A Denotational Engineering of Programming Languages

…

Part 7: Semantic correctness of programs

(Sections 7.1 – 7.6 of the book)

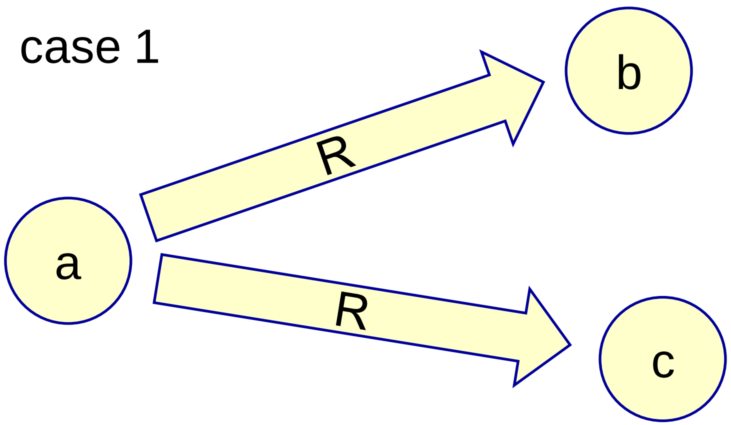
Andrzej Jacek Blikle

May 21st, 2021

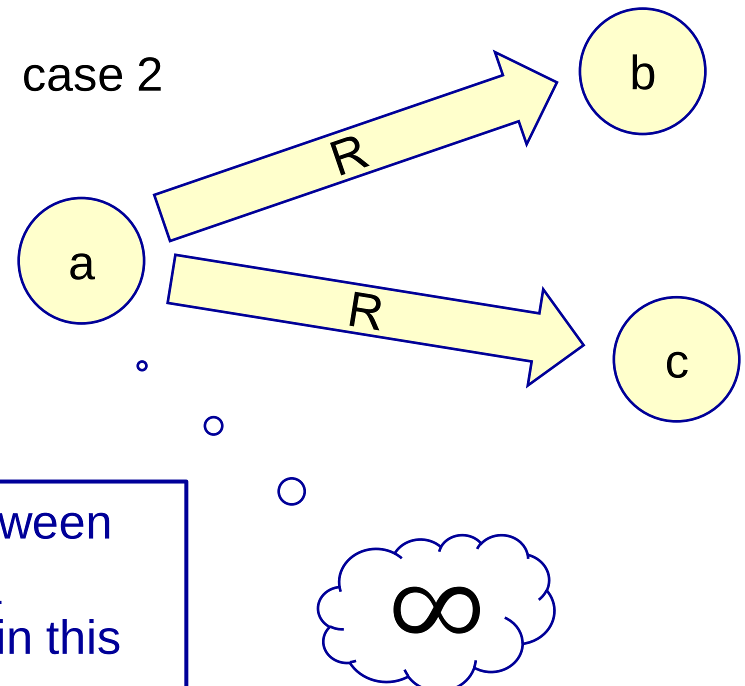
A relational model of nondeterministic programs

S	– a set of states
P, R ⊆ S x S	– the denotation of a nondeterministic program
a R b	– there is a finite (terminating) computation from a to b
P●R	= {(a, c) | (∃ b) a P b **and** b R c}

case 1



case 2



In both cases
a R b and a R c

The difference between
case 1 and case 2
is not expressible in this model.

F.a = ? means no finite run i.e. no run or infinite run

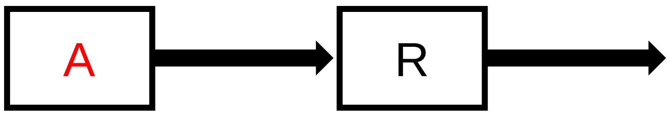
There are models for denotational treatment of infinite executions.

