Indice della terza lezione

LTL – Formule equivalenti

Definizione:

$$\alpha \equiv \beta \quad \text{se e solo se } \forall \pi : (\pi \models \alpha \longleftrightarrow \pi \models \beta)$$

$$\mathsf{F}\,\alpha \equiv \alpha \vee \mathsf{X}\mathsf{F}\,\alpha$$

$$\mathsf{G}\,\alpha \equiv \alpha \wedge \mathsf{X}\mathsf{G}\,\alpha$$

$$\alpha\,\mathsf{U}\,\beta \equiv \beta \vee (\alpha \wedge \mathsf{X}\,(\alpha\,\mathsf{U}\,\beta))$$

LTL – Formule equivalenti

Definizione:

$$\alpha \equiv \beta \quad \text{se e solo se } \forall \pi : (\pi \models \alpha \longleftrightarrow \pi \models \beta)$$

$F\,\alpha \equiv \alpha \vee XF\,\alpha$

$G\,\alpha \equiv \alpha \wedge XG\,\alpha$

$\alpha\,U\,\beta \equiv \beta \vee (\alpha \wedge X\,(\alpha\,U\,\beta))$

$FGF\,\alpha \equiv GF\,\alpha$

$GFG\,\alpha \equiv FG\,\alpha$

Insiemi minimali di operatori

$\mathsf{T}\,\mathcal{U}\,\alpha$

Insiemi minimali di operatori

$\mathsf{T}\,\mathsf{U}\,\alpha \equiv \mathsf{F}\,\alpha$

$\neg\mathsf{F}\neg\alpha$

Insiemi minimali di operatori

$$\mathsf{T}\,\mathsf{U}\,\alpha \equiv \mathsf{F}\,\alpha$$

$$\neg\mathsf{F}\,\neg\alpha \equiv \mathsf{G}\,\alpha$$

L'insieme $\{\mathsf{X}, \mathsf{U}\}$ forma un insieme minimale di operatori, dal quale possiamo derivare tutti gli altri: $\mathsf{F}, \mathsf{G}, \mathsf{W}, \mathsf{R}$.

Esercizio: trovare altri insiemi minimali di operatori.

La negazione in LTL

Che cosa significa “Non è vero che $\mathsf{F}\,\alpha$”?

La negazione in LTL

Che cosa significa “Non è vero che $\mathsf{F}\,\alpha$”?

Possiamo riformularla così: “Non è vero che in ogni cammino, prima o poi α diventa vera”, cioè “esiste un cammino nel quale α è sempre falsa”.

Che cosa significa $\neg\mathsf{F}\,\alpha$?

La negazione in LTL

Che cosa significa “Non è vero che $\mathsf{F}\,\alpha$”?

Possiamo riformularla così: “Non è vero che in ogni cammino, prima o poi α diventa vera”, cioè “esiste un cammino nel quale α è sempre falsa”.

Che cosa significa $\neg\mathsf{F}\,\alpha$?

“In ogni cammino non è vero che prima o poi α diventa vera”, cioè $\mathsf{G}\,\neg\alpha$

Attenzione: $\neg\mathsf{F}\,\alpha$ **non** è la negazione logica di $\mathsf{F}\,\alpha$.

Limiti espressivi di LTL

LTL non è in grado di esprimere proprietà del tipo

esiste un cammino in cui α

Limiti espressivi di LTL

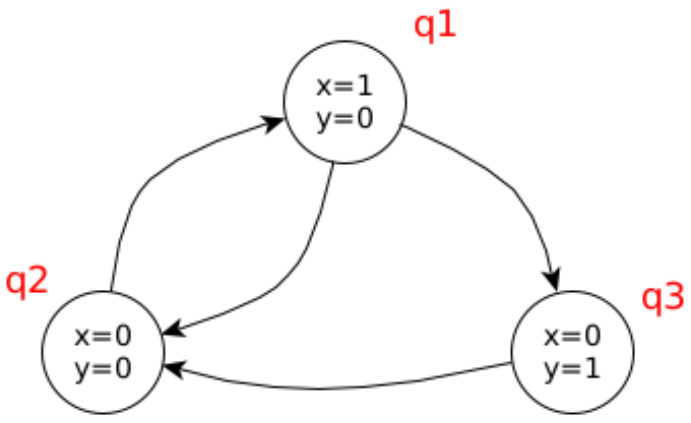
LTL non è in grado di esprimere proprietà del tipo

