# Building an Interactive Library of Formal Algorithmic Knowledge

Robert L. Constable

Cornell University

Hamilton, Ontario
June 2002

# Outline

- The ONR Digital Library Project
- Concepts for Formal Digital Library (FDL) design
- Current status of FDL
- Questions and issues

# Staff @ Cornell

**Direct**

- Robert Constable
- Stuart Allen
- Richard Eaton
- Lori Lorigo
- Christoph Kreitz
- Aleksey Nogin
- Eli Barzilay
- Amand Holland-Minkley
- Alexei Kopylov

**Indirect** (DARPA, IAI, Cornell)

- Mark Bickford (LPE/Cornell)
- Christoph Kreitz
- Robert Constable
- Amanda Holland-Minkley
- Brian Kulis
- Matt Fluet

PVS group help

Advisors:

| | |
|---|---|
| Arms | Kleinberg |
| Lagoze | Ginsparg |
| Demers | |

# Objective of ONR Program:

To create a digital library of algorithms and constructive mathematics useable for program and software construction.

# Characteristics of BAA

The research concentration areas have three aspects:

- Building infrastructure for a formal library of computational mathematics
- Creating formal content
- Applying formal content

# What Does “Formal” Mean?

The BAA refers to machine-checked mathematics presented in a consistent formal logical theory that is implemented.

This meaning of “formal” is technical. It is more narrow than what many people mean in daily use.

# Cornell, Cal Tech, Wyoming Proposal and Project

"Building Interactive Digital Libraries of Formal Algorithmic Knowledge"

# Goals

1. Build a semantics-based interactive logical library infrastructure
2. Create, collect and organize formal computational mathematics content
3. Apply the formal interactive DL in designing and creating reliable software (especially for CIP/SW)

# Benefits to Society

- Basis for highly reliable and responsive software
- Acceleration of scientific discovery

  mathematics

  computer science

  computational science

  metamathematics

- Wider access to content (participatory science)
- Topics in a new science of information

  formalized mathematics publication

  scholarly publication in general (arXiv)

  quantitative metamathematics

# Strategy

1. Attract a community of contributors who share formal knowledge and the connected mathematically literate articles

2. Account for correctness in a multi-logic, multi-prover (including tactic-style) environment

3. Provide semantics-based library services at many scales

## Challenges and Problems

1. Community using formal proofs is relatively small
    - Market for formal proofs is small
        - proof technology not widely used in software
        - proof technology not widely used in science and math
        - proof technology not widely used in education
    - Formal proving is still hard work
        - expansion factor
        - shallow base of basic mathematical facts
        - demanding skill set (programming + math + design)

# Challenges and Problems

2. Community is disconnected
   - Each group uses a different system
   - Almost no sharing (logical difficulties, practical ones)
   - Systems change or go extinct

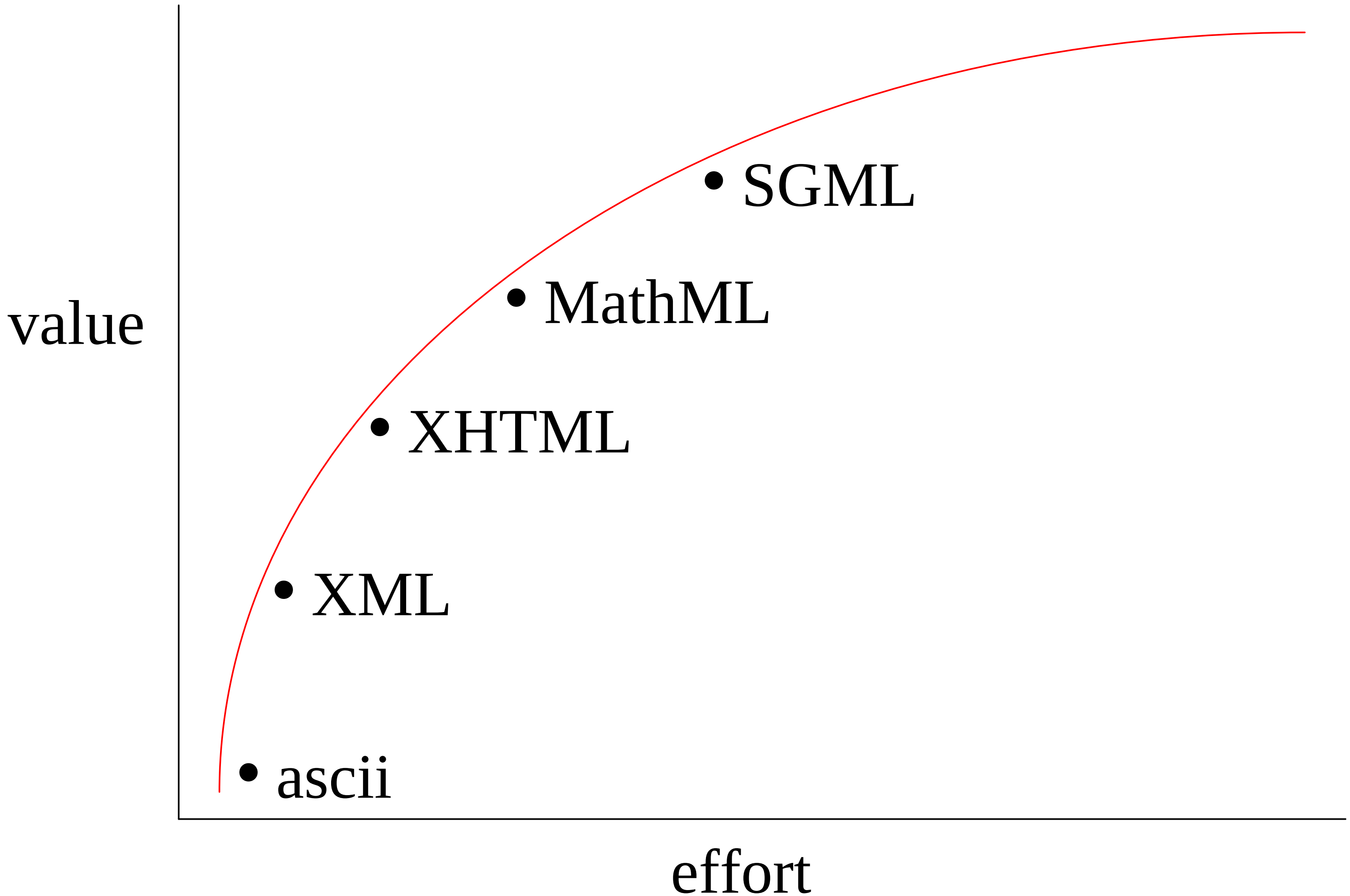
# Digital Library Approach to the Challenges

