IBM Research

A CHURCH-ROSSER PROPERTY OF CLOSED APPLICATIVE LANGUAGES

Paul McJones

May 23, 1975

RJ 1589
A CHURCH-ROSSER PROPERTY OF CLOSED APPLICATIVE LANGUAGES

by

Paul McJones

IBM Research Laboratory
San Jose, California 95193

ABSTRACT: In "Programming Language Semantics and Closed Applicative Languages" [IBM Research Report RJ1245], John Backus defines the class of closed applicative languages (of which his Red is a member), and states some of their properties. One of the most interesting is a Church-Rosser property: if an expression has a meaning, then every terminating sequence of reductions on it yields that meaning. The purpose of this report is to show how to prove this and other properties.

The approach will be to first construct the meaning function (mapping expressions into values) for a closed applicative language as the least fixed point of a continuous functional, thus establishing the "mathematical" semantics of the language. Then the "operational" notion of reduction will be defined in terms of the meaning function, and some of its properties verified using structural and computational induction.
1. Some machinery from the fixed point theory is necessary. [See for example: Manna, Z., Ness, S., and Vuilleman, J. Inductive methods for proving properties of programs. *CACM* 16, 8 (August 1973), 491-502.] A partial ordering \( \leq_S \) on a set \( S \) is a relation with the following three properties:

- reflexivity: \( a \leq_S a \)
- transitivity: \( a \leq_S b, b \leq_S c \implies a \leq_S c \) for all \( a, b, c \in S \)
- antisymmetry: \( a \leq_S b, b \leq_S a \implies a = b \) for all \( a, b, c \in S \)

A least upper bound for a subset \( X \) of a partially ordered set \([poset]\) \( S \) is an element \( a \in S \) such that:

1) \( x \leq_S a \) for all \( x \in X \)

and ii) \( a \leq_S b \) for all \( b \in S \) such that for all \( x \in X, x \leq_S b \).

If \( X \) has a least upper bound, we denote it by \( \text{lub}_S X \). A poset \( S \) is complete if it has a least element [usually denoted \( \mathbb{1}_S \) or, somewhat ambiguously, just \( 1 \)] and if \( \text{lub}_S L \) exists for every nonempty totally ordered \( L \subseteq S \).

If \( f: S_1 \rightarrow S_2 \), for complete posets \( S_1, S_2 \), then \( f \) is continuous if \( \text{lub}_S \{ f_{|L} : L \subseteq S_1 \} \) exists and equals \( f(\text{lub}_S L) \) for every nonempty totally ordered \( L \subseteq S_1 \). The set of all continuous functions from \( S_1 \) to \( S_2 \), for given complete posets \( S_1 \) and \( S_2 \), can itself be given a complete poset structure. We let \( f \leq g \) iff \( f(a) \leq_{S_2} g(a) \), for all \( a \in S_1 \). The least element is \( \Omega \), defined by \( \Omega(a) = \mathbb{1}_{S_2} \) for all \( a \in S_1 \).

A set \( S \) is a trivial complete poset if there exists an element \( \mathbb{1}_S \in S \) and a relation \( \leq_S \) satisfying \( a \leq_S a' \) iff \( a = \mathbb{1}_S \) or \( a = a' \), for all \( a, a' \in S \).

Suppose \( f: S_1^{n} \rightarrow S_2 \) for some trivial complete poset \( S_1 \), any complete poset
S_2$, and some n≥0. Then a condition sufficient to ensure f continuous is that f be \( i \)-preserving, that is, \( f(s_1,s_2,\ldots,s_n) = i_{S_2} \) whenever any \( s_i = i_{S_1} \) (\( s_i \in S_1 \), \( 1 \leq i \leq n \)).

2. Let \( E \) be a fixed set of expressions. Both "programs" and "meanings" of our closed applicative language [CAL] will lie in this set. We assume \( E \) is a trivial complete poset; intuitively \( i_E \) represents the result of a nonterminating evaluation.

