Lezione 19 14/12/2023

Esempi di formule

$FG\,\alpha$	α è invariante da un certo istante in poi	α è invariante da un certo istante in poi. Questa formula ci dice che la proprietà espressa dalla formula α, ad un certo punto dell'esecuzione, diventa invariante. (da un certo punto in poi alpha è vera)
$GF\,\alpha$	α è vera in un numero infinito di stati	Questa è soddisfatta se per esempio α è vera una volta si e una no, infinitamente. Non deve essere vera in tutti gli stati, ci devono solo essere stati infiniti in cui è vera.
$G\,\neg(cs_1 \wedge cs_2)$	Mutua esclusione	cs1 interpretato come "il processo 1 è nella sezione critica", idem per cs2. Qui per esempio cs1 e cs2 usano la stessa risorsa in modo condiviso. Questa formula dice che è sempre vera (ogni stato, ogni cammino massimale che parte dallo stato iniziale) la formula.
$G\,(req \longrightarrow XF\,ack)$		req dice che è stata presentata una richiesta. ack dice che il server ha ricevuto la richiesta e ha confermato la ricezione al client. La formula dice che se c'è una richiesta, allora prima o poi questa sarà riconosciuta, ma questo prima o poi è scalato di un istante nel tempo (con la X). Se non ci fosse la X, la formula sarebbe soddisfatta in un modello dove in uno stato compaiono vere simultaneamente req e ack. L'implicazione va letta come operatore di implicazione. L'implicazione è vera se l'antecedente è falso, indipendentemente dal conseguente. L'implicazione è banalmente vera in uno stato dove non c'è una richiesta pendente.

$G(req \longrightarrow (req \, U \, ack))$

è sempre vero che se c'è una richiesta pendente allora vale la formula che dice che la formula è pendente finché non c'è la risposta.

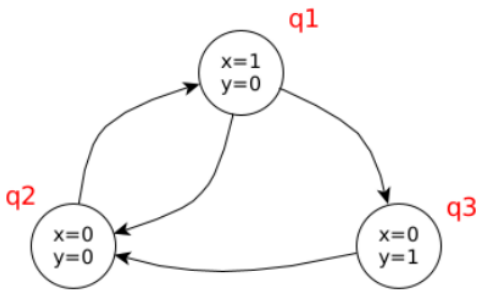
Nell'operatore until c'è nascosto l'operatore F, ovvero la richiesta che prima o poi sarà vero ack.

La prima formula sarebbe soddisfatta in un sistema in cui in un cammino compare una richiesta pendente, nello stato successivo questa scompare ma non c'è ancora l'ack che arriva dopo un po'. Questo non corrisponderebbe ad un comportamento desiderato. Nella seconda invece questa rimane vera finché non arriva la risposta (ack?).

$G(req \longrightarrow ((req \wedge \neg ack) \, U \, (ack \wedge \neg req)))$

Non solo se c'è una richiesta pendente prima o poi ci deve essere la risposta del server, ma la risposta deve anche avere come conseguenza che si cancella la richiesta pendente.

Esercizio



In questo modello di Keipke ci sono 3 stati. Possiamo interpretare la transizione da q2 a q1 come l'assegnamento della variabile x=1.

Non ci sono tutte le transizioni, da q3 non si può passare a q1, questo passaggio rappresenterebbe due assegnamenti contemporanei.

Tipico esercizio da esame:

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

Identificare i cammini massimali, e poi valutare le formule date. Questo viene fatto per ogni stato.

Dobbiamo stabilire se la formula 1 è valida in q1, in q2 e in q3.

La prima formula dice che è sempre vero che x=0 oppure y=0. Esprime una proprietà invariante del sistema. Per decidere se è vera o no, il primo passo che ci conviene fare è quello di osservare gli stati del sistema cercando stati che la violano. In questo caso questo è vero, quindi è inutile considerare i cammini massimali.

Questo sistema è fortemente reversibile perché da qualunque caso posso poi raggiungere qualunque altro stato.

