On testing and specifying

Dusko Pavlovic
(work in progress with Bart Jacobs)
September 2005

Some approaches to software development:

correct-by-construction:

spec $\rightarrow$ refine $\rightarrow \ldots \rightarrow$ prog $\surd$

hacking:

prog $\rightarrow$ test $\rightarrow$ prog $\rightarrow$ test $\ldots$

development cycle:

spec $\rightarrow$ refine $\rightarrow$ prog $\rightarrow$ test $\rightarrow$ spec $\ldots$

Testing goals:

correctness: "Does system satisfy spec?"

- "How closely?"

assurance:

bugs $\Rightarrow$

- "How bad?"
- "How likely am I to find more?"

no bugs $\Rightarrow$

- "How good?"
  - "How much assurance do I get from 20 tests?"
  - "Which tests are more informative?"

Task:

Define experimental method for software science:

- experiment design techniques for
  - test suites
  - blind sampling
- statistical data analysis for
  - *quantitative* view of
  - *computational* behaviors

Outline

1. Testing frameworks

2. Examples

3. Behaviors and representation

1. Testing frameworks

Facets of testing.

$$\text{Syst} \times \text{Test} \xrightarrow{\models} \text{Obs}$$

testing equivalence* on Syst:

$$R \sim S \qquad \iff \qquad \forall t \in \text{Test}.\ (R \models t) = (S \models t)$$

debugging:

$$\begin{aligned} R \#_\varepsilon S \qquad &\iff \qquad \exists bug \in \text{Test}.\ (R \models bug) > (S \models bug) + \varepsilon \vee \\ & \qquad\qquad \exists req \in \text{Test}.\ (R \models req) + \varepsilon < (S \models req) \\ &\iff \qquad \exists t.\ (R \models t) \notin \left[(S \models t) - \varepsilon,\ (S \models t) + \varepsilon\right] \end{aligned}$$

authentication:

$$\exists s_A \in \text{Test}.\ X \models s_A \implies X = A$$

*e.g., system R satisfies spec S
("real function" vs "ideal functionality")

1.1. Systems

given

- category $\mathcal{S}$ of "(state) spaces"
- monad $R : \mathcal{S} \longrightarrow \mathcal{S}$ of "next-state spaces"

represent

- systems as G-coalgebras $X \longrightarrow GX$ for
  - reactive (read): $G_r X = RX^A$
  - generating (write): $G_w X = R(A \times X)$
  - read-write: $G_{rw} X = R(A \times X)^A$
- We test reactive systems as the final G-coalgebra

$$\mathsf{Syst} \;=\; \nu X.\ RX^A$$

1.2. Tests

given

- category $\mathcal{T}$ of "(data) types"
- monad $L : \mathcal{S} \longrightarrow \mathcal{S}$ of "test algebras"
  - pointed by $1 \xrightarrow{0} L$

represent

- tests (for reactive systems) as the elements of F-algebras $FX \longrightarrow X$ for
  - $F_\ell X = LX + A \times X$
- type of tests as the initial F-algebra

$$\mathsf{Test} \;=\; \mu X.\ LX + A \times X$$

1.3. Connections and duality

Def. A *connection* is a contravariant adjunction

$$\mathsf{M} \dashv \mathsf{P} \;:\; \mathcal{S}^{op} \longrightarrow \mathcal{T}$$

where

- $\mathsf{P}X \subseteq \mathsf{Obs}^X$ represents a "type of predicates over the space X",

- $\mathsf{M}Y \subseteq \mathsf{Obs}^Y$ represents a "space of models over the type Y", i.e.

  – the underlying sets of predicates $\mathsf{P}X$ and of models $\mathsf{M}Y$ consist of functions $X \to \mathsf{Obs}_{\mathcal{T}}$ and $Y \to \mathsf{Obs}_{\mathcal{S}}$ respectively,

  – the space $\mathsf{Obs}_{\mathcal{S}} \in \mathcal{S}$ and the type $\mathsf{Obs}_{\mathcal{T}} \in \mathcal{T}$ have the same underlying set Obs of "observations"*

  (continued...)

*In $\mathcal{T}$ they form the type "propositions" or "truth values". In $\mathcal{S}$ they form the space of "coordinates".

. . .

- $\mathsf{Obs}_{\mathcal{S}}$ has an L-algebra structure,

- $\mathsf{Obs}_{\mathcal{T}}$ has an R-algebra structure.

A connection is a *duality* if it is an equivalence.

Examples of connections.

