Probabilità e Nondeterminismo nella teoria dei domini

Daniele Varacca

ENS, Paris

BRICS, Aarhus

Parma, 15 giugno 2004

Road Map

- Motivation
- Domain theory and denotational semantics
- Categorical Semantics and Monads
- Combining monads: Indexed valuations
- Orders on indexed valuation

The Nature of Computation

It's a wild world out there:

- complex needs: precision, speed, security
- complex tools: languages, protocols, networks, channels

What do computers do, really?

Semantics

Semantics is a tool to find our way

- programs are associated to mathematical objects (the meaning)
- mathematical objects are parts of structures (the models)

Goal: study the models to understand the programs

Semantics

Requirements for the models

- models should be complex enough to reflect interesting properties of real computation
- models should be simple enough to be studied

Right balance between abstraction and complexity

Compositionality

The meaning of a complex system is built using the meanings of simpler parts of the system.

Probability

A model is probabilistic when

- different alternatives are represented
- every alternative is assigned a probability

We need probability

- to model failures and unreliability
- to produce efficient algorithm
- to enhance security

Nondeterminism

A model is nondeterministic when

- different alternatives are represented
- the way one alternative is chosen over the others is not specified

We need nondeterminism

- to abstract away from details
- to achieve compositionality
- to model concurrency (sometimes)

Road Map

- Motivation
- Domain theory and denotational semantics
- Categorical Semantics and Monads
- Combining monads: Indexed valuations
- Orders on indexed valuation

Denotational semantics

Idea: programs take in input values of type X and return values of type Y
Programs are represented as function

$$X \xrightarrow{p} Y$$

How do we represent recursive programs?

$$fact(n) := \textbf{if } n == 0 \textbf{ then } 1 \textbf{ else } n \times fact(n-1)$$

Domain theory

(Scott-Strachey late ’60): recursive functions are fixed points of operators

$$\Xi(f)(n) := \textbf{if } n == 0 \textbf{ then } 1 \textbf{ else } n \times f(n-1)$$

Domain: a framework where fixed points exist

Domains

DCPO: a partial order $\langle D, \leq \rangle$ where every directed subset has a least upper bound.

Continuous function $f : D \to D'$ - monotone function that preserves least upper bounds of directed sets.

The Category of DCPO's and continuoud functions.

Fact: every continuous function $f : D \to D$ has a (least) fixed point.

Scott topology

Continuous function $f : D \longrightarrow D'$ = Continuous with respect to the Scott Topology.

Scott closed sets: downward closed sets, closed by least upper bounds of directed sets.
Scott open sets: upward closed sets, “inaccessible” from below.

This topology is T_0 but not T_1.

Road Map

- Motivation
- Domain theory and denotational semantics
- Categorical Semantics and Monads
- Combining monads: Indexed valuations
- Orders on indexed valuation

Categorical semantics

Idea: programs take in input values of type X and return values of type Y
Programs are represented as arrows

$$X \xrightarrow{p} Y$$

Programs compose:

$$\frac{X \xrightarrow{p} Y, Y \xrightarrow{q} Z}{X \xrightarrow{p;q} Z}$$

Examples: sets and functions — DCPOs and continuous functions

Computational effects

Programs can have effects

- nontermination
- nondeterminism
- probability
- permanent state

Idea: programs take in input values of type X and return computations of type Y

Monads

A monad consists of

- the operator T: if X represents values, $T(X)$ represents computations
- the unit $X \xrightarrow{\eta_X} T(X)$: values are special kinds of computations
- the extension $(-)^\dagger$: if $X \xrightarrow{f} T(Y)$, then $T(X) \xrightarrow{f^\dagger} T(Y)$

satisfying some axioms

Monads

- The extension models sequential composition:

$$\frac{X \xrightarrow{f} T(Y),\ Y \xrightarrow{g} T(Z)}{X \xrightarrow{f} T(Y) \xrightarrow{g^{\dagger}} T(Z)}$$

- Specific effects have also specific operations

Monads: examples

The nondeterministic monad

- the operator is the finite nonempty powerset $P(X)$
- the unit: $x \mapsto \{x\}$
- the extension: if $X \xrightarrow{f} P(Y)$, and if $A \in P(X)$, then

$$f^{\dagger}(A) = \bigcup_{x \in A} f(x)$$

A program returns a set of values

Choice is modelled by union of sets

Monads: examples

The probabilistic monad

- the finite valuation operator $V(X)$: functions $f : X \to \overline{\mathbb{R}^+}$ such that $f(x) \neq 0$ for only finitely many x
- the unit $x \mapsto \delta_x$
- the extension is obtained by weighted average