1. Widen the community by
    - library will increase the services provided
    - library will decrease the effort to create proofs (seen from experience)
2. Connect the community through a common service – the digital libraries approach

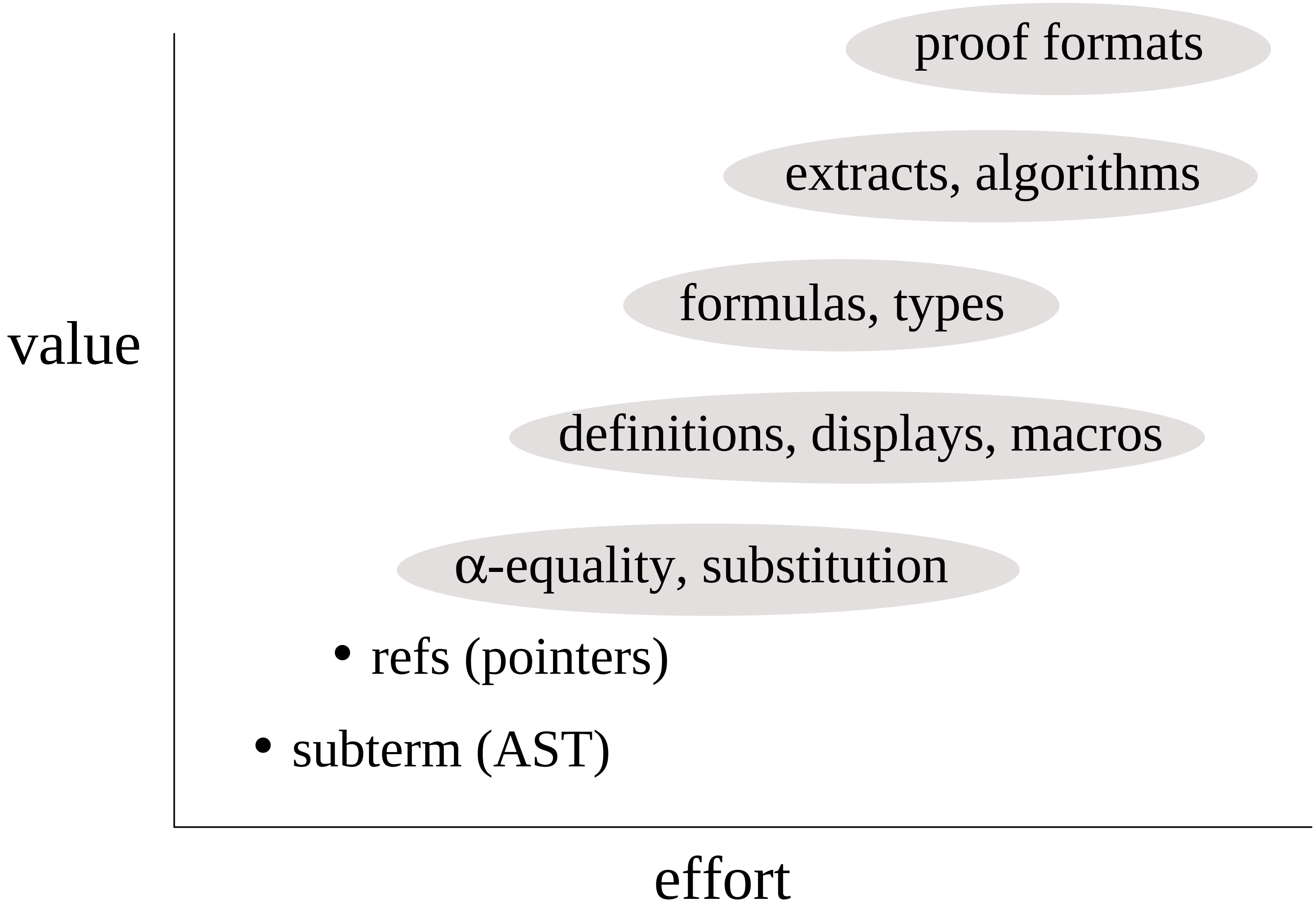
# Outline

- The ONR Digital Library Project
- Concepts for Formal Digital Library (FDL) design
- Current status of FDL
- Questions and issues

