A Survey of Command Algebra

Bernhard Möller

Institut für Informatik, Universität Augsburg

Collaborators: Walter Guttmann (Ulm)

Peter Höfner, Kim Solin (Augsburg)

Georg Struth (Sheffield)

1 Introduction

approaches to semantics of imperative programs

- partial/total correctness (wlp/wp) [Hoare 69, Dijkstra 74]
- general correctness [Morgan/Morris/Nelson 87, Doornbos 94]
- Kleene algebra with tests (partial correctness) [Kozen 97]
- demonic relational semantics [Nguyen 91, Backhouse 93, Desharnais 95, Desharnais/Mili/Nguyen 97, Desharnais/Möller/Tchier 02/04]
- Unifying Theories of Programs (UTP) [Hoare/He 98]
- omega algebra [Cohen 00]
- demonic refinement algebra (DRA) [von Wright 02]

how do all these interrelate?

2 Commands (General Correctness)

basic idea [Broy et al. 79, Berghammer/Zierer 86, Parnas 83]

- model a program as a pair (a, p) consisting of
- a transition relation a between states and
- a set p of states with guaranteed termination
- [Parnas 83] required $p \leq \mathsf{dom}\, a$ (= set of starting states of a)
- allows distinguishing the "must-termination" given by p from the "may-termination" given by $\mathsf{dom}\, a$
- excludes "miraculous" program behaviour
- [Morgan/Morris/Nelson 87] dropped this restriction

basic non-iterative commands

$$\begin{aligned}
\mathsf{fail} &\stackrel{\text{def}}{=} (0, 1) \\
\mathsf{skip} &\stackrel{\text{def}}{=} (1, 1) \\
\mathsf{loop} &\stackrel{\text{def}}{=} (0, 0) \\
(a, p) \mathbin{[\!]} (b, q) &\stackrel{\text{def}}{=} (a \vee b, p \wedge q) \\
(a, p) \,;\, (b, q) &\stackrel{\text{def}}{=} (a \wedge b, p \wedge [a]q)
\end{aligned}$$

where

- $0 \mathrel{\widehat{=}}$ empty transition relation/*false*
- $1 \mathrel{\widehat{=}}$ identical transition relation/*true*
- $[a]q \stackrel{\text{def}}{=} \neg \mathsf{dom}\,(a \wedge \neg q)$ (analogue of wlp)

algebraic properties:

- $(\mathrm{COM}(S), [\,], \mathsf{fail}, ;\, , \mathsf{skip})$ is a left semiring
- fail is only a left zero
- even right-distributive
- associated natural order on $\mathrm{COM}(S)$:

$$(a, p) \leq (b, q) \Leftrightarrow a \leq b \ \bigwedge\ p \geq q$$

- if S is a complete lattice then so is $\mathsf{COM}(S)$
- if S has a greatest element $\top$ then $\mathsf{chaos} \stackrel{\text{def}}{=} (\top, 0)$ is the greatest element of $\mathsf{COM}(S)$
- whereas $\mathsf{havoc} \stackrel{\text{def}}{=} (\top, 1)$ represents the most nondeterministic but everywhere terminating program

- *weakest (liberal) precondition*

$$\mathsf{wlp}.(a,p).q \stackrel{\text{def}}{=} [a]q$$

$$\mathsf{wp}.(a,p).q \stackrel{\text{def}}{=} p \wedge \mathsf{wlp}.(a,p).q$$

- then $p = \mathsf{wp}.(a,p).1$, so that, for command k,

$$\mathsf{wp}.k.q = \mathsf{wp}.k.1 \wedge \mathsf{wlp}.k.q$$

  (Nelson's pairing condition)

- by antitony of box: $k \leq l \Rightarrow \mathsf{wp}.k \geq \mathsf{wp}.l$

  (converse of the usual refinement relation)

3 wp is wlp

- definition of commands based on tests (abstract versions of assert-statements that characterise sets of states)
- analogous test commands: $(p, 1)$ where p is a test
- this admits a domain operation on commands:

$$\mathsf{dom}\, k = (\mathsf{grd}.k, 1)$$

where, as usual,

$$\mathsf{grd}.(a, p) \stackrel{\text{def}}{=} \neg \mathsf{wp}.(a, p).0 = p \rightarrow \mathsf{dom}\, a$$

