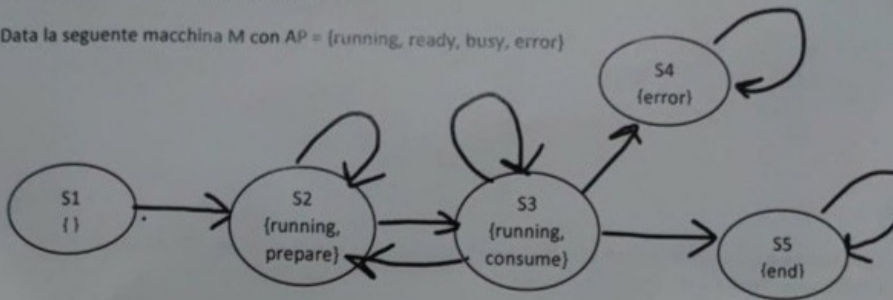


1. algoritmo di model checking

Data la seguente macchina M con $AP = \{\text{running, ready, busy, error}\}$



Mediante l'algoritmo di model checking trova in quali stati valgono queste proprietà.

1. $M \models AG(\text{running} \Rightarrow (\text{prepare or consume}))$
2. $M \models \text{error} \Rightarrow AG(\text{error})$
3. $M \models AG(EF(\text{error}))$
4. $M \models \text{running} \Rightarrow EX(\text{not running})$
5. $M \models A(\text{running} U (\text{end or error}))$
6. $M \models EX(\text{end or error})$
7. $M \models AG(\text{error} \Rightarrow \text{not running})$
8. $M \models \text{consume} \Rightarrow EG(\text{error})$

Nota che proprietà CTL potrebbero aver bisogno di essere trasformate per applicare l'algoritmo di model checking.

Commenta il perché alcune proprietà valgono in alcuni stati e in altri no come ti aspetteresti. Dà anche una traduzione di ogni proprietà in linguaggio naturale

$$1) AG(r \Rightarrow (P \text{ or } C)) = \neg E(\neg U \neg \underbrace{(r \Rightarrow (P \text{ or } C))}_{\varphi})$$

	r	P or C	$\underbrace{r \Rightarrow P \text{ or } C}_{\varphi}$	$E(\neg U \neg \varphi)$	$\neg E(\neg U \neg \varphi)$
S ₁	-	-	x	-	x
S ₂	x	x	x	-	x
S ₃	x	x	x	-	x
S ₄	-	-	x	-	x
S ₅	-	-	x	-	x

IN TUTTI I CASI GLOBALMENTE RUNNING IMPLICA O PREPARE O CONSUME

$$2) \text{ERROR} \Rightarrow \text{AG}(\text{Error}) \quad == \quad \text{Error} \Rightarrow \neg E(\neg \text{Error})$$

	er	$\neg \text{er}$	$E(\neg \text{er})$	$\neg E(\neg \text{er})$
s_1	-	x	x	-
s_2	-	x	x	-
s_3	-	x	x	-
s_4	x	-	-	x
s_5	-	x	x	-

ERROR IMPLICA PER OGNI CASO GLOBALMENTE ERROR

$$3) \text{AG}(EF(\text{Error})) \quad == \quad \neg E(\neg E(\neg \text{Error}))$$

	er	$E(\neg \text{er})$	$\neg E(\neg \text{er})$	$E(\neg \neg E(\neg \text{er}))$	$\neg E \dots$
s_1	-	x	-	x	-
s_2	-	x	-	x	-
s_3	-	x	-	x	-
s_4	x	x	-	-	x
s_5	-	-	x	x	-

PER OGNI CASO GLOBALMENTE ESISTE IN FUTURO ERROR

$$4) \text{RUNNING} \Rightarrow \text{EX}(\neg \text{Running})$$

	r	$\neg r$	$\text{EX}(\neg r)$	$r \Rightarrow \text{EX}(\neg r)$
s_1	-	x	- ?	x
s_2	x	-	- ?	-
s_3	x	-	x	x
s_4	x	-	x	x
s_5	-	x	x	x

ha senso?
 è perché error
 è "colpa"?
 penso di
 sì:

RUNNING IMPLICA ESISTE NEXT NOT RUNNING

5) A(ruhhiq u (ehor ennar))

	r	eh	er	ehorer	<u>ru(ehorer)</u>	A
S ₁	-	-	-	-	-	-
S ₂	x	-	-	-	x	x
S ₃	x	-	-	-	x	x
S ₄	-	-	x	x	x	x
S ₅	-	x	-	x	x	x

posso prelevare
soli?

Non vola perché
in S_1 non vola?

so Aress: Ax
variable?