esiste un cammino in cui α

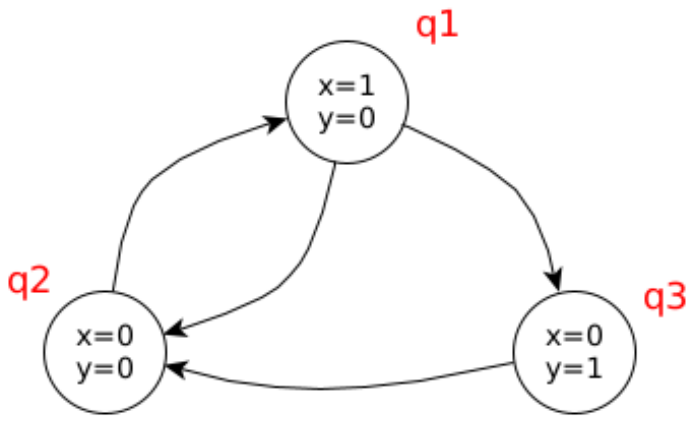
Esempi:

"è sempre possibile ritornare allo stato iniziale",
"se il processo P chiede la risorsa, è possibile raggiungere uno stato di *deadlock*"

Esercizio



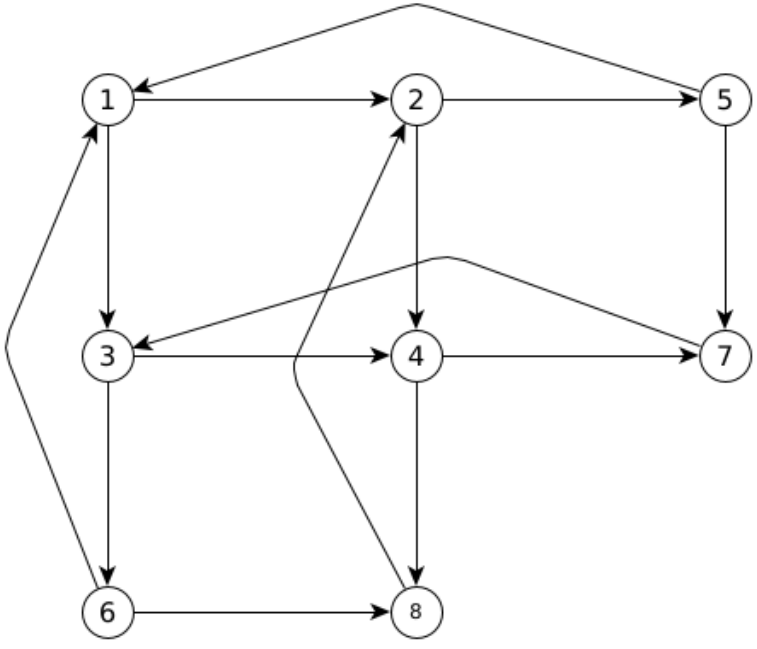
Esercizio



Stabilire in quali stati le seguenti formule sono vere

(1) $G(x = 0 \vee y = 0)$
(2) $GF(y = 0)$
(3) $GF(y = 1)$
(4) $G(x = 1 \longrightarrow F(y = 1))$

Il problema della mutua esclusione



1: rd, snc1, snc2
2: rd, req1, snc2
3: rd, snc1, req2
4: rd, req1, req2
5: sc1, snc2
6: snc1, sc2
7: sc1, req2
8: req1, sc2