Composition of a relations with a set

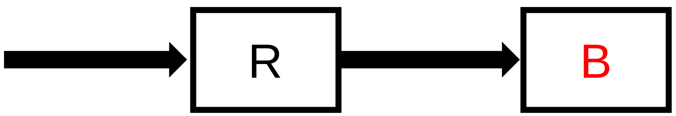
Let R : Rel(S,S) and A, B ⊆S

A R = {s | (∃a:A) a R s} – left composition; the image of A by R

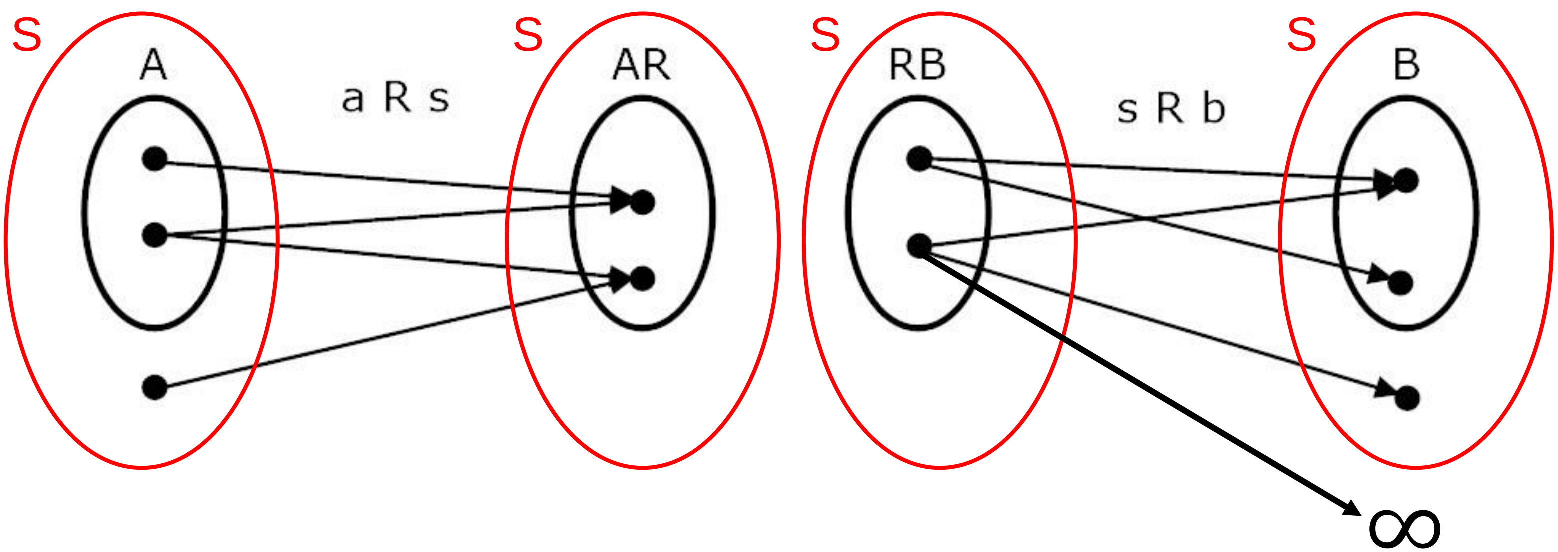
R B = {s | (∃b:B) s R b} – right composition; the coimage of B by R.



AR final states of R-executions that start in A



RB initial states of R-executions that end in B



Some properties of AR and RB

$A(RQ) = (AR)Q$ – associativity

$(RQ)B = R(QB)$

$(A \mid B)\, R = (AR) \mid (BR)$ – distributivity

$A\, (R \mid Q) = (AR) \mid (AQ)$

if $A \subseteq B$ then $AR \subseteq BR$ – monotonicity

if $R \subseteq Q$ then $AR \subseteq AQ$

$(U\, A_i)\, R = U\, (A_i\, R)$ – continuity

$A\, (U\, R_i) = U\, (A\, R_i)$

$R\, (U\, B_i) = U\, (R\, B_i)$ – continuity

$(U\, R_i)\, B = U\, (R_i\, B)$

Structured programs in a relational framework

$[A] : Rel(S,S)$ – an identity relation (function); $[A] = \{(a, a) \mid a : A\}$