3. The pair \( \langle A,K \rangle \) is a **constructor syntax** for \( E \) if the following hold [the idea is to give an "abstract syntax", postulating only those properties relevant to our subsequent discussion without specifying a particular representation]:

   CS0) \( i \in E \)

   CS1) \( a \in A \) [\( a \in A \) is called an *atom*]

   CS2) For each \( k \in K \), there is an integer \( n \geq 0 \) such that \( k \) is a \( i \)-preserving function from \( E^n \) into \( E \). [\( k \in K \) is called an \( n \)-place *constructor*]

   CS3) For every \( e \in E \), **exactly one** of the following holds:

   i) \( e = i \) [in which case \( e \) has no *components*]

   ii) \( e \in A \) [in which case \( e \) has no *components*]

   iii) there is a unique \( k \in K \) [hence, by CS2, a unique \( n \geq 0 \)] and unique \( e_1, \ldots, e_n \in E - \{ i \} \) such that \( e = k(e_1, \ldots, e_n) \) [in which case the *components* of \( e \) are \( e_1, \ldots, e_n \)]

   CS4) No expression has an infinite chain of components [so repeated applications of CS3 eventually results in atoms and 0-place constructors].
CS4 assures that we can make proofs by **structural induction**: if a property P holds for every expression e whenever it holds for all components of e, then P must hold for all expressions in E. [Since \( \bot \) and atoms have no components, P must hold for them.]

4. A specific **closed applicative language** is determined by two things:
   i) a specific constructor syntax \(<A,K>\) with a distinguished 2-place constructor \(\text{apcK}\) [called the **application** constructor]
   ii) a **representation function** \(\rho\) which is a function from E, to functions from E to E. To assure continuity, a legitimate \(\rho\) must satisfy:
      \[\rho_1) (\rho_\bot) e = \bot E \text{ for all } e \in E \text{ [i.e., } \rho_\bot = \Omega, \text{ the everywhere}-\bot \text{ function}]\]
      \[\rho_2) (\rho e) \bot E = \bot E \text{ for all } e \in E \text{ [i.e., } \rho e \text{ is always } \bot \text{-preserving}].\]

5. The **meaning function** for CALs is best given recursively, defining the meaning of a composite expression in terms of the meanings of its components. The fixed point theory provides a sound basis for such definitions: every continuous function \(f\) from a complete poset \(S\) into itself has a unique least fixed point \(p = \text{lub}_S \{i_S, f_i_S, f(f_i_S), \ldots\}\) [that is, \(fp = p\), and if \(fq = q\), then \(p \leq_S q\), all \(q \in S\)]. To see why this is useful, consider the following definition of a functional \(\tau\) mapping the set of continuous E to E functions into itself. [Notational convention: function applications associate to the right, so \(fghx = f(g(hx))\).] For every continuous E to E function \(f\) and for every \(e \in E\), let
4

\[ (\tau f)e \overset{\text{def}}{=} \begin{cases} 
1_E & \text{if } e = 1_E \\
e & \text{if } e \in \mathcal{A} \\
k(f(e_1, \ldots, e_n)) & \text{if } e = k(e_1, \ldots, e_n) \text{ and } k \neq \text{ap} \\
f(pfe_1)f_{e_2} & \text{if } e = \text{ap}(e_1, e_2) 
\end{cases} \]

It is tedious but not impossible to show that \( \tau \) is indeed continuous. [As discussed in Manna et al., this follows from its composition of known continuous functions and the function parameter \( f \).] Now we define \( \mu = f_\tau \) the unique least fixed point of \( \tau \) to be the meaning function of our CAL, and claim that it has the properties stated in Backus' report. To substantiate this claim, we will use the principle of computational induction: if a property \( P \) is true of \( \Omega \) [the everywhere-\( i \) function], and if \( P \) being true of \( f \) implies that \( P \) is true of \( \tau f \), then \( P \) must be true of \( f_\tau \), the least fixed point of \( \tau \). [Technically, \( P \) must be an admissible predicate, as discussed by Manna et al.]