Il problema della mutua esclusione

Requisiti

(1) I due processi non sono mai contemporaneamente nella sezione critica
(2) Se un processo richiede la risorsa, prima o poi entrerà nella sezione critica
(3) Se un solo processo richiede la risorsa, deve poter accedere alla sezione critica

Il problema della mutua esclusione

Requisiti

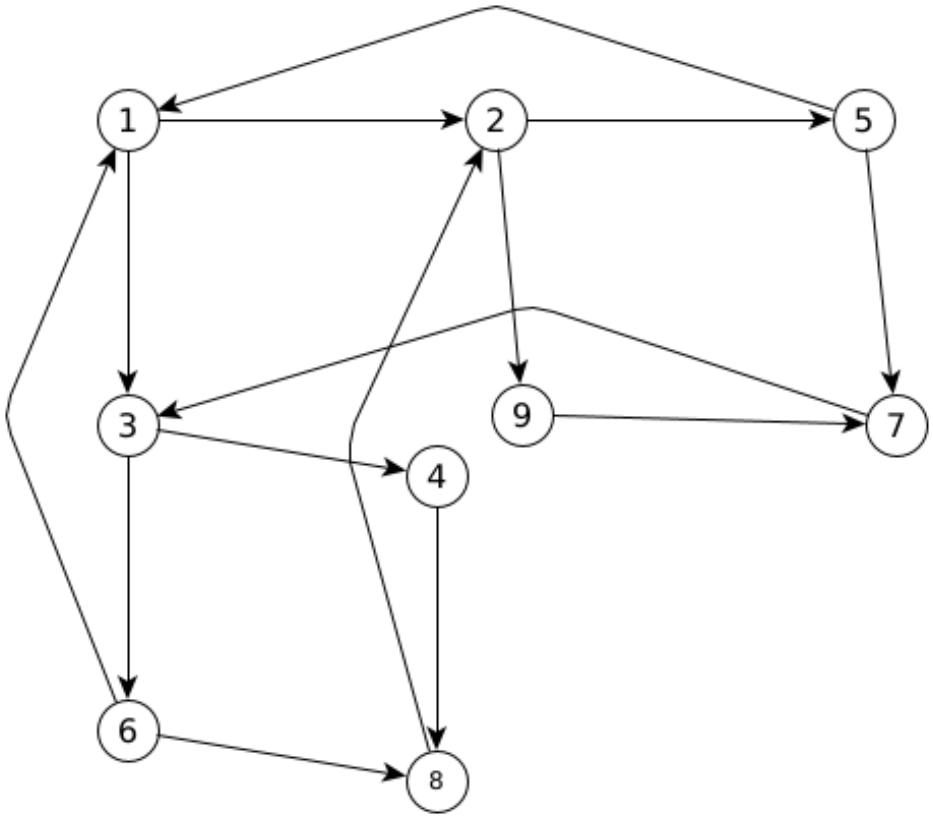
(1) I due processi non sono mai contemporaneamente nella sezione critica
(2) Se un processo richiede la risorsa, prima o poi entrerà nella sezione critica
(3) Se un solo processo richiede la risorsa, deve poter accedere alla sezione critica

