Automazione

20 Gennaio 2016

Esercizio 1

Un motore elettrico aziona attraverso un organo di trasmissione e riduzione del moto un carico costituito da un braccio robotico che ruota in un piano orizzontale. Sia θ la posizione angolare del braccio e $J = 10$ kgm^2 la sua inerzia intorno all'asse di rotazione. Un encoder incrementale misura la posizione angolare θ_m del motore e fornisce 2400 impulsi per giro. Un circuito digitale moltiplica per 4 il conteggio totale sul giro. L'organo di trasmissione tra motore e carico ha un rapporto di riduzione $N_r = (\dot{\theta}_m/\dot{\theta}) \gg 1$. Il motore produce una coppia massima pari a $\tau_m = 0.32$ Nm sul proprio asse di uscita (quindi una coppia $\tau = N_r\tau_m$ a valle del riduttore) e deve essere in grado di far partire da fermo il braccio con un'accelerazione non inferiore a $\ddot{\theta} = 1.6$ rad/s^2. Trascurando gli effetti dissipativi, si scelga un opportuno valore del rapporto di riduzione dell'organo di trasmissione e si determini di conseguenza la minima risoluzione angolare $\Delta\theta$ dal lato del carico.

Esercizio 2

Si consideri un sistema di automazione industriale di un mobilificio in cui, a livello di coordinamento, è necessario gestire i seguenti task periodici:

- ogni 4 t.u. viene effettuata la foratura, impiegando 1 t.u.
- ogni 6 t.u. viene eseguito l'assemblaggio in 2 t.u.
- ogni 9 t.u. viene eseguita la verniciatura in 3 t.u.

C'è n addizionale task aperiodico di controllo manuale della qualità che parte dopo che sono trascorse 2 t.u., ha una deadline relativa di 30 t.u. e un computation time di 3 t.u.. Si ipotizzi che i task periodici siano indipendenti dal punto di vista funzionale l'uno dall'altro. I task periodici devono essere gestiti con una modalità di scheduling hard real time.

1. Verificare se sussiste la condizione necessaria di schedulabilità dei task periodici.
2. Verificare se sussiste almeno una delle condizioni sufficienti di schedulabilità, utilizzando l'algoritmo RMPO.
3. Eseguire lo scheduling RMPO. Nel caso in cui RMPO non sia in grado di schedulare i task in maniera hard real time, eseguire lo scheduling utilizzando l'algoritmo EDF.
4. Utilizzando l'algoritmo di scheduling scelto nel punto precedente, verificare se il task aperiodico riesce ad essere eseguito in maniera hard real time utilizzando uno scheduling in background. Nel caso ciòn on sia possibile, verificare se il task aperiodico può essere eseguito in maniera hard real time utilizzando un processo polling server cosìcaratterizzato: $T_{\text{SRV}} = 12$ t.u. e $C_{\text{SRV}} = 1$ t.u.

Esercizio 3

In una rete industriale di comunicazione, il segnale al livello fisico sul canale di trasmissione utilizza una codifica binaria diretta di tipo polare: il livello H corrisponde al bit 1, il livello L al bit 0, e il livello Z ad assenza di segnale utile. In ricezione, un dispositivo esegue in linea il conteggio di parità del segnale, fornendo in uscita un valore P se il numero di 1 ricevuti fino al campione precedente è pari e un valore D altrimenti. Descrivere il comportamento di questo dispositivo mediante un automa ingresso-stato-uscita. Lo stato iniziale è l'assenza di segnale.

Esercizio 4

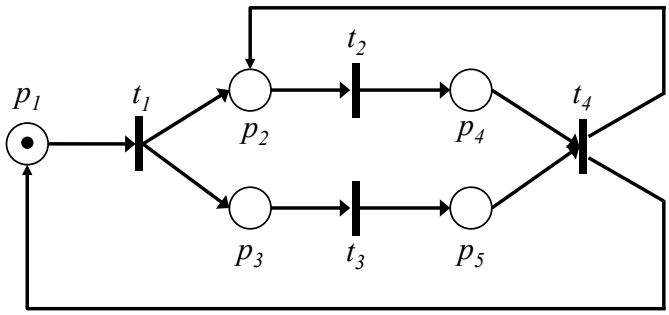

Figura 1: La rete di Petri PN con la marcatura iniziale $\boldsymbol{x}_0$

La rete di Petri PN mostrata in Fig. 1 con la sua marcatura iniziale $\boldsymbol{x}_0$ è una rete viva e illimitata.

