Overview of HOL Systems

Konrad Slind

School of Computing, University of Utah

Feb. 21, 2005

HOL Implementations

- LCF
  - Edinburgh LCF (1979)
  - Edinburgh/INRIA/Cambridge
- HOL
  - HOL88 (1988-1995)
  - ProofPower (1990-present)
  - Isabelle/HOL (1990-present)
  - HOL Light (1995-present)
  - HOL90/HOL98 (1990-2001)
  - HOL-4 (2001-present)

`http://hol.sourceforge.net/`

General Research Themes

- Hardware Verification
- Operational Semantics of Programming Languages
- Embedding other formalisms
- Theory formalization

Embedding Terminology

If you have a formalism and you want to formalize it, there are choices:

- Shallow (external meaning function)
- Deep (internal meaning function)

Example (Hoare Logic)

If you can prove the rules of Hoare Logic as theorems, you have a deep embedding. If you use a VCG to generate proof obligations, and the rules it uses are not theorems, you have a shallow embedding.

Embedding Terminology

If you have a formalism and you want to formalize it, there are choices:

- Shallow (external meaning function)
- Deep (internal meaning function)

Example (Hoare Logic)

If you can prove the rules of Hoare Logic as theorems, you have a deep embedding. If you use a VCG to generate proof obligations, and the rules it uses are not theorems, you have a shallow embedding.

Mature Technology

The following tools are **mature**, in the sense of being heavily used and having evolved through several generations:

- Simplification
- First-order proof search
- Datatype definitions
- Recursive function definitions
- Inductive definitions
- Arithmetic decision procedures (full Presburger)

Mature Technology

The following tools are **mature**, in the sense of being heavily used and having evolved through several generations:

- Simplification
- First-order proof search
- Datatype definitions
- Recursive function definitions
- Inductive definitions
- Arithmetic decision procedures (full Presburger)

Mature Technology

The following tools are **mature**, in the sense of being heavily used and having evolved through several generations:

- Simplification
- First-order proof search
- Datatype definitions
- Recursive function definitions
- Inductive definitions
- Arithmetic decision procedures (full Presburger)

Mature Technology

The following tools are **mature**, in the sense of being heavily used and having evolved through several generations:

- Simplification
- First-order proof search
- Datatype definitions
- Recursive function definitions
- Inductive definitions
- Arithmetic decision procedures (full Presburger)

Mature Technology

The following tools are **mature**, in the sense of being heavily used and having evolved through several generations:

- Simplification
- First-order proof search
- Datatype definitions
- Recursive function definitions
- Inductive definitions
- Arithmetic decision procedures (full Presburger)

Mature Technology

The following tools are **mature**, in the sense of being heavily used and having evolved through several generations:

- Simplification
- First-order proof search
- Datatype definitions
- Recursive function definitions
- Inductive definitions
- Arithmetic decision procedures (full Presburger)

Current Trends

- Large case studies in operational semantics
  - Thorough semantics of Java (Nipkow group, Isabelle/HOL)
  - Thorough semantics of UDP/IP/Sockets (Sewell & co.)
  - Large journal papers (70-100 pages)
- Moving on from case-studies. Use proof to implement interesting tools:
  - Original effort: **Prosper** (late 90's EU project)
  - Generate circuit checkers from formal semantics of PSL (Mike Gordon)
  - Model-checkers (HOL-BDD library integration) (Hasan Amjad)
  - Check conformance of network traces with formal model (Norrish)
  - Synthesize hardware from subset of HOL (Gordon/Slind)

Current Trends

- Large case studies in operational semantics
  - Thorough semantics of Java (Nipkow group, Isabelle/HOL)
  - Thorough semantics of UDP/IP/Sockets (Sewell & co.)
  - Large journal papers (70-100 pages)
- Moving on from case-studies. Use proof to implement interesting tools:
  - Original effort: **Prosper** (late 90's EU project)
  - Generate circuit checkers from formal semantics of PSL (Mike Gordon)
  - Model-checkers (HOL-BDD library integration) (Hasan Amjad)
  - Check conformance of network traces with formal model (Norrish)
  - Synthesize hardware from subset of HOL (Gordon/Slind)

Current Trends

- Large case studies in operational semantics
  - Thorough semantics of Java (Nipkow group, Isabelle/HOL)
  - Thorough semantics of UDP/IP/Sockets (Sewell & co.)
  - Large journal papers (70-100 pages)
- Moving on from case-studies. Use proof to implement interesting tools:
  - Original effort: **Prosper** (late 90's EU project)
  - Generate circuit checkers from formal semantics of PSL (Mike Gordon)
  - Model-checkers (HOL-BDD library integration) (Hasan Amjad)
  - Check conformance of network traces with formal model (Norrish)
  - Synthesize hardware from subset of HOL (Gordon/Slind)