# DL Shared Data Formats



# Formal DL Data Formats



# Terms (Abstract Syntax Trees)

$t = op(t;\cdots;t)$ for $t$ a term

$Term = op \times Term\ List$

with binding structure

$op(\bar{v}_1.t_1;\cdots;\bar{v}_n.t_n)$ $\bar{v}_i$ list of binding variables

$Op = OpName\{i_1 : F_1;\cdots;i_k : F_k\}$

$i$ can be reference objects or values

# Conceptual Basis for Design and Implementation

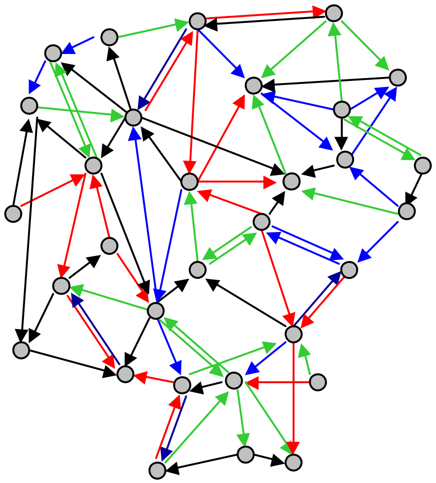
Important features

- Logical library keeps track of

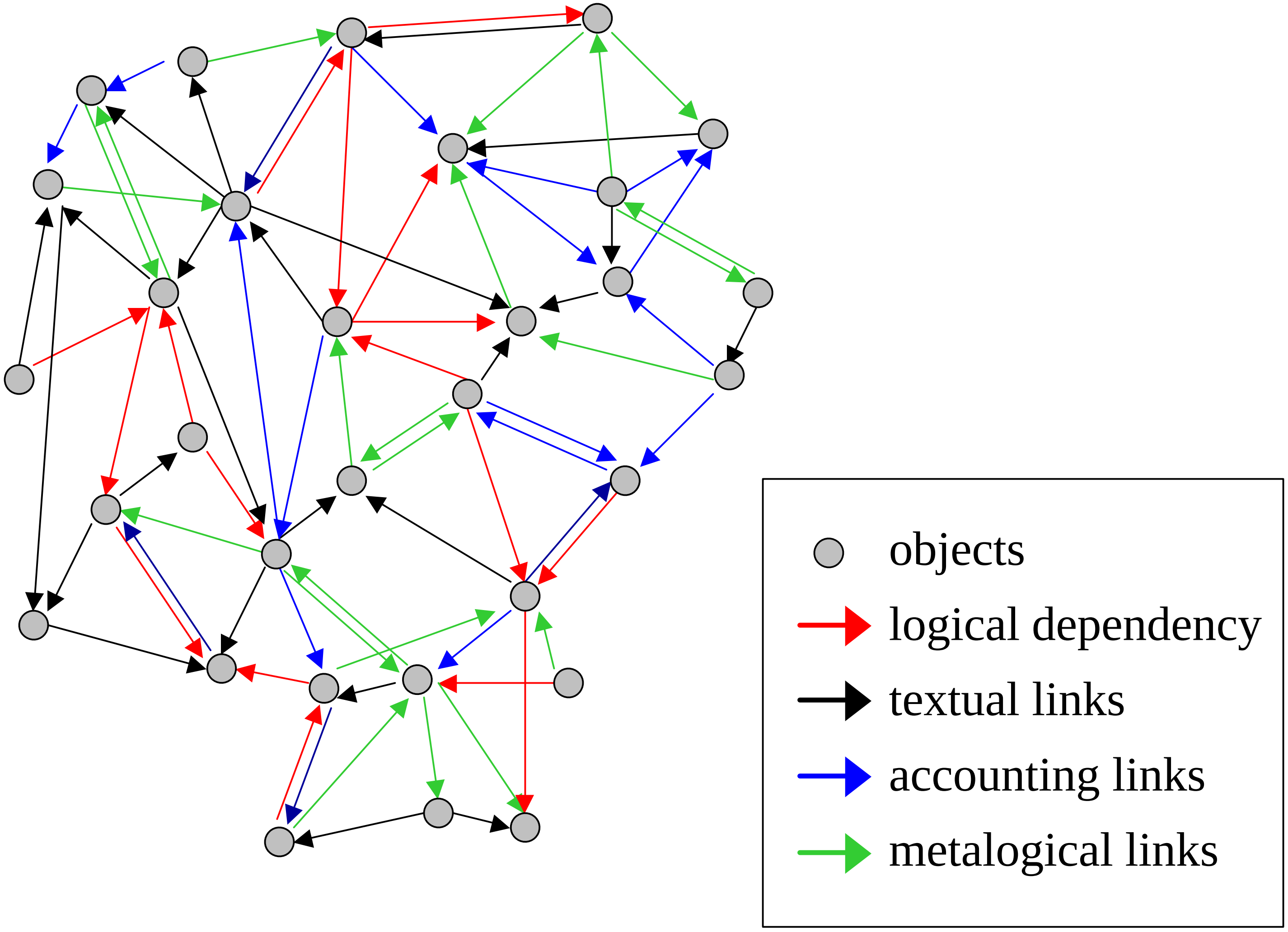
  evidence

  dependencies

  objects form a graph



# Information Graph of the FDL



# FDL contains formal objects

rules

definitions

algorithms, code

conjectures

specifications

theorems

inferences

proofs, partial proofs

certificates

# Inferences

$$\overline{H}_1 \vdash G_1 \text{ by } J_1,\ \overline{H}_2 \vdash G_2 \text{ by } J_2, \cdots, \overline{H}_n \vdash G_n \text{ by } J_n$$

