A Relational Theorem on the Correctness of General Recursive Programs

Jaime A. Bohórquez V.
jbohorqu@escuelaing.edu.co
Escuela Colombiana de Ingeniería

Mai 2, 2006

Agenda

Context

- Identify programs and specifications with their associated input-output relations on a space state Σ.
- A programming theory may be developed within the framework of Tarski's relational calculus where programs and specifications (possibly non-deterministic) are considered as logical predicates.
- Look at *recursive program definitions* as *fixed point equations* associated to *relational operators*, whose solutions include the input-output relations corresponding to such programs.
- *Conditions* (boolean expressions on Σ) correspond to relations whose domains coincide with the *set of those initial states in which the condition holds*, and which relates each member of its domain onto any final state whatsoever.

Continuous Operators

According to Knaster-Tarski theorem for a continuous relational operator f (function from relations to relations on Σ) the equation $X = f.X$ has a solution on Σ (a complete lattice).
There may be many solutions; the least of them is μf , *the least fixed point* of the operator f.
The continuity of f implies that μf can be constructed as the union of a series of approximations obtained by its iteration:

$$\mu f = \bigcup_{n>0} f^n. O$$

where $f^0.R = R$
and $f^{n+1}.R = f(f^n.R)$ for all $n \geq 0$ and relation R.

The Correctness Problem for a Recursive Program

Correctness Problem

Given the program (by its source text) which computes a function f with domain D ($D \subseteq \Sigma$) and given its specification in terms of a predicate S describing the desired relationship between its initial and final values, *find* non restrictive *regularity conditions* on its text to prove it correct with respect to S by a well-founded induction on a covering of D.

Agenda

Source Code of a Recursive Program

Every recursive program computing a function f with domain a subset of Σ, is an instance of the following abstract scheme:

$$\begin{array}{lll} f.s = & \textbf{if} & b.s \rightarrow q.s \\ & \square & c.s \rightarrow h(f, M.s) \\ & \textbf{fi} & \end{array}$$

where

- s symbolizes the initial state of program f.
- conditions b and c correspond (initially) to all non-recursive and all recursive cases of its domain,
- $q.s$ is the final value given by f in the non recursive cases
- $h(f, M.s)$ involves at least one recursive invocation of f on a set of states $M.s$.

Associated Operator

Abstract Recursive Scheme

$$\begin{array}{lll} f.s & = & \textbf{if } b.s \rightarrow q.s \\ & & \;\square\; c.s \rightarrow h(f,\, M.s) \\ & & \textbf{fi} \end{array}$$

For a *relational operator* $\mathfrak{f}$ corresponding to the program scheme above we may assume the existence of a relation Q and a relational operator $\mathfrak{h}$ fulfilling the following for all relation R,

$$\mathfrak{f}.R = Q \cup \mathfrak{h}.R$$

with

(a) $QL \cap (\mathfrak{h}.L)L = O$

(b) $\mathfrak{h}.O = O$

The Concept of Inductivity

The fact that the recursive program scheme above allows calculating the values of (program) function f by stages suggests both

i) requiring the associated operator f to be *continuous* and

ii) defining the following concept:

Inductivity

Given an operator f, a relation R and a sequence of conditions $\langle c_n \rangle_{n \geq 0}$, we say that f is *inductive* over R through the given sequence if

$$f.R \cap c_{n+1} = f(R \cap c_n)$$

for all $n \geq 0$.

Regularity Conditions

The regularity conditions we looked for are the following:

Definition of *Regular Operators*

A continuous operator f is *regular* if it fulfills the following *regularity conditions*:

There exist a relation Q and an operator h such that

(a) $f.R = Q \cup h.R$ for all relation R.

(b) $QL \cap (h.L)L = O$

(c) $h.O = O$

(d) h is inductive over *every fixed point* of f, through the sequence of conditions $\langle (f^n. O)L \rangle_{n>0}$.

The inductivity of h over any fixed point K of f through the ascending chain $\langle (f^m. O)\, L\rangle_{m>0}$ is equivalent to the inductivity of f over K through the same chain.

Inductivity of Regular Operators

Proposition 1

For a *continuous* operator f, satisfying regularity conditions (a), (b) and (c) and K *fixed point* of f, the following three properties are equivalent:

(i) f is inductive over K through ascending chain $\langle\, (f^n.\, O)L\,\rangle_{n>0}$.

(ii) $K \cap (f^n.\, O)L = f^n.\, O$ for all natural n.

(iii) $K \cap \mu f\, L = \mu f$, and sequence $\langle f^n.\, O\rangle_{n>0}$ is stable.

Definition of a *Stable* sequence of relations

A sequence of relations $\langle R_n\rangle_{n\geq 0}$ is *stable* whenever

$$m < n \;\Rightarrow\; R_n \cap R_m L = R_m$$

for all natural numbers m and n.