- Mediante lo studio dei P-invarianti della rete, determinare l'esistenza di cicli conservativi e indicare quali posti sono illimitati.
- Costruire l'albero di copertura della rete e descrivere in modo compatto l'insieme di tutte le marcature raggiungibili $\mathcal{R}(PN)$.
- Progettare un supervisore basato sull'introduzione di posti di controllo (*monitor*), definiti possibilmente mediante invarianti, in modo tale da limitare ad un valore intero $k \geq 1$ (arbitrario e finito) il numero totale di token nella rete, evitando al contempo situazioni di deadlock.
- Determinare l'insieme di tutte le marcature raggiungibili della rete di Petri supervisionata cosìottenuta.

[210 minuti per Automazione, 9 cfu; libri aperti]

Soluzioni

20 Gennaio 2016

Esercizio 1

Si tratta di operare le conversioni necessarie. La risoluzione dell'encoder incrementale dal lato del motore considerando la quadratura è pari a $\Delta\theta_m = 360°/(2400 \times 4) = 0.0375° \approx 6.545 \cdot 10^{-4}$ rad. In assenza di attrito e di altri fenomeni di dispersione/dissipazione, il bilanciamento dinamico del carico inerziale è dato da $\tau = J\ddot{\theta}$ (la gravità è assente perchè il moto è su un piano orizzontale). La coppia richiesta dal lato del carico è dunque $\tau = 10 \cdot 1.6 = 16$ [kgm$^2 \cdot$rad/s^2] $= 16$ [Nm]. Data la coppia massima erogabile dal motore sul suo asse di uscita, per realizzare tale coppia sul lato del carico occorre scegliere un rapporto di riduzione pari (almeno) a $N_r = \tau/\tau_m = 16/0.32 = 50$. Pertanto la risoluzione angolare dal lato del carico sarà $\Delta\theta = \Delta\theta_m/N_r = 0.00075° \approx 1.3 \cdot 10^{-5}$ rad.

Esercizio 2

Per verificare la condizione necessaria si calcola il fattore di utilizzazione dei task periodici hard real time:

$$U = \frac{1}{4} + \frac{2}{6} + \frac{3}{9} = \frac{33}{36} \simeq 0.92 < 1.$$

Verificata la condizione necessaria, controlliamo se esiste almeno una condizione sufficiente:

$$U_{lsm}(RMPO) = n\left(2^{1/n} - 1\right) = 3\left(2^{1/3} - 1\right) \simeq 0.78.$$

Dato che $U > U_{lsm}$, questa condizione sufficiente non è verificata. Inoltre i tre task non sono legati tra loro da relazioni armoniche. Non possiamo dire a priori se RMPO è in grado di schedulare i task. La soluzione dello scheduling RMPO è mostrata in Fig. 2, da cui segue che RMPO non è in grado di schedulare in maniera hard real time i task periodici come richiesto.