Current Trends

- Large case studies in operational semantics
    - Thorough semantics of Java (Nipkow group, Isabelle/HOL)
    - Thorough semantics of UDP/IP/Sockets (Sewell & co.)
    - Large journal papers (70-100 pages)
- Moving on from case-studies. Use proof to implement interesting tools:
    - Original effort: **Prosper** (late 90’s EU project)
    - Generate circuit checkers from formal semantics of PSL (Mike Gordon)
    - Model-checkers (HOL-BDD library integration) (Hasan Amjad)
    - Check conformance of network traces with formal model (Norrish)
    - Synthesize hardware from subset of HOL (Gordon/Slind)

Current Trends

- Large case studies in operational semantics
  - Thorough semantics of Java (Nipkow group, Isabelle/HOL)
  - Thorough semantics of UDP/IP/Sockets (Sewell & co.)
  - Large journal papers (70-100 pages)
- Moving on from case-studies. Use proof to implement interesting tools:
  - Original effort: **Prosper** (late 90's EU project)
  - Generate circuit checkers from formal semantics of PSL (Mike Gordon)
  - Model-checkers (HOL-BDD library integration) (Hasan Amjad)
  - Check conformance of network traces with formal model (Norrish)
  - Synthesize hardware from subset of HOL (Gordon/Slind)

Current Trends

- Large case studies in operational semantics
  - Thorough semantics of Java (Nipkow group, Isabelle/HOL)
  - Thorough semantics of UDP/IP/Sockets (Sewell & co.)
  - Large journal papers (70-100 pages)
- Moving on from case-studies. Use proof to implement interesting tools:
  - Original effort: **Prosper** (late 90's EU project)
  - Generate circuit checkers from formal semantics of PSL (Mike Gordon)
  - Model-checkers (HOL-BDD library integration) (Hasan Amjad)
  - Check conformance of network traces with formal model (Norrish)
  - Synthesize hardware from subset of HOL (Gordon/Slind)

Current Trends

- Large case studies in operational semantics
  - Thorough semantics of Java (Nipkow group, Isabelle/HOL)
  - Thorough semantics of UDP/IP/Sockets (Sewell & co.)
  - Large journal papers (70-100 pages)
- Moving on from case-studies. Use proof to implement interesting tools:
  - Original effort: **Prosper** (late 90's EU project)
  - Generate circuit checkers from formal semantics of PSL (Mike Gordon)
  - Model-checkers (HOL-BDD library integration) (Hasan Amjad)
  - Check conformance of network traces with formal model (Norrish)
  - Synthesize hardware from subset of HOL (Gordon/Slind)

Current Trends

- Large case studies in operational semantics
  - Thorough semantics of Java (Nipkow group, Isabelle/HOL)
  - Thorough semantics of UDP/IP/Sockets (Sewell & co.)
  - Large journal papers (70-100 pages)
- Moving on from case-studies. Use proof to implement interesting tools:
  - Original effort: **Prosper** (late 90's EU project)
  - Generate circuit checkers from formal semantics of PSL (Mike Gordon)
  - Model-checkers (HOL-BDD library integration) (Hasan Amjad)
  - Check conformance of network traces with formal model (Norrish)
  - Synthesize hardware from subset of HOL (Gordon/Slind)

Interfaces to the Outside World

- Building-in (in a principled way) external provers
  - BDDs
  - SAT
- External proof format (consumer: Isabelle/HOL)

Open Source Development of HOL

HOL-4 is open source software (BSD license)

- We do not worry about being ripped off
- Please rip us off!
- We want to spread the ideas and procedures and theorems in the system as widely as possible
- Almost anybody can become a HOL developer ... just ask
- But still need > 1 and < 3 main developers responsible for correctness-critical code.
- Also need a leader-figure

Open Source Development of HOL

HOL-4 is open source software (BSD license)

- We do not worry about being ripped off
- Please rip us off!
- We want to spread the ideas and procedures and theorems in the system as widely as possible
- Almost anybody can become a HOL developer ... just ask
- But still need > 1 and < 3 main developers responsible for correctness-critical code.
- Also need a leader-figure

Open Source Development of HOL

HOL-4 is open source software (BSD license)

- We do not worry about being ripped off
- Please rip us off!
- We want to spread the ideas and procedures and theorems in the system as widely as possible
- Almost anybody can become a HOL developer ... just ask
- But still need > 1 and < 3 main developers responsible for correctness-critical code.
- Also need a leader-figure

Challenges

- We must teach more people theorem proving
- We must continue to attack big problems

Challenges

- We must teach more people theorem proving
- We must continue to attack big problems

Challenges

- We must teach more people theorem proving
- We must continue to attack big problems

THE END