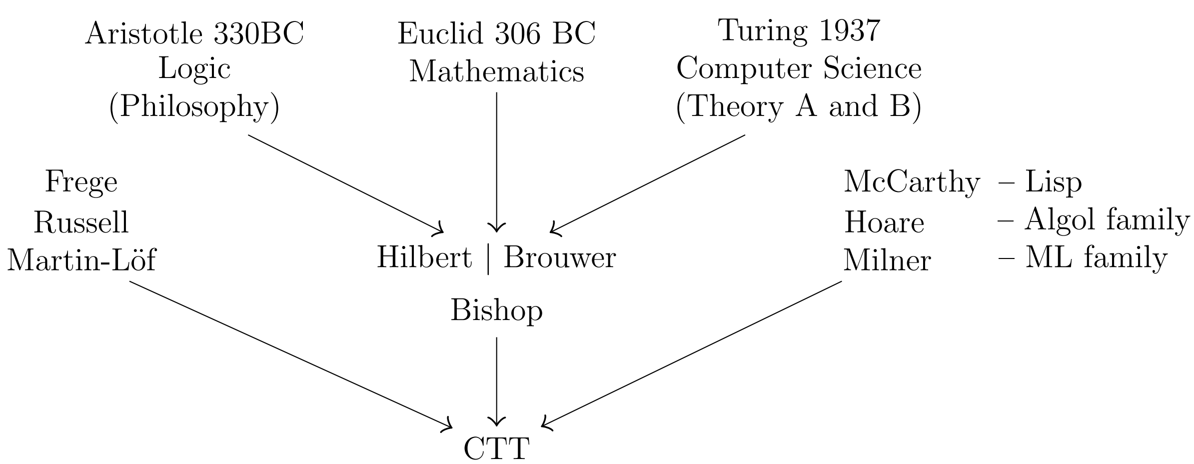
Advanced Progamming Languages **Lecture 35**
CS 6110 Spring 2015 **Wed. April 22, 2015**

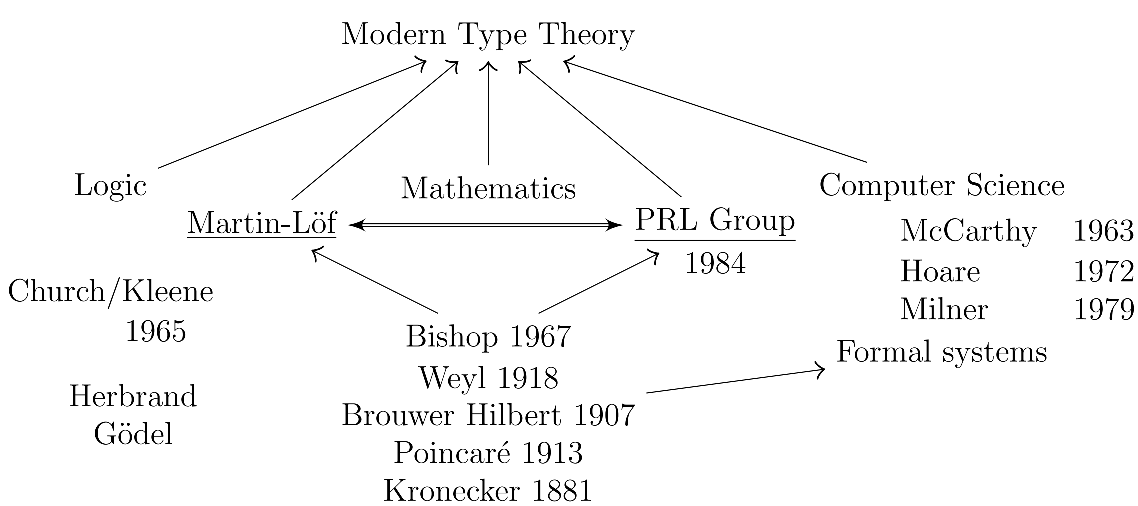
Lecture 35

Topics

1. There will be a small PS5 by Monday.
2. Background on type theory – historical, philosophical
3. Core of Nuprl's Constructive Type Theory (CTT) circa 2008 to 2014. The *CTT15* is "bolder", covers all of intuitionistic mathematics thanks to the current PRL group.

Background on Constructive type theories





Types in a sensible order – a dozen types

Universes	$\mathbb{U}_1,\ \mathbb{U}_2,\ \mathbb{U}_3, ...$		
	$universe\{j:\ell\}()$		
Empty type	$Void\{\}()$		
Atom	$Atom\{\}()$		
$\mathbb{Z}$	$int\{\}()$	the integers	
Equality	$equal\{\}(a;b;T)$	$a = b \in T$	
Less	$less\{\}(u;v)$	$u < v$	
Function	$function\{\}(A;x.B)$	$x : A \to B$	
	$function\{\}(A;B)$	$A \to B$	special case
Product	$product\{\}(A;x.B)$	$x : A \times B$	
	$product\{\}(A;B)$	$A \times B$	special case
Union	$union\{\}(A;B)$	$A + B$	
List	$list\{\}(A)$		
Set	$set\{\}(A;x.B)$	$\{x : A \vert B\}$	
Quotient	$quotient\{\}(A;x,y.E)$	$A//E(x,y)$	

In OCaml the type $A+B$ is called a *variant* and the types are labeled as in $x_1 : A_1 + x_2 : A_2 + ... + x_n : A_n$.

OCaml actually writes $x_1 : A_1 \,|\, x_2 : A_2 \,|\, ... \,|\, x_n : A_n$.

This special case is $inl : A \,|\, inr : B$.

Presentation of type theory – in Martin-Löf style

Canonical terms

Noncanonical terms

Based on Bishop's informal definition of a *constructive set* (1967).

The canonical terms are irreducable expressions, the primitive names for mathematical objects, e.g. 0, 1, -1, 2, *int*, ...

$0 = 0$ in *int*, $0 < 1$ are canonical "propositions".

The noncanonical terms can be reduced by *computation rules*,

e.g. $1 + 2 \to 3$

$apply(\lambda(x.0); 1) \to 0.$

To define a type we give it a canonical name and say what the canonical elements are *and what it means for two elements to be equal.* This is the Martin-Löf *semantic method.*

Nuprl uses a different method due to Stuart Allen [1, 2] and recast by Bob Harper [6]. Allen defines a type as a *partial equivalance relation* on terms. This is called the *per semantics*. This method was partially inspired by E. Bishops method of presenting constructive analysis.

Bishop *Foundations of Constructive Analysis* 1967 [3, 5, 4], p.2:

"To define a set we prescribe, at least implicitly, what we (the constructing intelligence) must do in order to construct an element of the set, and what we must do to show that two elements of the set are equal."

References

[1] Stuart F. Allen. A Non-type-theoretic Definition of Martin-Löf's Types. In D. Gries, editor, *Proceedings of the 2^{nd} IEEE Symposium on Logic in Computer Science*, pages 215–224. IEEE Computer Society Press, June 1987.

[2] Stuart F. Allen. *A Non-Type-Theoretic Semantics for Type-Theoretic Language*. PhD thesis, Cornell University, 1987.

[3] E. Bishop. *Foundations of Constructive Analysis*. McGraw Hill, NY, 1967.

[4] E. Bishop. Mathematics as a numerical language. In *Intuitionism and Proof Theory*, pages 53–71. North-Holland, NY, 1970.

[5] E. Bishop and D. Bridges. *Constructive Analysis*. Springer, New York, 1985.

[6] Robert Harper. Constructing type systems over an operational semantics. *J. Symbolic Computing*, 14(1):71–84, 1992.