1. $\wp^{op} \dashv \wp \ : \ \mathsf{Set}^{op} \longrightarrow \mathsf{Set}$

2. Stone duality

3. $pt \dashv \mathcal{O} : \mathsf{Esp}^{op} \longrightarrow \mathsf{Frm}$

4. $C \dashv S : \mathsf{Esp}^{op} \longrightarrow \mathsf{Rng}$

5. Priestley duality, and the various lattice correspondences

6. Scott duality: injective spaces and domains

$\boxed{\textbf{1.4 Behaviors}}$

Def. A *testing framework* consists of

- a system monad $R : \mathcal{S} \longrightarrow \mathcal{S}$
- a test monad $L : \mathcal{T} \longrightarrow \mathcal{T}$, and
- a connection $\mathsf{M} \dashv \mathsf{P} : \mathcal{S}^{op} \longrightarrow \mathcal{T}$

Def. For a given testing framework, with the final coalgebra Syst of systems and the initial algebra Test of tests, the space Behv of *behaviors* is defined by

$$\begin{array}{c}
\begin{tikzcd} \mathsf{Test} \arrow[r, "\mathrel{=\!|}"] & \mathsf{P(Syst)} \subseteq \mathsf{Obs}^{\mathsf{Syst}} \end{tikzcd} \\
\hline
\begin{tikzcd} \mathsf{Syst} \arrow[r, "\models"] \arrow[d, two heads] & \mathsf{M(Test)} \subseteq \mathsf{Obs}^{\mathsf{Test}} \\ \mathsf{Behv} \arrow[ur, hook] & \end{tikzcd}
\end{array}$$

Since the test algebra is $\mathsf{Test} = \mu X.\ LX + A \times X$, a test t must be in the form

$$t \ ::= \ c \mid f(t_0 \ldots t_n) \mid a.t$$

where c is a constant and f an operation from the signature of the algebraic theory of the monad L.

Testing semantics $\models$ is defined by combining induction over Test and coinduction over Syst

$$
\begin{aligned}
\Big(P \models c\Big) &= c \\
\Big(P \models f(t_0 \ldots t_n)\Big) &= f\Big((P \models t_0) \ldots (P \models t_n)\Big) \\
\Big(P \models a.t\Big) &= \Big(\varrho(P, a) \models t\Big)
\end{aligned}
$$

where

$$\varrho : \mathsf{Syst} \times A \longrightarrow R(\mathsf{Syst})$$

is (the transpose of) the final G-coalgebra structure on Syst, and $\models$ extends along

$$
\begin{tikzcd}
\mathsf{Syst} \arrow[rr, "\eta"] \arrow[dr, "\models"'] & & R(\mathsf{Syst}) \arrow[dl, dashed, "\models"] \\
& \mathsf{Obs}^{\mathsf{Test}} &
\end{tikzcd}
$$

because Obs is an R-algebra.

1.5 Metrics

Indistinguishability = testing equivalence

$$R \sim S \iff \forall t \in \textsf{Test}.\ (R \models t) = (S \models t)$$

refines to

$$d(R,S) \;=\; \bigvee_{t \in \textsf{Test}} \Big|(R \models t) - (S \models t)\Big|$$

($R \#_\varepsilon S$ becomes $d(R,S) > \varepsilon \ldots$)

2. Examples

2.1. Linear time – branching time

With

$$\begin{aligned} \mathcal{S} &= \mathsf{Set}_{<\omega} \\ R &= \wp : \mathsf{Set}_{<\omega} \longrightarrow \mathsf{Set}_{<\omega} \end{aligned}$$

we capture possibilistic nondeterminism*

$$X \longrightarrow (\wp X)^A$$

where A is a fixed set of actions.

The space of systems

$$\mathsf{Syst} = {}_A\mathsf{HSet}_{<\omega}$$

consists of finite A-labelled hypersets.

(It lives in Set, not in $\mathsf{Set}_{<\omega}$.)

*reading = writing, because $(\wp X)^A \cong \wp(A \times X)$

Moreover, in all of the following examples, take

$$\begin{aligned}
\mathcal{T} &= \mathsf{Set}_{<\omega} \\
\mathsf{Obs} &= 2 \ = \ \{0,1\} \text{ and} \\
T &= \wp \ : \ {\mathsf{Set}_{<\omega}}^{op} \longrightarrow \mathsf{Set}_{<\omega} \\
S &= \wp^{op} \dashv \wp
\end{aligned}$$

2.1.1. Testing with traces: $LX = 1 = \{\langle\rangle\}$

$$\begin{aligned} \mathsf{Test} &= A^* \\ (P \models \langle\rangle) &= 1 \\ (P \models a.t) &= \bigvee_{Q \in \varrho(P,a)} (Q \models t) \end{aligned}$$

