Problem Sheet 9: Software Model Checking

Chris Poskitt[*]
ETH Zürich

The exercises in this problem sheet are based on the software model checking slides:

http://se.inf.ethz.ch/courses/2014b_fall/sv/slides/12-SoftwareModelChecking.pdf

1 Predicate Abstraction

Recall that:

- $Pred(f)$ denotes the weakest under-approximation of the expression f expressible as a Boolean combination of the given predicates.
- The Boolean abstraction of an `assume` c `end` statement is `assume not` $Pred($`not` $c)$ `end` followed by a parallel conditional assignment updating the predicates with respect to the original `assume` statement.
- The Boolean abstraction of an `assert` c `end` statement is `assert` $Pred(c)$ `end`.

Exercises

i. Justify, using Venn diagrams, the use of double negation in the Boolean abstraction of `assume` statements.

ii. Consider the following code snippet C_1, where x, y, z are integer variables:

```
assume x > 0 end
z := (x*y) + 1
assert z >= 1 end
```

Build the Boolean abstraction A_1 of the code snippet C_1 with respect to the following set of predicates:

$$p \triangleq x > 0$$
$$q \triangleq y > 0$$
$$r \triangleq z > 0$$

[*] Some exercises adapted from ones written by Stephan van Staden and Carlo A. Furia.

iii. Consider the following code snippet C_2, where x, y are integer variables:

```
if x > 0 then
     y := x + x
else
     if x = 0 then
          y := 1
     else
          y := x * x
     end
end
assert y > 0 end
```

(a) Normalise the guards of conditionals using nondeterminism and `assume` statements.

(b) Build the Boolean abstraction A_2 of the normalised code snippet C_2 with respect to the following set of predicates:

$$p \triangleq x > 0$$
$$q \triangleq y > 0$$

2 Error Traces

i. Provide an annotated trace for the Boolean abstraction A_1, and a corresponding annotated trace for the concrete program C_1 that is feasible such that `assert z >= 1 end` evaluates to false when reached.

ii. Can you verify the Boolean abstraction A_2? If not, give a counterexample path and prove whether or not it is *spurious*.