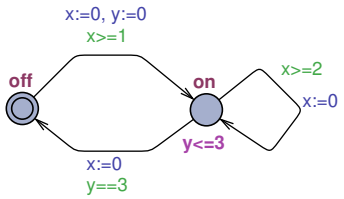
Esercizi su automi temporizzati e Uppaal

Per la sintassi utilizzata da Uppaal consultare l'Help del sistema stesso.

1 Switch

Si consideri il processo **switch** rappresentato dall'automa:



Formulare query Uppaal per verificare se:

1. l'interruttore verrà sempre acceso prima o poi
2. esiste la possibilità che l'interruttore sia sempre spento, per tutta la durata di un'esecuzione
3. ogni volta che l'interruttore è acceso verrà spento prima o poi
4. esiste la possibilità che l'interruttore resti acceso per più di 10 time units

Verificare in Uppaal se le proprietà sono soddisfatte o no.

2 Elevator

Modellare un ascensore che opera tra due piani soltanto (piano terra e primo piano). Quando l'ascensore arriva a un certo piano, la porta si apre automaticamente tra 2 e 5 secondi da quando è arrivato. Quando la porta è aperta possono entrare, uno ad uno, i passeggeri (non ci sono limiti di spazio!). La porta poi si chiude esattamente 6 secondi dopo l'ingresso dell'ultimo passeggero. Quando la porta si è chiusa, l'ascensore aspetta tra 2 e 4 secondi, poi si muove verso l'altro piano. L'ascensore è inizialmente al piano terra con la porta chiusa.

Suggerimento: utilizzare 3 stati per ogni piano, uno di arrivo, uno di partenza, e uno con un autociclo in cui la porta è aperta e possono entrare i passeggeri.

Formulare una query Uppaal che esprima il fatto che in ogni esecuzione esiste uno stato in cui l'ascensore arriva al primo piano e verificare che il sistema non la soddisfa. Esaminare la traccia generata da Uppaal per capirne il motivo.

3 Satellite

Un satellite può trovarsi in diversi stati, che chiamiamo: Earth (il satellite è orientato verso una stazione ricevente di terra ma non trasmette dati), Comm (il satellite trasmette dati alla stazione di terra), Slewing (il satellite ruota per cambiare il suo orientamento) e Science (il satellite effettua attività scientifiche, come fare fotografie al pianeta a cui è vicino). Inizialmente il satellite si trova nello stato Earth.

Modellare tale sistema mediante un TA in modo tale che:

- l'automa possa restare nella locazione Comm tra 30 e 50 unità di tempo (u.t.)
- l'automa resti sempre nella locazione Slewing esattamente 30 u.t. tempo
- l'automa possa restare nella locazione Science tra 40 e 60 u.t.

Formulare e verificare in Uppaal se il sistema soddisfa le proprietà seguenti: (1) ogni volta che il satellite si trova nello stato Science, si troverà prima o poi nello stato Comm; (2) esiste almeno un'esecuzione in cui il satellite si trova almeno una volta nello stato Comm; (3) in tutte le esecuzioni il satellite si trova almeno una volta nello stato Comm; (4) in ogni esecuzione, il satellite non resta nello stato Science per più di 60 u.t.

4 Cane e pancia

Modellare mediante una rete di TA un sistema biologico costituito da un cane e la sua pancia.

- Il cane può essere in uno degli stati seguenti: gioca (stato iniziale), mangia, dorme. Le transizioni sono da gioca a mangia, da mangia a dorme e da dorme a gioca.

  Per mangiare ci mette da 2 a 4 unità di tempo e non può dormire più di 8 unità di tempo.

- La pancia può essere piena (stato iniziale) o vuota, e le transizioni sono da piena a vuota e da vuota a piena. La pancia non resta piena per più di 10 unità di tempo.

- I due automi sono sincronizzati in questo modo:

  - quando la pancia non è più piena manda al cane il messaggio "fame" e, se il cane sta giocando, inizia allora a mangiare;
  - quando il cane finisce di mangiare manda alla pancia il messaggio "mangiato" e la pancia passa allora nello stato piena.

Formulare e verificare in Uppaal se il sistema soddisfa le proprietà seguenti: (1) in ogni esecuzione esiste almeno uno stato in cui il cane mangia; (2) esiste un'esecuzione in cui il cane gioca con la pancia vuota almeno una volta; (3) esiste un'esecuzione in cui il cane gioca sempre.