(1): $\mathsf{G}\,\neg(\mathrm{sc1} \wedge \mathrm{sc2})$

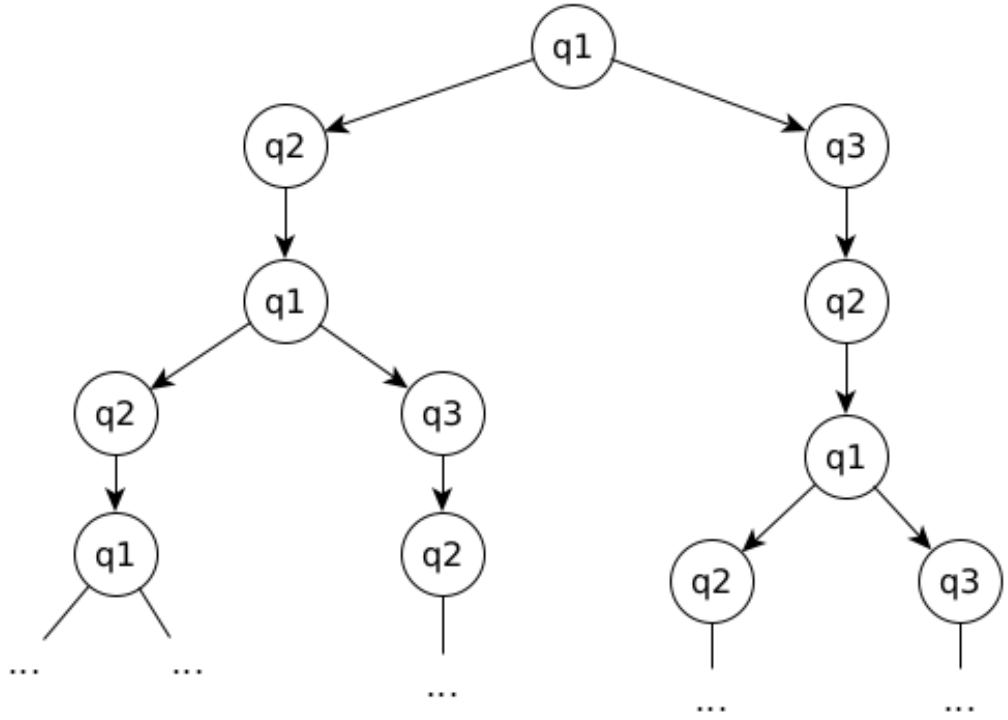
(2): $\mathsf{G}\,(\mathrm{req1} \longrightarrow \mathsf{F}\,\mathrm{sc1})$

(3): ??

Il problema della mutua esclusione



Alberi di computazione



Computazione: cammino nell'albero a partire dalla radice

Computation Tree Logic – sintassi

Proposizioni atomiche: $AP = \{p_1, p_2, \cdots, q, r, \cdots\}$

Definiamo FBF_{CTL}

Computation Tree Logic – sintassi

Proposizioni atomiche: $AP = \{p_1, p_2, \cdots, q, r, \cdots\}$

Definiamo FBF_{CTL}

per ogni $p \in AP$, $p \in FBF_{CTL}$

Per ogni $\alpha, \beta \in FBF_{CTL}$

1. $\neg\alpha$, $\alpha \vee \beta \in FBF_{CTL}$

Computation Tree Logic – sintassi

Proposizioni atomiche: $AP = \{p_1, p_2, \cdots, q, r, \cdots\}$

Definiamo FBF_{CTL}

per ogni $p \in AP$, $p \in FBF_{CTL}$

Per ogni $\alpha, \beta \in FBF_{CTL}$

1. $\neg\alpha$, $\alpha \vee \beta \in FBF_{CTL}$
2. $AX\,\alpha, EX\,\alpha \in FBF_{CTL}$
3. $AF\,\alpha, EF\,\alpha \in FBF_{CTL}$
4. $AG\,\alpha, EG\,\alpha \in FBF_{CTL}$
5. $A(\alpha\,U\,\beta) \in FBF_{CTL}, E(\alpha\,U\,\beta) \in FBF_{CTL}$

Computation Tree Logic – sintassi

A e E sono quantificatori sui cammini:

A: per ogni cammino

E: esiste un cammino tale che

Computation Tree Logic – sintassi

A e E sono quantificatori sui cammini:

A: per ogni cammino

E: esiste un cammino tale che

“Dopo l’accensione della spia, sarà sempre possibile riportare il sistema allo stato iniziale”

Computation Tree Logic – sintassi

A e E sono quantificatori sui cammini:

A: per ogni cammino

E: esiste un cammino tale che

"Dopo l'accensione della spia, sarà sempre possibile riportare il sistema allo stato iniziale"

s: la spia è accesa
init: il sistema si trova nello stato iniziale

$$\mathrm{AG}\,(s \longrightarrow \mathrm{AX}\,\mathrm{EF}\,\mathrm{init})$$