Basic notions of programming

1 Program function

Definition: *The weak program function of an $S \subseteq A \times (\bar{A} \cup \{fail\})^{**}$ program is a relation $\tilde{p}(S) \subseteq A \times (A \cup \{fail\})$ such that*

1. $\mathcal{D}_{\tilde{p}(S)} = \{a \in A \mid S(a) \cap (\bar{A} \cup \{fail\})^* \neq \emptyset\}$
2. $\forall a \in \mathcal{D}_{\tilde{p}(S)} \colon \tilde{p}(S)(a) = \{b \in A \cup \{fail\} \mid \exists \alpha \in S(a) \cap (\bar{A} \cup \{fail\})^* \colon b = \alpha_{|\alpha|}\}$

Informally, if given a program S, then its weak program function $\tilde{p}(S)$ maps from A to $A \cup \{fail\}$, and contains all the (a, b) pairs so that there is at least one finate execution of the program S starting from state a, that ends up in state b (b is the special $fail$ state or an element of the statespace A). In other words, for any state a, the weak program function $\tilde{p}(S)$ gives all the states where the program S can terminate starting its execution from state a. Note that executions of S that end up in the $fail$ state are not excluded here.

Definition: *The program function of an $S \subseteq A \times (\bar{A} \cup \{fail\})^{**}$ program is a relation $p(S) \subseteq A \times A$ such that*

1. $\mathcal{D}_{p(S)} = \{a \in A \mid S(a) \subseteq \bar{A}^*\}$
2. $\forall a \in \mathcal{D}_{p(S)} \colon p(S)(a) = \{b \in A \mid \exists \alpha \in S(a) \colon b = \alpha_{|\alpha|}\}$

Informally, if given a program S, then its program function $p(S)$ maps from A to A, and contains all the (a, b) pairs so that there are only finate and faultless executions of the program S starting from state a, and at least one of those finite executions ends up in state b. In other words, for any state a, the program function $p(S)$ gives all the states where the program S can terminate starting its execution from state a, but $p(S)$ is applicable only in states from where there are no faulty or endless executions the program can produce.

Notation: *Notice that the weak program function of a given S program is denoted by $\tilde{p}(S)$, whereas the program function of S is denoted by $p(S)$.*

2 Solution

Definition: *Let S be a program and F be a problem. We say that program S is totally correct with respect to F (or solves the problem F) if and only if*

1. $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$
2. $\forall a \in \mathcal{D}_F \colon p(S)(a) \subseteq F(a)$

Remark: *Notice that if S_1 and S_2 are such programs that their program functions equal ($p(S_1) = p(S_2)$) then they solve the same problems. In this case we say that S_1 and S_2 programs are equivalent.*

Example 1: *Consider the following problem: given a natural number n, our task is to find one of its positive divisors, d. The specification of the problem can be given in the following form:*
$A = (n : \mathbb{N}, d : \mathbb{N})$
$Pre = (n = n')$
$Post = (Pre \wedge d \mid n)$

For example, in the case of the "input" state $\{n : 10, d : 7\}$, the only good states are the elements of the statespace where the value of n is not changed and d is a divisor of n. So the problem (remember: a problem is a relation) assigns four states to the state $\{n : 10, d : 7\}$, namely $\{n : 10, d : 1\}, \{n : 10, d : 2\}, \{n : 10, d : 5\}$ and $\{n : 10, d : 10\}$.
What do we expect from a program to solve this given problem? If we consider the state $\{n : 10, d : 7\}$, we want our program not to abort but surely terminate and end up in any of the listed four states starting its execution from the state $\{n : 10, d : 7\}$. In general, we expect the program to surely terminate and end up in a "good" state (that is assigned to the given state by the problem) starting its execution from any state of the domain of the problem. In practice, we say that our program finds a divisor of 10 if the value of d is 1, 2, 5 or 10 after the execution of the program. In case our program is nondeterministic, and it ends up in a state where the value of d is either 2 or 10, but never is 1 or 5, we can say the program found a divisor of 10 (dispite the fact, that there is no execution of the program that yields the divisor 5). However, if there is an infinite execution assigned to the state $\{n : 10, d : 7\}$ by the program, or the program can terminate in a state where d is not a divisor of n, then the program does not solve the problem.

Informally, we say that a program solves the problem if

- for any state in the domain of the problem, it is guaranteed that the program terminates faultlessly starting its execution from the given state
- for any state in the domain of the problem, staring its execution there, the program ends up in "good" states (that are assigned to the given state by the problem)

Definition: *Let S be a program and F be a problem. We say that program S is partially correct with respect to F if and only if*

1. $\forall a \in \mathcal{D}_F \colon \tilde{p}(S)(a) \subseteq F(a)$

Remark: *Notice that if a state a is not in the domain of the weak program function $\tilde{p}(S)$ (that means only infinite sequences are assigned to a by the program S) then $\tilde{p}(S) = \emptyset$, and*

$\tilde{p}(S)(a) \subseteq F(a)$ is satisfied by default. Briefly saying: partial correctness = assuming the program terminates, it terminates in good states.

When one wants to define the notion of solution, the notion of weak program function is not useful for this purpose. Instead, the notion of (strong) program function has to be used. For any state a the weak program function $\tilde{p}(S)$ of the program S gives the states where the program may terminate starting its execution from state a. Notice, that $\tilde{p}(S)$ does not guarantee anything: if the pair (a, b) is an element of $\tilde{p}(S)$, then there is at least one execution of S that ends in state b, but it is not ensured that S faultlessly terminates whenever it is executed starting from state a.
Let us suppose $\tilde{p}(S)(a) = \{b\}$. In case there is an infinite sequence assigned to a by S, or a sequence that ends in the $fail$ state, it is not guaranteed that S terminates in state b.
In contrast, if $p(S)(a) = \{b\}$ then every possible execution of S leads b from state a.