5 Printer and user

Definire una rete di TA che rappresenti un sistema costituito da una stampante e un utente. La stampante può trovarsi nello stato idle (stato iniziale) o nello stato printing. Per stampare ci mette tra 5 e 10 unità di tempo (u.t.). L'utente può lavorare o aspettare il termine della stampa, ma non lavora per più di 100 u.t. Inizialmente lavora.

I due automi sono sincronizzati: la stampante inizia a stampare quando riceve un comando dall'utente, e l'utente esce dallo stato di attesa quando la stampante l'avverte che la stampa è terminata.

Formulare (1) una query Uppaal per verificare se ogni esecuzione ha almeno uno stato in cui la stampante stampa, e (2) una per verificare se ogni stato in cui l'utente lavora è sempre seguito da uno stato in cui aspetta la stampa.

6 Satellite (2)

Considerare l'automa del satellite del punto 3 e modificarlo in modo tale che $satellite.Science \rightsquigarrow satellite.Comm$ sia soddisfatta.

Per garantire questo, fare in modo che:

- Il satellite non possa restare all'infinito nello stato Earth, ponendo un limite massimo di 200 unità di tempo.
- Se il satellite ha acquisito dei dati (è passato per lo stato Science), sia obbligato a tornare in Earth e poi passare in Comm.

Formulare una query Uppaal per verificare che la prima proprietà è soddisfatta.

Per garantire la seconda proprietà, utilizzare una variabile booleana "done" che rappresenta il fatto che è stata fatta un'attività di Science. La variabile è inizialmente falsa, diventa vera quando si esce dallo stato Science e torna falsa quando si esce dallo stato Comm.

Per utilizzare la variabile booleana inizialmente falsa, si deve inserire nella sezione Declarations di Uppaal:

```
bool done=false;
```

Garantire inoltre che:

1. Il satellite possa comunicare dati (passare da Earth a Comm) soltanto dopo che ha eseguito un'attività Science, e
2. possa fare scienza (passare da Earth a Slewing, e poi a Science) soltanto se non ha dati da comunicare.
3. Dallo stato Slewing non possa tornare allo stato Earth senza prima passare per Science;

Formulare due query Uppaal per verificare che la prima e la seconda proprietà sono soddisfatte (ricorrendo alla variabile booleana).

È possibile esprimere nel linguaggio di Uppaal una proprietà che consenta di verificare la terza proprietà?

7 Ground station

Il satellite può comunicare dati alla stazione di terra, solo durante un intervallo di disponibilità della stazione stessa.

Modellare il comportamento della stazione di terra mediante un TA, i cui stati sono NotVisible (iniziale) e Visible. L'automa può passare da NotVisible a Visible e viceversa; può restare NotVisible tra 50 e 80 unità di tempo e può restare Visible tra 70 e 100 unità di tempo.

Modificare l'automa che modella il comportamento del satellite in modo tale che possa trovarsi nello stato Comm soltanto quando la stazione di terra è nello stato Visible (utilizzare una variabile booleana **visible** che è vera solo quando la stazione di terra è Visible).

Formulare query Uppaal per verificare se:

1. In ogni stato di ogni esecuzione la stazione di terra è visibile quando il satellite sta comunicando dati
2. È possibile che la stazione di terra sia visibile mentre il satellite sta effettuando attività scientifiche

E verificare se il sistema soddisfa queste proprietà.

8 Elevator (2)

Modificare il sistema del punto 2 in modo tale che nell'ascensore non possano entrare più di 5 persone, e verificare che ora la proprietà è soddisfatta.

9 Worker and door

Un lavoratore lavora (stato "work") esattamente per 8 u.t. e, quando ha finito, deve prendere l'ascensore per uscire. Se la porta dell'ascensore è chiusa deve aspettare che si apra (stato "wait"). Quando esce, ha finito (stato "finish"), e non ci sono transizioni uscenti da "finish".

La porta dell'ascensore si apre e si richiude, restando in ciascuno degli stati "open" e "closed" tra 1 e 3 u.t. Inizialmente è chiusa.

Il sistema dovrebbe essere modellato in modo che il lavoratore prenda l'ascensore appena possibile.

Dare tre diversi modelli per questo sistema:

Sistema n. 1: il modello del lavoratore ha solo due transizioni, da "work" a "wait" e da "wait" a "finish". Utilizzare una variabile booleana "ready" inizialmente falsa, che diventa vera quando il lavoratore è pronto a uscire (è nello stato "wait"), e definire due transizioni dallo stato "closed" allo stato "open" della porta: una è eseguibile solo se il lavoratore non è pronto a uscire, e l'altra è invece sincronizzata con la transizione del lavoratore da "wait" a "finish".