$$\overline{H} \vdash G \text{ by } J$$

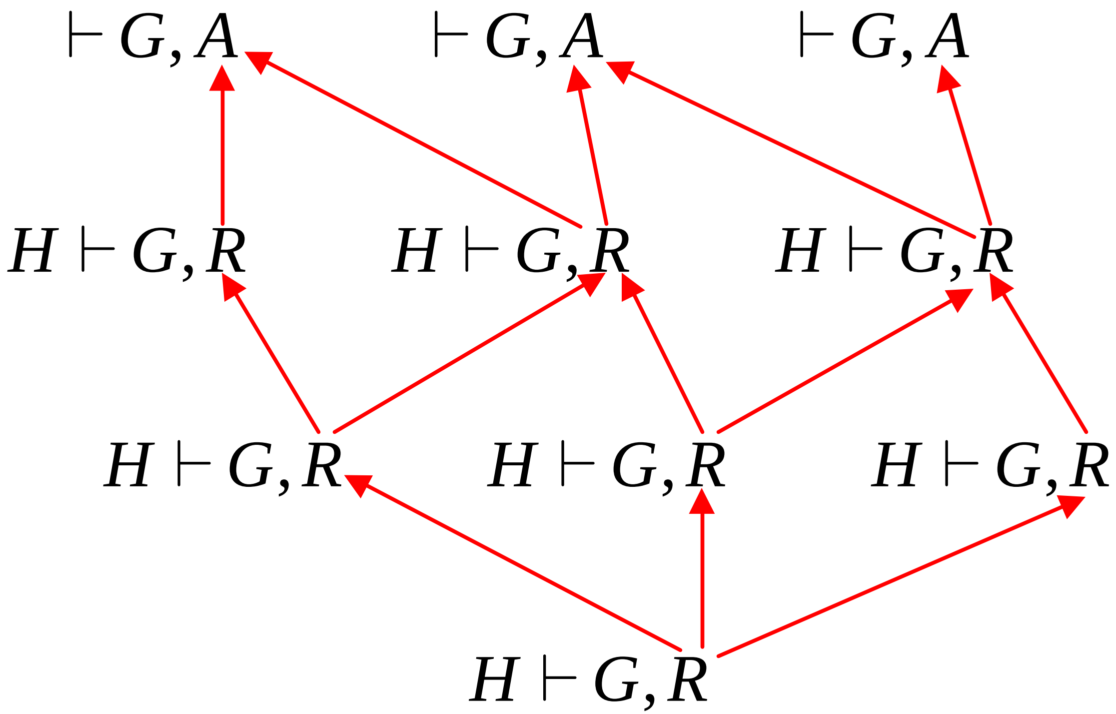
$\overline{H}_i$ a list of formulas (terms)

$G_i$ a formula (term)

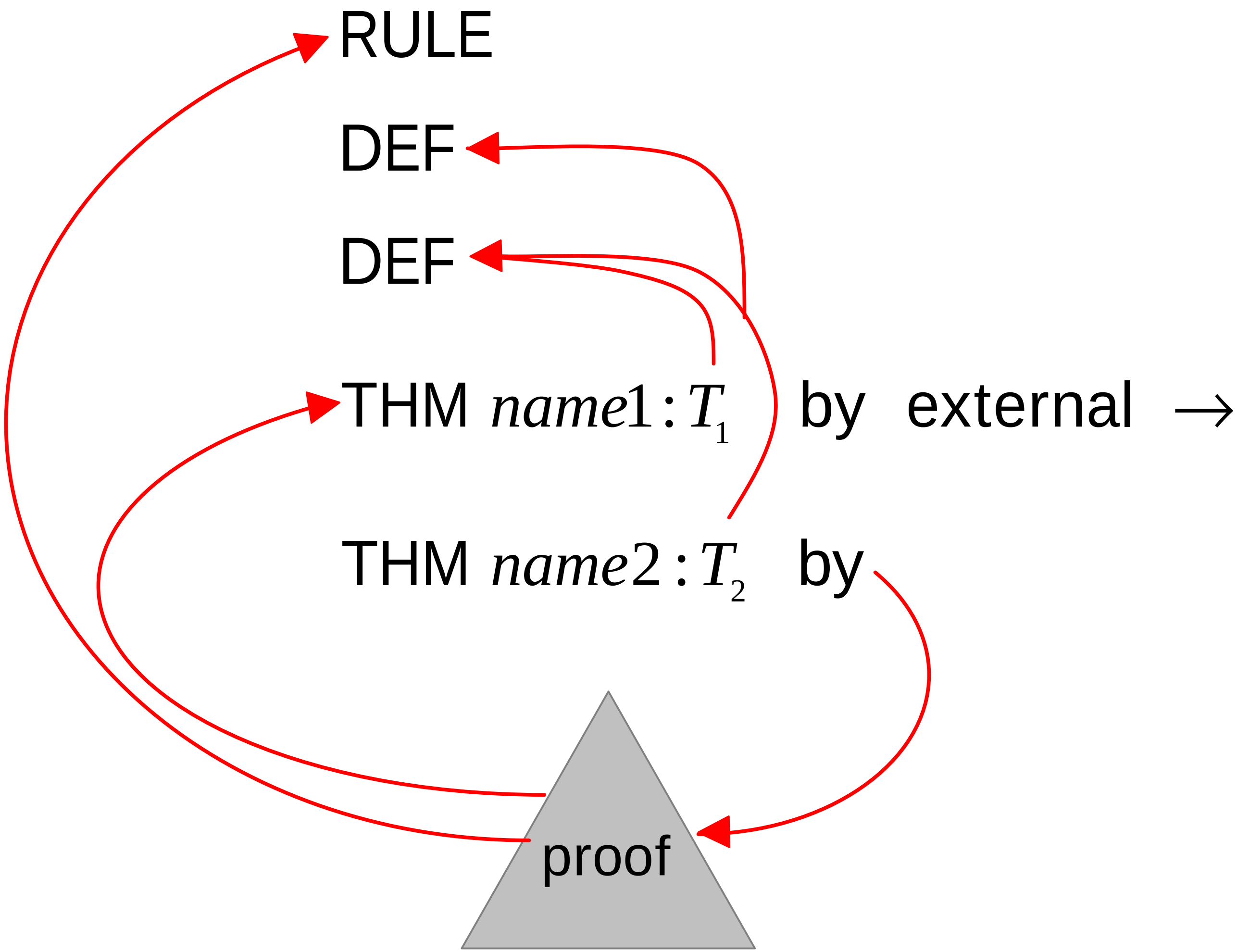
$J_i$ a justification (rule, tactic)

