Il model checker UPPAAL

UPPAAL: http://www.uppaal.org/

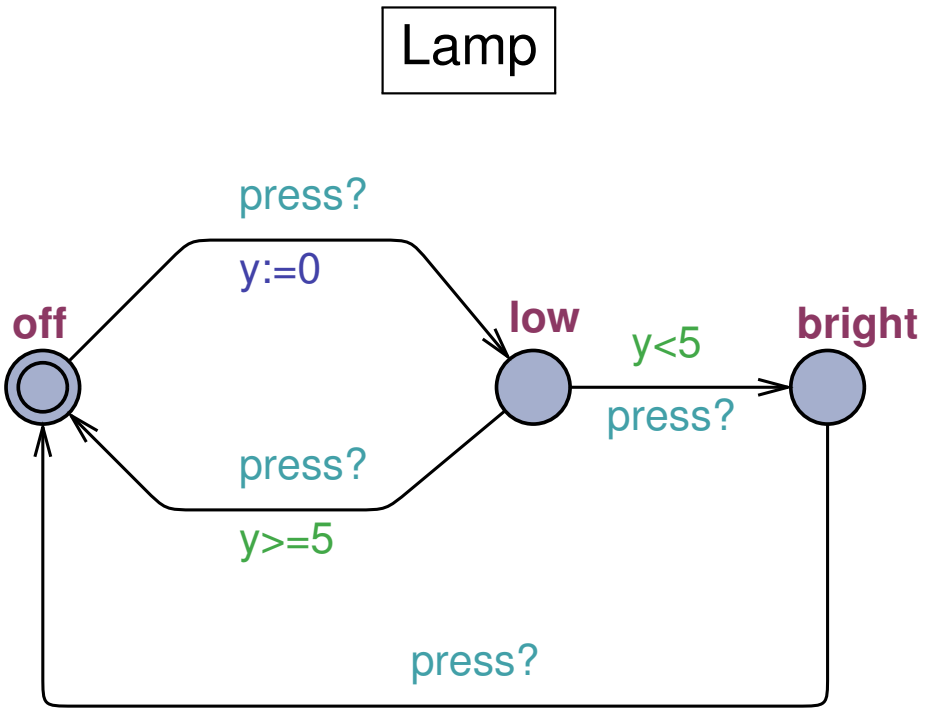
Language Reference:
http://www.uppaal.com/index.php?sida=217&rubrik=101

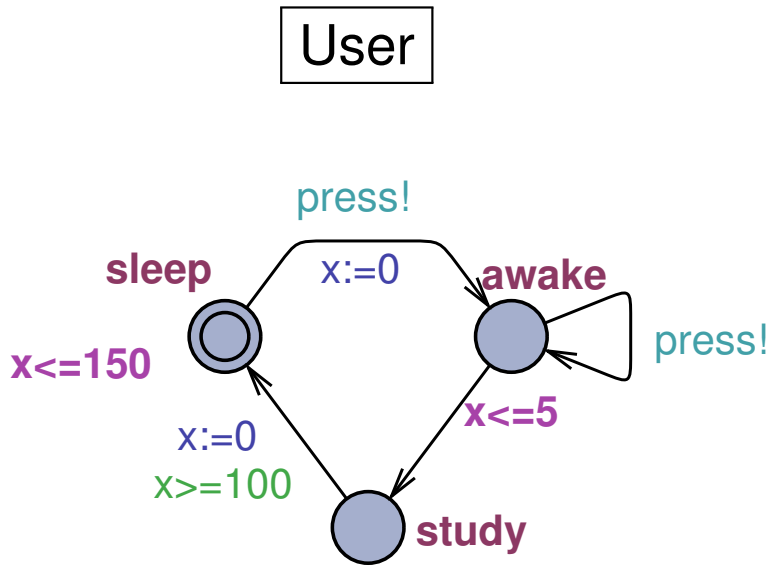
I sistemi sono modellati come reti di timed automata, con l'aggiunta di

- variabili
- tipi di dati strutturati (array, record, . . .)
- broadcast channels
- .....

Il **linguaggio di specifica** per le proprietà da verificare è un sottoinsieme di **CTL** (Computational Tree Logic).

Esempio





Declarations: chan press;

lamp Declarations: clock y;

user Declarations: clock x;

System declarations: system lamp, user;

Formato dei file: XML, XTA

lamp.xta

```
// Place global declarations here.
chan press;

process lamp() {
clock y;
state
    bright, low, off;
init off;
trans
    bright -> off { sync press?;  },
    low -> bright { guard y<5; sync press?;  },
    low -> off { guard y>=5; sync press?;  },
    off -> low { sync press?; assign y:=0;  };
}

process user() { ..... }

system lamp, user;
```

Queries

lamp.q

```
/* ogni run del sistema ha almeno uno stato in cui
   il processo lamp e` nello stato low */
A<> lamp.low

/* esiste una run con uno stato in cui il processo lamp
   e` nello stato bright */
E<> lamp.bright

/* ogni run e` deadlock-free */
A[] not deadlock

/* ogni run ha almeno uno stato in cui il processo lamp
   e` nello stato bright */
A<> lamp.bright
```

Computation Tree Logic (CTL)

Il linguaggio usato da UPPAAL per le queries è una forma semplificata di CTL.

CTL appartiene alla famiglia delle **branching time logics**.