Deterministic Programs are (easily) Regular

Inductivity condition is non restrictive for recursive programs defining (partial) functions:
if in scheme equation

$$f.R = Q \cup h.R$$

Q is a *univalent* relation and h is *closed on univalent relations*; necessarily, f is also closed on these relations and μf becomes a partial function. Therefore,

- if every fixed point of f is univalent (at least on the domain of μf) then necessarily, all of them coincide with μf on $\mu f L$ and
- since the chain of *partial functions* $f^n.O$ must be stable,

by last proposition f satisfies inductivity condition (d).

Proof of Proposition 1

(i) f is inductive over K through ascending chain $\langle (f^n . O)L \rangle_{n>0}$.

(ii) $K \cap (f^n . O)L = f^n . O$ for all natural n.

Proof of (i) $\equiv$ (ii):

"$\Rightarrow$" By induction on n. Case $n=1$ is easy. Suppose $n>1$.

$$
\begin{array}{ll}
 & K \cap (f^n . O)L \\
= & \quad \langle\ K \text{ fixed point of } f\ \rangle \\
 & f.K \cap (f^n . O)L \\
= & \quad \langle\ \text{Hypothesis (i)}\ \rangle \\
 & f(K \cap (f^{n-1} . O)L) \\
= & \quad \langle\ \text{Inductive hypothesis}\ \rangle \\
 & f(f^{n-1} . O) \\
= & \quad \langle\ \text{Definition of } f^n\ \rangle \\
 & f^n . O
\end{array}
$$

Proof of Proposition 1

(i) f is inductive over K through ascending chain $\langle (f^n . O)L \rangle_{n>0}$.

(ii) $K \cap (f^n . O)L = f^n . O$ for all natural n.

Proof of (i) $\equiv$ (ii):

" $\Leftarrow$ "

$$
\begin{array}{ll}
 & f(K \cap (f^m . O)L) \\
= & \quad \langle \text{ (ii) } \rangle \\
 & f . f^m . O \\
= & \quad \langle\, f \circ f^m = f^{m+1} \text{ ; (ii) } \rangle \\
 & K \cap (f^{m+1} . O)L \\
= & \quad \langle\, K \text{ fixed point of } f \,\rangle \\
 & f . K \cap (f^{m+1} . O)L
\end{array}
$$

Proof of Proposition 1

(ii) $K \cap (f^n. O)L = f^n. O$ for all natural n.
(iii) $K \cap \mu f L = \mu f$, and sequence $\langle f^n. O\rangle_{n>0}$ is stable.

Proof of (ii) $\equiv$ (iii):

"$\Rightarrow$" a)

$$
\begin{array}{ll}
 & K \cap \mu f L \\
= & \langle\ f \text{ continuous operator } \rangle \\
 & K \cap (\bigcup_{m>0} f^m. O)L \\
= & \langle\ \text{‘}\circ\text{’ and ‘}\cap\text{’ distributes over ‘}\cup\ \rangle \\
 & (\bigcup_{m>0} K \cap (f^m. O)L) \\
= & \langle\ \text{Hypothesis (ii)} \ \rangle \\
 & (\bigcup_{m>0} f^m. O) \\
= & \langle\ f \text{ continuous operator } \rangle \\
 & \mu f
\end{array}
$$

Proof of Proposition 1

(ii) $K \cap (f^n.\, O)L = f^n.\, O$ for all natural n.
(iii) $K \cap \mu f\, L = \mu f$, and sequence $\langle f^n.\, O \rangle_{n>0}$ is stable.

Proof of (ii) $\equiv$ (iii):

"$\Rightarrow$" b) Suppose $m < n$ then

$$
\begin{aligned}
& f^n.\, O \cap (f^m.\, O)L \\
= \quad & \langle \text{ (ii) } \rangle \\
& K \cap (f^n.\, O)L \cap (f^m.\, O)L \\
= \quad & \langle\ \langle (f^n.\, O)L \rangle_{n>0} \text{ ascending chain ; } m < n\ \rangle \\
& K \cap (f^m.\, O)L \\
= \quad & \langle \text{ (ii) } \rangle \\
& f^m.\, O
\end{aligned}
$$

Proof of Proposition 1

(ii) $K \cap (f^n.\, O)L = f^n.\, O$ for all natural n.
(iii) $K \cap \mu f L = \mu f$, and sequence $\langle f^n.\, O\rangle_{n>0}$ is stable.