Exercise 1: *Let $A = [1..5]$ be a statespace, $S \subseteq A \times (A \cup \{fail\})^{**}$ a program over the statespace A.*

$$S = \begin{Bmatrix} 1 \to < 1,2,5,1 > & 1 \to < 1,4,3,5,2 > & 1 \to < 1,3,2,3,\dots > \\ 2 \to < 2,1 > & 2 \to < 2,4 > & 3 \to < 3,3,3,\dots > \\ 4 \to < 4,1,5,4,2 > & 4 \to < 4,3,1,2,5,1 > & 5 \to < 5,2,3,4 > \\ 5 \to < 5,2,fail > & 5 \to < 5,3,4 > & \end{Bmatrix}$$

Let $F \subseteq A \times A$ denote the following problem: $F = \{\, (2,1), (2,4), (4,1), (4,2), (4,5) \,\}$

1. *Determine the program function of S and its domain.*
2. *Decide whether S is totally correct with respect to the given problem F.*

1. Determine the program function of S and its domain.
   $\mathcal{D}_{p(S)}$ is a set of states to which only finite and faultless sequences are assigned. For example, $5 \notin \mathcal{D}_{p(S)}$ since there exists a sequence linked to 5 that ends in the $fail$ state. However, at least one of the sequences linked to 5 is finite, so $5 \in \mathcal{D}_{\tilde{p}(S)}$.
   In case of state 1, there is an infinite sequence (namely $< 1,3,2,3,\dots >$) linked to 1. This is why $1 \notin \mathcal{D}_{p(S)}$, but $1 \in \mathcal{D}_{\tilde{p}(S)}$ and $\tilde{p}(S)(1) = \{1,2\}$ since there is a finite sequence assigned to 1 that ends in 1 and an other one that ends in 2, respectively.

   $\mathcal{D}_{p(S)} = \{2,4\} \qquad p(S) = \{(2,1),(2,4),(4,2),(4,1)\}$

   The weak program function of S is the following relation:
   $\mathcal{D}_{\tilde{p}(S)} = \{1,2,4,5\} \qquad \tilde{p}(S) = \{(1,1),(1,2),(2,1),(2,4),(4,2),(4,1),(5,4),(5,fail)\}$

2. Decide whether S is totally correct with respect to the given problem F.
   We have to check the two conditions of the definition of total correctness:

   - $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$
     Only the states 2 and 4 are in the domain of the problem: $\mathcal{D}_F = \{2,4\}$. We already

calculated $\mathcal{D}_{p(S)}$. The domain of the problem and the domain of the program function are equal, so the condition $\mathcal{D}_F \subseteq \mathcal{D}_{p(S)}$ holds.

$\{2,4\} = \mathcal{D}_F \subseteq \mathcal{D}_{p(S)} = \{2,4\} \qquad \checkmark$

- $\forall a \in \mathcal{D}_F\colon p(S)(a) \subseteq F(a)$

  $\mathcal{D}_F = \{2,4\}$
  In case $a = 2$, then $\{1,4\} = p(S)(2) \subseteq F(2) = \{1,4\}$. ✓
  In case $a = 4$, then $\{2,1\} = p(S)(4) \subseteq F(4) = \{1,2,5\}$. ✓

As both of the two conditions are satisfied, due to the definition of solution, S program solves the problem F.

3 Partial correctness vs total correctness

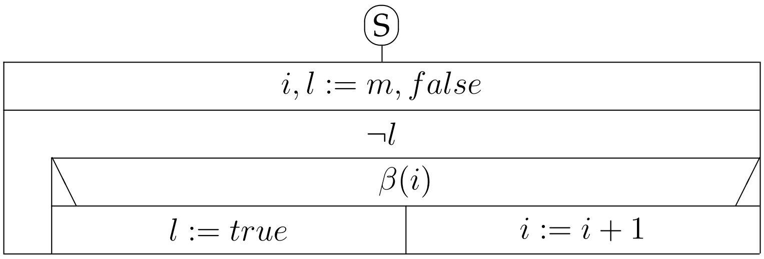
Example 2: *Consider the following program S, where the statespace is $A = (i\colon \mathbb{Z})$*

```
while TRUE do
          i := i +1
od
```

As the logical function $TRUE$ is always $true$, the loop is an endless loop. For any state σ, S never terminates starting its execution from σ. The only problem S solves is the empty problem where $F = \emptyset$. Nevertheless, S is partially correct with respect to any problem F, as $\tilde{p}(S) = \emptyset$ and therefore $\tilde{p}(S)(a) \subseteq F(a)$ holds for any a in the domain of F.

Example 3: *Consider the following problem: given an integer number m, find an integer number i such that i is the first integer number that is not less than m and for which a given β property holds.*

Consider the following program S, where the statespace is $A = (m\colon \mathbb{Z}, i\colon \mathbb{Z}, l\colon \mathbb{L})$.



As we do not know whether β holds for any integer number greater than or equal to m, it is not guaranteed that S terminates, so S does not solve the problem. However S is partially correct with respect to the problem, as if it terminates, it ends up in a state where i is the first integer number that is not less than m and for which β holds.