Logic and Languages

Robert Harper

Programming Languages Mentoring Workshop

Philadelphia, January 2012

Traditional Logic

- There are only 2 propositions, 0 and 1.
- Entailment: $A \leq B$ iff $A=1$ implies $B=1$.
  - A true iff $1 \leq A$ (i.e., $A=1$)
  - A false iff $A \leq 0$ (i.e., $A=0$)
- Truth tables define the connectives.

Boolean Algebra

Boolean Algebra

- Conjunction defines the *meet* (aka *glb*):

- Disjunction defines the *join* (aka *lub*).

- Complement: $C \leq \neg A$ iff $C \wedge A \leq 0$

Boolean Algebra

- Conjunction defines the *meet* (aka *glb*):

$$\begin{tikzcd} & A \wedge B \arrow[dl] \arrow[dr] & \\ A & & B \end{tikzcd}$$

- Disjunction defines the *join* (aka *lub*).
- Complement: $C \leq \neg A$ iff $C \wedge A \leq 0$

Boolean Algebra

- Conjunction defines the *meet* (aka *glb*):

$$\begin{tikzcd}
& C \arrow[ddl] \arrow[ddr] & \\
& A \wedge B \arrow[dl] \arrow[dr] & \\
A & & B
\end{tikzcd}$$

- Disjunction defines the *join* (aka *lub*).

- Complement: $C \leq \neg A$ iff $C \wedge A \leq 0$

Boolean Algebra

- Conjunction defines the *meet* (aka *glb*):

$$\begin{tikzcd}
& C \arrow[ddl] \arrow[d, dashed] \arrow[ddr] & \\
& A \wedge B \arrow[dl] \arrow[dr] & \\
A & & B
\end{tikzcd}$$

- Disjunction defines the *join* (aka *lub*).

- Complement: $C \leq \neg A$ iff $C \wedge A \leq 0$

Logic As If People Matter

- Standard logic gives no account of how knowledge is obtained or communicated.
  - A true iff there is a proof of A.
  - A false iff there is a refutation of A.
- But what is a proof?
- When are two proofs the same?

Truth as Provability

- Connectives are defined by *proof* conditions:
  - *Intro*: if A true and B true, then A∧B true.
  - *Elim*: if A∧B true, then A true and B true.
- A false means A true is refutable.
- A is *open* iff neither A true nor A false (i.e., A has neither a proof nor a refutation).

Entailment

- A_1 true, ..., A_n true $\vdash$ B true means there is a proof of B, given proofs of A_1, ..., A_n.
- Reflexivity / Identity: Γ, A true $\vdash$ A true
- Transitivity / Composition:
  If $\Gamma \vdash$ A true and Γ, A true $\vdash$ B true, then
  $\Gamma \vdash$ B true.
- Irrelevance / Weakening:
  If $\Gamma \vdash$ B true, then Γ, A true $\vdash$ B true.

Provability, Redux

- Rules may be expressed as entailments:
  - $A \wedge B$ true $\vdash$ A true
  - $A \wedge B$ true $\vdash$ B true
  - if C true $\vdash$ A true and C true $\vdash$ B true, then $C \vdash A \wedge B$ true
- Essentially a re-expression of meet conditions!

Structure of Proofs

- By instrumenting the provability rules we obtain a *grammar of proof*.
    - $M : A$ means M is a proof of A
    - Structure of M determined by form of A.
- More generally, $x_1 : A_1, \ldots, x_n : A_n \vdash M : A$ expresses entailment.

Variables: Algebra of Proof

- Assumption = Reflexivity:
  $\Gamma, x{:}A \vdash x : A.$
- Substitution = Transitivity:
  if $\Gamma, x{:}A \vdash M : B$ and $\Gamma \vdash N : A$, then
  $\Gamma \vdash [N/x]M : B$
- Proliferation = Weakening:
  if $\Gamma \vdash N : B$, then $\Gamma, x{:}A \vdash N : B.$

Structure of Proofs

- Proof objects for connectives:
  - $x : A \wedge B \vdash \text{fst } x : A$
  - $x : A \wedge B \vdash \text{snd } x : B$
  - if $\Gamma \vdash M : A$ and $\Gamma \vdash N : B$, then $\Gamma \vdash \langle M, N \rangle : A \wedge B$.
- And similarly for other connectives.

Proof Theory

- *Proof theory* is the study of these proof objects.
  - Considered boring among logicians.
  - Of the essence for computer scientists!
- Key idea: proofs are mathematical objects.
  - Mechanizable.
  - Computational.

Brouwer's Dictum

Logic is part of mathematics, rather than mathematics being derived from logic.

- The concept of a *construction* (*program!*) is the primitive notion.
- Proofs are particular constructions, which is to say programs.
- All mathematical objects are constructions.