Sistema n. 2: modificare il sistema precedente, utilizzando inoltre una variabile booleana "chiusa" che è vera quando la porta è chiusa, e vincolare il lavoratore a restare nello stato di attesa solo se la porta è chiusa.

Sistema n. 3: modificare il secondo sistema, eliminando la variabile "ready" e la transizione della porta da "closed" a "open" sincronizzata con quella del lavoratore da "wait" a "finish" (eliminando ovviamente anche la sincronizzazione). Far sì che il lavoratore aspetti solo se la porta è chiusa (come nel sistema n. 2), ma aggiungere una transizione da "work" a "finish" eseguibile solo se la porta è aperta.

Formulare queries Uppaal per esprimere le proprietà seguenti:

1. In nessuna esecuzione il lavoratore aspetta quando la porta dell'ascensore è aperta. Verificare che questa proprietà non è soddisfatta dal sistema n. 1 (spiegare perché), e lo è invece dai sistemi n. 2 e 3.

2. Nessuna esecuzione si trova mai in uno stato di deadlock (in cui nessun processo può compiere alcuna *action transition*). Questa proprietà è soddisfatta dal sistema n. 3, ma non dai sistemi n. 1 e 2 (spiegare perché).

3. Ogni esecuzione ha almeno uno stato in cui il lavoratore ha finito. Questa proprietà è soddisfatta dai sistemi n. 1 e 3, ma non dal sistema n. 2 (spiegare perché).

4. Esiste un'esecuzione che abbia almeno uno stato in cui il lavoratore lavora e la porta è aperta. Questa proprietà è soddisfatta da tutti e tre i sistemi.

5. Ogni esecuzione ha almeno uno stato in cui il lavoratore lavora e la porta è aperta. Questa proprietà è soddisfatta da tutti e tre i sistemi.

6. Ogni stato in cui il lavoratore lavora è sempre seguito da uno stato in cui ha finito. Questa proprietà è soddisfatta dai sistemi n. 1 e 3, ma non dal sistema n. 2 (spiegare perché).

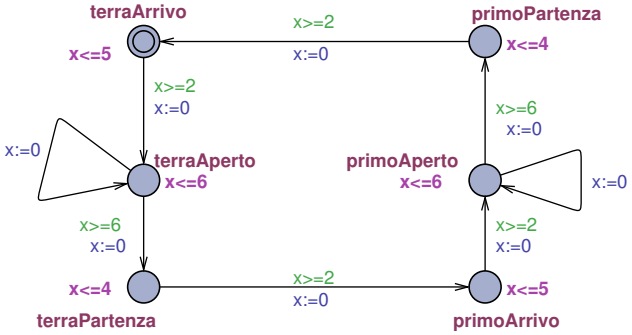
Soluzioni proposte

1 Switch

Queries:

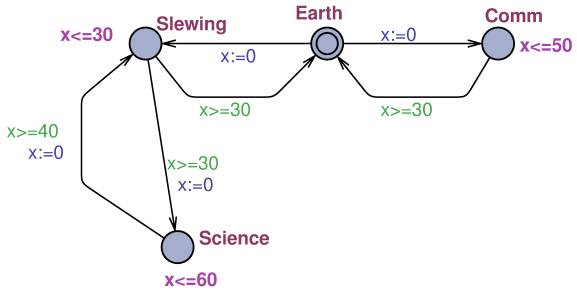
1. A$\Diamond$ *switch.on*: property is not satisfied
2. E$\Box$ $switch.off$: property is satisfied
3. $switch.on \rightsquigarrow switch.off$: property is satisfied
4. E$\Diamond(switch.on \wedge y > 10)$: property is not satisfied

2 Elevator



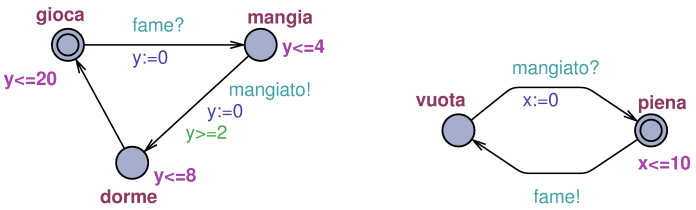
Query: A$\Diamond$*elevator.primoArrivo*

3 Satellite



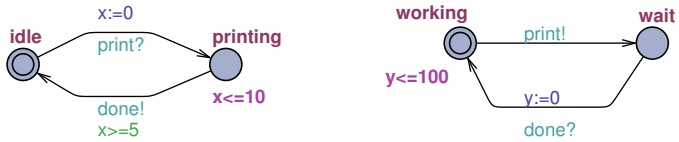
Queries: (1) $satellite.Science \rightsquigarrow satellite.Comm$; (2) E$\Diamond satellite.Comm$; (3) A$\Diamond satellite.Comm$; (4) A$\Box(satellite.Science \rightarrow satellite.x \leq 60$ (in Uppaal: `A[](satellite.Science imply satellite.x <= 60)`). Si assume che il clock x sia dichiarato localmente all'automa satellite, per cui occorre riferirvisi come "$satellite.x$".