6. The subset \( C \) of \( E \) constructed without \( \text{ap} \) we call constants [as justified by proposition 3 below]. We can define \( C \) recursively as follows:

\begin{enumerate}
\item \( A \subset C \)
\item \( \text{If } k \in K, k \neq \text{ap}, \text{ and } e_1, \ldots, e_n \in C, \text{ then } k(e_1, \ldots, e_n) \in C \)
\item \( e \in C \) only by virtue of \( C1, C2 \) above.
\end{enumerate}

7. Based on the definition of \( \mu \), we can define a relation \( R \) of reducibility. An expression \( e \) is [directly] reducible to \( e' \), \( eRe' \), iff:

\begin{enumerate}
\item \( e = \text{ap}(c_1, c_2) \) and \( e' = (pc_1)c_2 \), for some \( c_1, c_2 \in C \)
\item or
\end{enumerate}
R2) \( e = k(e_1, \ldots, e_i, \ldots, e_n) \) and \( e' = k(e_1, \ldots, e'_i, \ldots, e_n) \) and \( e \quad \text{Re}_{i} \), for some \( k \in K \) and some \( i, 1 \leq i \leq n \).

Intuitively \( e \quad \text{Re}_{i} \) means \( e' \) results from \( e \) by "performing some innermost application".

8. We will need an additional function \( \delta \) mapping expressions into the trivial complete poset \( N = \{ i_N, 0, 1, 2, \ldots \} \) [with the obvious \( \leq_N \)].

Intuitively, \( \delta e \) gives the number of reductions needed to evaluate \( e \). Using the fixed point theory, we take \( \delta \) to be \( f_0 \), the unique least fixed point of the [reputedly] continuous functional \( \sigma \) defined as follows: For every continuous \( E \) to \( E \) function \( f \) and for every \( e \in E \), let

\[
(\sigma f)e \stackrel{\text{def}}{=} \begin{cases} 
1_N & \text{if } e = i_E \\
0 & \text{if } e \in A \\
f(e_1) + \cdots + f(e_n) & \text{if } e = k(e_1, \ldots, e_n), k \neq \text{ap} \\
1 + f(e_1) + f(e_2) + f(\rho e_1) \mu e_2 & \text{if } e = \text{ap}(e_1, e_2)
\end{cases}
\]

The propositions below give various properties of \( \mu \), \( R \), and \( \delta \), culminating in the "Church-Rosser property" for CALs.
Prop. 1. For all \( e \in E \), \( \mu e \in Cu\{\mu\} \).

Proof. We use computational induction.

**Basis:** \( \Omega e = 1 \) for all \( e \in E \), by definition of \( \Omega \).

**Induction:** Assume that \( fe \in Cu\{\mu\} \) for all \( e \in E \). We show that \( (\tau f)e \in Cu\{\mu\} \) for all \( e \in E \). By definition CS3, there are three distinct cases.

- **Case 1:** \( e = \lambda \). Then \( (\tau f)e = \lambda \) by definition of \( \tau \).
- **Case 2:** \( e \in A \). Then \( (\tau f)e = e \), and \( e \in C \) by definition Cl.
- **Case 3:** \( e = k(e_1, \ldots, e_n) \) for a unique \( n \)-place \( k \in K \) and unique \( e_1, \ldots, e_n \in E \). Two subcases arise.
  1) \( k \not\in ap \). Then \( (\tau f)e = k(fe_1, \ldots, fe_n) \) by definition of \( \tau \), and by induction \( fe \in Cu\{\mu\} \). Thus by definitions C2 and CS2, \( (\tau f)e \in Cu\{\mu\} \).
  2) \( k \in ap \), so \( e = ap(e_1, e_2) \). Then \( (\tau f)e = f(pfe_1)fe_2 \) by definition of \( \tau \). By induction, \( f(pfe_1)fe_2 \in Cu\{\mu\} \).