3-valued partial predicates p on S will be represented by two disjoint sets of states

$C = \{s \mid p.s = tt\}$, $\quad C \cap \neg C = \varnothing$

$\neg C = \{s \mid p.s = ff\}$ $\quad C \mid \neg C \subseteq S$

$S – (C \mid \neg C)$ – the set of states that lead to abortion (error) or infinite executions

To distinguish between abortion and infinite execution we would need a third set:

$eC = \{s \mid p.s : Error\}$

We shall not exploit this option since in the construction of correct programs we want to avoid both – abortion and looping.

Constructors of structured programs:

P ; Q = P Q

if (C,¬C) **then** P **else** Q **fi** = [C] P | [¬C] Q

while (C,¬C) **do** P **od** = ([C]P)*[¬C]

i.e. the least solution of X = [C](PX) | [¬C]

A

Program correctness

general case – possibly nondeterministic

$AR \subseteq B$ – <u>partial correctness</u> wrt <u>precondition</u> A and <u>postcondition</u> B

$(\forall a : A)$ **if** $(\exists b)\ a\ R\ b$ **then** $b : B$

For every a : A, **every** a-execution of R which terminates, terminates in B.

$A \subseteq RB$ – <u>weak total correctness</u> wrt <u>precondition</u> A and <u>postcondition</u> B

$(\forall a : A)\ (\exists b)\ a\ R\ b$ **and** $b : B$

For every a : A, **there is** a-execution of R that terminates in B but there may be other executions, that do not terminate in B or do not terminate at all.

None of these properties is stronger than the other!

A

Program correctness in deterministic case

deterministic case – F is a function

$AF \subseteq B$ – partial correctness: for every a : A, if F.a = ! then F.a : B

$A \subseteq FB$ – clean total correctness: for every a : A, F.a = ! and F.a : B

No abortion

No error

$A \subseteq FB$ iff $AF \subseteq B$ and $F : A \mapsto S$

Clean partial correctness

Termination in A

Clean termination = termination (no infinite execution) & non-abortion

Expressible as a property of a function due to abstract errors (as in **Lingua**).

halting property
własność stopu

A

Halting property of deterministic programs

In the general case halting property of programs is not decidable, and sometimes may be very difficult to prove.

```
pre n > 0
  x := n;
  while x > 1 do;
  if even(x) then x := x/2 else x := 3x + 1 fi
post x = 1
```

Collatz hypothesis formulated in 1937.

So far proved only for $n < 5*2^{86}$.

# of years > age of universe x 10^{65} with 1 ns program execution

```
pre n, m > 0
  x := 1; y := m;
  while x < n do;
  x := x+1;  y := y*m
post y = m^n
```

In some practical situations halting property may be quite obvious.

This makes us interested in partial correctness

Proof rules for partial correctness

No recursion or iteration

Sequential composition

$$\begin{array}{l} AP \subseteq B \\ B \subseteq C \\ CQ \subseteq D \\ \hline A(P;Q) \subseteq D \end{array}$$

Strengthening precondition

$$\begin{array}{l} AP \subseteq B \\ C \subseteq A \\ \hline CP \subseteq B \end{array}$$

Conditional composition; $C \cap \neg C = \varnothing$

$$\begin{array}{l} (A \cap C)P \subseteq B \\ (A \cap \neg C)Q \subseteq B \\ \hline A\ \textbf{if}\ (C,\neg C)\ \textbf{then}\ P\ \textbf{else}\ Q\ \textbf{fi} \subseteq B \end{array}$$

Weakening postcondition

$$\begin{array}{l} AP \subseteq B \\ B \subseteq C \\ \hline AP \subseteq C \end{array}$$

The general case of (mutually) recursive procedures

$X_1 = \Psi_1.(X_1,\dots,X_n)$
…
$X_n = \Psi_n.(X_1,\dots,X_n)$

Ψ_i – polynomials, e.g.

