Esame di testing e verifica del software – 14/6/2021 3.15h

Senza appunti.

Senza eclipse

1. Testing di un programma

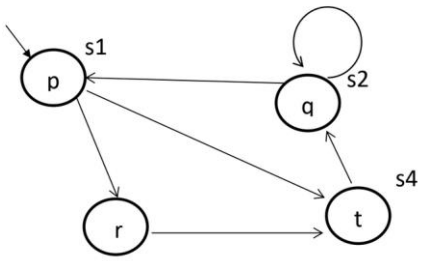
Dato il seguente programma in Java, che controlla se un array contiene un numero pari maggiore di b;

```
boolean check(int[] a, int b) {
	if (a != null && a.length > 0) {
		for (int x : a) {
			if (x > b && x % 2 == 0)
				return true;
		}
	}
	return false;
}
```

Scrivi i casi di test per: la copertura delle istruzioni (1), delle decisioni (2) e MCDC (3). Cerca di **minimizzare** i casi di test necessari per avere queste coperture. Non è necessario che tu scriva i test come junit (però deve essere possibile scrivere i test junit dai tuoi test).

2. algoritmo di model checking

Data la seguente macchina M



Mediante l'algoritmo di model checking, dire in quali stati s valgono le proprietà:

- M,s |= AF(p or q)
- M,s |= AG(t)

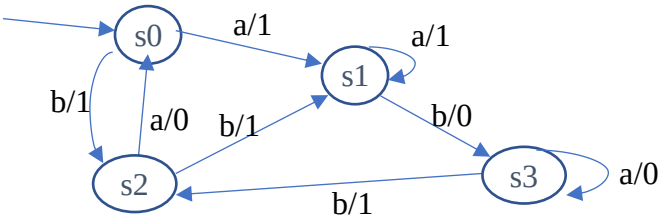
3. combinatorial testing

Date quattro variabili con i loro domini S: a,b,c , T: d,e,f, V:1,2,3, W: 4,5,6

Costruisci la test suite combinatoriale pairwise usando l'algoritmo IPO

4. conformance testing

Data la seguente FSM con 4 stati, due input a e b e due output 1 e 0



La macchina è correttamente definita? (giustifica la risposta)

Scrivi due test sequences, una per la copertura degli stati e una per la copertura delle transizioni.

Fa due esempi di errori (di quelli visti) e controlla se riesci a scoprirli con i test che hai trovato tu (può succedere che tu non riesca a scoprirli?) Giustifica la risposta.

Con eclipse

5. JML e KEY

Riscrivi il programma dell'esercizio 1 in cui metti gli opportuni contratti e provi la correttezza. Puoi semplificare il codice se i contratti che metti rendono superflui alcuni controlli. Puoi riscrivere il codice a tuo piacimento per portare a termine la dimostrazione. Ricordati la sintassi dei quantificatori:

(\forall <dominio>;<range_valori>;<condizione>)

(\exists <dominio>;<range_valori>;<condizione>)

Proof Result
Closed

6. logica temporale con asmeta

Un forno può essere in standby (chiuso e spento), con la porta aperta (ma spento) oppure acceso (la porta deve essere chiusa). Modella questi stati e le opportune azioni con Asmeta.

Prova le seguenti proprietà con LTL o CTL a tua scelta:

1. Quando è acceso la porta è sempre chiusa.
2. Prima o poi si può accendere in qualsiasi momento in futuro.
3. La porta può essere aperta dopo che viene acceso.
4. Quando è acceso, la porta rimane chiusa fino quando rimane acceso (usa Until).

C'è qualche proprietà che invece è giustamente falsa e il cui controesempio ti aiuta a capire come funziona il forno?