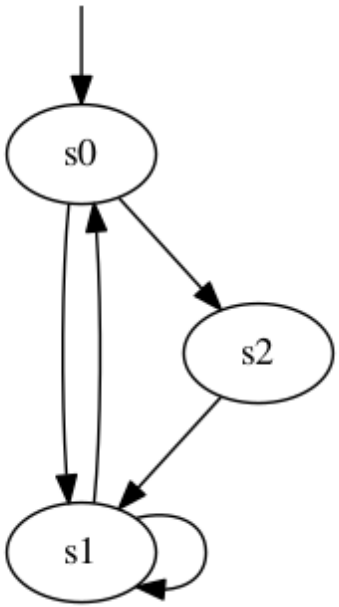
Descrive proprietà di un **computation tree**

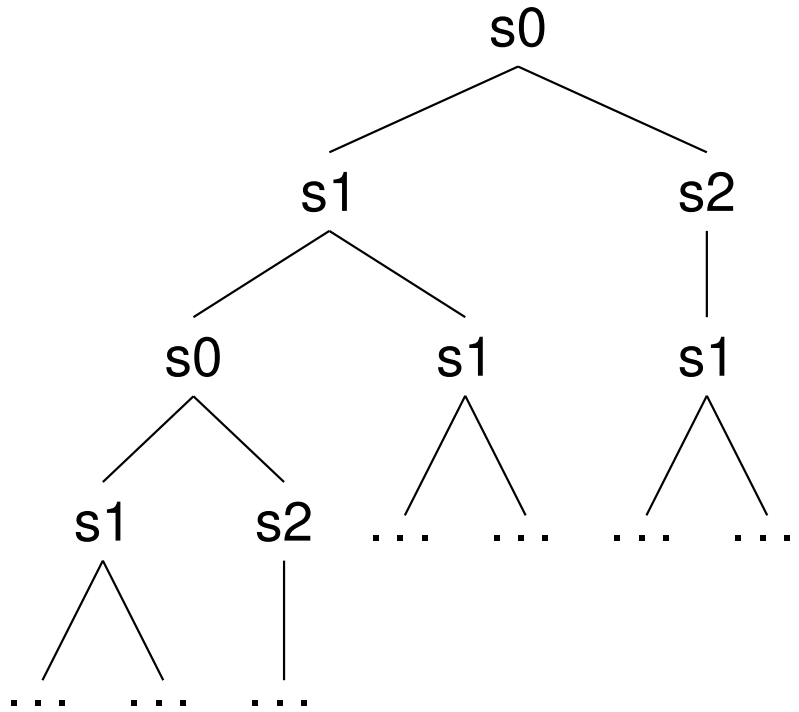
Operatori temporali in CTL: combinazione di un quantificatore (su path) con un operatore di LTL (relativo al path)

Computation tree

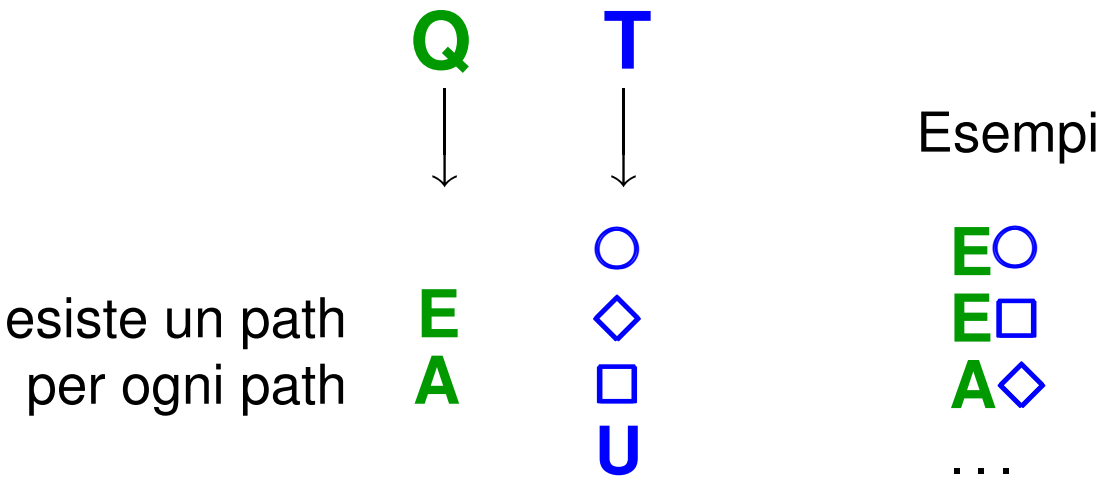
Dato un sistema di transizioni $\mathcal{T} = \langle S, \longrightarrow, s_0 \rangle$, l'albero di computazione di $\mathcal{T}$ è l'**unfolding** di $\mathcal{T}$: albero (finito o infinito) in cui:

- ogni nodo è etichettato da un elemento di S
- la radice è etichettata da s_0
- per ogni nodo n: se n è etichettato da s e $s \longrightarrow s'$, allora esiste un figlio di n etichettato da s'





Operatori temporali in CTL



Path: sequenza **massimale** di stati

Queries in UPPAAL: state formulae

- **State formulae**, che descrivono singoli stati.
- **Path formulae**, che quantificano sulle run (path) del modello

Una state formula è un'espressione che viene valutata in uno stato

```
x>3    y==2    x<=3 and y==52    x<=3 imply y==5
```

Inoltre:

- se P è un processo e l una locazione di P:
  $P.l$ = il TA identificato dal processo P è nella locazione l
- **deadlock** è vera in ogni stato di deadlock della rete (nessuna transizione abilitata)

Path formulae

Operatori utilizzati in UPPAAL (F e G sono state formulae)

- **A$\Box F$**: per ogni path P e per ogni stato s di P, $s \models F$ (safety)

  A [ ] F
- **A$\Diamond F$**: per ogni path P esiste uno stato s di P tale che $s \models F$ (liveness)

  A <> F
- **E$\Box F$**: esiste un path P tale che per ogni stato s di P, $s \models F$ (safety)

  E [ ] F
- **E$\Diamond F$**: esistono un path P e uno stato s di P tali che $s \models F$ (reachability)