A program returns a valuation over values
Probabilistic choice $x +_p y$ is modelled by convex combination

$$x +_p x = x$$

Road Map

- Motivation
- Domain theory and denotational semantics
- Categorical Semantics and Monads
- Combining monads: indexed valuations
- Orders on indexed valuation

Combining monads

We want a program return a set of valuations.

- the operator is the the composition of the operators $P(V(X))$
- the unit is the composition of the units
- the extension

Combining monads

We want a program return a set of valuations.

- the operator is the the composition of the operators $P(V(X))$
- the unit is the composition of the units
- the extension does not work!

We need to change something

Solutions

Solution 1 (Tix, Mislove): change the sets.
Use convex sets of distributions
The operator $P_{convex} \circ V$ form a monad

Solutions

Solution 1 (Tix, Mislove): change the sets.
Use convex sets of distributions
The operator $P_{convex} \circ V$ form a monad

Solution 2 (Varacca, Winskel): change the valuations.
Use indexed valuations

Indexed valuations

An indexed valuation over a set X is given by

- a (finite) indexing set I
- an indexing function $ind : I \to X$
- a valuation over I

Notation

$$\nu = (x_i, p_i)_{i \in I}$$

Indexed valuations

Intuition:

- elements of X represent observations
- elements of I represent computations
- the indexing function associates computations to observations

The probability on observations is gotten from the computations

Like in a... random variable!

The monad

We have a monad:

- the operator is $IV(X) = \{$indexed valuations over $X\}$
- the unit: $x \mapsto (x, 1)_{* \in \{*\}}$
- the extension is obtained via dosjoint union of indices

The probabilistic choice is modelled by duplicating the indices

$$x +_p x \neq x$$

Composing the monads

We can compose IV and P and get a monad
We can now model a programming language with

- sequential composition
- nondeterministic choice
- probabilistic choice

Ordering

Which order on indexed valuations?
Idea: the equation $x +_p x = x$ is weakened to
$x +_p x \geq x$
Morally: the more indices, the better
This combines well with the Hoare order on sets
$A \cup B \geq A$

$\leadsto$ domain theory

Semantics

A language

$$c ::= \mathbf{skip} \mid X := a \mid c; c \mid \mathbf{if}\ b\ \mathbf{then}\ c\ \mathbf{else}\ c\,.$$

States: contents of the locations
Semantics

$$[\![c]\!] : \Sigma \to \Sigma$$

Semantics

A nondeterministic language

$$c ::= \ldots \mid c \textbf{ or } c\,.$$

Semantics

$$[\![c]\!] : \Sigma \to P(\Sigma)$$

Semantics

A probabilistic language

$$c ::= \ldots \mid X := random\,.$$

Semantics

$$[\![c]\!] : \Sigma \to V(\Sigma)$$

Semantics

A nondeterministic and probabilistic language

$$c ::= \ldots \mid c \textbf{ or } c \mid X := random\,.$$

Semantics

$$[\![c]\!] : \Sigma \to P(V(\Sigma))$$

Semantics

A nondeterministic and probabilistic language

$$c ::= \ldots \mid c \ \mathbf{or} \ c \mid X := random\,.$$

Semantics

$$[\![c]\!] : \Sigma \to P(V(\Sigma))$$

Semantics

A nondeterministic and probabilistic language

$$c ::= \ldots \mid c \ \textbf{or} \ c \mid X := random\,.$$

Semantics

$$[\![c]\!] : \Sigma \to P(IV(\Sigma))$$

Semantics

A language with recursion

$$c ::= \ldots \mid \mathbf{while}\ b\ \mathbf{do}\ c\,.$$

$\leadsto$ domain theory

Other issues

- equational theories
- other orderings
- relation with continuous valuations
- the problem of the concrete characterization
- operational intuition

Related work

- CSP group in Oxford
- Tix' thesis and Mislove's paper
- Plotkin, Power, Hyland on monads and operations

Future work

- concrete characterization
- presheaf models?

Grazie