# Domain Theory

Hongfei Fu

John Hopcroft Center for Computer Science
Department of Computer Science
Shanghai Jiao Tong University

# Domain Theory

- advanced constructions on complete partial orders (cpo's)
- a meta-language for cpo's and continuous functions

# Today's Topic

- a meta-language for cpo's and continuous functions

# A Metalanguage

# A Metalanguage

## Motivation

- an programming-language-like syntax for continuous functions
- guaranteed continuity from the syntax

# A Metalanguage

## $\lambda$-Notation

- $D, E$: cpo's
- $x$: a variable representing an element in $D$
- $e$: an expression that represents an element in $E$ (e.g. $x + 1$)

# A Metalanguage

## $\lambda$-Notation

- $D, E$: cpo's
- $x$: a variable representing an element in $D$
- $e$: an expression that represents an element in $E$ (e.g. $x + 1$)

We use the notation

$$\lambda x \in D.e \text{ (or simply } \lambda x.e)$$

for the function $h : D \to E$ such that $h(d) := e\,[d/x]$ for all $d \in D$.

# A Metalanguage

## $\lambda$ Notation

- $D_1, D_2, E$: cpo's
- $e$: an expression with variables $x, y$

# A Metalanguage

## $\lambda$ Notation

- $D_1, D_2, E$: cpo's
- $e$: an expression with variables $x, y$

We write

- $\lambda z \in D_1 \times D_2.(e[\pi_1(z)/x, \pi_2(z)/y])$

# A Metalanguage

## $\lambda$ Notation

- $D_1, D_2, E$: cpo's
- $e$: an expression with variables $x, y$

We write

- $\lambda z \in D_1 \times D_2.(e[\pi_1(z)/x, \pi_2(z)/y])$
- $\lambda(x, y) \in D_1 \times D_2.e$

# A Metalanguage

## $\lambda$ Notation

- $D_1, D_2, E$: cpo's
- $e$: an expression with variables $x, y$

We write

- $\lambda z \in D_1 \times D_2.(e[\pi_1(z)/x, \pi_2(z)/y])$
- $\lambda(x, y) \in D_1 \times D_2.e$
- $\lambda x \in D_1, y \in D_2.e$

# A Metalanguage

## $\lambda$ Notation

- $D_1, D_2, E$: cpo's
- $e$: an expression with variables $x, y$

We write

- $\lambda z \in D_1 \times D_2.(e[\pi_1(z)/x, \pi_2(z)/y])$
- $\lambda(x, y) \in D_1 \times D_2.e$
- $\lambda x \in D_1, y \in D_2.e$
- $\lambda x, y.e$

# A Metalanguage

## Continuity of Expressions

- $D, E$: cpo's
- $e$: an expression representing an element in $E$
- $x$: a variable ranging over elements from $D$

The expression $e$ is continuous in the variable $x$ if the function

$$\lambda x \in D.e : D \to E$$

is continuous no matter which values the other free variables take.

# A Metalanguage

## Continuity of Expressions

- $D, E$: cpo's
- $e$: an expression representing an element in $E$

Then $e$ is continuous in its variables if $e$ is continuous in all its variables.

# A Metalanguage

## The Roadmap

- expressions for continuous functions
- recursive construction

# A Metalanguage

## Variables

Each single variable $x$ is continuous in its variables.

# A Metalanguage

## Variables

Each single variable $x$ is continuous in its variables.

## Proof

- $\lambda x.x$ (the identity function)

# A Metalanguage

## Variables

Each single variable $x$ is continuous in its variables.

## Proof

- $\lambda x.x$ (the identity function)
- $\lambda y.x$ $(y \neq x)$ (a constant function)

# A Metalanguage

## Constants

Constant expressions are continuous in their variables since they represent constant functions.

# A Metalanguage

## Constants

Constant expressions are continuous in their variables since they represent constant functions.

## Examples

- a bottom element $\bot_D$ of a cpo $D$
- truth values **true**, **false**
- projections functions $\pi_i$'s
- function application *apply*
- the operator $(-)^*$
- ...

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$
- $(e_1, \ldots, e_k)$: the tuple expression for elements of $E_1 \times \cdots \times E_k$

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$
- $(e_1, \ldots, e_k)$: the tuple expression for elements of $E_1 \times \cdots \times E_k$

Then the expression $(e_1, \ldots, e_k)$ is continuous in its variables iff every $e_i$ is continuous in its variables.