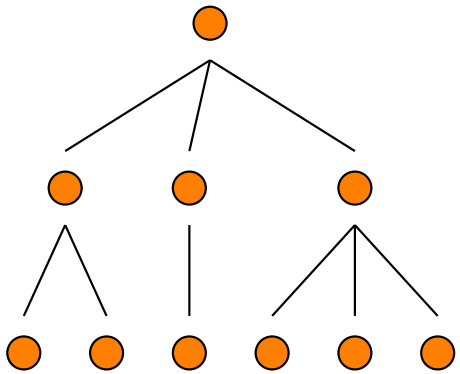
  E <> F
- $F \rightsquigarrow G$: per ogni path P e per ogni stato s di P, se $s \models F$ allora ogni path che inizia da s ha uno stato s' tale che $s' \models G$ (liveness)

  F **-->** G

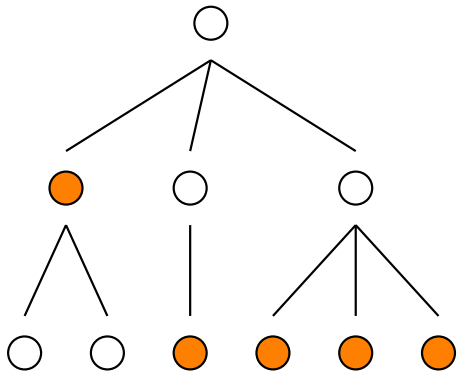
$$A\Box(F \rightarrow A\Diamond G)$$

Intuitivamente...