6) Ex(eho or ennon)

	eh	er	eh or en	Ex(eh or en)
S ₁	-	-	-	x
S ₂	-	-	-	x
S ₃	-	-	-	x
S ₄	-	x	x	x
S ₅	x	-	x	x

0 cos: ?

ESISTE 1 PERCORSO NEL PROX STATO DOVE END OPPURE ERROR

7) $AG(\underbrace{error \Rightarrow \neg running}_{\varphi}) \equiv \neg E(\neg \varphi)$

	er	t	$\neg r$	φ	$\neg \varphi$	$E(\pi \cup \neg \varphi)$	$\neg E(\pi \cup \neg \varphi)$
s_1	-	-	x	x	-	-	x
s_2	-	x	-	x	-	-	x
s_3	-	x	-	x	-	-	x
s_4	x	-	x	x	-	-	x
s_5	-	-	x	x	-	-	x

IN OGNI CASO GLOBALMENTE ERROR IMPLICA NOT RUNNING

8) Consue $\Rightarrow EG(annon) == ?$

	er	Co	EG(er)	Co \Rightarrow EG(er)
s ₁	-	-	-	x
s ₂	-	-	-	x
s ₃	-	x	-	-
s ₄	x	-	x	x
s ₅	-	-	-	x

2. combinatorial testing

Date 3 variabili con i loro domini $x: 1,2,3$ $y: a,b,c$ $z: F,G$

Costruisci la test suite combinatoriale pairwise usando l'algoritmo IPO (cerca di minimizzare la dimensione della test suite).

Se ci fosse il vincolo che le tuple $[x=3 \text{ e } z = F]$ e $[x=3 \text{ e } z = G]$ sono vietate, dovresti modificare la test suite?
Se sì come, se no perché?

iPO Bahale

x	y	z
1	A	F
2	B	G
3	C	F
1	B	G
2	C	F
3	A	G
1	C	F
2	A	G
3	B	F
3	C	G

iPO Miglione

x	y	z
1	A	F
2	B	G
3	C	F
1	B	G
2	C	F
3	A	G
1	C	G
2	A	F
3	B	G

iPO

Vincolato

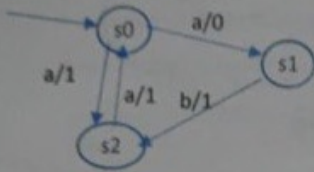
x	y	z
1	A	F
2	B	G
3	C	F
1	B	G
2	C	F
3	A	G
1	C	G
2	A	F
3	B	G

=>

x	y	z
1	A	F
2	B	G
1	C	F
2	A	G
1	B	F
2	C	G
1	-	G
2	-	F

3. conformance testing

Data la seguente FSM con 3 stati, due input a e b e due output 1 e 0



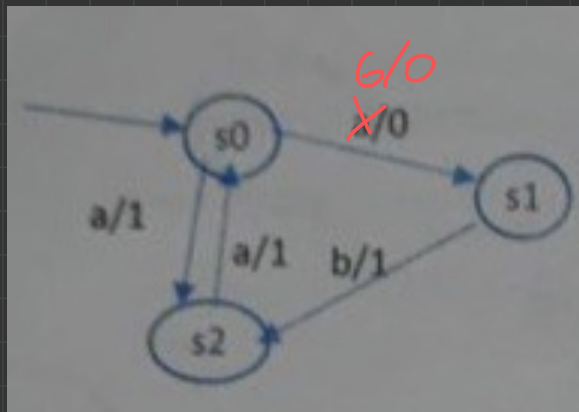
La macchina è correttamente definita? (giustifica la risposta – se non lo è, correggi la macchina modificandola come vuoi tu anche sul foglio)

1. Scrivi due test sequences, una per la copertura degli stati e una per la copertura delle transizioni.
2. Fa due esempi di errori (di quelli visti) e scrivi se possibile, per ognuno di essi, una test sequence che pur coprendo tutte le transizioni non li scopre (senza usare status message). Se aggiungi lo status, il test scopre l'errore? Se non è possibile trovare tale sequenza spiega perché.

Indica bene le test sequences come test di input e anche gli output attesi.

OUTPUT TRANSITION

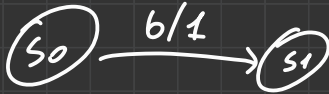
3) No, ogni stato deve usare tutti gli input (a,b)
Comunque così, e poi ho una macchina deterministica

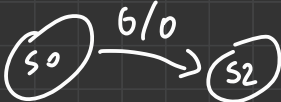


è un
problema?
Pensò di
sì

3.1) $S_{TATi} = 662$, 011
 $T_{RANS:Z:OH} = 6622$, 0111

3.2) errorne OUTPUT e errorne TRANSFER,

OUTPUT  BOH

TRANSFER  ...