Nel caso di una formula Gα questa indagine preliminare può essere utile solo quando ci sono alcuni stati non raggiungibili da altri stati. (per esempio se non posso tornare allo stato iniziale)

Seconda formula: è sempre vero che prima o poi y=0. Qui dobbiamo studiare i cammini massimali. Osserviamo quali sono gli stati in cui y=0, ovvero q2 e q1. Studiamo cammini massimali che non passano mai da questi stati. Possiamo osservare che non ci sono (nel grafo) dei cicli che non passano mai per q1 o per q2. Se ci fosse un cappio su q3 allora in q3 non sarebbe valida questa formula, perché potremmo fare un cammino massimale su q3. Di conseguenza nel nostro caso questa formula è vera per tutti e 3 gli stati.

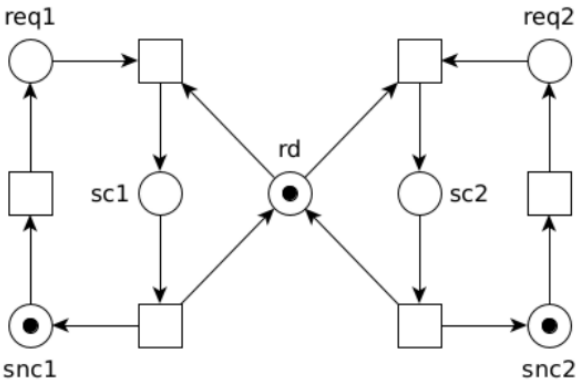
Terza formula: è sempre vero che prima o poi y=1. In quali stati è verificata questa formula? Dobbiamo considerare tutti i cammini massimali che partono da q3, e vedere se è vero che prima o poi raggiungerò uno stato in cui y=1. Questo non è vero perché c'è un cammino critico che parte da q3, andando a q2 e poi percorrendo all'infinito il ciclo q1-q2, qui y non avrà mai più il valore 1 e quindi è falsa. Quindi in q3 non vale. A maggior ragione non vale neanche in q1 e q2. GF ci richiede di considerare tutti i possibili cammini massimali.

Quarta formula: è sempre vero che se x=1 allora prima o poi y diventerà 1. Questo non è vero perché per esempio in q1 non è verificata, potrei rimanere nel ciclo q1-q2. Qui possiamo prima studiare l'argomento della formula, cioè x=1 implica f(y=1). In q1 non è vera perché c'è il cammino ciclico q1-q2. In q2 dovresti passare da q1 ma poi la formula completa non sarà valida. In q3 è valida perché sto considerando la formula senza la G e quindi nel caso iniziale x=1 non è vera, e quindi la formula sarà vera. Rimane però che non è vera l'intera formula.

La formula interna è vera in q2 e q3 perchè è sbagliato l'antecedente. è falsa in q1 perchè rimane nel ciclo.
La formula completa è falsa per ogni stato perchè a partire da ognuno posso raggiungere q1 e quindi la formula interna non sarà sempre vera.

Esercizio 2

Partendo da una rete di Petri:



rd: risorsa disponibile

snc: sezione non critica

sc: sezione critica

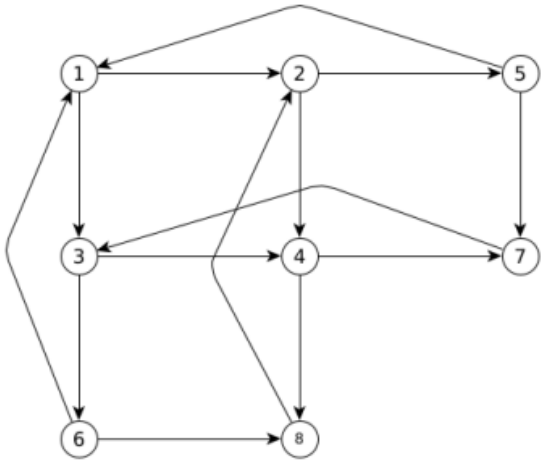
req: richiesta pendente

Ogni singola condizione di una rete elementare può essere vista come una proposizione che è vera quando la proposizione è marcata e falsa quando non è marcata.