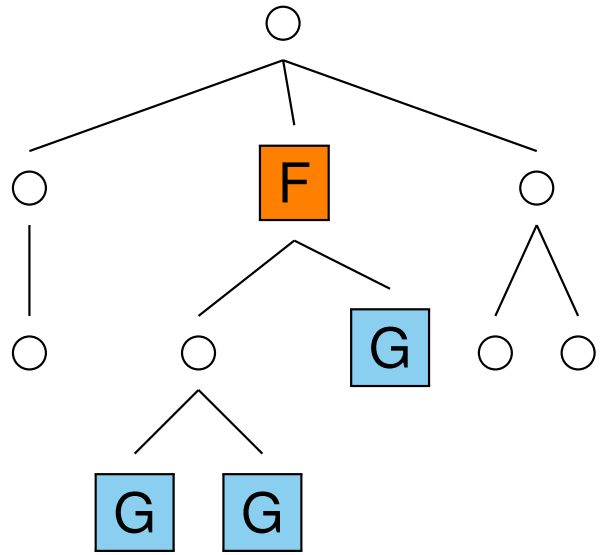
$A\Box F$



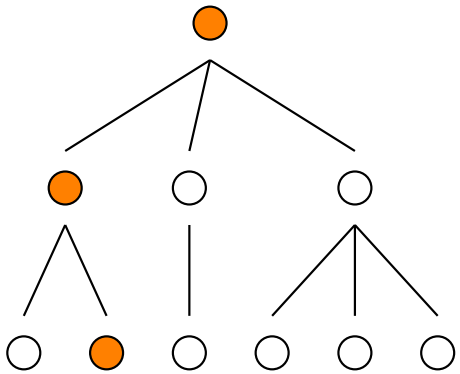
$A\Diamond F$



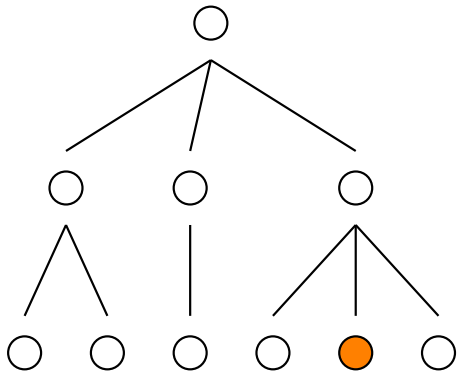
$F \rightsquigarrow G$



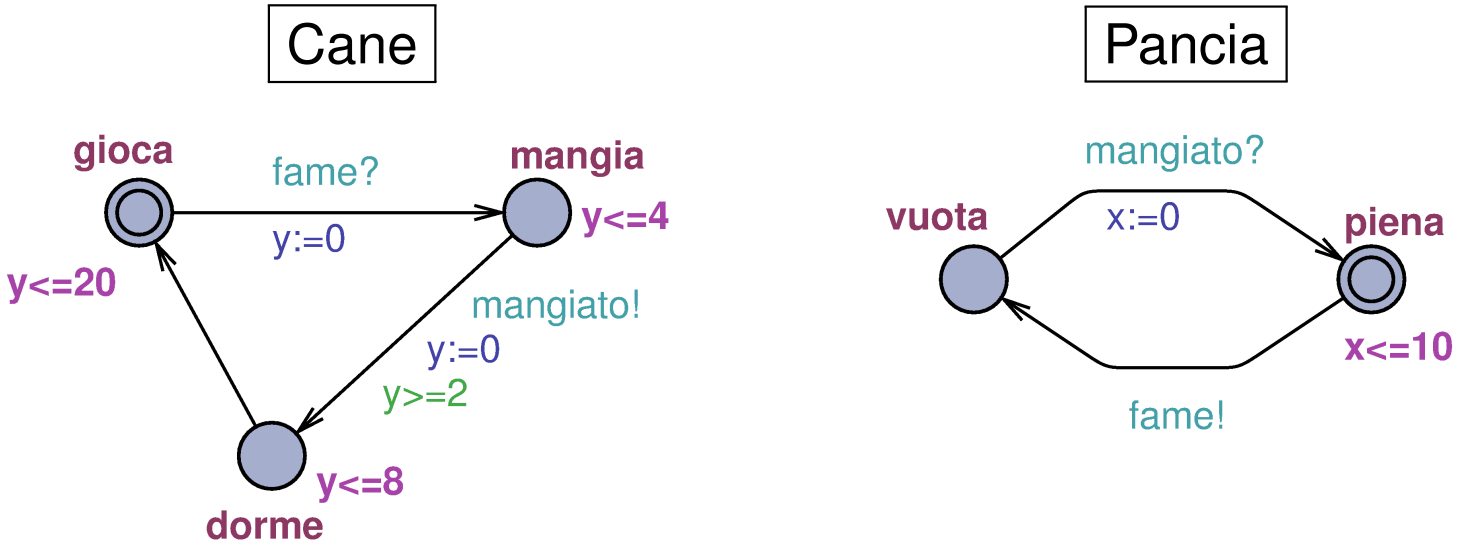
$E\Box F$



$E\Diamond F$

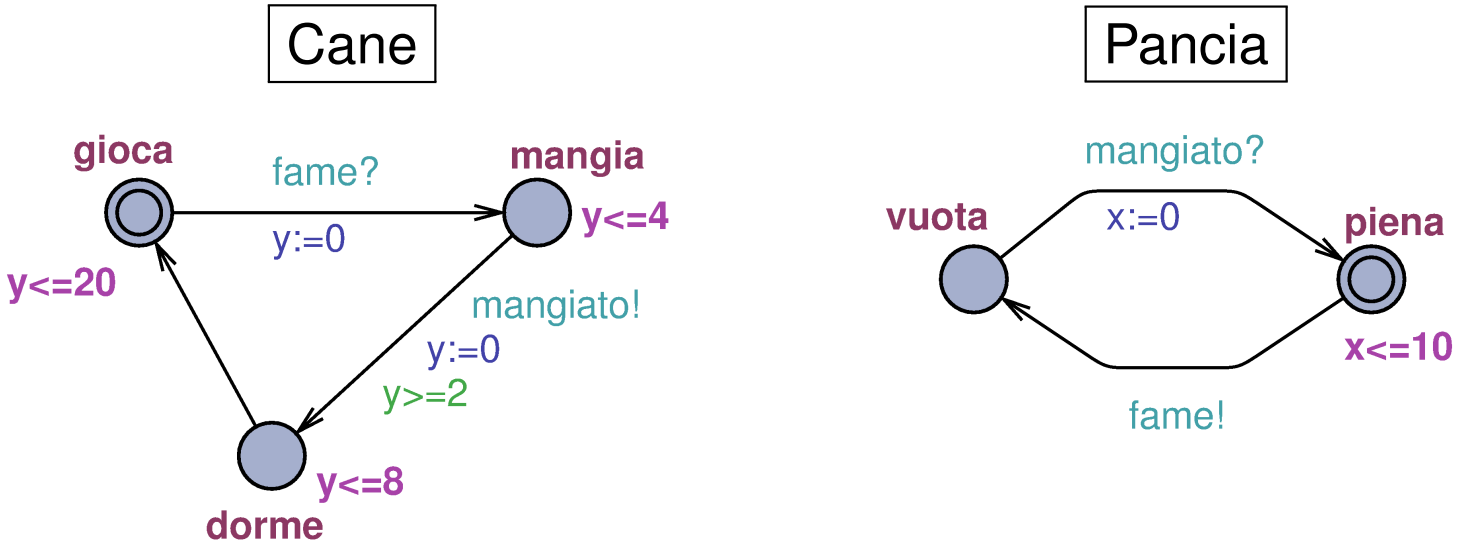


Esempio: la pancia del cane



In ogni esecuzione esiste almeno uno stato in cui il cane mangia:

Esempio: la pancia del cane

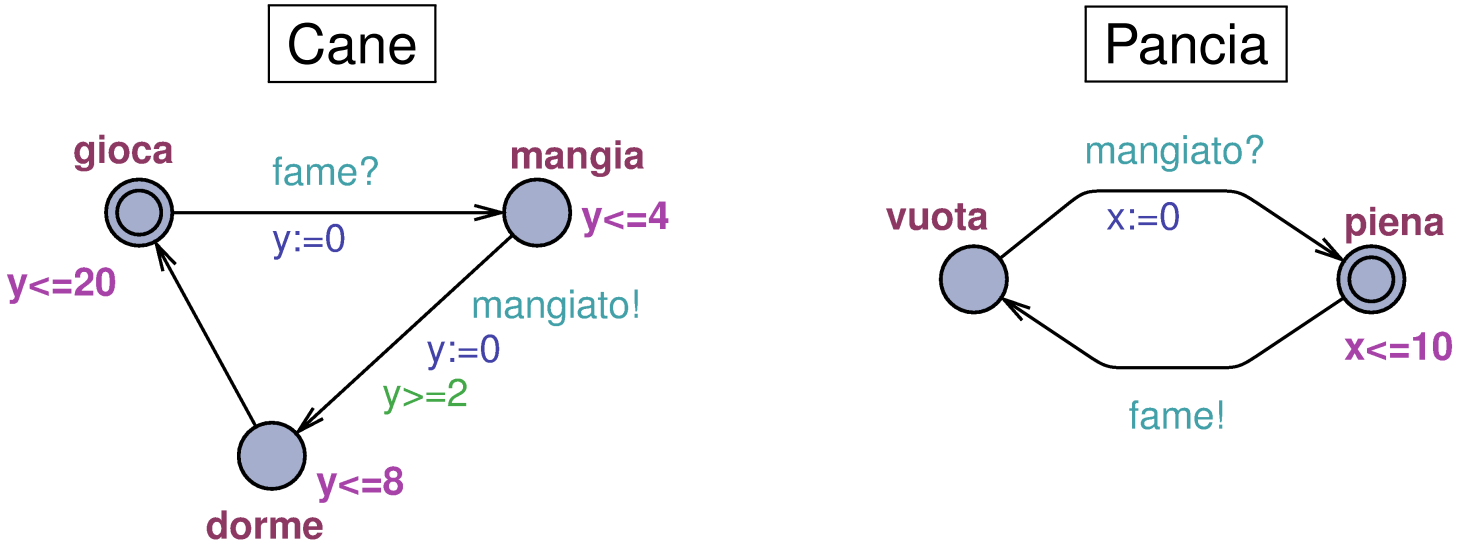


In ogni esecuzione esiste almeno uno stato in cui il cane mangia:

$$A\Diamond \mathit{cane.mangia}$$

In ogni esecuzione, nel futuro di ciascuno stato in cui il cane mangia c'è uno stato in cui il cane dorme:

Esempio: la pancia del cane



In ogni esecuzione esiste almeno uno stato in cui il cane mangia:

$$A\Diamond \mathit{cane.mangia}$$

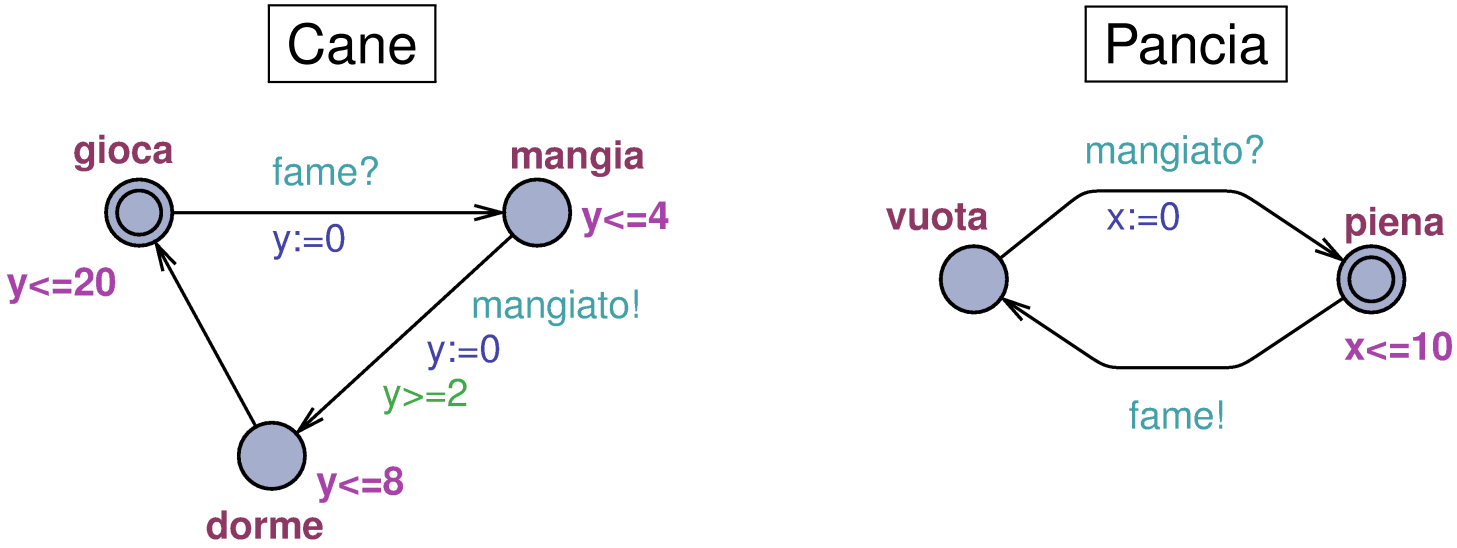
In ogni esecuzione, nel futuro di ciascuno stato in cui il cane mangia c'è uno stato in cui il cane dorme:

$$\mathit{cane.mangia} \leadsto \mathit{cane.dorme}$$

Ogni esecuzione è deadlock-free: $A\Box\neg \mathit{deadlock}$

Esiste un'esecuzione con uno stato in cui il cane gioca e la pancia è vuota:

Esempio: la pancia del cane



In ogni esecuzione esiste almeno uno stato in cui il cane mangia:

$$A\Diamond \mathit{cane.mangia}$$

In ogni esecuzione, nel futuro di ciascuno stato in cui il cane mangia c'è uno stato in cui il cane dorme:

$$\mathit{cane.mangia} \rightsquigarrow \mathit{cane.dorme}$$

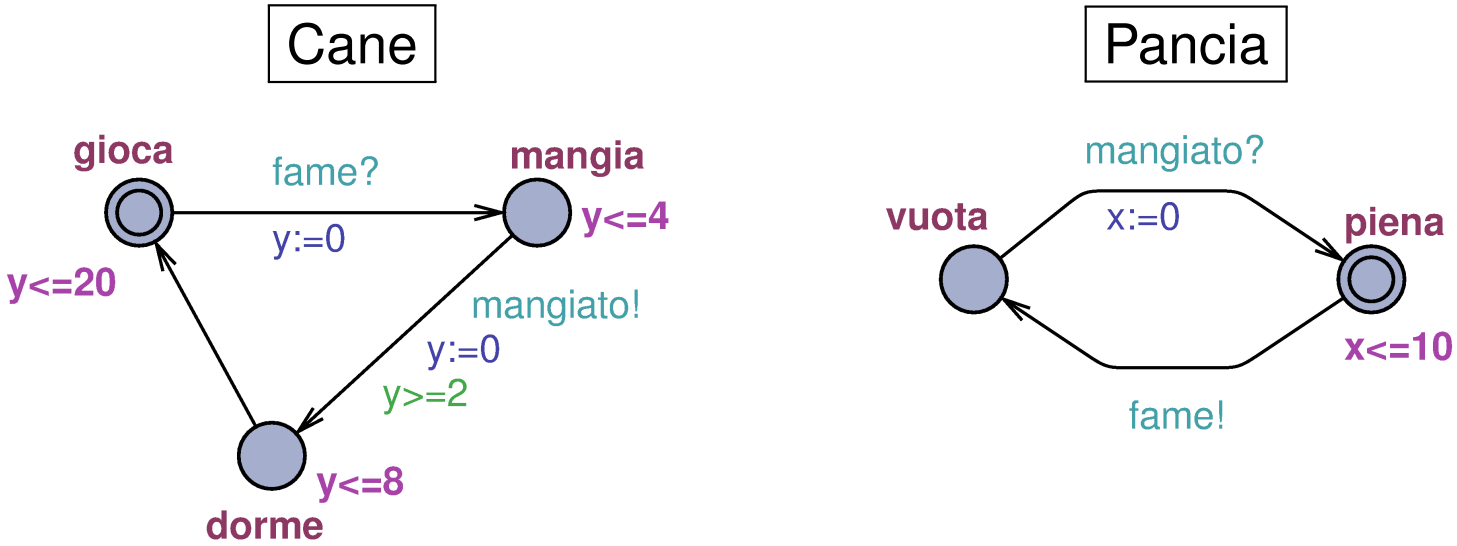
Ogni esecuzione è deadlock-free: $A\Box\neg \mathit{deadlock}$