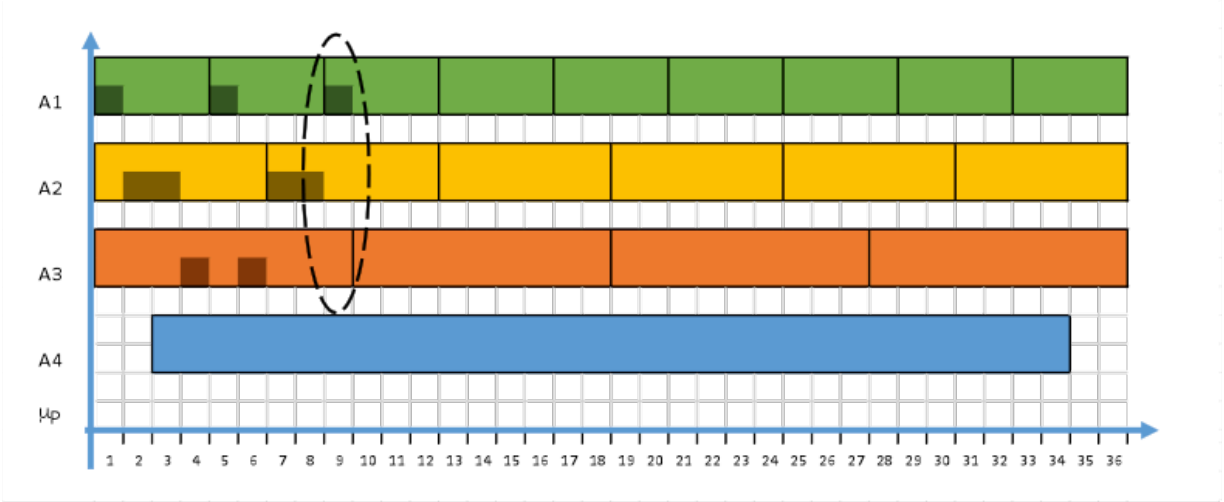

Figura 2: Scheduling RMPO

La soluzione dello scheduling EDF è riportata in Fig. 3. Il task aperiodico ($A4$) non riesce ad essere eseguito in tempo se si utilizza una politica in background. Aggiungendo il processo server

(A_{SRV}), il coefficiente di utilizzazione diventa:

$$U = \frac{1}{4} + \frac{2}{6} + \frac{3}{9} + \frac{1}{12} = \frac{36}{36} = 1.$$

Verificata la condizione necessaria e sufficiente per EDF, la soluzione dello scheduling è riportata in Fig. 4. Da questa si evince che le 3 t.u. del task aperiodico non possono essere eseguite in questo modo entro la deadline assoluta.

