

Master M1 informatique
Vérification – contrôle continu

Durée : 85 minutes – documents autorisés – sans moyen électronique

Répondre dans les cadres prévus à cet effet en gérant l'espace

Nom :

Signature :

Prénom(s) :

Exercice 1 (4 ●) Quels étaient les deux thèmes de recherche abordés lors de la présentation du LIM à la FST ?

Quels étaient les deux principaux concepts développés par Gilles Dowek lors de son exposé au LIM ?

Liveness Analysis

- A variable is *live* at a program point if its current value may be read in the remaining execution
- This is clearly undecidable, but the property can be conservatively approximated
- The answer "*dead*" must be the true one
 - dead variables may be ignored



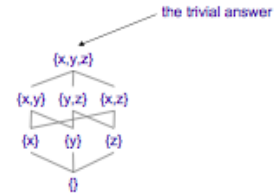
A Lattice for Liveness

- A subset lattice of program variables

```

var x, y, z;
x = input;
while (x>1) {
  y = x/2;
  if (y>3) x = x-y;
  z = x-4;
  if (z>0) x = x/2;
  z = z-1;
}
output x;
    
```

$$L = (2^{\{x,y,z\}}, \subseteq)$$



Setting Up

- For every CFG node, v , we have a variable $[[v]]$:
 - the subset of program variables that are live at the program point *before* v
- Since the analysis is conservative, the computed sets may be *too large*
- Auxiliary definition:

$$JOIN(v) = \bigcup_{w \in succ(v)} [[w]]$$



Liveness Constraints

- For the exit node: $vars(E) = \text{variables occurring in } E$
 $[[exit]] = \{\}$
- For conditions and output:
 $[[v]] = JOIN(v) \cup vars(E)$
- For assignments:
 $[[v]] = JOIN(v) \setminus \{id\} \cup vars(E)$
- For variable declarations:
 $[[v]] = JOIN(v) \setminus \{id_1, \dots, id_n\}$
- For all other nodes:
 $[[v]] = JOIN(v)$

right-hand sides are monotone since JOIN is monotone

Classifying Analyses

	forwards	backwards
may	reaching definitions $[[v]]$ describes state <i>after</i> v $JOIN(v) = \bigsqcup_{w \in pred(v)} [[w]] = \bigcup_{w \in pred(v)} [[w]]$	liveness $[[v]]$ describes state <i>before</i> v $JOIN(v) = \bigsqcup_{w \in succ(v)} [[w]] = \bigcup_{w \in succ(v)} [[w]]$
must	available expressions $[[v]]$ describes state <i>after</i> v $JOIN(v) = \bigsqcup_{w \in pred(v)} [[w]] = \bigcap_{w \in pred(v)} [[w]]$	very busy expressions $[[v]]$ describes state <i>before</i> v $JOIN(v) = \bigsqcup_{w \in succ(v)} [[w]] = \bigcap_{w \in succ(v)} [[w]]$

Initialized Variables Analysis

- Compute for each program point those variables that have *definitely* been initialized in the *past*
- \Rightarrow *forwards must analysis*
- Reverse subset lattice of all variables
- $JOIN(v) = \bigcap_{w \in pred(v)} [[w]]$
- $[[entry]] = \{\}$
- For assignments: $[[v]] = JOIN(v) \cup \{id\}$
- For all others: $[[v]] = JOIN(v)$



Nom :

Signature :

Prénom(s) :

Exercice 2 (8 ●) Vivacité. Pour le programme ci-dessous, déterminez le CFG, le système de contraintes de vivacité, calculez le plus petit point fixe et reportez les résultats sur le CFG.

```
var x,y,z;  
x = 27; y = input; z =2*x+z;  
if (0 > x) {y=z-3;} else {y=12;}
```

Exercice 3 (8 ●) Variables initialisées. Pour le programme ci-dessous, déterminez le CFG, le système de contraintes pour l'analyse des variables initialisées, calculez le plus petit point fixe et reportez les résultats sur le CFG. Que pouvez-vous déduire de cette analyse ?

```
var r,s,t,u,v;  
r = 0; s = 1; t = input;  
while (t > s*s) { u = s; s = s+1; }  
output u;
```