Questo è un sistema formato da 2 processi ciclici (rettangolo sx e dx). Questi due processi ciclicamente cercano di impossessarsi di una risorsa condivisa che non può essere usata in contemporanea.

snc1 è la situazione non critica perchè ha già la risorsa. req1 richiede la risorsa.

Dobbiamo scrivere una serie di formule che descrivono la nostra idea di correttezza del sistema.



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

Questo è il grafo dei casi raggiungibili della rete: è il primo passo da fare.

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): $G\,\neg(sc1 \wedge sc2)$

(2): $G\,(req1 \longrightarrow F\,sc1)$

(3): ??

(1) è sempre vero che non valgono contemporaneamente sc1 e sc2, quindi due processi non sono mai contemporaneamente nella zona critica.

(2) Se ha fatto la richiesta di entrare nella sezione critica, prima o poi entrerà nella sezione critica. Scritta per 1, stessa cosa anche per la seconda risorsa

lasciamo in sospeso la (3)

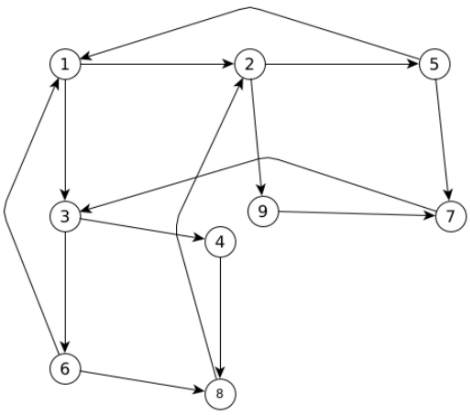
Dobbiamo verificare le formule in ciascuno stato.

Possiamo osservare la tabella dei 8 casi raggiungibili, e vedere che tra quelli non ci sono stati in cui valgono simultaneamente sc1 e sc2, quindi la formula è valida.

La seconda formula è espressa come invariante (è sempre vero che ...), per dimostrare che non è vera in un certo stato è sufficiente trovare un cammino massimale che parte da quello stato in cui la formula non è verificata. Guardiamo il cammino che parte da 1 (qui è vera perchè l'antecedente dell'implicazione è vera), va in 2 e fa richiesta e qui l'antecedente è vero, ma da qui noi potremmo andare in 4 (secondo processo fa richiesta della risorsa) e poi in 8 (l'arbitro assegna la risorsa al secondo processo). Da 8 dobbiamo passare a 2 (il processo 2 ha liberato la risorsa ed è tornato sulla sezione non critica, mentre il proccesso 1 è ancora in attesa). Questa sezione 4-2-8 può essere ripetuta all'infinito, e quindi in questo caso il processo 1 rimarrà in attesa all'infinito e la seconda formula non sarà verificata per questo cammino.

Il problema della mutua esclusione

Questa è una proposta di correzione, per cercare di soddisfare la seconda richiesta.



Qui lo stato centrale è stato sdoppiato.

1→2 richiesta process 1

2→9 richiesta processo 2

L'arbitro privilegia chi ha fatto la richiesta per primo. Quando il processo 1 libera la risorsa, dallo stato 7 si passa allo stato 3 (è ancora pendente la richiesta del processo 2, la risorsa è libera, il processo 1 è nella sezione non critica). Da 3 può capitare che l'arbitro conceda la risorsa al processo 2, ma è anche possibile che il processo 1 ripresenti la sua richiesta. In questo secondo caso, non torniamo allo stato 9, ma in uno stato dove entrambe le richieste sono pendenti, la risorsa è disponibile, ma l'arbitro da precedenza al processo 2.