2.1.2. . . . complete traces: $LX = \{\langle\rangle\}$ again, but A is extended to $A + \kappa$, i.e.

$$\text{Test} \;=\; (A+\kappa)^*$$

Extend each system by a final state $\surd$, so that each run must be completed by κ:

$$\frac{X \times A \xrightarrow{\varrho} \wp X}{(X+\surd) \times (A+\kappa) \xrightarrow{\varrho_\kappa} \wp(X+\surd)}$$

by setting

$$\varrho_\kappa(P,a) = \begin{cases} \varrho(P,a) & \text{if } P \in X \wedge a \in A \\ \{\surd\} & \text{if } P \in X \wedge a = \kappa \wedge \vec{P} = \emptyset \\ \emptyset & \text{otherwise} \end{cases}$$

where $\vec{P} = \{a \in A \mid \varrho(P,a) \neq \emptyset\}$.

The semantics definition

$$\begin{aligned}(P \models \langle\rangle) &= 1 \\ (P \models a.t) &= \bigvee_{Q \in \varrho_\kappa(P,a)} (Q \models t)\end{aligned}$$

now unfolds to

$$(P \models \kappa.t) = \begin{cases}(\surd \models t) & \text{if } \vec{P} = \emptyset \\ 0 & \text{otherwise}\end{cases}$$

where

$$(\surd \models t) = \begin{cases}1 & \text{if } t = \langle\rangle \\ \bigvee \emptyset = 0 & \text{otherwise}\end{cases}$$

i.e. to

$$
\begin{aligned}
(P \models \langle\rangle) &= 1 \\
(P \models a.t) &= \bigvee_{Q \in \varrho(P,a)} (Q \models t) \\
(P \models \kappa.t) &= \begin{cases} 1 & \text{if } \vec{P} = \emptyset \wedge t = \langle\rangle \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

2.1.3. Failures $LX = \{\langle\rangle\}$ again, but A is extended to $A + \wp A$, i.e.

$$\text{Test} \;=\; (A + \wp A)^*$$

The final state $\surd$ is now reached reached by testing, at the end of a run, by a failure set $\alpha \in \wp A$:

$$\frac{X \times A \xrightarrow{\varrho} \wp X}{(X + \surd) \times (A + \wp A) \xrightarrow{\varrho_{fail}} \wp(X + \surd)}$$

by setting

$$\varrho_{fail}(P, \alpha) = \begin{cases} \varrho(P, \alpha) & \text{if } P \in X \wedge \alpha \in A \\ \{\surd\} & \text{if } P \in X \wedge \alpha \in \wp A \wedge \alpha \cap \vec{P} = \emptyset \\ \emptyset & \text{otherwise} \end{cases}$$

where $\vec{P} = \{a \in A \mid \varrho(P, a) \neq \emptyset\}$.

The semantics definition

$$
\begin{aligned}
(P \models \langle\rangle) &= 1 \\
(P \models \alpha.t) &= \bigvee_{Q \in \varrho_{fail}(P,\alpha)} (Q \models t)
\end{aligned}
$$

now unfolds to

$$
(P \models \alpha.t) = \begin{cases}
\bigvee_{Q \in \varrho(P,\alpha)} (Q \models t) & \text{if } \alpha \in A \\
1 & \text{if } \begin{cases} \alpha \in \wp A \ \wedge \\ \alpha \cap \vec{P} = \emptyset \ \wedge \\ t = \langle\rangle \end{cases} \\
0 & \text{otherwise}
\end{cases}
$$

2.1.4. Refusal $LX = \{\langle\rangle\}$ again, but A is extended to $A + \wp A$, i.e.

$$\text{Test} \;=\; (A + \wp A)^*$$

Test systems not only by the accepted actions $a \in A$, but also by the refused sets $\alpha \in \wp A$:

$$\frac{X \times A \xrightarrow{\varrho} \wp X}{X \times (A + \wp A) \xrightarrow{\varrho_{ref}} \wp X}$$

by setting

$$\varrho_{ref}(P, \alpha) = \begin{cases} \varrho(P, \alpha) & \text{if } \alpha \in A \\ \{P\} & \text{if } \alpha \in \wp A \wedge \;\; \alpha \cap \vec{P} = \emptyset \\ \emptyset & \text{otherwise} \end{cases}$$

where $\vec{P} = \{a \in A \mid \varrho(P, a) \neq \emptyset\}$.