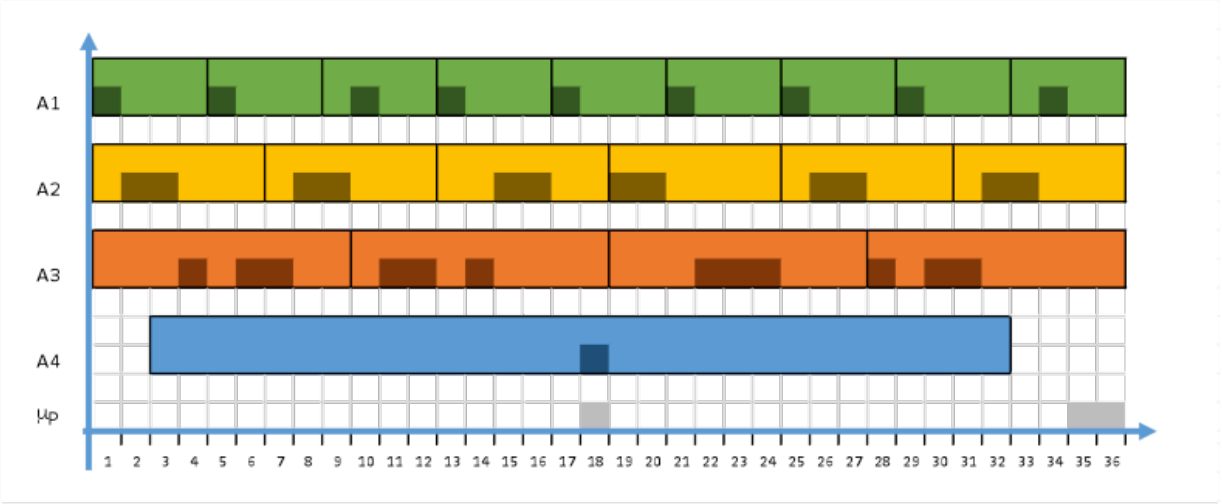

Figura 3: Scheduling EDF con politica in background

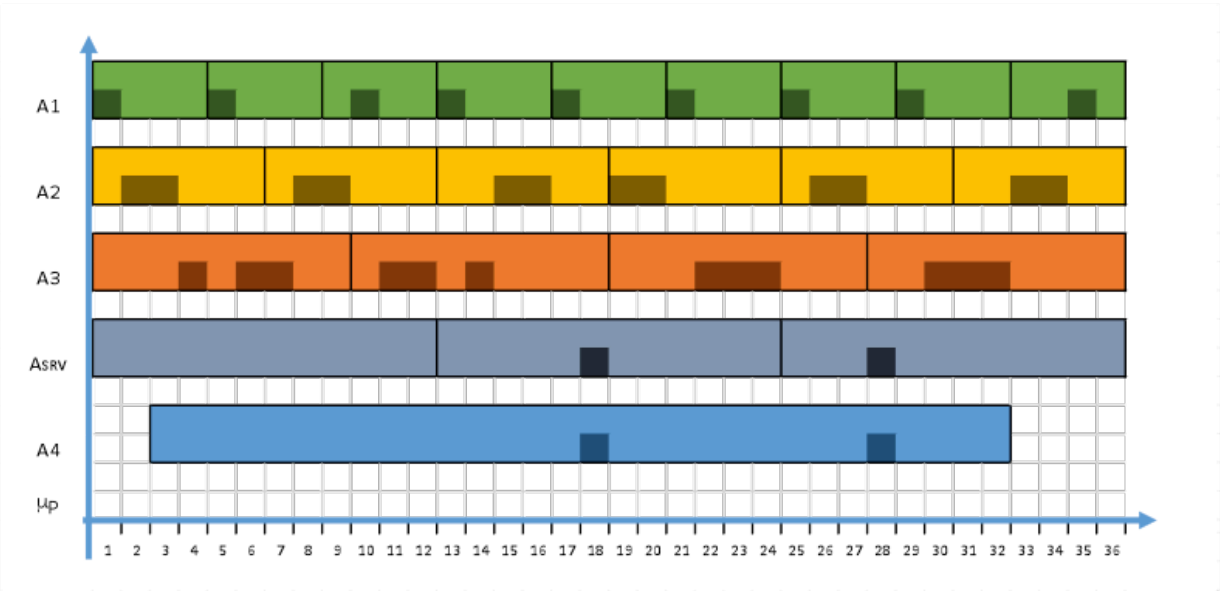

Figura 4: Scheduling EDF con politica Polling Server

Se invece di utilizzare una politica Polling Server si fosse invece scelta una politica Deferring Server, si sarebbe ottenuto lo scheduling in Fig. 5. Da quest'ultima si può concludere che le 3 t.u. del task aperiodico possono in effetti essere eseguite entro la deadline assoluta.

