Esecuzione simbolica

1

1

Esecuzione Simbolica

- Il programma non è eseguito con i valori effettivi ma con valori simbolici dei dati di input.
- L'esecuzione procede come una esecuzione normale ma non sono elaborati valori bensì formule formate dai valori simbolici degli input
- Gli output sono formule dei valori simbolici degli input
- L'esecuzione simbolica anche di programmi di modeste dimensioni può risultare molto difficile.
  - ✓ dovuto all'esecuzione delle istruzioni condizionali: deve essere valutato ciascun caso (vero e falso);
  - ✓ necessità di theorem proving
  - ✓ in programmi con cicli ciò può portare a situazioni difficilmente gestibili.

2

2

Esecuzione simbolica: esempio

```
1    int prod (int x, int y, int z)
2    {
2       int tmp1, tmp2, product;
4       tmp1 = x*y;
5       tmp2 = y*z;
6       product = tmp1 * tmp2 / y;
7       return product;
7    }
```

Stm.	x	y	z	tmp1	tmp2	product
1	X	Y	Z	?	?	?
4	X	Y	Z	X*Y	?	?
5	X	Y	Z	X*Y	Y*Z	?
6	X	Y	Z	X*Y	Y*Z	(X*Y)*(Y*Z)/Y

3

3

Esecuzione simbolica e condizioni

```
cin >> x >> y >> z;
t = x + y;
x = x + t - 1;
z = t * y
y = y + 1
if (x>=0) then
        t = x + y + z
cout << t;
```

x = X, y = Y, z = Z
t = $X + Y$
x = $2*X + Y - 1$
z = $Y*(X + Y)$
y = $Y + 1$

$2*X + Y - 1 >= 0$
t = $(Y+2)*(X + Y)$

$2*X + Y - 1 < 0$
t = $X + Y$

4

4

Path Conditions

- Diremo ***path condition*** (*pc*) la relazione composta (deducibile dai predicati che giacciono sul cammino) che deve essere soddisfatta dai dati d'ingresso del programma perché la sequenza di istruzioni associata ai nodi del cammino sia eseguita.
  - ✓ Una *pc* è un'espressione Booleana sugli input simbolici di un programma.
  - ✓ All'inizio dell'esecuzione simbolica essa assume il valore vero (*pc* := true).
  - ✓ Per ogni condizione che si incontrerà lungo l'esecuzione *pc* assumerà differenti valori a seconda dei differenti casi relativi ai diversi cammini dell'esecuzione.

5

5

Costruzione di path conditions: esempio

…...	*pc* = true	
if (C) S1	*pc1* = *pc* ^ C	[cammino con S1]
else S2;	*pc2* = *pc* ^ (not C)	[cammino con S2]
…...		

6

6

Esempio

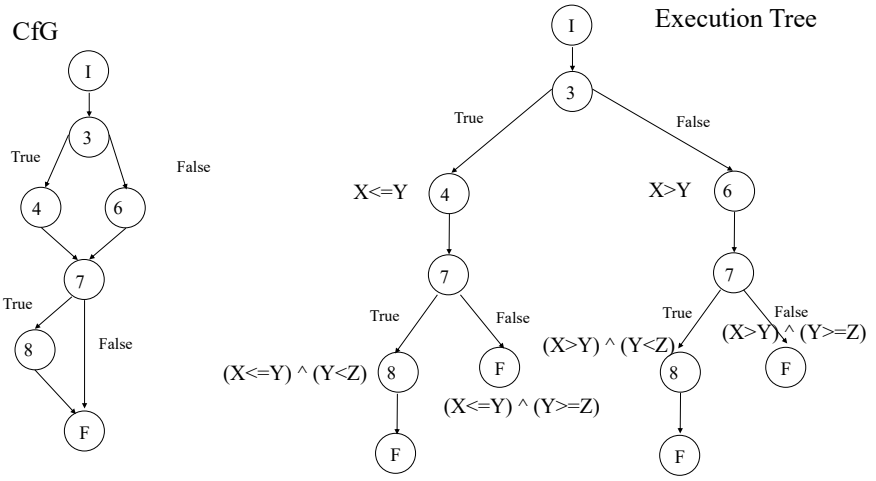
```
1   int max3 (int x, int y, int z) {
2   int max;
3       if (x <= y)
4               max = y;
5           else
6                max = x;
7       if (max < z)
8               max = z;
9       return max;
10  }
```