The semantics definition

$$\begin{array}{rcl} (P \models \langle\rangle) & = & 1 \\ (P \models \alpha.t) & = & \displaystyle\bigvee_{Q \in \varrho_{ref}(P,\alpha)} (Q \models t) \end{array}$$

now unfolds to

$$(P \models \alpha.t) \;=\; \begin{cases} \bigvee_{Q \in \varrho(P,\alpha)} (Q \models t) & \text{if } \alpha \in A \\ (P \models t) & \text{if } \begin{cases} \alpha \in \wp A \;\wedge \\ \alpha \cap \vec{P} = \emptyset \end{cases} \\ 0 & \text{otherwise} \end{cases}$$

2.1.5. Acceptance-refusal $LX = \{\langle\rangle\}$ again, but A is extended to $A + 2 \times \wp A$, i.e.

$$\text{Test} \;=\; (A + 2 \times \wp A)^*$$

Extend each system to test it not only by the accepted actions $a \in A$, but also by the refused sets $\alpha \in \{0\} \times \wp A$ and by the accepted sets $\alpha \in \{1\} \times \wp A$:

$$\frac{X \times A \xrightarrow{\varrho} \wp X}{X \times (A + 2 \times \wp A) \xrightarrow{\varrho_{ar}} \wp X}$$

by setting

$$\varrho_{ar}(P, \alpha) = \begin{cases} \varrho(P, \alpha) & \text{if } \alpha \in A \\ \{P\} & \text{if } \begin{cases} \alpha = \langle 0, \alpha' \rangle \;\wedge\; \alpha' \cap \vec{P} = \emptyset \\ \text{or } \alpha = \langle 1, \alpha' \rangle \;\wedge\; \alpha' \subseteq \vec{P} \end{cases} \\ \emptyset & \text{otherwise} \end{cases}$$

where $\vec{P} = \{a \in A \mid \varrho(P, a) \neq \emptyset\}$.

The semantics definition

$$
\begin{array}{rcl}
(P \models \langle\rangle) & = & 1 \\
(P \models \alpha.t) & = & \displaystyle\bigvee_{Q \in \varrho_{ar}(P,\alpha)} (Q \models t)
\end{array}
$$

now unfolds to

$$
(P \models \alpha.t) \;=\; \begin{cases}
\bigvee_{Q \in \varrho(P,\alpha)} (Q \models t) & \text{if } \alpha \in A \\
(P \models t) & \text{if } \begin{cases} \alpha = \langle 0, \alpha' \rangle \\ \wedge\ \alpha' \cap \vec{P} = \emptyset \\ \text{or } \alpha = \langle 1, \alpha' \rangle \\ \wedge\ \alpha' \subseteq \vec{P} \end{cases} \\
0 & \text{otherwise}
\end{cases}
$$

2.1.6. Simulation testing $LX = \wp X$

$$
\begin{aligned}
\text{Test} &= A - \text{edge labelled sets}^* \\
&= \text{positive Hennessy-Milner formulas}
\end{aligned}
$$

$$
\begin{aligned}
(P \models \emptyset) &= 1 \\
(P \models \{t_1 \dots t_n\}) &= \bigwedge_{i=1}^{n} (P \models t_i) \\
(P \models a.t) &= \bigvee_{Q \in \varrho(P,a)} (Q \models t)
\end{aligned}
$$

2.1.7. Bisimulation testing $LX = 2 \times \wp X$

$$
\begin{aligned}
\text{Test} \;&=\; A - \text{edge labelled} \\
&\;\; 2 - \text{node labelled sets} \\
&=\; \text{Hennessy-Milner formulas}
\end{aligned}
$$

$$
\begin{aligned}
(P \models \langle \iota, \emptyset \rangle) &= \iota \\
(P \models \langle \iota, \{t_1 \ldots t_n\} \rangle) &= \iota \oplus \bigwedge_{i=1}^{n} (P \models t_i) \\
(P \models a.t) &= \bigvee_{Q \in \varrho(P,a)} (Q \models t)
\end{aligned}
$$

2.2. Probabilistic systems

$$X \times A \xrightarrow{\varrho} \mathcal{V}X$$

where for finite X

$$\mathcal{V}X \;=\; \{\mu : X \to [0,1] \mid \sum_{x \in X} \mu(x) \leq 1\}$$

or for general measurable X, and $\mathcal{V} : \mathsf{Mes} \longrightarrow \mathsf{Mes}$

$$\mathcal{V}X \;=\; \{\mu : \mathcal{O}(X) \longrightarrow [0,1] \mid \mu(X) \leq 1\}$$