Type Theory

- Types *classify* constructions (programs).
  - Specify a problem to be solved.
  - Categorize objects of study.
- Types encompass proofs *and* data.
  - All objects are classified by types.

Propositions as Types

	Proposition	*Type*	
truth	$\top$	1	*unit*
falsity	$\bot$	0	*void*
conjunction	$\wedge$	$\times$	*product*
disjunction	$\vee$	$+$	*sum*
implication	$\supset$	$\rightarrow$	*function*
		nat	*number*

Propositions as Types

	Proposition	*Type*	
negation	$\neg$	cont	*continuation*
universal	$\forall$	Π	*product*
existential	$\exists$	Σ	*sum*
necessity	$\Box$	$\Box$	*mobility*
possibility	$\Diamond$	$\Diamond$	*locality*
laxity	$\bigcirc A$	$\{A\}$	*monad*

Logic and Languages

- *First dictum:* logic and languages coincide.
    - Logical concepts suggest and inform language concepts, and *vice versa*.
    - Long-term goal is a grand unification of logic and computation.
- *Second dictum:* languages are for people, not computers.

Proof Equivalence

- When are two proofs (programs) the *same*?
  - $M = N : A$ means M and N are the same proof of A / program of type A
  - Reflexive, symmetric, and transitive, and a congruence.
- What are the principles of proof equivalence?

Gentzen's Principle

- Introduction and elimination are inverses.
  - $\mathrm{fst}\ \langle M,N\rangle = M : A$
  - $\mathrm{snd}\ \langle M,N\rangle = N : A$

  "local soundness"

  - $N = \langle \mathrm{fst}\ M, \mathrm{snd}\ N\rangle : A \wedge B$ "local completeness"
- Local soundness gives rise to an equational dynamics of proofs (execution as programs).

Proofs as Maps

- Proofs can be thought of as mappings.
  - $\Gamma \vdash M : A$ as map $M : \Gamma \to A$ (given by substitution)
  - Reflexivity: $id : A \to A$
  - Transitivity: $N \circ M : A \to C$
    if $M : A \to B$ and $N : B \to C$
- Generalizes pre-orders to categories.

Lawvere's Principle

- Maximality generalizes to universality:
  - $\langle M,N\rangle : \Gamma \to A \wedge B$ is "universal" among $M : \Gamma \to A$ and $N : \Gamma \to B$
- Pictorially:

Lawvere's Principle

- Maximality generalizes to universality:
  - $\langle M,N \rangle : \Gamma \to A \wedge B$ is "universal" among $M : \Gamma \to A$ and $N : \Gamma \to B$
- Pictorially:

$$\begin{tikzcd} & A \wedge B \arrow[dl, "fst"'] \arrow[dr, "snd"] & \\ A & & B \end{tikzcd}$$

Lawvere's Principle

- Maximality generalizes to universality:
  - $\langle M,N\rangle : \Gamma \to A \wedge B$ is "universal" among $M : \Gamma \to A$ and $N : \Gamma \to B$
- Pictorially:

$$\begin{tikzcd}
& \Gamma \arrow[ddl, "M"'] \arrow[ddr, "N"] & \\
& A \wedge B \arrow[dl, "fst"'] \arrow[dr, "snd"] & \\
A & & B
\end{tikzcd}$$

Lawvere's Principle

- Maximality generalizes to universality:
  - $\langle M,N \rangle : \Gamma \rightarrow A \wedge B$ is "universal" among $M : \Gamma \rightarrow A$ and $N : \Gamma \rightarrow B$
- Pictorially:

$$\begin{tikzcd}
 & \Gamma \arrow[ddl, "M"'] \arrow[ddr, "N"] \arrow[d, dashed, "!\ \langle M,N \rangle"] & \\
 & A \wedge B \arrow[dl, "\mathrm{fst}"] \arrow[dr, "\mathrm{snd}"'] & \\
A & & B
\end{tikzcd}$$

Lawvere's Principle

- Universality expresses Gentzen equivalences:
  - $\mathrm{fst} \circ \langle M, N \rangle = M : A \wedge B \to A$
  - $\mathrm{snd} \circ \langle M, N \rangle = N : A \wedge B \to B$
  - $M = \langle \mathrm{fst} \circ M, \mathrm{snd} \circ N \rangle : \Gamma \to A \wedge B$
- Yields an *algebra of proofs* (and data).