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$
- $(e_1, \ldots, e_k)$: the tuple expression for elements of $E_1 \times \cdots \times E_k$

Then the expression $(e_1, \ldots, e_k)$ is continuous in its variables iff every $e_i$ is continuous in its variables.

## Proof

For all variables $x$, we have

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$
- $(e_1, \ldots, e_k)$: the tuple expression for elements of $E_1 \times \cdots \times E_k$

Then the expression $(e_1, \ldots, e_k)$ is continuous in its variables iff every $e_i$ is continuous in its variables.

## Proof

For all variables $x$, we have

$$(e_1, \ldots, e_k) \text{ is continuous in } x$$

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$
- $(e_1, \ldots, e_k)$: the tuple expression for elements of $E_1 \times \cdots \times E_k$

Then the expression $(e_1, \ldots, e_k)$ is continuous in its variables iff every $e_i$ is continuous in its variables.

## Proof

For all variables $x$, we have

$$\begin{aligned} & (e_1, \ldots, e_k) \text{ is continuous in } x \\ \Leftrightarrow \quad & \lambda x.(e_1, \ldots, e_k) \text{ is continuous} \end{aligned}$$

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$
- $(e_1, \ldots, e_k)$: the tuple expression for elements of $E_1 \times \cdots \times E_k$

Then the expression $(e_1, \ldots, e_k)$ is continuous in its variables iff every $e_i$ is continuous in its variables.

## Proof

For all variables $x$, we have

$$
\begin{aligned}
& (e_1, \ldots, e_k) \text{ is continuous in } x \\
\Leftrightarrow \quad & \lambda x.(e_1, \ldots, e_k) \text{ is continuous} \\
\Leftrightarrow \quad & \pi_i \circ (\lambda x.(e_1, \ldots, e_k)) \text{ is continuous for all } i
\end{aligned}
$$

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$
- $(e_1, \ldots, e_k)$: the tuple expression for elements of $E_1 \times \cdots \times E_k$

Then the expression $(e_1, \ldots, e_k)$ is continuous in its variables iff every $e_i$ is continuous in its variables.

## Proof

For all variables $x$, we have

$$
\begin{aligned}
& (e_1, \ldots, e_k) \text{ is continuous in } x \\
\Leftrightarrow \quad & \lambda x.(e_1, \ldots, e_k) \text{ is continuous} \\
\Leftrightarrow \quad & \pi_i \circ (\lambda x.(e_1, \ldots, e_k)) \text{ is continuous for all } i \\
\Leftrightarrow \quad & \lambda x.e_i \text{ is continuous for all } i
\end{aligned}
$$

# A Metalanguage

## Tupling

- $E_1, \ldots, E_k$: cpo's
- $e_i$ $(1 \leq i \leq k)$: expressions for elements of $E_i$
- $(e_1, \ldots, e_k)$: the tuple expression for elements of $E_1 \times \cdots \times E_k$

Then the expression $(e_1, \ldots, e_k)$ is continuous in its variables iff every $e_i$ is continuous in its variables.

## Proof

For all variables $x$, we have

$$
\begin{aligned}
& (e_1, \ldots, e_k) \text{ is continuous in } x \\
\Leftrightarrow \quad & \lambda x.(e_1, \ldots, e_k) \text{ is continuous} \\
\Leftrightarrow \quad & \pi_i \circ (\lambda x.(e_1, \ldots, e_k)) \text{ is continuous for all } i \\
\Leftrightarrow \quad & \lambda x.e_i \text{ is continuous for all } i \\
\Leftrightarrow \quad & e_i \text{ is continuous in } x \text{ for all } i
\end{aligned}
$$

# A Metalanguage

## Application

- $K$: a constant continuous function (e.g., $\pi_i$, *apply*)
- $e$: an expression

Then the expression $K(e)$ is continuous in its variables if the expression $e$ is continuous in its variables.

# A Metalanguage

## Application

- $K$: a constant continuous function (e.g., $\pi_i$, *apply*)
- $e$: an expression

Then the expression $K(e)$ is continuous in its variables if the expression $e$ is continuous in its variables.