Proof of (ii) $\equiv$ (iii):

"$\Leftarrow$"

$$
\begin{array}{ll}
 & K \;\cap\; (f^n.\, O)L \\
= & \langle \text{ "}f\text{ continuous" } \Rightarrow (f^n.\, O)L \subseteq \mu f L \rangle \\
 & K \cap \mu f L \cap (f^n.\, O)L \\
= & \langle \text{ (iii) } \rangle \\
 & \mu f \cap (f^n.\, O)L \\
= & \langle \text{ "}f\text{ continuous" ; range splitting } \rangle \\
 & ((\bigcup_{m\leq n} f^m.\, O) \cup (\bigcup_{m>n} f^m.\, O)) \;\cap\; (f^n.\, O)L \\
= & \langle \cap \text{ distributes over } \cup \text{ twice ; (iii) } \rangle \\
 & (\bigcup_{m\leq n} f^m.\, O) \cup f^n.\, O \\
= & \langle \; \langle (f^n.\, O)L\rangle_{n>0} \text{ ascending chain } \rangle \\
 & f^n.\, O
\end{array}
$$

Agenda

Regular Recursive Schemes

Definition

Relation P is defined by a *regular recursive scheme* associated to operator f, whenever

- P is a relation representing a program, and
- $P = \mu f$ i.e. P is the *minimum solution* of the fixed point equation associated to a *regular operator* f.

If relation P represents a *program* and a relation S represents a *specification*, the expression $P \subseteq S$ means that program P *meets* specification S.

Correctness Criterion

Proposition 2

If P is a program defined by a *regular recursive scheme* associated to an operator f via equation $f.R = Q \cup h.R$ (i.e. $P = \mu f$) then, there exists an ascending chain of conditions $\langle b_n \rangle_{n \geq 0}$ with union equal to PL, $b_0 = QL$, and such that

$$P \subseteq S \;\equiv\; Q \subseteq S \wedge (\forall n \mid n \geq 0 : P \cap b_n \subseteq S \Rightarrow h(P \cap b_n) \subseteq S)$$

This proposition gives an induction scheme for proving that program P satifies S.
In fact, this scheme may be generalized to well founded relations on the classes determined by a covering on the domain of P.

Correctness Criterion

Partial Proof of prop. 2

Let $b_n = (f^{n+1}. O)L$ for all non negative n.
The equivalence is proved as follows:

$$
\begin{array}{ll}
 & P \subseteq S \\
\equiv & \langle\ P = \mu f\ ;\ \text{definition of } b_n\ ;\ f \text{ regular}\ ;\ \text{prop. } 1\ \rangle \\
 & (\bigcup n \mid n \geq 0 : P \cap b_n) \subseteq S \\
\equiv & \langle\ \text{Set Theory}\ \rangle \\
 & (\forall n \mid n \geq 0 : P \cap b_n \subseteq S) \\
\equiv & \langle\ \text{Induction on } n\ \rangle \\
 & P \cap b_0 \subseteq S \ \wedge\ (\forall n \mid n \geq 0 : P \cap b_n \subseteq S \Rightarrow P \cap b_{n+1} \subseteq S) \\
\equiv & \langle\ P \cap b_0 = Q\ ;\ P = \mu f\ ;\ \text{proposition } 1\ \rangle \\
 & Q \subseteq S \ \wedge\ (\forall n \mid n \geq 0 : P \cap b_n \subseteq S \ \Rightarrow\ f(P \cap b_n) \subseteq S) \\
\equiv & \langle\ f.R = Q \cup \hbar.R\ \rangle \\
 & Q \subseteq S \ \wedge\ (\forall n \mid n \geq 0 : P \cap b_n \subseteq S \ \Rightarrow\ \hbar(P \cap b_n) \subseteq S)
\end{array}
$$

General Correctness Theorem

If P is a program defined by a recursive scheme associated to a monotonic operator f via equation $f.R = Q \cup h.R$, then, for all well founded set $(\mathcal{C}, \sqsubset)$ with $\mathcal{C}$ a countable family of conditions with union equal to PL, and '$\sqsubset$' a well founded relation on $\mathcal{C}$ such that

(i) $(\forall u \in \mathcal{C} \mid : \; u \subseteq QL \;\vee\; u \subseteq PL \cap \overline{QL})$

(ii) $(\forall u \in \mathcal{C} \mid u \subseteq PL \cap \overline{QL} : P \cap u \;\subseteq\; h(\bigcup v \mid v \sqsubset u : P \cap v))$

we have the equivalence of the following two propositions:

a) Program P satisfies specification S.

b) $(\forall v \mid v \subseteq QL : P \cap v \subseteq S) \wedge$
$(\forall v \mid v \subseteq PL \cap \overline{QL} :$
$(\forall u \mid u \sqsubset v : P \cap u \subseteq S) \Rightarrow h(\bigcup u \mid u \sqsubset v : P \cap u) \subseteq S)$