By computational induction \( \mu e \in Cu\{\mu\} \), where \( \mu = \) least fixed point of \( \tau \).

Prop. 2. a) \( \mu \lambda = \lambda \)

b) \( \mu a = a \), all \( a \in A \)

c) \( \mu k(e_1, \ldots, e_n) = k(\mu e_1, \ldots, \mu e_n) \), \( k \not\in ap \)

d) \( \mu ap(e_1, e_2) = \mu (\mu e_1) \mu e_2 \)

Proof. Use the fact that \( \mu = \tau \mu [\mu \text{ is a fixed point of } \tau] \), together with the definition of \( \tau \).
Prop. 3. For all \( e \in \mathcal{C}_\mu \), \( \mu e = e \).

Proof. We show that if for every component \( e' \) of \( e \), \( e' \in \mathcal{C}_\mu \) implies \( \mu e' = e' \), then \( \mu e = e \) for all \( e \in \mathcal{C}_\mu \). The proposition follows by structural induction. Definition CS3 leads to three distinct cases.

Case 1: \( e = 1 \). Then \( \mu e = e \) by proposition 2a.

Case 2: \( e \in A \). Then \( \mu e = e \) by proposition 2b.

Case 3: \( e = k(e_1, \ldots, e_n) \) for a unique \( n \)-place \( k \in \mathcal{K} \) and unique \( e_1, \ldots, e_n \in \mathcal{E} \), and we assume the proposition holds for each \( e_i, 1 \leq i \leq n \). If \( k = \alpha p \) then \( e \in \mathcal{C} \), so assume \( k \neq \alpha p \). By definition Cl-C3, if \( e \in \mathcal{C} \) then each \( e_i \in \mathcal{C}, 1 \leq i \leq n \). Then by induction \( \mu e_i = e_i, 1 \leq i \leq n \). Thus \( \mu e = k(\mu e_1, \ldots, \mu e_n) = k(e_1, \ldots, e_n) = e \).

Prop. 4. \( \mathcal{C}_\mu \) is the set of fixed points of \( \mu \), so \( \mu = \mu \circ \mu \) [\( \mu \) is idempotent].

Proof. By proposition 3, \( \mu \) and the elements of \( \mathcal{C} \) are all fixed points of \( \mu \). Conversely, if \( e = \mu e \) is a fixed point of \( \mu \), then by proposition 1, \( e \in \mathcal{C}_\mu \). Thus \( \mu \), whose range is equal to the set of its own fixed points, is idempotent.

Prop. 5. \( e \) is reducible [for some \( e', e \in e' \)] if and only if \( e \in \mathcal{C}_\mu \).

Proof. The proof is by structural induction. We must show that the proposition holds for each expression, provided that it holds for all components of the given expression. By definition CS3, there are three distinct cases to consider.

Case 1: \( e = 1 \). By inspection of definition R1-R2, \( e \) is not
reducible. Further, $e \in C \cup \{i\}$. Thus $e$ satisfies the proposition.

**Case 2:** $e \in A$. [Same as case 1 above.]

**Case 3:** $e = k(e_1, \ldots, e_n)$, for a unique $n$-place $k \in K$ and unique $e_1, \ldots, e_n \in E$. There are two subcases of interest.

i) $k \neq \text{ap}$. Suppose $e$ is reducible [we show $e \notin C \cup \{i\}$]. By definition R1-R2, some component $e_i$ of $e$ is also reducible. By induction, $e_i \notin C \cup \{i\}$. Thus by definition C1-C3, $e \notin C$ [and of course $e \neq i$ by assumption].