4 Cane e pancia



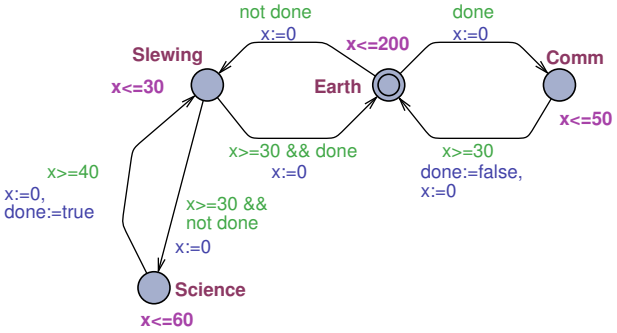
Queries (1) A$\Diamond$*cane.mangia*; (2) E$\Diamond$(*cane.gioca*$\wedge$*pancia.vuota*); (3) E$\Box$*cane.gioca*.

5 Printer and user



Queries: (1) A$\Diamond$*printer.printing*; (2) *user.working* $\rightsquigarrow$ *user.wait*.

6 Satellite (2)



Queries:

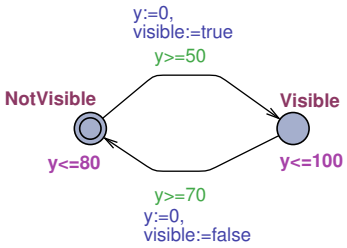
1. A$\Box$(*satellite.Comm* $\rightarrow$ *done*)
2. A$\Box$(*satellite.Science* $\rightarrow$ $\neg$ *done*)

La terza proprietà non può essere verificata mediante una query Uppaal: occorrerebbe infatti far riferimento alle locazioni in cui si trova il satellite dopo essere uscito dalla locazione Slewing. In CTL la proprietà si potrebbe esprimere in uno dei due modi seguenti:

- A$\Box$(*satellite.Slewing* $\rightarrow$ A$\bigcirc$(*satellite.Earth* $\rightarrow$ *done*): per ogni stato di ogni esecuzione, se il satellite è in Slewing, allora se la locazione successiva, in qualsiasi esecuzione che inizia da tale stato, è Earth, allora `done` è true;
- A$\Box$(*satellite.Slewing* $\rightarrow$ A($\neg$*satellite.Earth* U *done*) (dove U è l'operatore "until"): in ogni stato di ogni esecuzione, se il satellite è in slewing allora in ogni esecuzione che parte da tale stato il satellite non sarà in Earth finché `done` non è true.

Queste sono entrambe formule CTL, ma il linguaggio delle query di Uppaal è un sottoinsieme di CTL, che non contiene gli operatori "next" e "until" e non consente di nidificare operatori temporali (con l'eccezione di quello nascosto in $\rightsquigarrow$: ricordiamo che $F \rightsquigarrow G$ è un'abbreviazione di $\mathsf{A}\Box(F \rightarrow \mathsf{A}\Diamond G)$).

7 Ground station (ground)



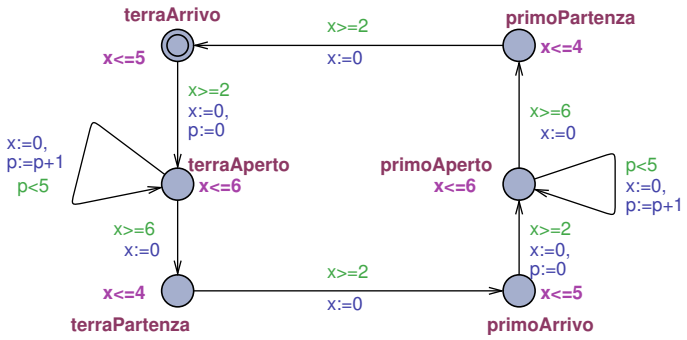
L'automa che modella il comportamento del satellite va modificato sostituendo l'invariante dello stato Comm con $x \leq 50 \wedge visible$.