Agenda

Recursive program scheme

Remember our recursive program scheme:

Abstract Recursive Scheme

$$\begin{array}{lll} f.s & = & \textbf{if } b.s \rightarrow q.s \\ & & \square \; c.s \rightarrow h(f, M.s) \\ & & \textbf{fi} \end{array}$$

Applying the Correctness Theorem

Restricting the application of previous theorem to functions (like f) requires finding a well founded relation '$\sqsubset$' on the domain of f for which

$$c.s \wedge t \in M.s \Rightarrow t \sqsubset s \quad \text{for all states } s \text{ and } t.$$

Besides this, regularity conditions above reduce to two (reasonable) conditions holding for all state $s \in \Sigma$:

1. $\neg(b.s \wedge c.s)$
2. $b.s \Rightarrow dom\, q.s$

Ackermann function

This function $A(x, y)$ is defined for natural numbers x and y by

$$A(x,y) = \begin{cases} y+1 & \text{if } x=0 \\ A(x-1,1) & \text{if } y=0 \\ A(x-1, A(x,y-1)) & \text{otherwise} \end{cases}$$

This *function* is clearly defined by a regular inductive scheme. The usual lexicographic order relation on $\mathbb{N}\times\mathbb{N}$ (noted with '$\prec$') is a well founded relation satisfying the requirement given above:

$$\begin{array}{ll} (x-1,1) \prec (x,y) & \text{if } \ x\neq 0 \ \wedge \ y=0. \\ (x,y-1),(x-1,A(x,y-1)) \prec (x,y) & \text{if } \ x\neq 0 \ \wedge \ y\neq 0. \end{array}$$

McCarthy's 91 function

This function is defined for natural number x by

$$\begin{array}{lll} g.x = & \mathbf{if} \; x > 100 & \rightarrow \; x - 10 \\ & \square \; x \leq 100 & \rightarrow \; g(g(x+11)) \\ & \mathbf{fi} & \end{array}$$

This recursive scheme is regular. Partial order '$\sqsubset$' on $\mathbb{Z}$ defined as

$$x \sqsubset y \;\equiv\; y \leq 100 \wedge y < x \;\text{ for all integers } x \text{ and } y,$$

allows to apply our theorem. $\sqsubset$ is a well founded relation such that

- if $x > 100$ then x is $\sqsubset$-minimal,
- $x+11 \sqsubset x$ if $x \leq 100$ and
- $g(x+11) \sqsubset x$ if $x \leq 100$.

McCarthy's 91 function

If

$$\begin{array}{rlll} g.x = & \textbf{if} & x > 100 & \rightarrow \; x - 10 \\ & \square & x \leq 100 & \rightarrow \; g(g(x+11)) \\ & \textbf{fi} & & \end{array}$$

we may prove by induction on $(\mathbb{Z}, \sqsubset)$ that for all integer x, $g.x = f.x$ where

$$\begin{array}{rlll} f.x = & \textbf{if} & x > 101 & \rightarrow \; x - 10 \\ & \square & x \leq 101 & \rightarrow \; 91 \\ & \textbf{fi} & & \end{array}$$

Since $g.101 = 101 - 10 = 91 = f.101$, by definition of f and g, it is enough to show that $g.x = f.x$ for $x \leq 100$.

$$
\begin{array}{ll}
 & g.x \\
= & \langle \text{ Definition of } g \text{ ; } x \leq 100 \rangle \\
 & g(g(x+11)) \\
= & \langle\, x+11 \sqsubset x \text{ ; Inductive Hypothesis } \rangle \\
 & g(f(x+11)) \\
= & \langle \text{ Definition of } f \text{ ; } x \leq 100 \rangle \\
 & \begin{cases} g(x+1) & \text{if } 100 < x+11 \leq 111 \\ g(91) & \text{if } x < 90 \end{cases} \\
= & \langle\, x < 90 \Rightarrow 91 \sqsubset x \text{ ; } x \leq 100 \Rightarrow x+1 \sqsubset x \text{ ; Ind. Hypothesis } \rangle \\
 & \begin{cases} f(x+1) & \text{if } 90 < x+1 \leq 101 \\ f(91) & \text{if } x < 90 \end{cases} \\
= & \langle \text{ Definition of } f \rangle \\
 & 91 \\
= & \langle \text{ Definition of } f \text{ ; } x \leq 100 \rangle \\
 & f.x
\end{array}
$$

Agenda

Final Conclusions

- We have found reasonable *regularity conditions* to ask of the code of a recursive program to prove it correct with respect to a given specification.
- (terminating) *Deterministic* recursive programs fulfill these regularity conditions.
- The correctness proof of a recursive program may be done by *induction on a well founded relation on its domain* induced by the values on which the program recurs (according to its code).