Conversely, suppose $e \in C \cup \{i\}$ [we show $e$ is reducible]. Inspection of definition C1-C3 shows we must have $e_i \notin C$ for some component $e_i$ of $e$. Further, $e_i \neq i$ since $k$ is $i$-preserving [definition CS2]. By induction, $e_i \in R_i$ for some $e' \in E$, so $e \in R_k(e_1, \ldots, e_i', \ldots, e_n)$ follows from definition R2.

ii) $k = \text{ap}$, so $e = \text{ap}(e_1, e_2) \neq i$. Then $e \notin C \cup \{i\}$, so we must find $e' \in E$ with $e \in R e'$. If both $e_1 \in C$ and $e_2 \in C$, definition R1 gives us $e \in R(\varepsilon e_1) e_2$. So suppose $e_1 \notin C$. By definition CS2, $e_1 \neq i$ is impossible, so by induction there exists $e_1' \in E$ with $e_i \in R_1$. Thus $e \in R_k(e_1', e_2) \in C \cup \{i\}$ by definition R2.

[The final possibility, $e_1 \in C$ but $e_2 \notin C$, is treated the same way.]
Prop. 6. If $e$ is reducible to $e'$, then $\mu e = \mu e'$ [R preserves $\mu$].

Proof. We show the proposition holds for every expression whose components all satisfy the proposition. Then we can conclude by structural induction that the proposition holds for all $e \in E$.

Definition CS3 partitions $E$ into three classes: $\{i\}$, $A$, and the "constructed expressions".

1) $e = i$. Then by proposition 5, $e$ is irreducible.
2) $e \in A$. Then $e \in C$, so again by proposition 5, $e$ is irreducible.
3) $e = k(e_1, \ldots, e_n)$ for a unique $n$-place $k \in K$ and unique $e_1, \ldots, e_n \in E$. We consider two subcases.
   i) $k \neq ap$. Suppose $eR e'$. Then by definition $R_1-R_2$, we must have $e_i R e'_i$ for some component of $e_i$ of $e$, and $e' = k(e_1, \ldots, e'_i, \ldots, e_n)$. Now $\mu e_i = \mu e'_i$ by induction, so $\mu e = k(\mu e_1, \ldots, \mu e_i, \ldots, \mu e_n) = k(\mu e_1, \ldots, \mu e'_i, \ldots, \mu e_n) = \mu e'$.
   ii) $k = ap$, so $e = ap(e_1, e_2)$. If both $e_1 \in C$ and $e_2 \in C$, then $eR (pe_1) e_2$ by definition $R_1$. Further, $\mu e_1 = e_1$ and $\mu e_2 = e_2$ [proposition 3], so $\mu e = \mu (pe_1) e_2 = p(\mu e_1) e_2$. On the other hand, if one of the components $e_1, e_2$ is nonconstant then it must be reducible. Reasoning similar to case i above shows that $e$ is reducible to an $e'$ with $\mu e = \mu e'$.

Prop. 7. If $\mu e \neq i$, then $\delta e \neq i$.

Proof. The proof is by "parallel computational induction": if $P(\Omega, \Omega)$, and $P(f, g)$ implies $P(tf, cg)$ for an admissible predicate $P$, we can
deduce $P(f, g)$, where $f, g$ are the least fixed points of $\tau, \sigma$ respectively. Our predicate $P(f, g)$ is: if $fe\not=\bot$, then $fe=\mu e$ and $ge\not=\bot$. [The clause relating $f$ and $\mu$ is needed in the induction step.] $\tau$ and $\sigma$ are the defining functionals of $\mu$ and $\delta$, respectively.

**Basis:** $[P(\Omega, \Omega)]. fe=qe=\bot$ for all $e\in E$, by definition of $\Omega$.

**Induction:** Assume $P(f, g)$ holds; we must show $P(\tau f, \sigma g)$, that is, if $(\tau f)e\not=\bot$, then $(\tau f)e=\mu e$ and $(\sigma g)e\not=\bot$. By definition CS3, there are three distinct cases to consider.

