HDCC, NASA Ames, FM group report; 12 January 2001

Formal Methods: Modeling, Specification, Verification

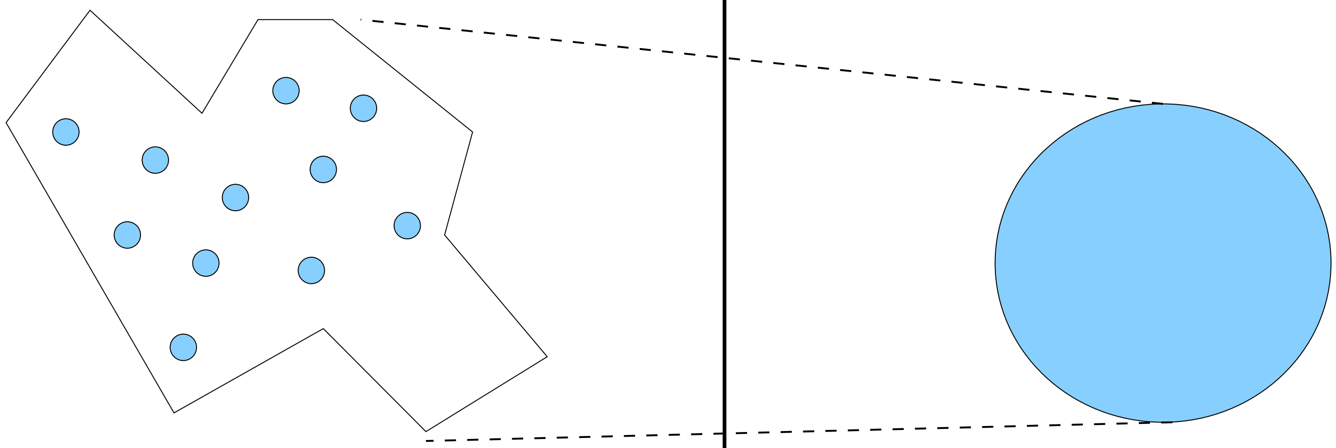
Jeanne Behnke, Rick Butler, Dan Clancy, Ed Clarke, Rance Cleaveland, Tom Henzinger, John Kelly, Brad Martin, Martin Mohrenschildt, Peter Norvig, John Penix, John Rushby, Willem Visser, Chuck Weinstock, Jeannette Wing

Group Report

HDCC, NASA Ames, 12 January 2001

Formal Analysis: The Idea

Testing/Simulation	Formal Analysis



Real System

Formal Model

- Partial coverage

- Complete coverage (of the modeled system)

  Accurate model: verification

  Approximate model:debugging

State of the Practice

Verification: Modest success

- FP units: Intel, AMD, Motorola
- Nuclear shutdown: Ontario Hydro
- Crypto processor: NSA/Motorola
- Avionics: Aerospatiale, Collins/NASA
- Railroads: Stålmarck, GEC/Alsthom

