Formal Methods of Software Engineering

Robert Dewar and Amir Pnueli
Course: G22.3033.007

Thursdays, 5-7 PM

Copies of presentations and Lecture Notes will be
available at
http://cs.nyu.edu/courses/fall01/G22.3033-007/index.htm

Formal Methods

The Formal Methods approach to software construction is based on viewing a program and its execution as mathematical objects and applying mathematical and logical techniques to specify and analyze the properties and behavior of these objects.

The main advantages of the formal approach to software construction is that, whenever applicable, it can lead to an increase of the reliability and correctness of the resulting programs by several orders of magnitude. Again, the main potential beneficiary of such an increase is the class of safety critical systems.

The Questions Asked by Formal Methods

Formal methods deal with two distinct descriptions of the same system. A more abstract description S, called a specification, and a more detailed (concrete) description I, called an implementation. The main questions asked are:

- Verification — Given S and I, validate that I is a correct implementation of S or, more generally, that S and I are compatible.
- Synthesis — Given one of the descriptions, produce the other at the appropriate level of abstraction. Usually, we are given S and asked to construct I. However, in some reverse engineering applications, we may be asked to produce a specification S for a given implementation I .

These questions can be asked in

- A Two Languages Framework — The specification is presented in a specification language, usually a logic or an equivalent declarative formalism. The implementation is presented in a programming-like language.
- A Single Language framework — S and I ae presented in the same language. We then talk about establishing refinement, abstraction, equivalence, etc. This framework can support a multistep development process:

$$S_1 \sqsupseteq S_2 \sqsupseteq \cdots \sqsupseteq S_n = I$$

So, How Successful is the Application of Formal Methods to System Development?

- Much more successful in the development of hardware. Many of the large semi-conductors manufacturers: Intel, IBM, AMD, use formal verification (mainly model checking) as part of their development process.
- Partially, in some large software projects.

One of the expected benefits of studying formal methods is the acquired state-of-mind by which every software component should have its own well defined interface specification, independently of whether we attempt to formally verify it, or even write it down explicitly.