Stmt	pc	max	Stmt	pc	max
1	true	?	**1**	true	?
Case (x>y)			*Case (x<=y)*		
6	X > Y	X	**4**	X < = Y	Y
case (max< z)			*case (max< z)*		
8	(X>Y) ^ (X<Z)	Z	**8**	(X=<Y) ^ (Y<Z)	Z
ritorna questo valore per max			ritorna questo valore per max		
case (max => z)			*case (max => z)*		
8λ	(X>Y) ^ (X>=Z)	X	**8λ**	(X=<Y) ^ (Y>=Z)	Y
ritorna questo valore per max			ritorna questo valore per max		

7

7

Execution Tree (1)



Ciascuna foglia dello execution tree rappresenta un cammino che sarà percorso per una certa pc

8

8

Execution Tree (2)

- Ogni foglia dello execution tree rappresenta un cammino che sarà percorso per certi insiemi di valori di input
- Le *pc* associate a due differenti foglie sono distinte; ciascuna foglia dello execution tree rappresenta un cammino che sarà percorso per la pc ad essa associata.
- Non esistono esecuzioni per cui sono vere contemporaneamente più *pc* (per linguaggi di programmazione sequenziali).
- Se l'output ad ogni foglia è corretto allora il programma è corretto.
  - ✓ Ma, quanti rami può avere un execution tree?

9

9

Feasible and Infeasible paths

- Diremo cammino eseguibile (***feasible path***) un cammino per il quale esiste un insieme di dati in ingresso che soddisfa la path condition
- Diremo cammino non eseguibile (***infeasible path***) un cammino per il quale non esiste un insieme di dati di ingresso che soddisfa alla path condition
- L'uso di un theorem prover durante l'esecuzione simbolica può semplificare le path conditions ed eliminare infeasible paths, ma …

10

10

Altri problemi indecidibili derivati dalla indecidibilità dell'equivalenza di funzioni

- Il problema di dimostrare che due cammini di un programma calcolano la stessa funzione è indecidibile
- Il problema di dimostrare l'equivalenza di due espressioni simboliche di due cammini di un programma è indecidibile

11

11

Eseguibilità e Cammini

- Davis 1973: il problema di stabilire se esiste una soluzione per un sistema di diseguaglianze è indecidibile
- Un cammino è eseguibile se esiste un punto nel dominio di ingresso che rende soddisfattta la sua path condition
  - ✓ ... un sistema di diseguaglianza ...
- La determinazione della feasibility o infeasibility di un cammino è indecidibile

12

12

Tuttavia …

se si riesce a dimostrare che ciascun predicato nella path condition è dipendente linearmente dalle variabili di ingresso allora il problema è risolubile con algoritmi di programmazione lineare

13

13

Problemi di raggiungibilità indecidibili (Weyuker ’79) – (1)

- Stabilire se:
  - ✓una istruzione sarà eseguita per un punto del dominio di ingresso
  - ✓un ramo (branch: corrisponde ad un arco del GFC) sarà attraversato per un punto del dominio di ingresso
  - ✓un cammino sarà eseguito per un punto del dominio di ingresso

14

14

Problemi di raggiungibilità indecidibili (Weyuker ’79) – (2)

- Stabilire se :
  - ✓per ogni istruzione di un programma esiste un insieme definito di punti del dominio di ingresso che ne implica l'esecuzione
  - ✓per ogni ramo di un programma esiste un insieme definito di punti del dominio di ingresso che ne implica l'esecuzione
  - ✓per ogni cammino di un programma esiste un insieme definito di punti del dominio di ingresso che ne implica l’esecuzione

15

15