**Case 1:** $e=\bot$. Then $(\tau f)e=\bot$ by definition of $\tau$.

**Case 2:** $e\in A$. Then $(\tau f)e=e$ by definition of $\tau$, and $e=\mu e$, by proposition 2b. Also $(\sigma g)e=\not=\bot$, by definition of $\sigma$.

**Case 3:** $e=k(e_1, \cdots, e_n)$ for a unique $n$-place $k\in K$ and unique $e_1, \cdots, e_n \in E$. Two subcases arise.

i) $k\not=ap$. Assume $(\tau f)e\not=\bot$. Then

$(\tau f)e=k(fe_1, \cdots, fe_n)$ [def. of $\tau$; note $fe_i\not=\bot$ by CS2]

$=k(\mu e_1, \cdots, \mu e_n)$ [by the inductive assumption, $P(f, g)$]

$=\mu e$ [by proposition 2c].

Additionally, $(\sigma g)e\not=\bot$, for $(\sigma g)e=ge_1+\cdots+ge_n$ by definition of $\tau$, and $ge_i\not=\bot$ by inductive assumption.

ii) $k=ap$, so $e=ap(e_1, e_2)$. Note that $f$ must be $\bot$-preserving.

[By inductive assumption, either $f_1=\bot$ or $f_1=\mu \bot$. But $\mu \bot=\bot$, by proposition 2a.] Thus if we assume $(\tau f)e =f(pfe_1)fe_2\not=\bot$, then we must have $(pfe_1)fe_2\not=\bot$. By our
general assumptions on \( p, f_1 \neq i \) and \( f_2 \neq i \). \( f_1 = \mu e_1, f_2 = \mu e_2 \), and \( f(pf_1)f_2 = \mu (pf_1)f_2 = \mu (\mu e_1)\mu e_2 \) follow by induction and substitution. This proves \((\tau f)e=\mu e\).

We must still show that \((\sigma g)e=1+ge_1+ge_2+g(\mu e_1)\mu e_2 \neq i\).

Since \( f_1 \neq i \) and \( f_2 \neq i \), \( ge_1 \neq i \) and \( ge_2 \neq i \) follow by induction. Also \( g(\mu e_1)\mu e_2 \neq i \), because \( f(pf_1)f_2 \neq i \), \( f_1 = \mu e_1 \), and \( f_2 = \mu e_2 \) [as was shown above]. Thus \((\sigma g)e \neq i\).

Prop. 8. For all \( e \in C \), \( \delta e=0 \)

Proof. By structural induction on \( e \). There are, by CS3, three distinct cases.

Case 1: \( e=1 \). Then \( e \notin C \), by definition C1-C3.

Case 2: \( e \in A \). Then \( \delta e=(\sigma \delta)e \), since \( \delta \) is a fixed point of \( \sigma \).

But \((\sigma \delta)e=0 \) for all \( e \in A \), by definition of \( \sigma \).

Case 3: \( e=k(e_1, \ldots, e_n) \), for unique \( n \)-place \( k \in K \) and \( e_1, \ldots, e_n \in E \). Two subcases arise.

i) \( k \neq \alpha p \). If \( e \in C \) then by definition C1-C3, \( e_i \in C \) for \( 1 \leq i \leq n \).

By induction \( \delta e_i = 0 \), \( 1 \leq i \leq n \). So \( \delta e=\delta e_1+\cdots+\delta e_n=0 \).

ii) \( k=\alpha p \). Then \( e=\alpha p(e_1, e_2) \notin C \).

Prop. 9. If \( \mu \neq i \) and \( e \neq \epsilon' \), then \( \delta e=\delta e'+1 \)

Proof. Suppose \( \mu \neq i \) and \( e \neq \epsilon' \). By proposition 6, \( \mu \epsilon'=\mu \neq i \); hence by proposition 7, \( \delta e \neq i \) and \( \delta e' \neq i \). We prove the relationship \( \delta e=\delta e'+1 \) using structural induction on \( e \). By CS3, there are
three distinct cases to consider.