LTL - formule equivalenti

Definizione:

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

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

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

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

Due formule LTL sono equivalenti se per ogni cammino, alpha è vera in quel cammino sse beta è vero nello stesso cammino.

Prima formula (prima o poi alpha), la formula a destra dice che alpha o XFalpha. Sostengo che siano equivalenti. Questa dice che a partire dallo stato iniziale che stiamo considerando, prima o poi è vera alpha. La seconda dice che alpha è vero subito, oppure mi sposto nello stato successivo, e a partire da questo stato è vera Falpha. Queste due formule dicono la stessa cosa.

Terza formula: alpha until beta significa che adesso è vera beta, oppure adesso è vera alpha e se mi sposto nello stato successivo è vera alpha until beta.

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

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

Prima o poi sarà vero che è sempre vero che prima o poi alpha. (prima o poi raggiungo uno stato dove è sempre vero che prima o poi alpha). Sostengo che questo primo F è inutile.

Simmetricamente anche la seconda.

Operatori derivati

Until debole (*weak until*)

$$\alpha\,W\,\beta \equiv G\,\alpha \vee (\alpha\,U\,\beta)$$

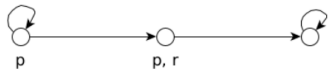
L'until normale significa: prima o poi beta e in tutti gli istanti precedenti, alpha.

Nell'until debole vogliamo che o valga sempre alpha, oppure alpha until beta.

Release — $\alpha\,R\,\beta$

$$\pi \models \alpha\,R\,\beta \text{ sse } \forall k \geqslant 0 : (\pi^{(k)} \models \beta \vee \exists h < k : \pi^{(h)} \models \alpha$$

$$\alpha\,R\,\beta \equiv \beta\,W\,(\alpha \wedge \beta)$$



In questo modello è verificata la formula r R p

alpha release beta viene interpretata come alpha svincola(libera) beta. C'è un segmento iniziale del cammino dove vale beta, e dall'istante successivo in quello dove diventa vera alpha, allora anche beta può diventare falsa.

Alpha release beta è verificata in un cammino pigreco se è vera questa disgiunzione: per ogni istante k nel cammino pigreco o beta è sempre vera nel cammino, oppure deve esistere un'istante h che precede strettamente k, tale che il suffisso h di k ???

O beta è sempre vera, oppure c'è un indice k dove beta è falsa, e quindi c'è un istante prima di k dove alpha è diventata vera.

Se beta non è vera per l'intero cammino, allora ci deve essere uno stato dove beta è ancora vera, ma è vera anche alpha. ?

Nello stato iniziale del modello è verificata la formula. Verifichiamolo. Per ogni cammino massimale, dobbiamo stabilire se la formula è valida. Nel primo cammino

(stato iniziale, va a destra, e poi rimane nell'ultimo stato all'infinito) è verificata. Nel cammino dove rimane all'infinito nel primo stato è verificata, perché rientriamo nel primo caso. Infine dobbiamo considerare i cammini che girano per un po' nello stato iniziale e poi vanno avanti.

Formule particolari

$$T \, U \, \alpha$$

Prima o poi alpha, e in tutti gli stati precedenti deve valere true (che è vera in tutti gli stati). Questa formula equivale a Falpha.

$$T \, U \, \alpha \equiv F \, \alpha$$

$$\neg F \neg \alpha \equiv G \, \alpha$$

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

Esercizio: trovare altri insiemi minimali di operatori.

La negazione in LTL

Che cosa significa "Non è vero che $F \, \alpha$"?

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 $\neg F \, \alpha$?

"In ogni cammino non è vero che prima o poi α diventa vera", cioè $G \neg \alpha$

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

Concludiamo che non F alpha non è la negazione logica di F alpha. Succede questo perchè nelle LTL dobbiamo sempre pensare che ci sia davanti un quantificatore universale implicito, non si può avere un quantificatore esistenziale.

Limiti espressivi di LTL

LTL non è in grado di esprimere proprietà del tipo

esiste un cammino in cui α