

# Dependence and Non-Termination

Plid 2005

Imperial College

David Clark (King's),  
Sebastian Danicic (Goldsmiths),  
Mark Harman (King's),  
Rob Hierons (Brunel),  
John Howroyd (@UK PLC),  
Mike Laurence (King's)

September 6, 2005

What is Dependence?

What is Dependence?

```
z := y + 1;  
x := z * 2;
```

program  $P$

What is Dependence?

```
z := y + 1;  
x := z * 2;
```

program  $P$

What does it mean for variable  $x$  to depend on variable  $y$  in program (fragment)  $P$ ?

What is Dependence?

```
z := y + 1;  
x := z * 2;
```

program  $P$

What does it mean for variable  $x$  to depend on variable  $y$  in program (fragment)  $P$ ?

There exist two initial states differing only on  $y$  such that if we execute  $P$  in either of these states, the corresponding final state have different values for  $x$ ?

What is Dependence?

$z := y + 1;$ $x := z * 2;$
--------------------------------

program  $P$

What does it mean for variable  $x$  to depend on variable  $y$  in program (fragment)  $P$ ?

There exist two initial states differing only on  $y$  such that if we execute  $P$  in either of these states, the corresponding final state have different values for  $x$ ?

More formally,  $x$  is dependent on  $y$  in  $P$  if and only if there exist two states,  $\sigma_1$  and  $\sigma_2$  differing only on  $y$  such that  $M[|P|]\sigma_1 x \neq M[|P|]\sigma_2 x$ .

Dependence is not decidable.

Dependence is not decidable.

So we can only conservatively approximate to it.

What about

```
while (h>0)  $h := h + 1$ ;  
 $x := 1$ ;
```

program  $Q$

What about

```
while (h>0)  $h := h + 1$ ;  
 $x := 1$ ;
```

program  $Q$

Who thinks  $x$  depends on  $h$  in  $Q$  here?

What about

```
while (h>0) h:=h + 1;  
  
x := 1;
```

program  $Q$

Who thinks  $x$  depends on  $h$  in  $Q$  here?

Clearly, there exist two states,  $\sigma_1$  and  $\sigma_2$  differing only on  $y$  such that  $M[|P|]\sigma_1 x \neq M[|P|]\sigma_2 x$ .

What about

```
while (h>0) h:=h + 1;  
  
x := 1;
```

program  $Q$

Who thinks  $x$  depends on  $h$  in  $Q$  here?

Clearly, there exist two states,  $\sigma_1$  and  $\sigma_2$  differing only on  $y$  such that  $M[|P|]\sigma_1 x \neq M[|P|]\sigma_2 x$ .

I get the feeling that most people here want to ignore this dependence!

What about

```
while (h>0) h:=h + 1;  
  
x := 1;
```

program  $Q$

Who thinks  $x$  depends on  $h$  in  $Q$  here?

Clearly, there exist two states,  $\sigma_1$  and  $\sigma_2$  differing only on  $y$  such that  $M[|P|]\sigma_1 x \neq M[|P|]\sigma_2 x$ .

I get the feeling that most people here want to ignore this dependence!

Why? Is it because this form of dependence is much *smaller*.  
With the general form, every variable depends on all loops.

So perhaps my definition of dependence should read:

there exist two initial states differing only on  $y$  such that if we execute  $P$  in either of these states,  $P$  **terminates** in both cases and ends up in final states with a different values of  $x$ ?

So perhaps my definition of dependence should read:

there exist two initial states differing only on  $y$  such that if we execute  $P$  in either of these states,  $P$  **terminates** in both cases and ends up in final states with a different values of  $x$ ?

So what semantics is being captured (approximated to) by this form of dependence which ignores non-termination?

So perhaps my definition of dependence should read:

there exist two initial states differing only on  $y$  such that if we execute  $P$  in either of these states,  $P$  **terminates** in both cases and ends up in final states with a different values of  $x$ ?

So what semantics is being captured (approximated to) by this form of dependence which ignores non-termination?

Answer: Lazy Semantics

# Lazy Semantics

## Lazy Semantics

Using Lazy Semantics, states can be partially defined. i.e. some variables are mapped to proper values and others to  $\perp$ .

## Lazy Semantics

Using Lazy Semantics, states can be partially defined. i.e. some variables are mapped to proper values and others to  $\perp$ .

```
while (y>0) y:=y + 1;
```

```
x := 1;
```

program  $Q$

## Lazy Semantics

Using Lazy Semantics, states can be partially defined. i.e. some variables are mapped to proper values and others to  $\perp$ .

```
while (y>0) y:=y + 1;  
  
x := 1;
```

program  $Q$

In all states,  $x$  has the final value 1. So  $x$  does not depend on  $y$  as required. In states where  $y > 0$  the final value of  $y$  is  $\perp$ .

## Lazy Semantics of Loops

$$\mathcal{M}_L[\textit{while} (B) S] = \lambda\sigma \cdot \bigsqcap_{i=0}^{\infty} G_i\sigma$$

$$\text{where } G_i\sigma = \bigsqcap_{n=i}^{\infty} \mathcal{M}_L[w_n(B, S)]\sigma.$$

The lazy meaning of *while* loop is the limit of the meet of the lazy meaning of all its corresponding unfoldings:

Given a state  $\sigma$  and a variable  $x$  the *final lazy value* of  $x$  after executing a while loop starting in state  $\sigma$  is the limit of all the values of  $x$  after executing each of the unfoldings. If the limit does not exist, then we define the final lazy value to be  $\perp$ . Here we mean the limit with respect to a discrete metric i.e. for the

limit to exist, there must exist an  $N \in \mathbb{N}$  such that all unfoldings greater than  $N$  give the same value for  $x$  in  $\sigma$ . If this is the case we say the value of  $x$  *stabilises* after  $N$  unfoldings.

## **Applications and Future Work**

## **Applications and Future Work**

We have proved the correctness of certain slicing algorithms using our Lazy Semantics.

## **Applications and Future Work**

We have proved the correctness of certain slicing algorithms using our Lazy Semantics.

We want to extend the Lazy Semantics to other language features

## **Applications and Future Work**

We have proved the correctness of certain slicing algorithms using our Lazy Semantics.

We want to extend the Lazy Semantics to other language features

We want to Prove that Standard Semantics is an Abstract interpretation of Lazy Semantics.

## Observations and Questions

## Observations and Questions

Do we care about non-termination?

## Observations and Questions

Do we care about non-termination?

What sort of dependence is required for interference?