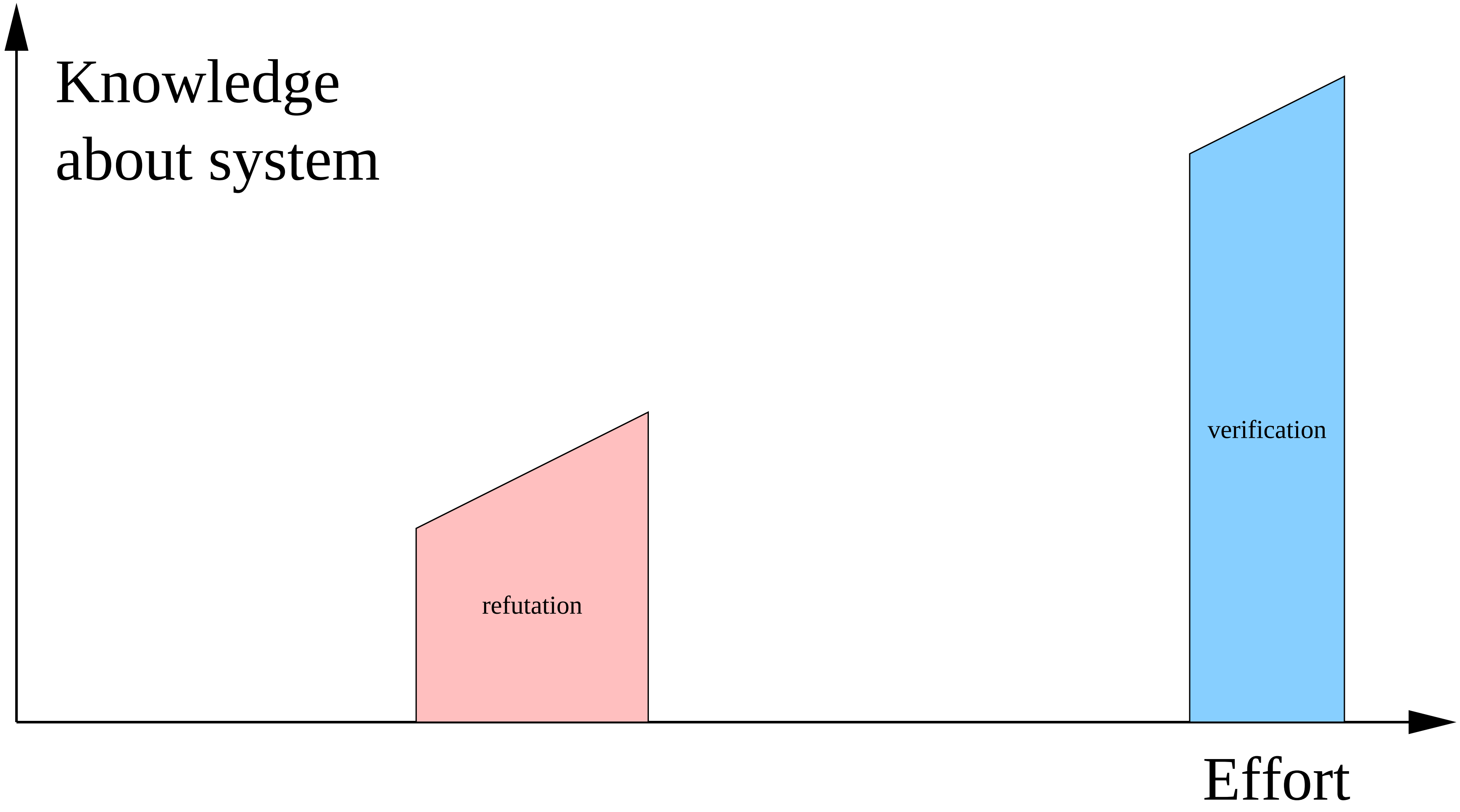
Refutation (debugging): Larger success

- Hardware (incl. equivalence checking)
- Protocols
- Statecharts
- Avionics: Honeywell/NASA

Why Were Those Applications Successful?

- Not too difficult to extract/construct the formal models
  - Mathematical domain (hardware)
  - Someone else had introduced a halfway decent notation and design method (Statecharts)
- Can get good payoff with existing tools
  - Relatively small systems
  - Other methods don't work well
    (intricate, critical systems)
  - Faults have high cost

Summary (now)



Near-Term Progress

Reducing the modelization gap (invisible formal methods)

- Extract models automatically from existing notations
  - C++ (Verisoft)
  - Java (JPF, Bandera)
  - Machine code (SLAM)
- Analyze weaker properties
  - Stronger typechecking (ESC, Lackwit)
  - Static analysis (Prefix, Logiscope)
- Build languages around architectural frameworks (APIs of well-designed middleware)
  - TTA (Giotto)
  - Publish/Subscribe