Esiste un'esecuzione con uno stato in cui il cane gioca e la pancia è vuota:

$$E\Diamond(\mathit{cane.gioca} \wedge \mathit{pancia.vuota})$$

Esiste almeno un'esecuzione in cui il cane gioca sempre:

Esempio: la pancia del cane



In ogni esecuzione esiste almeno uno stato in cui il cane mangia:

$$A\Diamond \mathit{cane.mangia}$$

In ogni esecuzione, nel futuro di ciascuno stato in cui il cane mangia c'è uno stato in cui il cane dorme:

$$\mathit{cane.mangia} \rightsquigarrow \mathit{cane.dorme}$$

Ogni esecuzione è deadlock-free: $A\Box\neg \mathit{deadlock}$

Esiste un'esecuzione con uno stato in cui il cane gioca e la pancia è vuota:

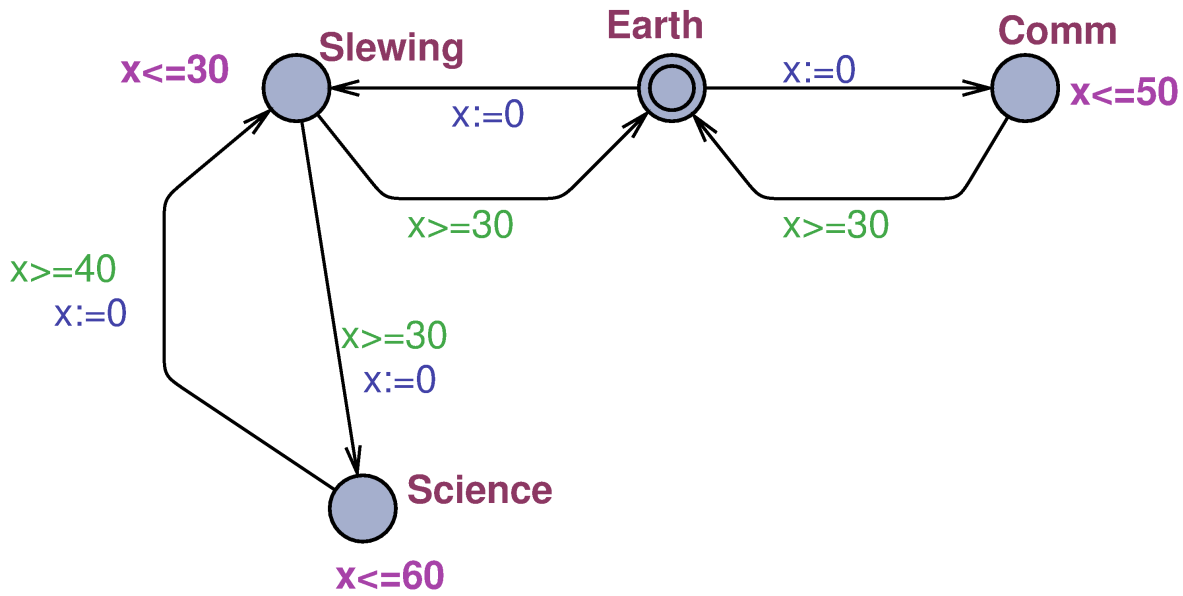
$$E\Diamond(\mathit{cane.gioca} \wedge \mathit{pancia.vuota})$$

Esiste almeno un'esecuzione in cui il cane gioca sempre:

$$E\Box \mathit{cane.gioca}$$

Il satellite

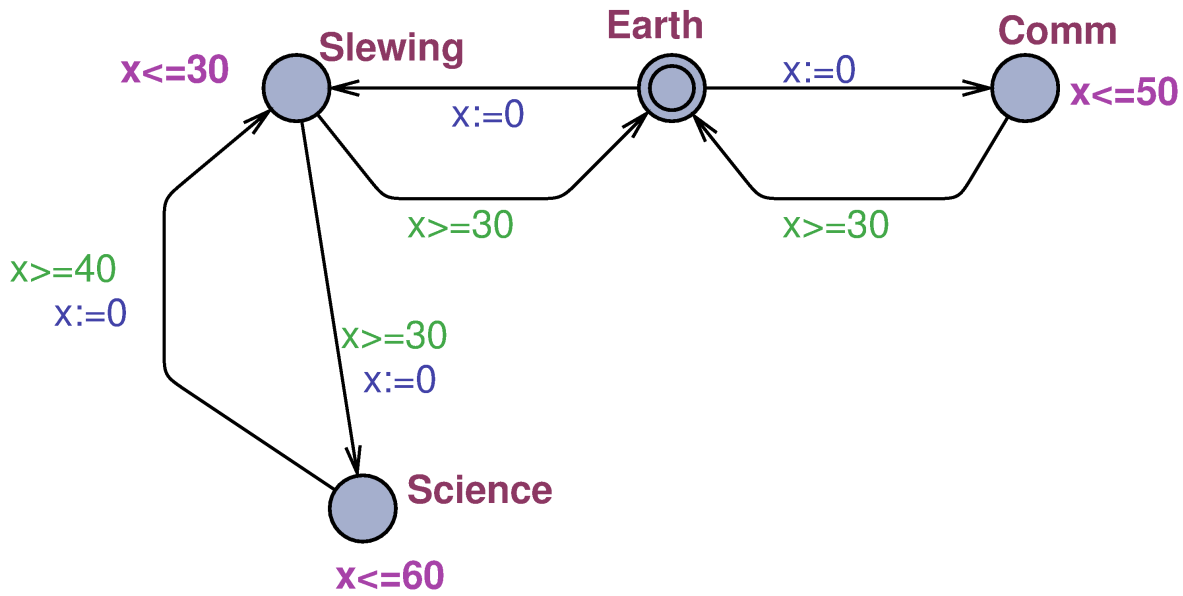
(x: variabile locale al processo)