# Proof

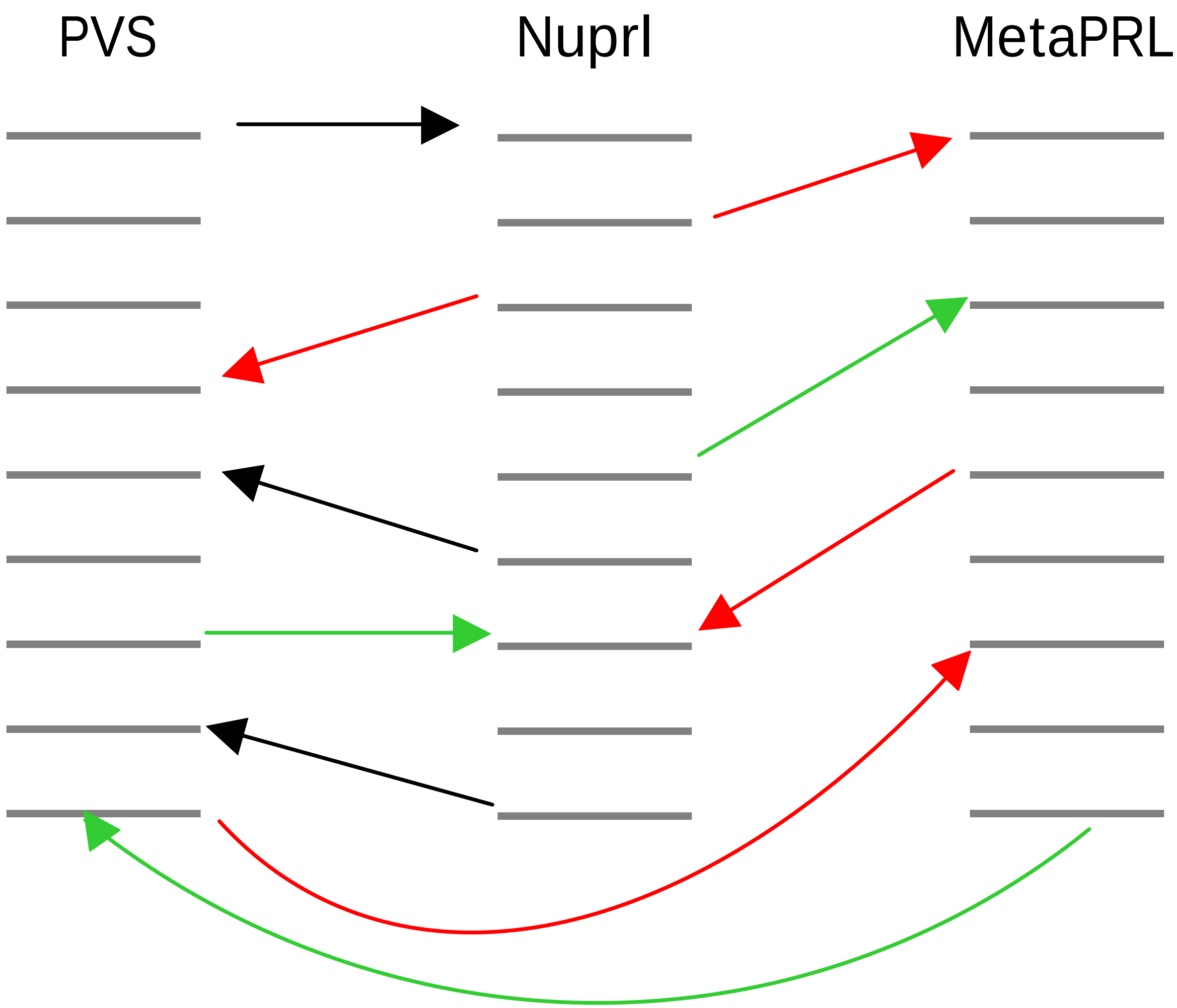
A proof is a dag of inferences



# Example of Dependencies



# FDL allows sharing among collections

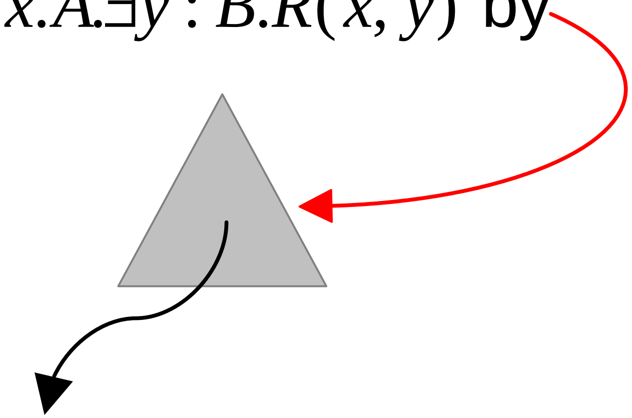


# FDL is interactive

- Can create new definitions, claims, conjectures
- Can interactively build proofs
- Can execute algorithms, extracts
- Can search for information
- Can display dependencies
- Can transform entire collections, theories

# FDL supports algorithmic mathematics

THM: $\forall x.A.\exists y:B.R(x,y)$ by



THM: $\exists f:A \to B.\forall x.A.R(x,f(x))$

THM: $\forall x:A.R(x,f_0(x))$

THM: $f_0$ in $\{g:A \to B \mid P(g)\}$

# Concepts for FDL Design

- FDL provides an experimental publication medium

  Can solicit exemplary contributions

  hybrid articles – formal and informal

  elegant formalizations

  challenging formalizations

  expository articles

  hypothetical formalizations

  Articles directly include shared material

# FDL performs archival functions

Automath system Auto QE checked the following formalization of Landau's Grundlagen (August 17, 2004).

Coq 5.0 created the following extract for the Fundamental Theorem of Algebra (June 14, 2003).

Nuprl 5 checked that Total Order (TO) protocol satisfies P (June 5, 2003).

MetaPRL compiler produced C code from TO, and P is preserved (October 19, 2003).

PVS 2.4 proved Menger's theorem (September 15, 2003).