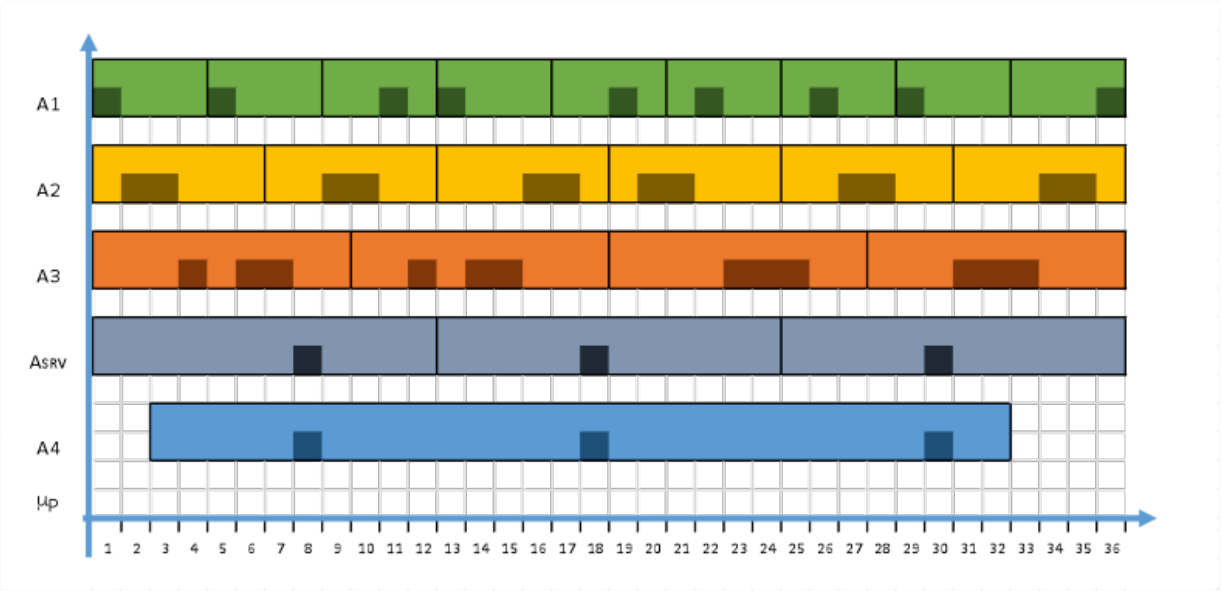

Figura 5: Scheduling EDF con politica Deferring Server

Esercizio 3

La Fig. 6 mostra un automa a tre stati che realizza il comportamento desiderato per il conteggio di parità di un segnale codificato in binario con polarità. Lo stato iniziale x_0 di assenza di segnale è evidenziato da una freccia in ingresso. Poichè l'uscita dipende solo dallo stato (D in x_1, P in x_2 e nulla in x_0), l'automa è convenientemente rappresentato come macchina di Moore. Si noti che l'arrivo di un livello Z successivo all'arrivo di cifre del segnale utile (combinazioni di '1' e '0') non dovrebbe, come scelta di progetto, resettare il conteggio di parità ossia riportarlo nello stato iniziale x_0: il segnale utile potrebbe infatti non essere ancora completo (pausa di trasmissione).

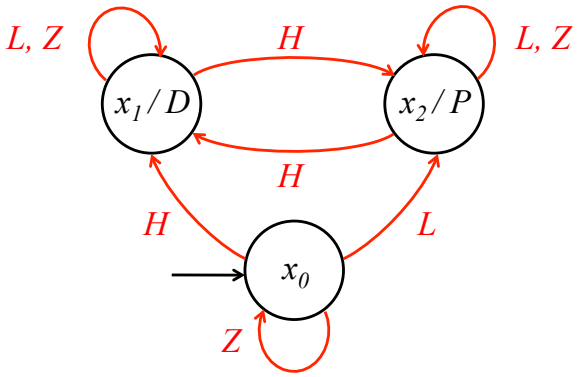

Figura 6: Un automa a stati finiti per il conteggio di parità

Esercizio 4

La matrice di incidenza $\boldsymbol{C}$, di dimensioni (5×4), della rete di Petri di Fig. 1 è

$$\boldsymbol{C} = \begin{pmatrix} -1 & 0 & 0 & 1 \\ 1 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

E' immediato verificare (ad esempio con la funzione `rank` in Matlab, ma non solo!) che la matrice $\boldsymbol{C}$ ha rango massimo, pari a 4. La dimensione dello spazio nullo di $\boldsymbol{C}^T$ è pari a $5-4=1$. L'equazione matriciale che definisce i P-invarianti,

$$\boldsymbol{\gamma}^T \boldsymbol{C} = \boldsymbol{0}^T \quad \Rightarrow \quad \boldsymbol{C}^T \boldsymbol{\gamma} = \boldsymbol{0},$$

si traduce nelle quattro equazioni scalari

$$\begin{aligned} -\gamma_1 + \gamma_2 + \gamma_3 &= 0 \\ -\gamma_2 + \gamma_4 &= 0 \\ -\gamma_3 + \gamma_5 &= 0 \\ \gamma_1 + \gamma_2 - \gamma_4 - \gamma_5 &= 0, \end{aligned}$$

che ammettono dunque ∞^1 soluzioni. E' semplice determinare la struttura di tutti i P-invarianti, parametrizzati dallo scalare $\alpha \in \mathbb{Z}^+$:

$$\boldsymbol{\gamma}^T = \begin{pmatrix} \alpha & 0 & \alpha & 0 & \alpha \end{pmatrix}.$$