$A\Box(satellite.Science \rightarrow satellite.x \leq 60)$

Il satellite

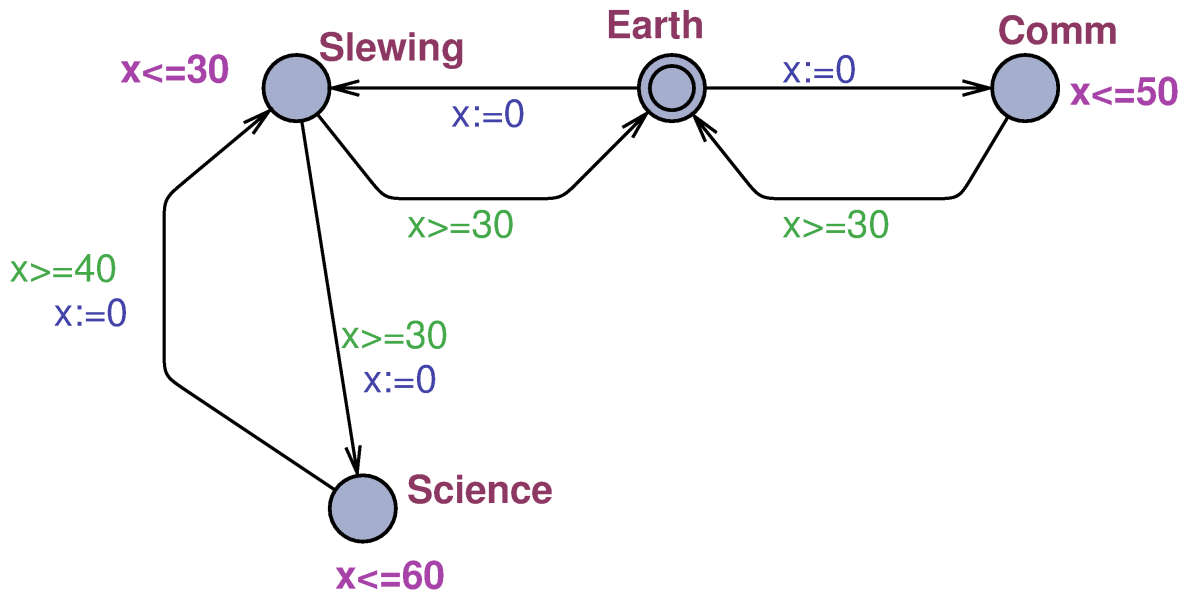
(x: variabile locale al processo)



A$\Box(satellite.Science \rightarrow satellite.x \leq 60)$ property is satisfied
E$\Diamond$ $satellite.Comm$

Il satellite

(x: variabile locale al processo)



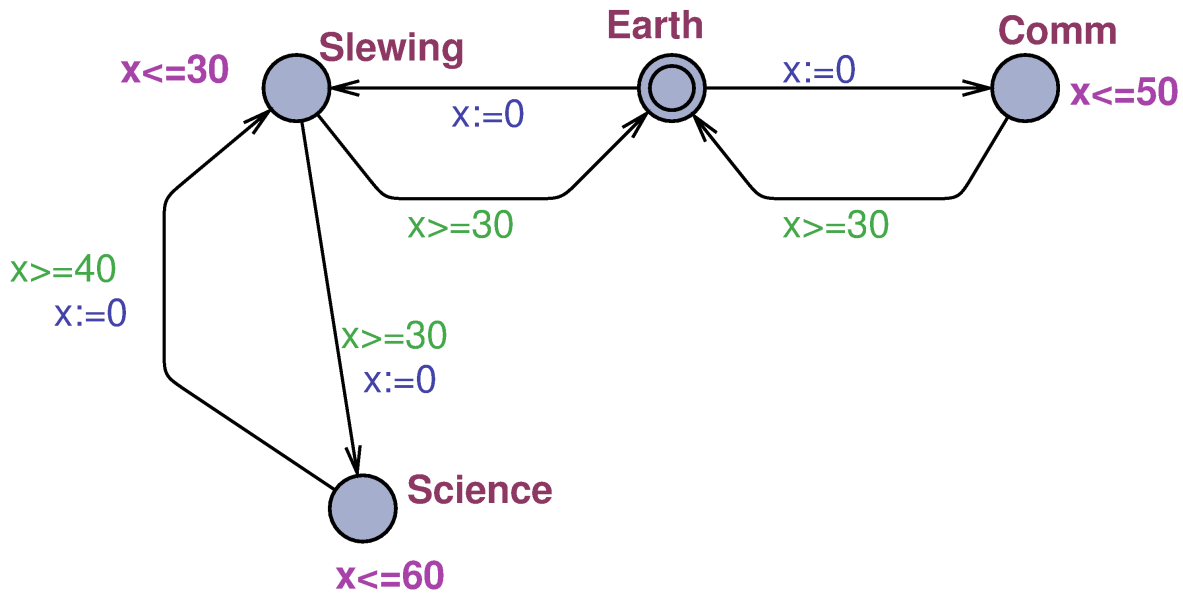
$A\Box(satellite.Science \rightarrow satellite.x \leq 60)$ property is satisfied