2.2.1. Possibilistic observations

Reduce finite $X \times A \longrightarrow \mathcal{V}X$ to the framework

$$\begin{aligned} \mathcal{S} \ = \ \mathcal{T} \ &= \ \mathsf{Set}_{<\omega} \\ S \dashv T \ &= \ \wp^{op} \dashv \wp : {\mathsf{Set}_{<\omega}}^{op} \longrightarrow \mathsf{Set}_{<\omega} \\ \mathsf{Obs} \ &= \ 2 \\ R \ &= \ \wp : \mathsf{Set}_{<\omega} \longrightarrow \mathsf{Set}_{<\omega} \end{aligned}$$

by setting

$$\frac{X \times A \xrightarrow{\ \varrho\ } \mathcal{V}X}{X \times A \times [0,1] \xrightarrow{\ \overline{\varrho}\ } \wp X}$$

$$\overline{\varrho}(P, a, p) \ = \ \{Q \in X \mid \varrho(P, a)(Q) \geq p\}$$

With the labels from $A \times [0,1]$ and $LX = \wp X$, we get

$$\text{Test} \;=\; A \times [0,1] - \text{edge labelled sets}$$

and semantics

$$\begin{aligned}
(P \models \emptyset) &= 1 \\
(P \models \{t_1 \ldots t_n\}) &= \bigwedge_{i=1}^{n} (P \models t_i) \\
(P \models \langle a, p \rangle . t) &= \bigvee_{Q \in \overline{\varrho}(P,a,p)} (Q \models t) \\
&= \bigvee_{\varrho(P,a)(Q) \geq p} (Q \models t)
\end{aligned}$$

2.2.2. Probabilistic observations

For $\mathcal{V}$: Mes$\longrightarrow$Mes and Obs $= [0,1]$, testing by $LX = \mathcal{M}X$ suffices:

$$\text{Test} \;=\; A - \text{edge labelled wf-trees}$$

and semantics

$$\begin{aligned}
(P \models \emptyset) &= 1 \\
(P \models \{t_1 \dots t_n\}) &= \prod_{i=1}^{n} (P \models t_i) \\
(P \models a.t) &= \sum_{Q \in X} (Q \models t) \varrho(P,a)(Q)
\end{aligned}$$

2.2.2. Probabilistic observations

For $\mathcal{V} : \mathsf{Mes} \longrightarrow \mathsf{Mes}$ and Obs $= [0,1]$, testing by $LX = \mathcal{M}X$ suffices:

$$\text{Test} \;=\; A - \text{edge labelled wf-trees}$$

and semantics

$$\begin{aligned}
(P \models \emptyset) &= 1 \\
(P \models \{t_1 \ldots t_n\}) &= \prod_{i=1}^{n} (P \models t_i) \\
(P \models a.t) &= \int_{Q \in X} (Q \models t)\, d\varrho_{(P,a)}
\end{aligned}$$

Remarkably,

$$\begin{array}{rcl} \forall t \in \mathsf{Test}_{poss}.\ (R \models t) & = & (S \models t) \\ & \Updownarrow & \\ \forall t \in \mathsf{Test}_{prob}.\ (R \models t) & = & (S \models t) \end{array}$$

although, of course

$$\begin{array}{c} d_{poss}.(R, S) \\ \neq \\ d_{prob}(R, S) \end{array}$$

2.3. Cryptanalysis as testing

Secrecy is indistinguishability

2.4. Quantum systems

3. Behaviors and representation

$$\frac{\mathsf{Test} \xrightarrow{\;\Dashv\;} \mathsf{Obs}^{\mathsf{Syst}}}{\begin{tikzcd}[column sep=small] \mathsf{Syst} \arrow[rr, "\models"] \arrow[dr] & & \mathsf{Obs}^{\mathsf{Test}} \\ & \mathsf{Behv} \arrow[ur, hook] & \end{tikzcd}}$$

Thm. [FoSSACS04] The bisimilarity classes of probabilistic systems (LMPs) correspond to the monoid homomorphisms $\mathsf{Test}\longrightarrow[0,1]$.

The category of LMPs is dual to the category of PMLs.

Proof sketches. [FoSSACS] Generate the free C^*-algebra over the monoid Test and use Stone-Gelfand duality. (The states of a probabilistic system are the characters of this C^*-algebra. Their weak topology is compact Hausdorff.)

[Soft proof.] Use $LX = \mathbb{Z}[X]$ and develop testing framework. . .

Gaussian error estimate (central limit theorem) to determine how much more testing is needed for how much assurance.