Per $\alpha = 1$, il P-invariante $\boldsymbol{\gamma}^T = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \end{pmatrix}$ è un vettore di supporto minimo e canonico. Esso indica la conservazione del numero di token iniziali (pari a $\boldsymbol{\gamma}^T \boldsymbol{x}_0 = 1$) nel ciclo $\{p_1, p_3, p_5\}$ durante una qualsiasi evoluzione ammissibile della rete. Viceversa, i posti p_2 e p_4 non compaiono in nessun P-invariante e sono quelli in cui i token possono diventare illimitati.

Verifichiamo la non limitatezza (e la vivezza) della rete di Fig. 1 andando a costruire l'albero delle marcature raggiungibili. Notando la presenza di marcature crescenti che dominano alcune delle precedenti, viene ricavato di fatto l'albero delle coperture come mostrato in Fig. 7. Tale albero è costruito in modalità depth-first (in profondità), espandendo le transizioni abilitate in una data marcatura in ordine lessicografico (t_i prima di t_j, se $i < j$). Dato l'algoritmo di visita, le considerazioni riportate in rosso nella figura sono fatte solo dopo aver raggiunto le varie foglie dell'albero. La presenza della notazione ω rende dunque la rete *illimitata*. L'intera rete è inoltre *viva* in quanto ogni transizione può essere abilitata allo scatto a partire da tutte le marcature raggiungibili, dopo opportune sequenze ammissibili.

Dall'analisi dell'albero, eliminando le marcature che vengono ricoperte da quelle con la notazione ω, è possibile descrivere l'insieme delle marcature raggiungibili a partire da $\boldsymbol{x}_0$ in modo compatto come

$$\mathcal{R}(PN) = \left\{ \boldsymbol{x}_A,\ \boldsymbol{x}_B,\ \boldsymbol{x}_C \right\} = \left\{ \begin{pmatrix} 0 \\ \omega \\ 1 \\ \omega \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \omega \\ 0 \\ \omega \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ \omega \\ 0 \\ \omega \\ 0 \end{pmatrix} \right\}.$$

