The Development of a Toolkit to Support Probabilistic B-Method

Thai Son Hoang

Introduction

B Method (B) is a formal development method that facilitates the refinement of specification to code. Our *probabilistic B* (pB) replaces Boolean predicates by real valued probabilities in the range 0..1. This allows probabilistic uncertainty to be modelled.

Probabilistic choice

$$S \ {}_p\!\oplus \ T$$

represents a choice between S and T in which S is taken with probability p and T is taken with probability $1-p$.

Questions?

- ⋆ What is the expected running cost of a system?
- ⋆ What is the reliability for a system given some information about its components?

Probability-one termination

What is termination with probability one?

Consider the following programs:

$n := 2$;
while $n \neq 0$ **do** $n := n-1$ **end**
Program A: *Absolute* correctness

$n := 1$;
while $n \neq 0$ **do** $n := n-1 \ [] \ \texttt{skip}$ **end**
Program B: *Demonic* incorrectness

$n := 1$;
while $n \neq 0$ **do** $n := n-1 \ {}_{0.5}\!\oplus \ \texttt{skip}$ **end**
Program C: *Almost-certain* correctness

Consider Program D:

$n := 1$;
while $n \neq 0$ **do** $n := n-1 \ {}_p\!\oplus \ \texttt{skip}$ **end**
Program D

Let t be the probability of termination for Program D:

$t = p + (1-p) \times t$
$\equiv \ p = p \times t$
$\equiv \ t = 1$ provided that $p \neq 0$

Abstract probabilistic choice substitution

Program D will still terminate with probability one without knowing the actual probability p, provided it is not 0 or 1.

$n := 1$;
while $n \neq 0$ **do** $n := n-1 \ \oplus \ \texttt{skip}$ **end**
Program E

Consider Program E, where $S \oplus T$ is the abstract probabilistic choice substitution between S and T. The $\oplus$ should be implemented by "concrete" probabilistic choices ${}_p\!\oplus$ that are bounded away from 0 and 1.

Termination with probability-one proof rules for loops

For loops, we have to denote the upper bound B for the variant and prove the following:

T1: Invariant is established	P	$\Rightarrow$	$[init]I$
T2: Invariant is maintained during executions	$G \wedge I$	$\Rightarrow$	$\lfloor S \rfloor I$
T3: Post-condition is established on termination	$\neg G \wedge I$	$\Rightarrow$	Q
T4: Variant is a natural number (bound below)	I	$\Rightarrow$	$V \in \mathbb{N} \wedge V \leq B$
T5: Variant decreases with non-zero probability	$G \wedge I$	$\Rightarrow$	$[n := V]\lceil S \rceil (V < n)$

Where $\lfloor S \rfloor$ is the substitution S with the abstract choice interpreted demonically, and $\lceil S \rceil$ is the substitution S with the abstract choice interpreted angelically.

Probabilistic invariants

Example of probabilistic Library

The following illustrates a simple library in which books are lost with probability p.

StartLoan $\widehat{=}$
pre $booksInLibrary > 0$ **then**
  $booksInLibrary := booksInLibrary - 1 \ \|$
  $loansStarted := loansStarted + 1$
end

EndLoan $\widehat{=}$
pre $loansEnded < loansStarted$ **then**
  $booksLost := booksLost + 1 \ {}_p\!\oplus \ booksInLibrary := booksInLibrary + 1 \ \|$
  $loansEnded := loansEnded + 1$
end

Invariants are replaced by *expectations*. For this specification, the expectation is defined by **expectation** clause

$$0 \Rrightarrow p * loansEnded - booksLost \ .$$

We can conclude that the expected number of books lost is bounded above by $p * loansEnded$.

Proof obligations

Proof obligations are similar to those for standard specifications with the exception that the invariant is now a real-value, not a Boolean. In order to prove that the *real* invariant is bounded below, we have to prove the following:

P1: The initialisation needs to establish the lower bound of the probabilistic invariant, given the context of the machine (information about sets and constants): $e \Rrightarrow [Init]I$.

P2: The operations do not decrease the expected value of the probabilistic invariant, i.e. the expected value of the invariant after the operation is at least the expected value before the operation: $I \Rrightarrow [Op]I$.

Probabilistic specification substitution

Specification substitution

$v{:}\{A, B\}$, where A and B are expectations over the state x and $v \subseteq x$. B can refer to the original state by x_0.
Probabilistic specification substitution

The semantics of the specification $v : \{A \ , \ B\}$ with respect to arbitrary post-expectation C is given by

$$[v : \{A \ , \ B\}]C \quad \widehat{=} \quad A \ \times \ [x_0 := x] \left(\sqcap x \cdot \left(\frac{C}{B^w} \right) \right) \ ,$$

where $B^w \widehat{=} B \times \langle w = w_0 \rangle$ (w are unchanged variables in the substitution).
This definition is constructed to extend the standard case and it satisfies the fundamental theorem.
Semantics of Probabilistic specification substitution

Fundamental theorem

Assume that $S \ \widehat{=} \ v : \{A \ , \ B\}$.
For any program T, $S \sqsubseteq T$ if and only if $A \ \Rrightarrow \ [x_0 := x][S]B^w$, where $B^w \widehat{=} B \times \langle w = w_0 \rangle$.
Probabilistic specification substitution

Terminating probabilistic specification substitution

It turns out that we need to specify that the specification does in fact terminate.

Terminating specification substitution

$v : \{\!\{p, \langle Q \rangle\}\!\}$, where p is a real expression between 0 and 1 and Q is a predicate over the state x and $v \subseteq x$. Q can refer to the original state by x_0.
Probabilistic specification substitution

The semantics of the specification $v : \{\!\{p, \langle Q \rangle\}\!\}$ is given as:

$$v : \{\!\{p, \langle Q \rangle\}\!\} \quad \equiv \quad v : \{1 \ , \ \langle Q \rangle\} \ {}_p\!\oplus \ v : \{1 \ , \ 1\} \ ,$$

where $v : \{1 \ , \ 1\}$ is the program to specify termination.
Semantics of Probabilistic specification substitution

Fundamental theorem

Assume that $S \ \widehat{=} \ v : \{\!\{p, \langle Q \rangle\}\!\}$.
For any program T, $S \sqsubseteq T$ if and only if $v : \{p \ , \ \langle Q \rangle\} \sqsubseteq T$ and $v : \{1 \ , \ 1\} \sqsubseteq T$ (i.e. T terminates).
Probabilistic specification substitution

Conclusion and future work

Work done

The *extended B (pB)* and modified *B-Toolkit* supports the following:

- ⋆ Termination with probability-one;
- ⋆ Probabilistic invariant;
- ⋆ Probabilistic specification substitution;
- ⋆ Fundamental theorem for refinement;
- ⋆ Probabilistic loops.

Future work

More research needs to be done on:

- ⋆ Specification with multiple expectations;
- ⋆ Fundamental theorem for multiple expectation specification;
- ⋆ General data refinement for probabilistic programs.

This project was supported by the Australian Research Council under the large grant A00103115.

THE UNIVERSITY OF NEW SOUTH WALES SYDNEY · AUSTRALIA

School of Computer Science and Engineering, UNSW, Sydney, Australia
email: htson@cse.unsw.edu.au web: http://www.cse.unsw.edu.au/∼htson

NATIONAL ICT AUSTRALIA LIMITED