Queries:

1. $\mathsf{A}\Box\ (satellite.Comm \rightarrow ground.Visible)$
2. $\mathsf{E}\Diamond(ground.Visible \wedge satellite.Science)$

8 Elevator (2)

Dichiarare una variabile intera per memorizzare il numero di passeggeri (inserire nelle dichiarazioni: `int p=0;`). La variabile sarà reinizializzata a 0 nelle transizioni che entrano in terraAperto e primoAperto, sarà incrementata di 1 in ciascuna delle transizioni terraAperto $\longrightarrow$ terraAperto e primoAperto $\longrightarrow$ primoAperto, e le stesse transizioni avranno la guardia $p < 5$:



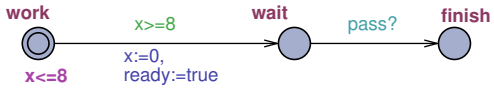
9 Worker and door

Sistema n. 1

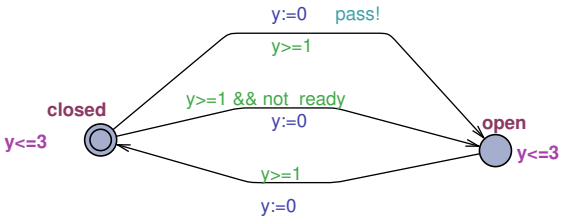
Oltre agli orologi, vanno dichiarati il canale di sincronizzazione e la variabile booleana:

```
chan pass;
bool ready = false;
```

worker:

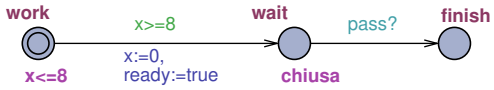


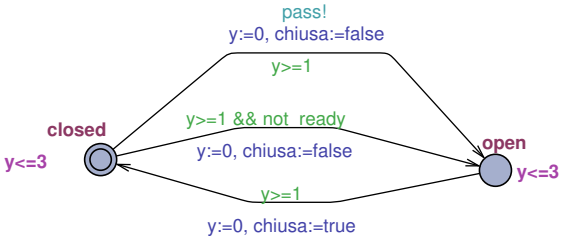
door:



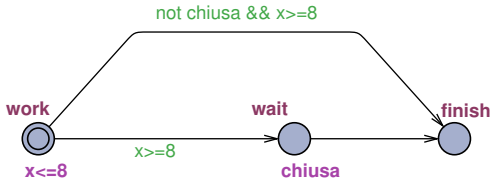
Sistema n. 2

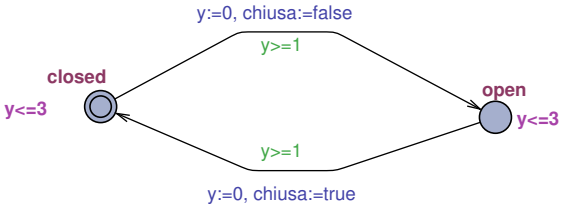
Va dichiarata anche la variabile "chiusa": `bool chiusa=true;`





Sistema n. 3





Queries

1. $\mathsf{A}\Box(worker.wait \rightarrow \neg\, door.open)$. Il sistema n. 1 non la soddisfa perché il lavoratore potrebbe entrare in "wait" quando la porta è già aperta, e in tal caso deve aspettare che si chiuda e si apra di nuovo per rispettare la sincronizzazione.

2. $\mathsf{A}\Box\neg\, deadlock$. I sistemi n. 1 e 2 entrano in deadlock quando il lavoratore è nello stato finish e la porta è chiusa: infatti la porta non può più aprirsi, dato che non può sincronizzarsi con il passaggio del lavoratore da "wait" a "finish". Nel terzo sistema invece l'apertura e chiusura della porta non è sincronizzata con alcuna transizione del lavoratore, quindi la porta può continuare ad aprirsi e chiudersi all'infinito.

3. $\mathsf{A}\Diamond worker.finish$. La proprietà non è soddisfatta dal sistema n. 2 perché le 8 ore di lavoro possono scadere quando la porta è aperta, e il lavoratore non può quindi entrare in "wait" (questo è un altro caso di deadlock per il sistema n. 2).

4. $\mathsf{E}\Diamond(worker.work \wedge door.open)$.

5. $\mathsf{A}\Diamond(worker.work \wedge door.open)$.

6. $worker.work \rightsquigarrow worker.finish$. La proprietà non è soddisfatta dal sistema n. 2 per lo stesso motivo per cui non è soddisfatta la proprietà 3.