# Concepts for FDL Design

- FDL supports large-scale operations on collections

  theory translation, e.g. CZF to Type Theory

  cross linking via formal thesaurus

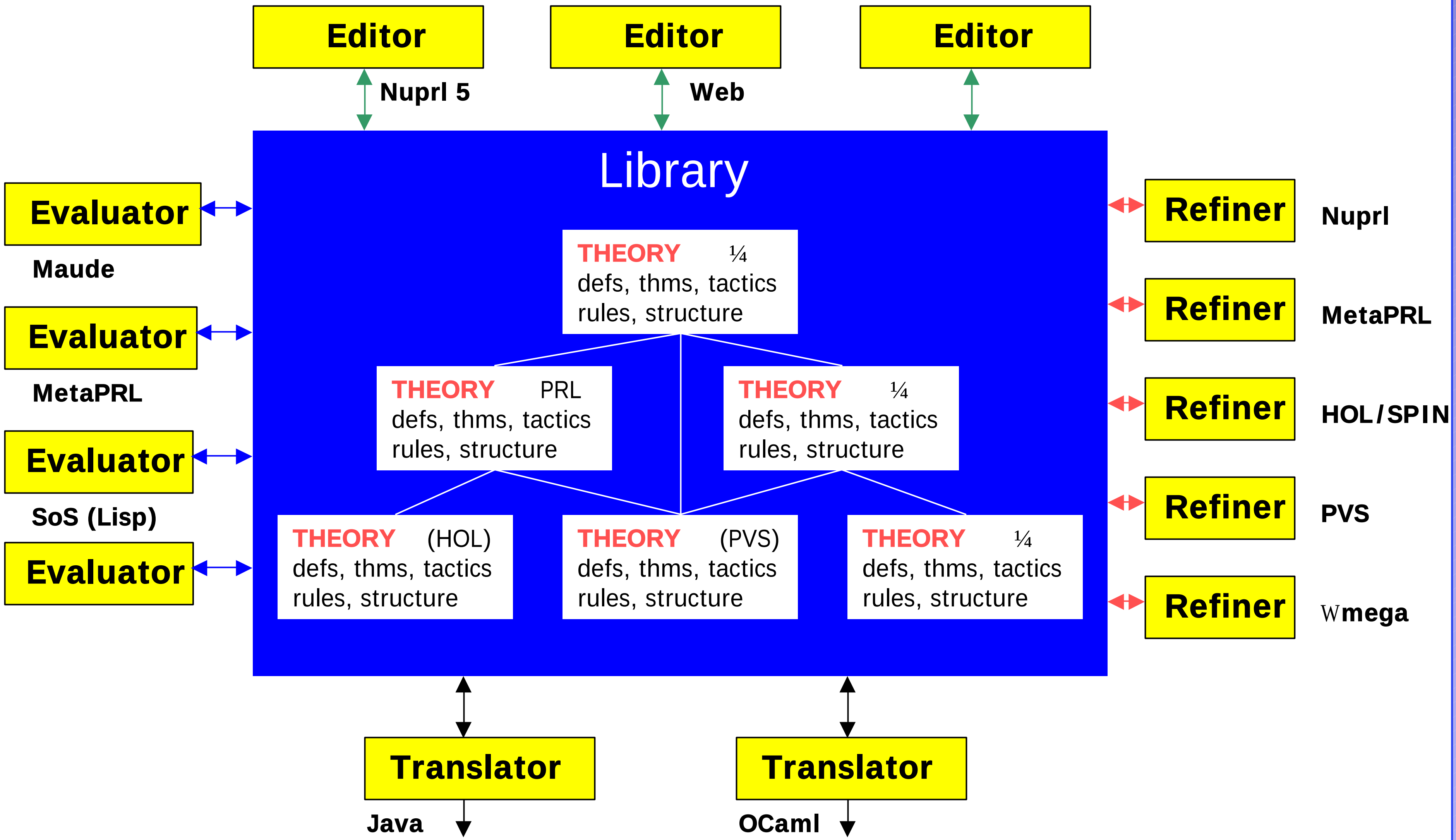
  transplanting theorems

  classical to constructive translations

# Outline

- The ONR Digital Library Project
- Concepts for Formal Digital Library (FDL) design
- Current status of FDL
- Questions and issues

# Formal Digital Library



# Features of Prototype FDL (see Description and Ref Manual)

LISP/ML based system

6,000 named functions

62,000 lines of code

22,000 lines of comments

Some code adapted from LPE and Nuprl currently stores many Nuprl, PVS objects. Limited service will be available from the Web over the course of the year, e.g. accepting PVS files.

We have a customer – ORA.

## Prototype FDL – Operations (Manual 3.2)

The basic operations are:

| | |
|---|---|
| bind id to object | unbind id from object |
| generate new object id | lookup object |
| activate an object | deactivate object |
| allow garbage collection | disallow collection |

# Prototype FDL – Data (Manual 3.1)

Organized to eventually support closed maps

$\boxed{D\text{®}Term(D)}$