$\Psi.(X,Y,Z) = P\ X\ Q\ Y\ |\ X\ Y\ |\ P\ Z\ P$

There is nothing like canonical equations for recursion.
Each case has to be considered (given a rule) separately

Simple recursion

$X = HXT \mid E$

H – head
T – tail
E - exit

while is a particular case of simple recursion

$X = [C]PX \mid [\neg C]$

$H = [C]P$
$T = [S]$
$E = [\neg C]$

Proof rules for partial correctness
General recursion

A componentwise CPO of vectors of relations

$\mathbf{R} = (R_1,\ldots,R_n) \qquad \mathbf{A} = (A_1,\ldots,A_n) \qquad \mathbf{B} = (B_1,\ldots,B_n) \qquad n \geq 1$

Let $\mathbf{R}$ be the least solution of $\mathbf{X} = \mathbf{\Psi}.\mathbf{X}$,

General recursion

there exists a family of (vectors of) preconditions $\{\mathbf{A}_i \mid i \geq 0\}$
and a family of (vectors of) postconditions $\{\mathbf{B}_i \mid i \geq 0\}$ such that

$$
\frac{
\begin{array}{ll}
(\forall i \geq 0)\ \mathbf{A} & \subseteq \mathbf{A}_i \\
(\forall i \geq 0)\ \mathbf{A}_i\ \mathbf{\Psi}^i.\mathbf{\emptyset} & \subseteq \mathbf{B}_i \\
\mathbf{U}\{\mathbf{B}_i \mid i \geq 0\} & \subseteq \mathbf{B}
\end{array}
}{
\mathbf{A}\ \mathbf{R} \subseteq \mathbf{B}
}
$$

Proof rules for partial correctness
simple recursion

If R is the least solution of $X = HXT \mid E$ then for any $A, B \subseteq S$ the following rules hold:

Version 1

there exists a family of preconditions $\{A_i \mid i \geq 0\}$
and a family of postconditions $\{B_i \mid i \geq 0\}$ such that

$$\frac{\begin{array}{ll} (\forall i \geq 0)\ A & \subseteq A_i \\ (\forall i \geq 0)\ A_i H^i E T^i & \subseteq B_i \\ \cup\{B_i \mid i \geq 0\} & \subseteq B \end{array}}{AR \subseteq B}$$

Version 2 For any $A, B \subseteq S$

$$\frac{\begin{array}{l} (\forall Q)\ (AQ \subseteq B \ \text{ implies } \ A(HQT) \subseteq B) \\ AE \subseteq B \end{array}}{AR \subseteq B}$$

Proof rules for partial correctness
while loop

Then for any A, B ⊆S, any disjoint C, ¬C ⊆S, and for any P ⊆Rel(S, S)

$$\frac{\begin{array}{ll}\text{there exists a family of postconditions } \{B_i \mid i \geq 0\} \text{ such that} & \\ (\forall i \geq 0)\ A\,([C]P)^i\,[\neg C] & \subseteq B_i \\ \cup\{B_i \mid i \geq 0\} & \subseteq B\end{array}}{A\ \textbf{while}\ (C,\neg C)\ \textbf{do}\ P\ \textbf{od} \subseteq B} \Updownarrow$$

$$\frac{\begin{array}{ll}\text{there exists } N \subseteq S \text{ (called } \underline{\textit{loop invariant}}\text{) such that} & \\ (N \cap C)\ P & \subseteq N \\ A & \subseteq N \\ N\,[\neg C] & \subseteq B\end{array}}{A\ \textbf{while}\ (C,\neg C)\ \textbf{do}\ P\ \textbf{od} \subseteq B} \Updownarrow$$

to prove ⬆ set
N = A([C]P)*


Thank you for
your attention