Formal Verification

Sumeet Agarwal
EE Dept., IIT Delhi

[Ref.: Huth and Ryan, *Logic in Computer Science*. Cambridge University Press, 2004.]

Why verification?

- Verifying correctness very valuable for hardware/software systems
- Especially safety-critical systems; also commercially or mission critical
- Formal verification methods a growing area and have become quite usable by industry

Components of formal verification

- **Framework for modelling systems**: some sort of *language* in which they can be described
- **Specification language**: to describe the properties to be verified
- **Verification method**: establishes whether the system description satisfies the specification

Approaches to verification

- **Proof-based vs. Model-based**
  - In proof-based, both description (Γ) and specification (ϕ) are formulae in a suitable logic. Verification method is proof-finding ($\Gamma \vdash \phi$). Usually needs human intervention.
  - In model-based, the system is represented by a model M for an appropriate logic. Specification is again a formula ϕ; verification is checking if model satisfies formula ($M \models \phi$). Usually automatic for finite models.

Approaches to verification

- Degree of automation
- Full verification vs. property-verification
- Domain of application: hardware/software; sequential/concurrent; reactive/terminating
- Pre-development vs. post-development (e.g., Intel Pentium FDIV error)

Model checking

- *Model checking* is automatic, model-based, property-verification; intended for concurrent, reactive systems; originated as a post-development method
- Concurrency bugs among the hardest to detect using testing: often non-reproducible or not covered by test cases
- Based on *temporal logic*: dynamic notion of truth; formulae aren't just statically true/false, but state-dependent

Model checking

1. Model the system using the description language (representing some kind of *transition system*), to arrive at model M
2. Code the property to be verified using the specification language (temporal logic), giving formula ϕ
3. Run the model checker with inputs M and ϕ, to check if $M \models \phi$: output either *yes* or *no* (along with system trace)