Equivalence as Structure

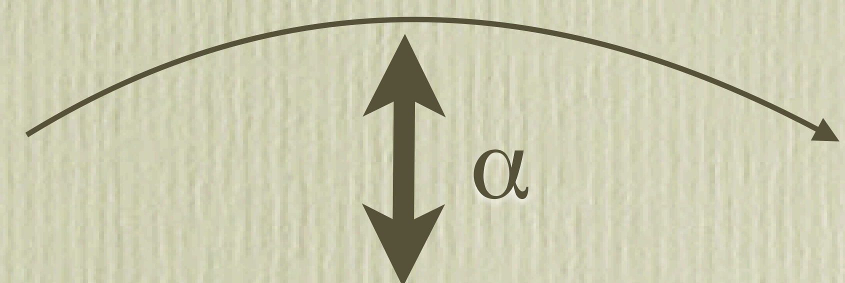
- Equivalences are symmetric preorders on maps.

$$\Updownarrow \alpha$$

- Maps can be equivalent for different reasons!
  - $\alpha :: M = N : A$ means α is evidence for equivalence of M and N in A.
  - “faces” or “2-cells”: maps between maps.

Equivalence as Structure

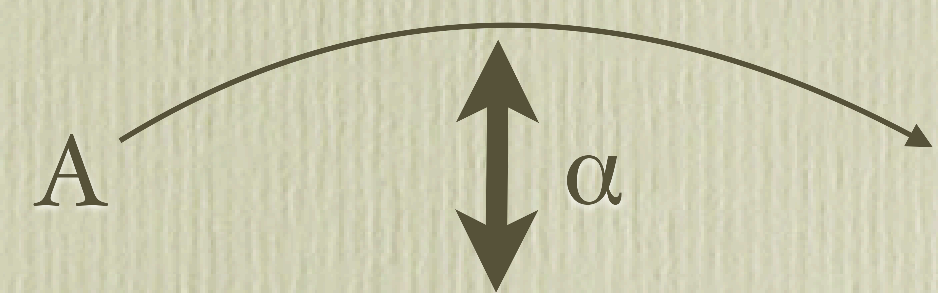
- Equivalences are symmetric preorders on maps.



- Maps can be equivalent for different reasons!
    - $\alpha :: M = N : A$ means α is evidence for equivalence of M and N in A.
    - “faces” or “2-cells”: maps between maps.

Equivalence as Structure

- Equivalences are symmetric preorders on maps.



- Maps can be equivalent for different reasons!
  - $\alpha :: M = N : A$ means α is evidence for equivalence of M and N in A.
  - "faces" or "2-cells": maps between maps.

Equivalence as Structure

- Equivalences are symmetric preorders on maps.

$$\begin{tikzcd} A \arrow[rr, bend left] \arrow[rr, phantom, "\Updownarrow \alpha"] & & B \end{tikzcd}$$

- Maps can be equivalent for different reasons!
  - $\alpha :: M = N : A$ means α is evidence for equivalence of M and N in A.
  - “faces” or “2-cells”: maps between maps.

Equivalence as Structure

- Equivalences are symmetric preorders on maps.

$$\begin{tikzcd} A \arrow[rr, bend left] \arrow[rr, bend right] \arrow[rr, phantom, "\Updownarrow\alpha"] & & B \end{tikzcd}$$

- Maps can be equivalent for different reasons!
  - $\alpha :: M = N : A$ means α is evidence for equivalence of M and N in A.
  - “faces” or “2-cells”: maps between maps.

Equivalence as Structure

- 2-cells form a *groupoid*:
  - Reflexive: $\mathrm{id} : M = M : A$
  - Transitive: $\beta \circ \alpha : M = P : A$ if $\alpha : N = P : A$ and $\beta : M = N : A$
  - Symmetric: $\alpha^{-1} : N = M : A$ if $\alpha : M = N : A$
- Groupoid is an "equivalence relation with evidence".

Equivalences of Equivalences

- Need equivalences between equivalences!
  - identity as unit of composition
  - associativity of composition
  - inverses compose to identity
- 3-cells witness equivalences of 2-cells, and so on through all dimensions.
  - "(weak) ∞-groupoid"

Topology of Proofs

- Consider a proposition to be a *space* of proofs.
  - M, N are "points" in the space.
- Equivalences are *paths* in the space.
  - $\alpha : M \rightarrow N : A$ deforms M into N
- Higher equivalences are *homotopies of paths*.
  - deformations of deformations