## Proof

For all variables $x$, we have:

$$
\begin{aligned}
& K(e) \text{ is continuous in } x \\
\Leftrightarrow \quad & \lambda x.K(e) \text{ is continuous} \\
\Leftrightarrow \quad & K \circ (\lambda x.e) \text{ is continuous} \\
\Leftarrow \quad & \lambda x.e \text{ is continuous} \\
\Leftrightarrow \quad & e \text{ is continuous in } x
\end{aligned}
$$

# A Metalanguage

## Application

- $e_1, e_2$: two expressions

Then the expression $e_1(e_2)$ is continuous in its variables if both $e_1, e_2$ are continuous in their variables.

# A Metalanguage

## Application

- $e_1, e_2$: two expressions

Then the expression $e_1(e_2)$ is continuous in its variables if both $e_1, e_2$ are continuous in their variables.

## Proof

We have that $e_1(e_2) = \mathit{apply}(e_1, e_2)$, a composition of tupling and $\mathit{apply}$.

# A Metalanguage

## $\lambda$-Abstraction

- $D, E$: cpo's
- $e$: an expression that represents an element in $E$
- $y$: a variable

Then $\lambda y.e$ is continuous in its variables if the expression $e$ is continuous in its variables.

# A Metalanguage

## $\lambda$-Abstraction

- $D, E$: cpo's
- $e$: an expression that represents an element in $E$
- $y$: a variable

Then $\lambda y.e$ is continuous in its variables if the expression $e$ is continuous in its variables.

## Proof

For in all variables $x$, we have:

- if $x = y$ then $\lambda x.\lambda y.e$ is a constant function;

# A Metalanguage

## $\lambda$-Abstraction

- $D, E$: cpo's
- $e$: an expression that represents an element in $E$
- $y$: a variable

Then the expression $\lambda y.e$ is continuous in its variables if the expression $e$ is continuous in its variables.

## Proof

- if $x \neq y$ then

# A Metalanguage

## $\lambda$-Abstraction

- $D, E$: cpo's
- $e$: an expression that represents an element in $E$
- $y$: a variable

Then the expression $\lambda y.e$ is continuous in its variables if the expression $e$ is continuous in its variables.

## Proof

- if $x \neq y$ then

$$\lambda x.\lambda y.e \text{ is continuous}$$

# A Metalanguage

## $\lambda$-Abstraction

- $D, E$: cpo's
- $e$: an expression that represents an element in $E$
- $y$: a variable

Then the expression $\lambda y.e$ is continuous in its variables if the expression $e$ is continuous in its variables.

## Proof

- if $x \neq y$ then

$$\begin{aligned} & \lambda x.\lambda y.e \text{ is continuous} \\ \Leftrightarrow \quad & \mathit{curry}(\lambda x, y.e) \text{ is continuous} \end{aligned}$$

# A Metalanguage

## $\lambda$-Abstraction

- $D, E$: cpo's
- $e$: an expression that represents an element in $E$
- $y$: a variable

Then the expression $\lambda y.e$ is continuous in its variables if the expression $e$ is continuous in its variables.

## Proof

- if $x \neq y$ then

$$\begin{aligned} & \lambda x.\lambda y.e \text{ is continuous} \\ \Leftrightarrow \quad & \mathit{curry}(\lambda x, y.e) \text{ is continuous} \\ \Leftarrow \quad & \lambda x, y.e \text{ is continuous} \end{aligned}$$

# A Metalanguage

## $\lambda$-Abstraction

- $D, E$: cpo's
- $e$: an expression that represents an element in $E$
- $y$: a variable

Then the expression $\lambda y.e$ is continuous in its variables if the expression $e$ is continuous in its variables.

## Proof

- if $x \neq y$ then

$$
\begin{aligned}
& \lambda x.\lambda y.e \text{ is continuous} \\
\Leftrightarrow \quad & \mathit{curry}(\lambda x, y.e) \text{ is continuous} \\
\Leftarrow \quad & \lambda x, y.e \text{ is continuous} \\
\Leftrightarrow \quad & e \text{ is continuous in } x, y
\end{aligned}
$$

# A Metalanguage

## $\lambda$-Abstraction

- $e_1, e_2$: expressions
- $e_1 \circ e_2 := \lambda x.(e_1(e_2))$

# A Metalanguage

## $\lambda$-Abstraction

- $e_1, e_2$: expressions
- $e_1 \circ e_2 := \lambda x.(e_1(e_2))$

Then $e_1 \circ e_2$ is continuous in its variables if both $e_1, e_2$ are continuous in their variables.

