T-79.5303 Safety Critical Systems

Case Study 1: Formal Methods - Introduction

Teemu Tynjälä

January 24, 2008

Formal Methods - Reasons for existence

What’s the purpose of formal methods? They are used in a similar way as prototypes in construction engineering. We make a series of verification and validation runs on a system’s formal model and if they are successful, we proceed with implementation.

What’s the current status of formal methods? In hardware engineering, methods such as *model checking* and *theorem proving* are much used in circuit design. Evidently, the interest to formal verification was spurred by the Pentium Bug.

Another field where formal methods have already proven themselves are safety critical systems. Railway interlocking systems and nuclear plant control programs must be formally verified before implementation. In effect, all computer programs used in the UK government must be formally verified! This is one reason why the formal methods community in the British Isles is so strong.

Formal Methods and Software.. Some thoughts

There is a great amount of interest in the software industry, but commercial companies are still hesitating to make a full-blown implementation.

Some problems with formal methods - There are too many formalisms available! It seems that there are about as many possible formalisms and their variants as there are researchers in the field.

How to survive in the jungle of formal methods? Develop your own methodology for learning formalisms, so that you can distill the essentials out of each one with a reasonable amount of effort. Most importantly - be open-minded! It's recommended (almost a must) to have your own formalism of choice, but the real winners in the field are able to understand other formalisms and follow their development.

Teemu Tynjälä

Formal Methods - What formalisms are out there??

Formal Methods can be grouped under 3 general headings:

Set based formalisms: These are good for describing systems in an ’object oriented’ way. Provides a high level view of a system that can be refined as specification proceeds. Examples of this are: Z, VDM, B

Logic based formalisms: This is a wide field which includes specification languages and property languages. There exist classical logic, predicate logic, modal logics, temporal logics, theorem provers. Some example theorem provers are: Isabelle, Coq, PVS.

Behavior based formalisms: Systems are described as states and transitions between states. A natural way of describing a system for a programmer. Examples in this area include process algebras (CCS, CSP), labelled transition systems, Petri Nets

Math essentials - necessary for survival in the jungle...

Sets: $A = \{a, b, c, \ldots\}$ and Multisets: $B = \{1`a, 3`b, 4`c, 2`d\}$. We may have finite and infinite sets and multisets. Notice that in a multiset, each element has a multiplicity.

Cartesian Products: $A \times A = \{\ \langle a1, a2 \rangle \mid a1 \in A \wedge a2 \in A\ \}$

Cartesian products are sometimes represented with 'exponents', i.e. if we write A^3 where A is a set, this actually means a Cartesian product $A \times A \times A$.

Relations. Relations are described as a subset of a Cartesian product of two sets. I.e. $R \subseteq A \times B$. A relation is described as a set of pairs. $R = \{\ (a1, b1),\ (a1, b2),\ (a1, b3),\ (a2, b1)\ \}$

More Math...

When talking about relations, we usually talk about whether they are *reflexive*, *symmetric* or *transitive*. Let’s define these formally.

reflexive: R is reflexive $\iff \forall a\, aRa$

symmetric: R is symmetric $\iff \forall a \forall b\, aRb \implies bRa$

transitive: R is transitive $\iff \forall a \forall b \forall c\, (aRb \wedge bRc) \implies aRc$

More Math... Total Order and Partial Order

Relations have two subtypes. A relation may be a partial order or a total order.

total order: $\forall x \forall y\ xRy \vee yRx$

partial order: $\neg(\forall x \forall y\ xRy \vee yRx)$

More Math... Operations

For example, $\bullet : A \times A \longrightarrow A$ describes an operation that takes two elements from set A and produces one element of set A.

commutativity: $\forall x \forall y \;\; x \bullet y \; = \; y \bullet x$

associativity: $\forall x \forall y \forall z \;\; (x \bullet y) \bullet z \; = \; x \bullet (y \bullet z)$

left identity e: $\forall x \;\; e \bullet x = x$

left absorbing a: $\forall x \;\; a \bullet x = a$

left inverse x' *of* x: $x' \bullet x = e$

Formal Methods in Action - Peano Arithmetic

No two integers are the same:

$$\forall x \quad \neg(0 = S(x))$$
$$\forall x \forall y \quad S(x) = S(y) \Longrightarrow x = y$$

Axioms of addition:

$$\forall x \quad x + 0 = x$$
$$\forall x \forall y \quad x + S(y) = S(x + y)$$

Axioms of multiplication:

$$\forall x \quad x.0 = 0$$
$$\forall x \forall y \quad x.S(y) = x + (x.y)$$

More Peano Arithmetic

Axioms of comparison:

$\forall x \quad \neg(x < 0)$

$\forall x \forall y \quad x < S(y) \Longleftrightarrow x < y \vee x = y$

Induction Schema:

$\phi(0) \wedge [\forall x \;\; \phi(x) \Longrightarrow \phi(S(x))] \Longrightarrow \forall x \;\; \phi(x)$

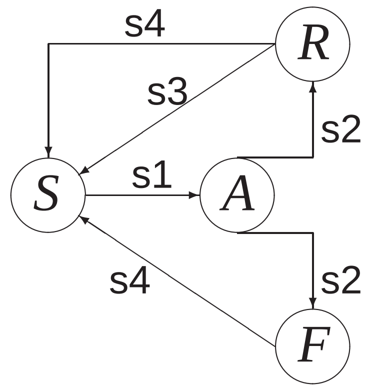
Formal Methods in Action - Transition Systems

A transition system (or labeled transition system) is described as a triple:

$\langle S, A, (\mathcal{R}_a)_{a \in A} \rangle$
S is a (finite or infinite) set of states
A is a (finite or infinite) set of actions
each $\mathcal{R}_a$ is a binary relation on S. ($\subseteq A \times S \times S$)

You might be wondering what's the subscript a on the R.. Don't worry, this will soon become evident.

One Small Transition System..



$T = \langle S, A, (\mathcal{R}_a)_{a \in A} \rangle$

$S = \{S, A, R, F\}$

$A = \{s1, s2, s3, s4\}$

$\mathcal{R}_{s1} = \{\langle S, A \rangle\}$

$\mathcal{R}_{s2} = \{\langle A, F \rangle, \langle A, R \rangle\}$

$\mathcal{R}_{s3} = \{\langle R, S \rangle\}$

$\mathcal{R}_{s4} = \{\langle R, S \rangle, \langle F, S \rangle\}$

A Small Example of Set-based Specification

Example Schema

$x, y : \mathbb{Z}$

$x \geq 0$
$y \geq 5$
$x + y = 10$

This is equivalent to

$$\{\langle x, y \rangle \in \mathbb{Z} \times \mathbb{Z} \mid x \geq 0 \wedge y \geq 5 \wedge x + y = 10\}$$

Plan of Action for Future Case Studies

In the following lectures, we will look at one formal method - B method - in detail, covering its syntax, semantics and common use. We will be using a model checking tool for B method, which enables you to construct machines and verify that they operate correctly.

THANK YOU!

References

The material in this presentation has been obtained from:

- Understanding Formal Methods. Jean-François Monin. Springer, 2003.