$E\Diamond\ satellite.Comm$ property is satisfied

$A\Diamond\ satellite.Comm$

Il satellite

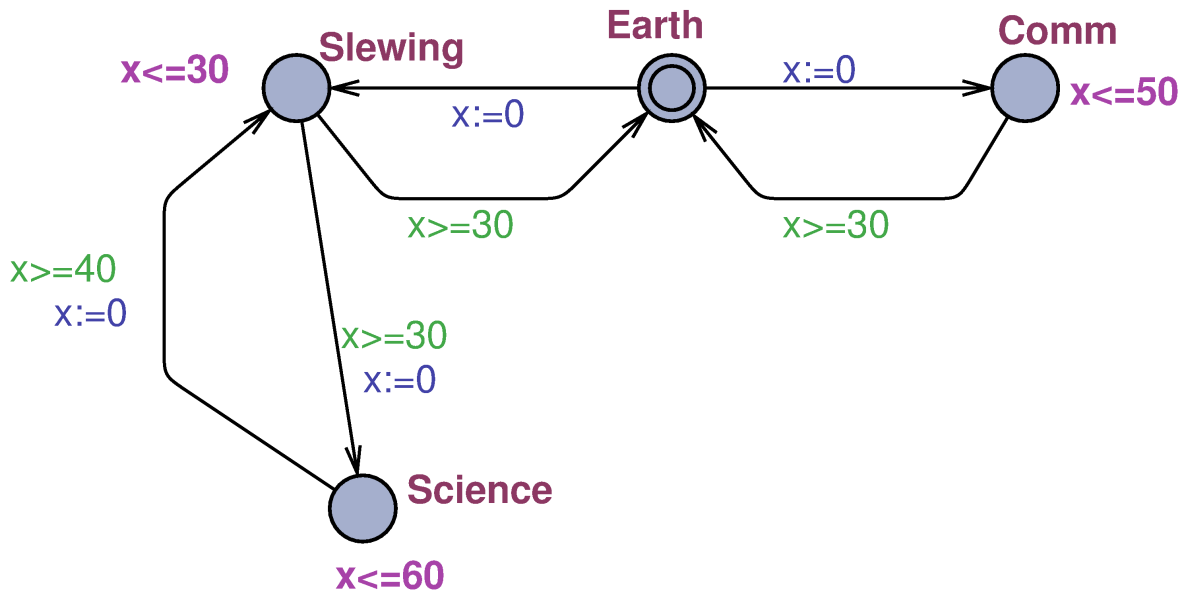
(x: variabile locale al processo)



A$\Box(satellite.Science \rightarrow satellite.x \leq 60)$	property is satisfied
E$\Diamond$ *satellite.Comm*	property is satisfied
A$\Diamond$ *satellite.Comm*	property is not satisfied
A$\Box$ *not deadlock*	

Il satellite

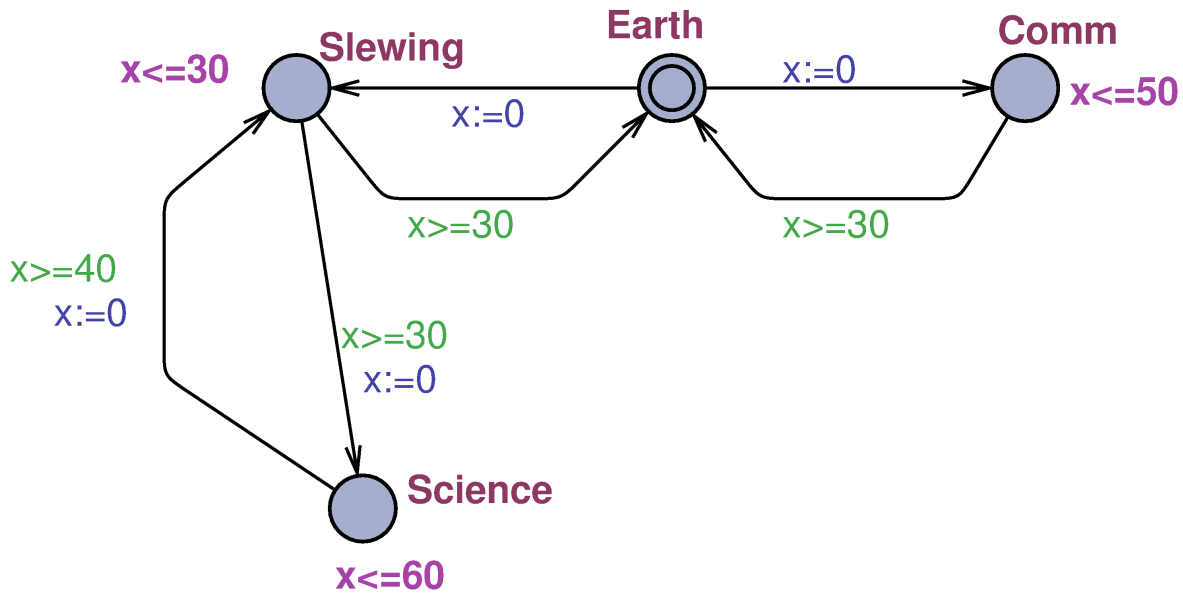
(x: variabile locale al processo)



$A\Box(satellite.Science \rightarrow satellite.x \leq 60)$	property is satisfied
$E\Diamond\ satellite.Comm$	property is satisfied
$A\Diamond\ satellite.Comm$	property is not satisfied
$A\Box\ not\ deadlock$	property is satisfied
$satellite.Science \rightsquigarrow satellite.Comm$	

Il satellite

(x: variabile locale al processo)



$A\Box(satellite.Science \rightarrow satellite.x \leq 60)$	property is satisfied
$E\Diamond\ satellite.Comm$	property is satisfied
$A\Diamond\ satellite.Comm$	property is not satisfied
$A\Box\ not\ deadlock$	property is satisfied
$satellite.Science \rightsquigarrow satellite.Comm$	property is not satisfied

Esercizi

Terminare gli esercizi proposti, considerando questa volta anche le richieste relative alla formulazione delle queries.

http://cialdea.dia.uniroma3.it/teaching/logica/slides/3-verifica/esercizi.pdf