- corresponding box operator

$$[\mathsf{k}]\,(\mathsf{q},1) \;=\; (\mathsf{wp.k.q},1)$$

- this equation explains the title of this section:
  wp *is nothing but* wlp *in the semiring of commands*
- except for fail the usual wp/wlp laws are just general laws for box operators
- moreover, we can re-use the general soundness and relative completeness proof for propositional Hoare logic from [Möller/Struth 04]
- this yields fairly quickly a sound and relatively complete proof system for wp

refinement relation:

$$(a, p) \sqsupseteq (b, q) \stackrel{\text{def}}{\Leftrightarrow} q \leq p \bigwedge q \wedge a \leq b$$

- $\sqsupseteq$ is a preorder
- associated equivalence:

$$(a, p) \equiv (b, q) \Leftrightarrow p = q \bigwedge p \wedge a = p \wedge b$$

4 Relation to UTP

- UTP specs and programs are predicates relating initial values v of variables with their final values v'
- $ok \leftrightarrow$ program has been started
- $ok' \leftrightarrow$ program has terminated
- both may occur freely in predicates

- set of all such predicates is too general
- subclass: *designs*

$$\mathrm{P} \vdash \mathrm{Q} \ \overset{\text{def}}{\Leftrightarrow}\ (ok \wedge \mathrm{P} \Rightarrow ok' \wedge \mathrm{Q})$$

  where ok and ok' do not occur in P or Q
- informal meaning: a computation is allowed by the design iff when started in a state satisfying P it will terminate in a state satisfying Q

- still narrower subclass: *normal* (or (H3)) designs
- where the precondition P may involve only initial values
- such a predicate is formally called a *condition*
- an (H3) design $p \vdash a$ can be modelled as the command (a, p)
- (actually as an equivalence class under refinement equivalence)
- the more general *normal prescriptions* of [Dunne 01] correspond precisely to the set of all commands (without a quotient formation)

- *feasible* (or (H4)) designs model programs that cannot "recover" from nontermination
- characterised by $\mathsf{chaos} \,;\, \mathrm{k} = \mathsf{chaos}$
- equivalent to Parnas's condition $\mathrm{p} \leq \mathsf{dom}\, \mathrm{a}$

general UTP predicates:

- can be modelled as 2×2-matrices that record the residual predicates for the four possible combinations of the values of ok and ok' [Möller 06]
- in this way the unobservables ok and ok' are truly hidden
- choice then becomes matrix addition
- and $;$ becomes matrix multiplication
- designs and prescriptions correspond to matrices of special shapes, from which many of the relevant laws can be derived more simply and concisely than from the original predicative specifications

5 Relation to Demonic Semantics

demonic semantics is a simplification of the general command semantics for feasible commands

- projection: $(a, p) \mapsto (p \wedge a,\ p \wedge \mathsf{dom}\, a)$
- for such commands the termination information coincides with the domain of the first component,
- hence can be omitted
- i.e., $(a, p) \mapsto p \wedge a$ suffices
- inverse operation (up to refinement equivalence) $b \mapsto (b, \mathsf{dom}\, b)$

- this is the view of demonic semantics:
- all states that have the possibility of triggering a non-terminating computation are considered “unsafe”, and hence all “proper” transitions for them are deleted as well
- hence all such states are excluded from the domain of the corresponding semantic element
- this means that the transition part alone is sufficient
- the demonic operators can now be *calculated* from the command versions using the above projection/injection pair [Guttmann/Möller 06]

6 Iteration and Demonic Refinement Algebra

- finite/infinite iteration: (left) Kleene/ω algebra
- DRA: strong iteration (finite or infinite iteration)
- connection [Höfner/Möller/Solin 06]

DRA	$=$	left ω algebra $+$ chaos is a left zero
strong iteration	$=$	$* \; + \; \omega$

- in particular, the commands form a DRA
- this can be non-extensional, hence not isomorphic to a predicate transformer model
- therefore the DRA axioms do not characterise predicate transformer models uniquely