1976, pp. 352-358.

3. Goguen, J. A. Abstract errors for abstract data types. Proc. IFIP Working Conference on Formal Description of Programming Language Concepts (St. Andrew's, Canada, 1977), pp. 21.1-21.32.

4. Goguen, J. A., Thatcher, J. W. and Wagner, E. G. An initial algebra approach to the specification, correctness, and implementation of abstract data types. IBM Research Report RC 6487 (1977).

5. Guttag, J. V., Horowitz, E., and Musser, D. R. Abstract data types and software validation. University of Southern California, Information Sciences Institute Research Report ISI/RR-76-48 (August, 1976).

6. MacLane, S. Categories for the Working Mathematician Springer-Verlag, New York, 1971.

7. O'Donnell, M. Subtree replacement systems: a unifying theory for recursive equations, LISP, Lucid, and combinatory logic. Proc. 9th ACM Symp. on Theory of Computing (Boulder, Co., 1977), pp. 295-305.

8. Pareigis, B. Categories and Functors Academic Press, New York, 1970.

9. Reynolds, J. C. Definitional interpreters for higher-order programming languages. Proc. ACM Nat'l. Conf. (1972), pp. 717-740.

10. Rosen, B. K. Tree manipulating systems and Church-Rosser theorems. J. ACM 20 (1973), pp. 160-187.

11. Vuillemin, J. Correct and optimal implementations of recursion in a simple programming language. J. Comp. Sys. Sci. 9 (1974), pp. 332-354.

12. Wand, M. Final algebra semantics and data type extensions. Indiana University, Computer Science Department, Technical Report #65 (July, 1977).

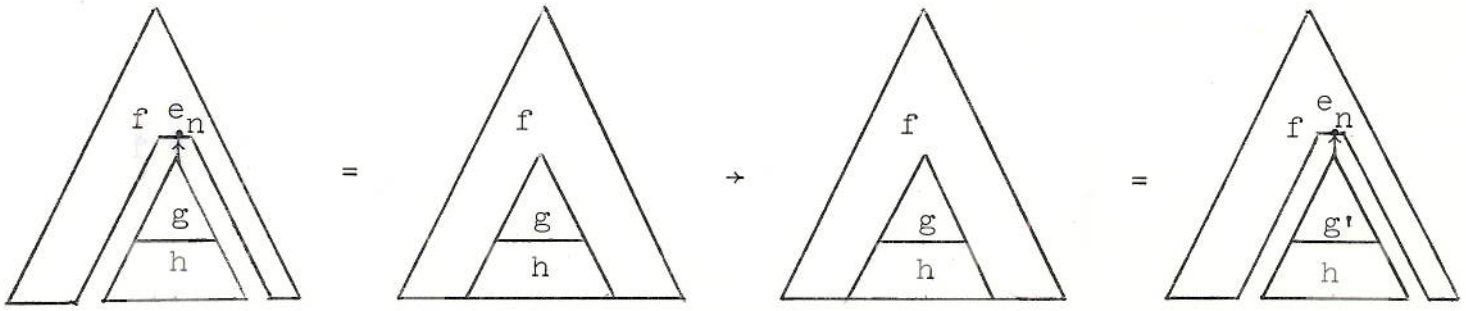

Figure 3.1 The single move f[1,gh]→f[1,g'h]