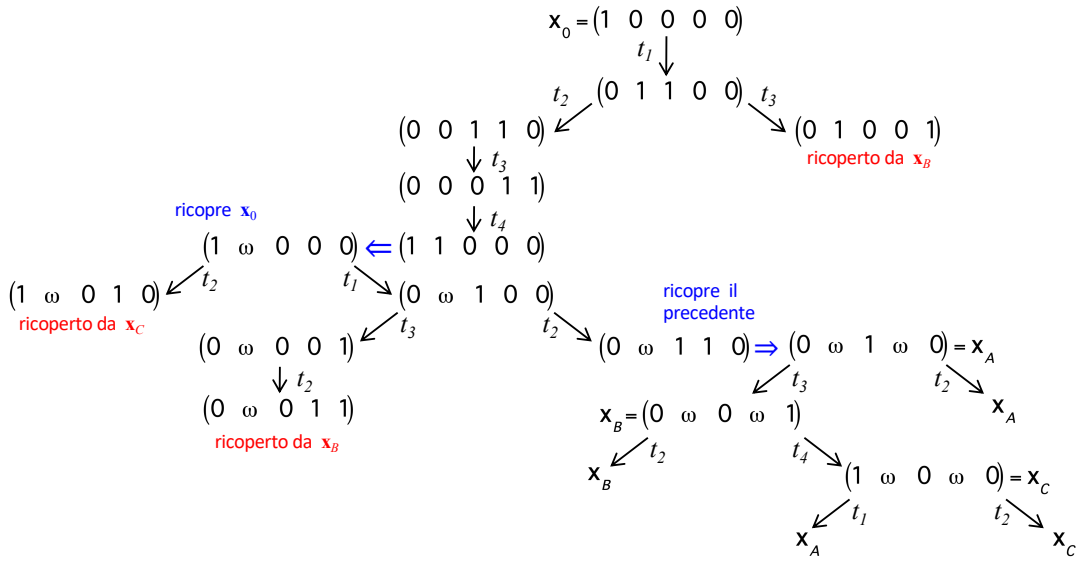

Figura 7: Albero di copertura della rete di Petri di Fig. 1

L'imposizione di un limite massimo pari a $k \geq 1$ al numero totale di token nella rete si può esprimere mediante il seguente vincolo di disuguaglianza sulle marcature raggiungibili da $\boldsymbol{x}_0$:

$$x(p_1) + x(p_2) + x(p_3) + x(p_4) + x(p_5) = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \end{pmatrix} \boldsymbol{x} = \boldsymbol{h}_1^T \boldsymbol{x} \leq k.$$

Poichè la marcatura iniziale $\boldsymbol{x}_0$ soddisfa il vincolo ($\boldsymbol{h}_1^T \boldsymbol{x}_0 = 1 \leq k$), un supervisore che mantenga il soddisfacimento di tale vincolo per tutte le evoluzioni possibili della rete si ottiene introducendo un posto monitor p_1^m, con associata riga aggiuntiva nella matrice di incidenza pari a

$$\boldsymbol{c}_1^m = -\boldsymbol{h}_1^T \boldsymbol{C} = \begin{pmatrix} -1 & 0 & 0 & 0 \end{pmatrix}$$

e inizializzazione

$$x(p_1^m) = k - \boldsymbol{h}_1^T \boldsymbol{x}_0 = k - 1.$$

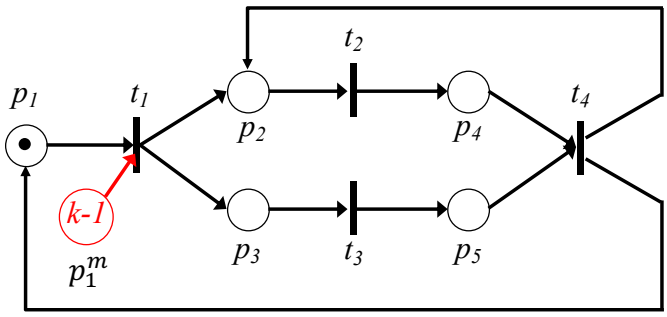

Figura 8: La rete di Petri di Fig. 1 supervisionata dal posto monitor p_1^m

La rete supervisionata risultante è mostrata in Fig. 8. Il posto monitor permette di avere solo $k-1$ scatti della transizione t_1. Si noti che a questo stesso supervisore si sarebbe arrivati anche

se si fosse voluto limitare il numero massimo di token presenti solo nei due posti originariamente illimitati, ossia p_2 e p_4 (con il vincolo $x(p_2) + x(p_4) \leq k$).

E' facile vedere che, una volta esauriti i token nel posto di controllo p_1^m, la rete può ancora effettuare solo un numero finito di scatti per poi bloccarsi in uno stato con tutti i token presenti nel posto p_4. Tale situazione di deadlock è legata al fatto che, quando t_1 non può più scattare, anche il cammino $\{p_3, t_3, p_5\}$ non può essere percorso da token e la transizione t_4 viene quindi disabilitata per sempre. Ragionando sulla logica di funzionamento della rete (o effettuando altri tentativi di supervisione), è possibile convincersi che non è possibile definire altrimenti un supervisore mediante la tecnica degli invarianti, nè con un posto monitor collegato in modo diverso alle transizioni nè utilizzando più posti monitor (ognuno associato a una disuguaglianza lineare su combinazioni di token presenti nei posti della rete).