Case 1: \( e = 1 \). Then \( \mu e = 1 \), by proposition 2a.

Case 2: \( e \in A \). Then \( e \in C \), so \( e \) is irreducible by proposition 5.

Case 3: \( e = k(e_1, \ldots, e_n) \) for unique \( n \)-place \( k \in K \) and \( e_1, \ldots, e_n \in E \). There are two subcases.

i) \( k \neq \text{ap} \). Suppose \( e \in R' \). By definition \( R2 \), \( e' = k(e_1, \ldots, e_i', \ldots, e_n) \) for some \( e_i' \in E \) with \( e_i \in R_i' \). Since \( \mu e = k(\mu e_1, \ldots, \mu e_i, \ldots, \mu e_n) \) and \( k \) is \( 1 \)-preserving [by CS2], \( \mu e_i \neq 1 \). By induction, \( \delta e_i = \delta e_i' + 1 \). Thus

\[
\delta e = \delta e_1 + \cdots + \delta e_i + \cdots + \delta e_n = \delta e_1 + \cdots + (\delta e_i' + 1) + \cdots + \delta e_n = (\delta e_1 + \cdots + \delta e_i' + \cdots + \delta e_n) + 1 = \delta e' + 1.
\]

ii) \( k = \text{ap} \), so \( e = \text{ap}(e_1, e_2) \) for unique \( e_1, e_2 \in E \). If \( e_i \notin C \) and/or \( e_2 \notin C \), the reasoning is analogous to case i above. So here assume \( e_1, e_2 \in C \) and \( e \in R' \). By definition \( R1 \), we must have \( e' = (pe_1)e_2 \). Thus

\[
\delta e = 1 + \delta e_1 + \delta e_2 + \delta (\mu e_1)\mu e_2 \quad [\text{since } \delta e = (o\delta)e]
= 1 + 0 + 0 + \delta (\mu e_1)\mu e_2 \quad [\text{by proposition 8}]
= 1 + \delta (p e_1)e_2 \quad [\text{by proposition 3}]
= 1 + \delta e'.
\]

Prop. 10. If \( \mu e \neq 1 \), then every sequence of reductions on \( e \) converges to \( \mu e \) [in exactly \( \delta e \) steps].

Proof. A sequence of reductions on \( e \) is a [finite or infinite] sequence of expressions \( e_0, e_1, e_2, \ldots \) with \( e_0 = e \) and, for all \( i \geq 0 \), \( e_i \rightarrow e_{i+1} \).
Thus we wish to show that if $\mu e \neq e$, the sequence is of length $\delta e$ [which, by proposition 7, is not 1], and that $e \delta e = \mu e$.

Since $\mu e_0 = \mu e \neq e$, and $R$ preserves $\mu$ [proposition 6], it follows by induction that the meanings of all expressions in the sequence are identical. If, for some $n \geq 0$, there is no $e'$ with $e_n R e'$, then $e_n \in C$ [proposition 5], so $\mu e = \mu e_n = e_n$ [proposition 3]. Thus if the sequence terminates, it terminates in $\mu e$.

But the sequence must terminate, for $\delta e_0, \delta e_1, \delta e_2, \ldots$ is a decreasing sequence of natural numbers [proposition 9]. Suppose $e_n$ is the last term, that is, $\mu e = e_n \in C$. Then $\delta e_n = 0$ [proposition 8], so $\delta e_0 = \delta e_1 + 1 = (\delta e_2 + 1) + 1 = \cdots = \delta e_n + n = n$, the number of steps in any reduction sequence for $e$.

ACKNOWLEDGEMENTS

I am indebted to John Backus and Jim Gray for their careful reading and constructive criticism of the manuscript, and to Barry Rosen for much information on complete posets and their application to closed applicative languages.