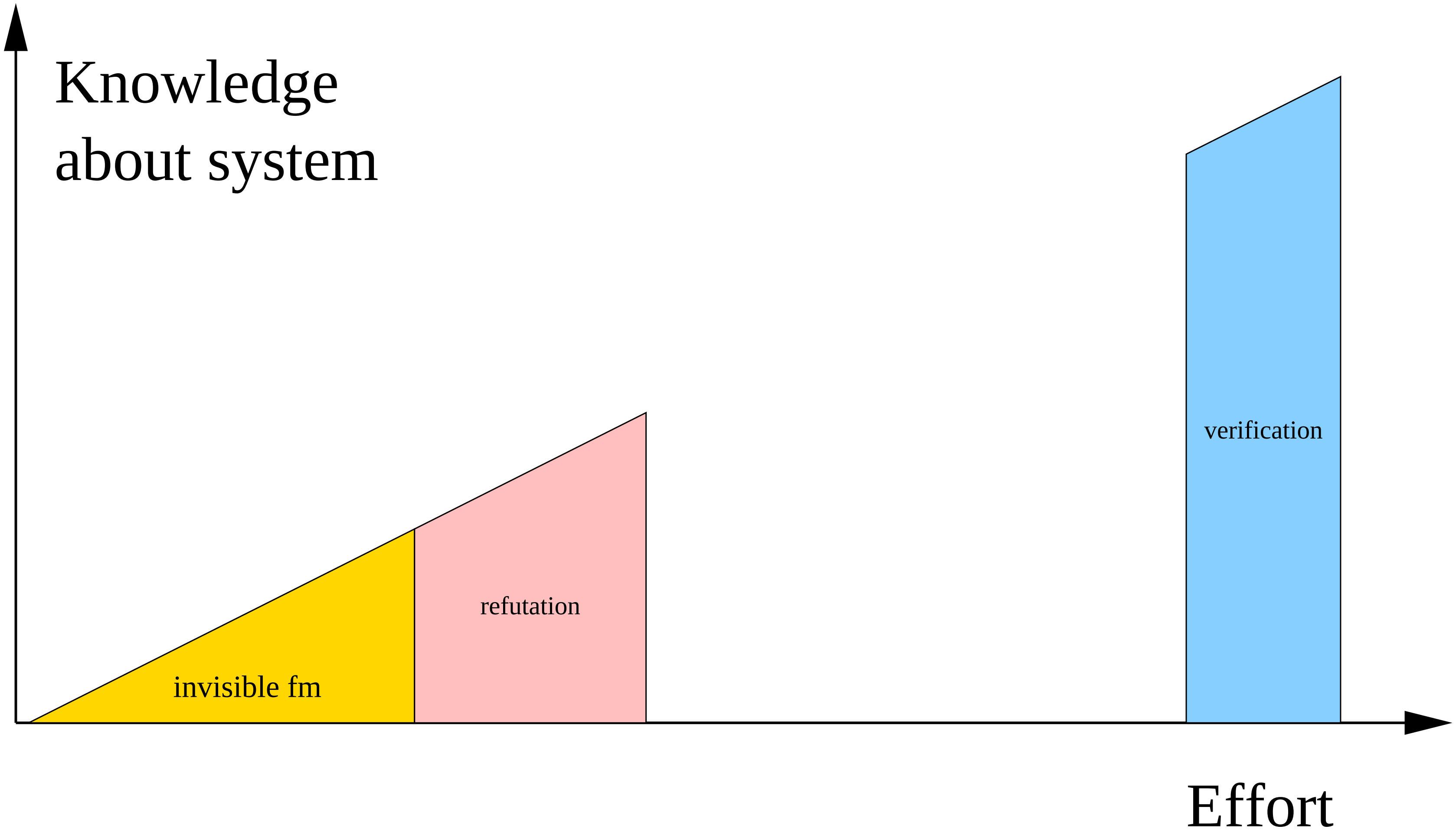
Near-Term Progress (ctd)

Better/different analysis

- Analyze different properties, for different purposes
  - Human factors (mental models)
  - Data, rather than control (Alloy)
  - Test case generation (TGV, UTwente)
  - Computer algebra systems
- Use more powerful analysis

  (don't have to simplify the models so much)
  - Infinite state-model checking
  - Hybrid systems
  - Automated abstraction
  - More automated theorem proving

Summary (Near-Term)



Near-Term Challenges

- Get involved in design of emerging industry-accepted notations to make them more amenable to effective analysis
  - UML (OCL)
  - System design languages
- Similarly for frameworks
- Build more powerful tools

Long-Term Challenges

- Cf. Challenge problems
- Dependability is often concerned with nonfunctional properties (security, partitioning, reliability)
- Undependability usually arises through unanticipated interactions among components
    - Or the system and its environment
- Will be concerned with large systems, often using COTS components, and nontraditional software (neural nets, learning)
- Formal methods must confront these realities

Long-Term Prospects

- Get basic technology into the mainstream:

  pervasive formal methods

  - Education
  - Tech transfer
    - ⋆ Eliminate 10% of testers (economic argument)
    - ⋆ Make same number more productive (dependability)

- Develop technology for the more challenging applications
  - Compositional design and verification
  - Runtime verification?
  - Smooth tradeoff between effort and return

Summary (Long-Term)