Functoriality

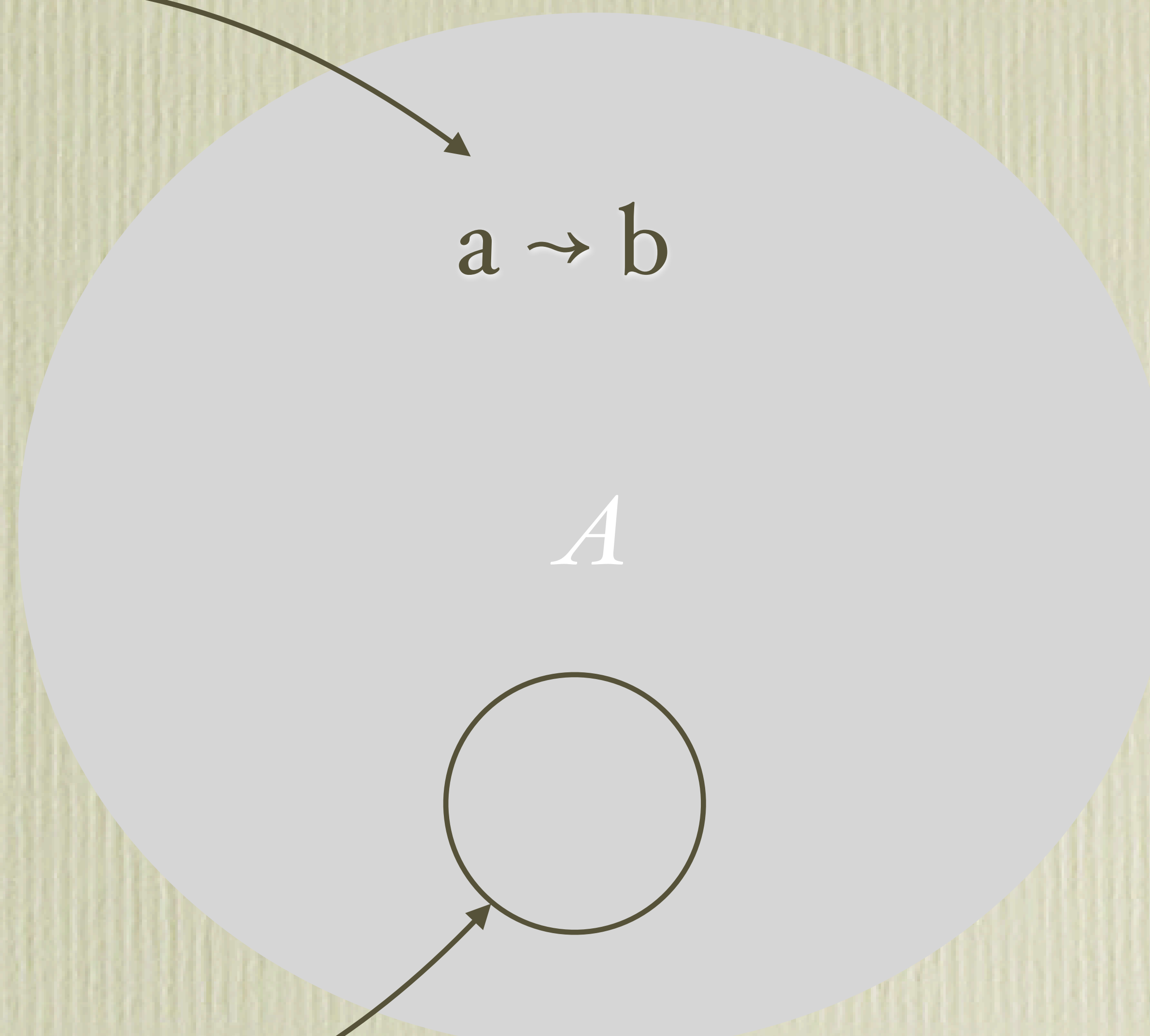
- Functionality: maps respect *equality*.
  - F M = F N : B   if M = N : A
- Functoriality: maps act on *equivalences*:
  - resp[α](F) : F M = F N : B   if α : M = N : A
  - action determined by α, a map between M and N in A
  - automatically respects composition, inverses

Higher Inductive Types

- Higher inductive definitions:
  - I type; $0, 1 : I$; $seg : 0 \rightsquigarrow 1 : I$
  - S^1 type; $b : S^1$; $loop : b \rightsquigarrow b : S^1$
- Program by pattern-matching:
  - $p : I \to A$ given by $p\ 0 = a$, $p\ 1 = b$, and $p\ seg = \alpha : a \rightsquigarrow b : A$
  - $c : S^1 \to A$ given by $c\ b = a$, $c\ loop = \alpha : b \rightsquigarrow b : A$

Higher Inductive Types

seg : $0 \to 1$



loop : $b \to b : S^1$

Logic, Types, Maps, Spaces

- *Third dictum:* logics and languages may be structured as *higher categories*, a natural setting for studying equivalences.

- *Fourth dictum:* you never know where logic will turn up next!

    - Personally, I was shocked by the natural connection to homotopy theory (though it's quite obvious once you see it).

The Holy Trinity of PL Research