Per ottenere contemporaneamente limitatezza e assenza di deadlock, è possibile introdurre una versione estesa di supervisore a eventi discreti che preveda al suo interno, oltre ai posti di controllo, anche transizioni aggiuntive (che si possono interpretare come dinamiche interne al sistema di controllo). In questo modo diventa possibile agire sulla rete originaria non solo connettendo i posti del supervisore alle transizioni della rete originaria, ma anche intervenendo in un senso o nell'altro sui posti della stessa. Ad esempio, il supervisore 'dinamico' riportato in Fig. 9 è una modifica del precedente che realizza l'obiettivo preposto. Viene di fatto sostituito il ciclo originario che passava attraverso l'abilitazione allo scatto della transizione t_1 con un nuovo ciclo attraverso il controllore, evitando al contempo l'accumulo di token in p_1. L'inizializzazione del secondo posto di controllo p_2^m non prevede token.

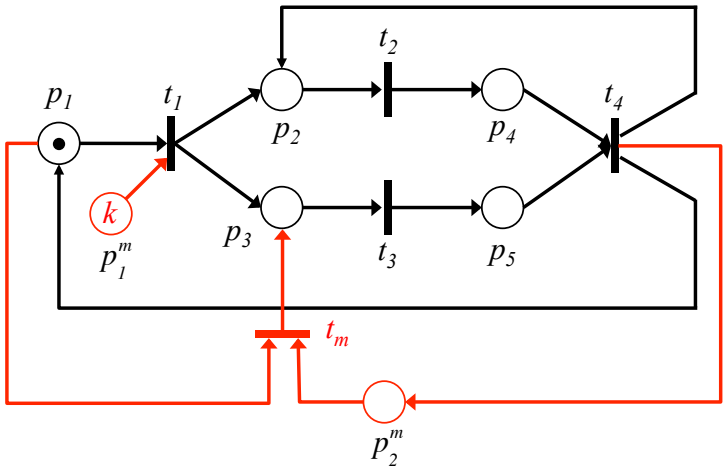

Figura 9: La rete di Petri di Fig. 1 controllata da un diverso tipo di supervisore con due posti di monitor risulta limitata e priva di deadlock

La nuova matrice di incidenza $\boldsymbol{C}^e$ complessiva (processo+controllore), ora di dimensioni (7×5), e

lo stato iniziale $\boldsymbol{x}_0^e$ sono:

$$\boldsymbol{C}^e = \begin{pmatrix} -1 & 0 & 0 & 1 & -1 \\ 1 & -1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix}, \qquad \boldsymbol{x}_0^e = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ k \\ 0 \end{pmatrix}.$$

Considerando per semplicità il caso $k = 1$, tutte le marcature raggiungibili a partire da

$$\boldsymbol{x}_0^e = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}^T$$

si possono determinare dall'albero di raggiungibilità riportato in Fig. 10. La rete supervisionata è limitata. Eccetto la t_1, tutte le altre transizioni (compresa la t_m del supervisore) sono vive.

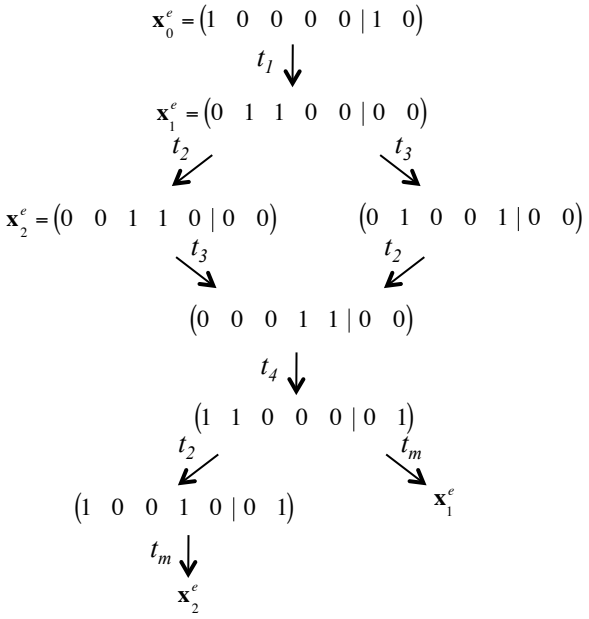

Figura 10: Albero di raggiungibilità della rete di Petri supervisionata di Fig. 9

* * * * *