$D$ are object names (abstract)

$Term(D)$ are objects with embedded references

Map is closed under object reference.

Working space is the current closed map.

Basic data structure is the library table.

# Closed Maps

$$D \rightarrow Term(D)$$

closed under reference (no dangling pointers)

**closed**

$$< id, term(\ ) >$$
$$< id, term(\ ) >$$
$$< id, term(\ ) >$$
$$< id, term(\ ) >$$
$$< id, term(\ ) >$$

**open**

$$< id, term(\ ) >$$
$$< id, term(\ ) >$$
$$< id, term(\ ) >$$

?

# Prototype FDL – Transaction System (Manual 6.2)

Operations on closed maps can be elegantly implemented by transactions.

For example, deleting an object from map f requires deleting all objects that depend on it (no dangling pointers).

Delete is a database transaction – all or nothing, leaving a closed map.

Transaction management allows crash recovery.

# Prototype FDL Content (Manual 7.2)

PVS libraries and refiner

- 20 libraries
- 400 theories
- 900 definitions
- 2,300 lemmas
- 300 theorems
- 200 postulates

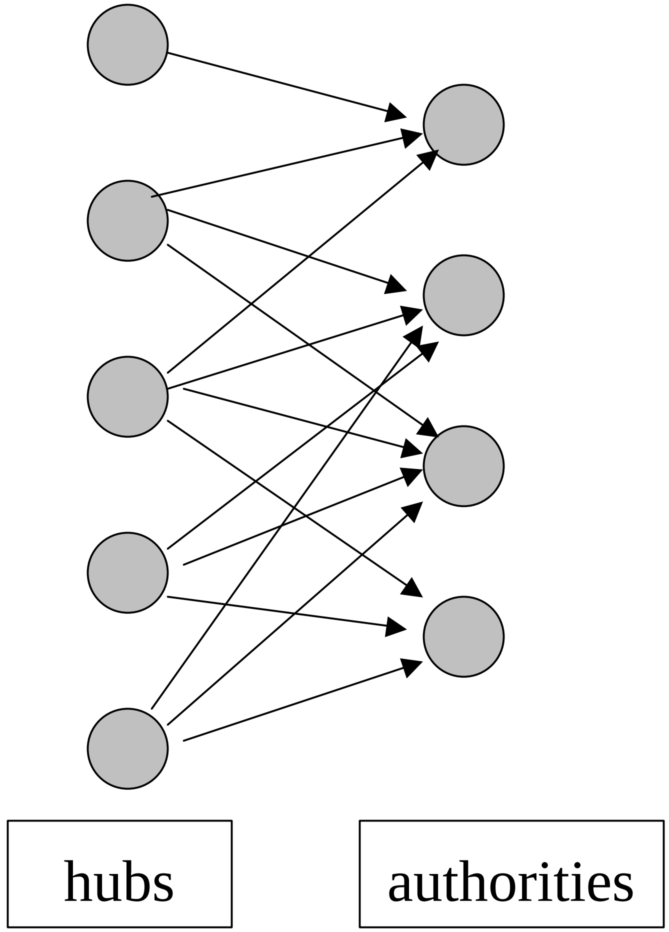
## Outline

- The ONR Digital Library Project
- Concepts for Formal Digital Library (FDL) design
- Current status of FDL
- Questions and issues