# A Metalanguage

## *let*-Construction

- $D$: a cpo
- $E$: a cpo with bottom
- $e_1$: a expression representing an element in $D_\perp$
- $e_2$: a expression representing an element in $E$

# A Metalanguage

## *let*-Construction

- $D$: a cpo
- $E$: a cpo with bottom
- $e_1$: a expression representing an element in $D_\perp$
- $e_2$: a expression representing an element in $E$

Then the expression

$$let\ x \Leftarrow e_1.e_2$$

is continuous in its variables if both $e_1, e_2$ are continuous.

# A Metalanguage

## *let*-Construction

- $D$: a cpo
- $E$: a cpo with bottom
- $e_1$: a expression representing an element in $D_\perp$
- $e_2$: a expression representing an element in $E$

Then the expression

$$\mathit{let}\ x \Leftarrow e_1.e_2$$

is continuous in its variables if both $e_1, e_2$ are continuous.

## Proof

We have $\mathit{let}\ x \Leftarrow e_1.e_2 = (\lambda x.e_2)^*(e_1)$.

# A Metalanguage

## *case*-Construction

- $E, D_1, \ldots, D_k$: cpo's
- $e$: an expression representing an element in $D_1 + \cdots + D_k$
- $e_1, \ldots, e_k$: expressions representing elements in $E$

# A Metalanguage

## *case*-Construction

- $E, D_1, \ldots, D_k$: cpo's
- $e$: an expression representing an element in $D_1 + \cdots + D_k$
- $e_1, \ldots, e_k$: expressions representing elements in $E$

Then the case expression

$$\begin{aligned} &\textbf{case } e \textbf{ of } in_1(x_1).e_1 \mid \\ &\qquad\qquad \vdots \\ &\qquad\qquad in_k(x_k).e_k \end{aligned}$$

is continuous if all $e, e_1, \ldots, e_k$ are continuous.

# A Metalanguage

## *case*-Construction

- $E, D_1, \ldots, D_k$: cpo's
- $e$: an expression representing an element in $D_1 + \cdots + D_k$
- $e_1, \ldots, e_k$: expressions representing elements in $E$

Then the case expression

$$\begin{array}{l} \textbf{case } e \textbf{ of } in_1(x_1).e_1 \mid \\ \vdots \\ in_k(x_k).e_k \end{array}$$

is continuous if all $e, e_1, \ldots, e_k$ are continuous.

## Proof

The case expression is defined to be $[\lambda x_1.e_1, \ldots, \lambda x_k.e_k](e)$.

# A Metalanguage

## Fixed-Point Operator

- $D$: a cpo with a bottom element $\bot$
- $fix : [D \to D] \to D$: the least-fixed-point operator $f \mapsto fix(f)$

Then $fix$ is a continuous function (i.e. $fix \in [[D \to D] \to D]$).

# A Metalanguage

## Fixed-Point Operator

- $D$: a cpo with a bottom element $\bot$
- $fix : [D \to D] \to D$: the least-fixed-point operator $f \mapsto fix(f)$

Then $fix$ is a continuous function (i.e. $fix \in [[D \to D] \to D]$).

## Proof

We have

$$fix = \bigsqcup_{n \in \omega} (\lambda f . f^n(\bot))$$

where

$$\lambda f . \bot \sqsubseteq \lambda f . f(\bot) \sqsubseteq \cdots \sqsubseteq \lambda f . f^n(\bot) \sqsubseteq \ldots \quad .$$

is an $\omega$-chain of continuous functions in $[[D \to D] \to D]$ (why?).

# Inductive Construction of Continuous Expressions

## Fixed-Point Operator

- $D$: a cpo with a bottom element $\bot$
- $fix : [D \to D] \to D$: the least-fixed-point operator $f \mapsto fix(f)$
- $e$: an expression representing an element in $D$

We define $\mu x.e := fix(\lambda x.e)$ .

# Inductive Construction of Continuous Expressions

## Fixed-Point Operator

- $D$: a cpo with a bottom element $\bot$
- $fix : [D \to D] \to D$: the least-fixed-point operator $f \mapsto fix(f)$
- $e$: an expression representing an element in $D$

We define $\mu x.e := fix(\lambda x.e)$ .

## Proposition

The fixed-point expression $\mu x.e$ is continuous in its variables if $e$ is continuous in its variables.

# Summary

## A Metalanguage for Continuous Functions

- variables
- constants
- tupling
- application
- $\lambda$-abstraction
- *let*-construction
- case-construction
- fixed-point operator

Questions?