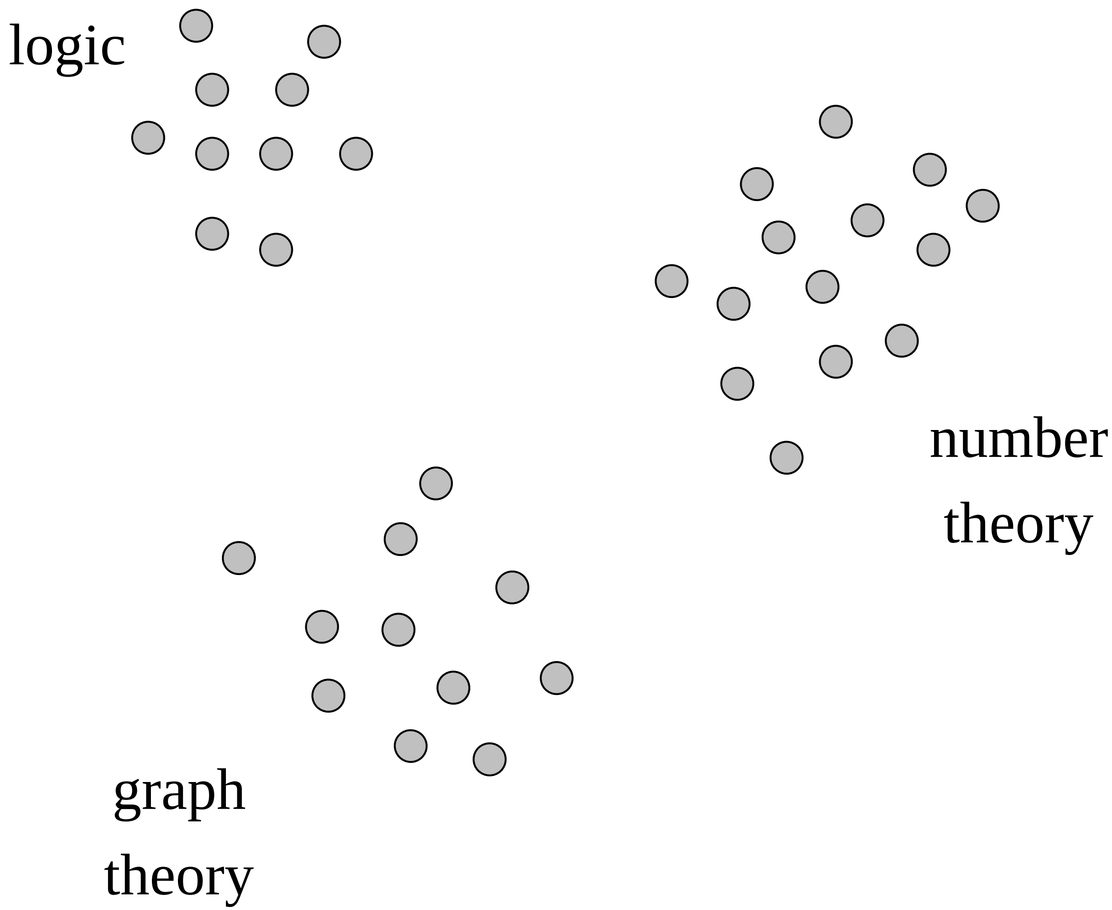
# Questions and Issues

- What minimal set of services should an FDL provide?
- What community would be well served by an FDL?
- How can users contribute to an FDL?

# Using Kleinberg's Hubs and Authorities



# Classifying by Eigenvectors



# Rehosting Larch and Larch/VHDL Libraries

- "Legacy" system, developed at ORA
- Large database of Larch definitions/theorems/proofs
- Verification that VHDL code meets Larch/VHDL spec
- Larch proof editor/checker implemented with Synthesizer Generator
  - hard to maintain
  - expensive to license
  - monolithic editor/refiner/library

# Rehosting Strategy

- Import Larch theories and proofs as FDL terms
  - generic Yacc/Lex to FDL tool
  - C/C++ connection to FDL
- Build only the Larch prover’s “refiner”
  - port from SSL to C++ using existing code
- Make display forms for Larch and Larch/VHDL
  - FDL provides editor attachments

# FDL Capabilities – Formal Metamathematics

Deep sharing requires metamathematical results such as

Howe: Classical Nuprl is consistent with HOL

Smith: Nuprl domain theory is not consistent with HOL, PVS

Moran: Extended Classical Nuprl is consistent with HOL and PVS

## Services

- Can we justify our data format as essential to a minimal set of services?
- How to search?
- How to justify proofs with code?

# Technical Challenges: How to Increase the Value of Formal Material

- Increase access
  - for computing, math, science
  - for publication and dissemination
  - for information science studies
  - for education
- Account for trust
  - store evidence (proofs, dependencies)
  - third-party validation
  - certificates
- Track dependencies
  - logical dependence
  - relevance
- Insure stability of stored objects
  - replayability
  - stable proofs
  - promote stable code