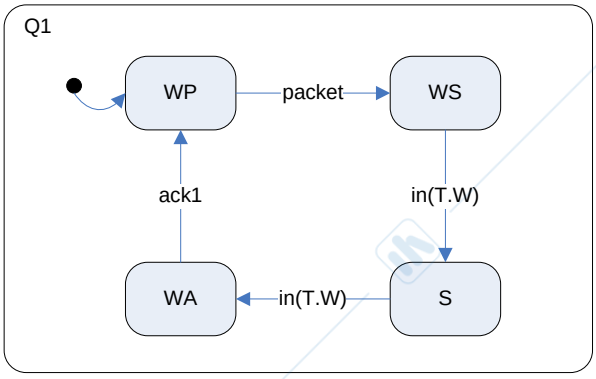
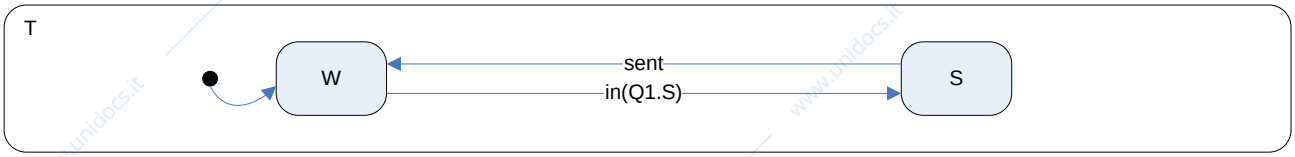
	Politecnico di Milano Facoltà di Ingegneria dell'Automazione **INFORMATICA INDUSTRIALE** Appello 12 luglio 2007	COGNOME E NOME
	RIGA / COLONNA	MATRICOLA

- Il presente plico pinzato, composto di quattro fogli (fronte/retro), deve essere debitamente compilato con cognome, nome, numero di matricola, posizione durante lo scritto, e deve essere firmato.
- I compiti non compilati, non firmati o con fogli mancanti non saranno considerati validi e quindi non saranno corretti.
- Sarà valutato solo quanto scritto su questi fogli.
- Non è consentito consultare testi né appunti.
- Sul tavolo non devono essere presenti telefoni cellulari, né astucci, né custodie di altro tipo.
- Le risposte devono essere scritte negli appositi riquadri, qualsiasi testo esterno a tali riquadri non verrà preso in considerazione.
- Se lo spazio per la soluzione degli esercizi non fosse sufficiente, si può scrivere sull'ultimo foglio.

FIRMA

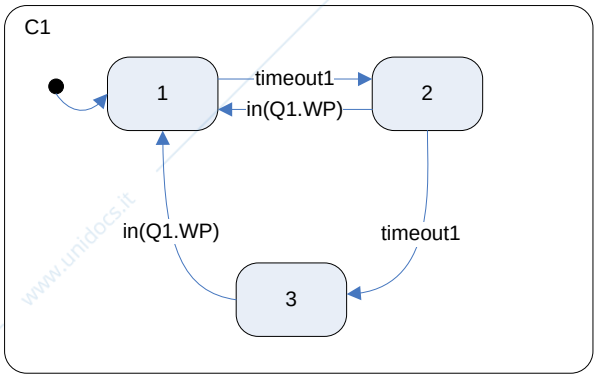
Esercizio 1 (10 punti). Si vuole specificare tramite il formalismo degli Statechart senza l'utilizzo di variabili la trasmissione di pacchetti TCP su rete da parte un singolo host. Il mezzo trasmissivo è costituito da un trasmettitore *T* che può essere in attesa (*W* - waiting) oppure in invio (*S* - sending). Il trasmettitore può trasmettere un solo pacchetto per volta e, qualora la trasmissione del singolo pacchetto si concluda, viene generato un evento *sent*. Il mezzo trasmissivo viene utilizzato con mutua esclusività da alcune code a priorità (*Q* – queue) il cui obiettivo è gestire l'invio di un pacchetto: recuperare il pacchetto, inviarlo al mezzo trasmissivo, aspettare il messaggio che confermi la corretta consegna del pacchetto e, qualora non arrivi tale messaggio, rinviare il pacchetto. In figura è fornito lo statechart che specifica la situazione in cui sia presente una sola coda *Q1* e la rete sia affidabile. La coda si può trovare: in attesa dell'arrivo di un pacchetto (*WP* – waiting for a packet), in attesa di accedere al mezzo trasmissivo (*WS* – waiting for sending), in invio (*S* - sending) e in attesa del messaggio di corretta consegna del pacchetto (*WA* – waiting for acknoledgment). Ogni volta che almeno una coda (in questo caso solo *Q1*) è in *WP* ed è presente un pacchetto da inviare, viene generato un evento *packet* che viene comunicato a tutte le code. Nel caso ci sia una sola coda, il pacchetto viene automaticamente assegnato a *Q1* e quindi questa transisce da *WP* a *WS*. Una volta in *WS*, la coda *Q1* transisce in *S* qualora il mezzo trasmissivo sia disponile, cioè quando è in *W*. Una volta terminata la trasmissione il mezzo trasmissivo ritorna nello stato *W* e conseguentemente la coda transisce in *WA*. A fronte dell'evento *ack1* la coda transisce da *WA* a *WP*.

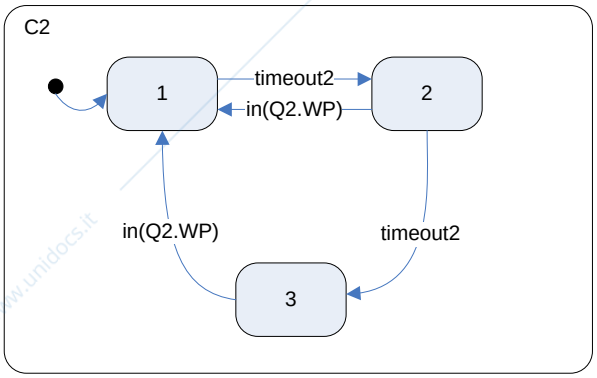


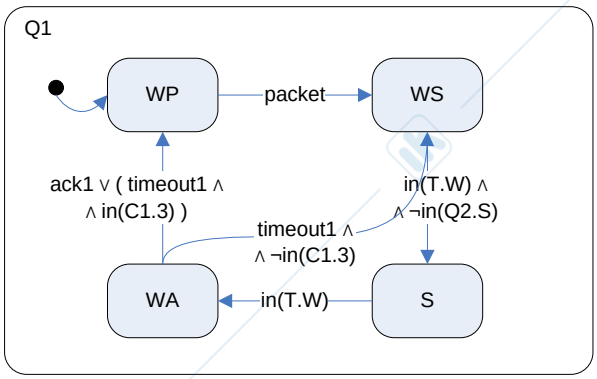


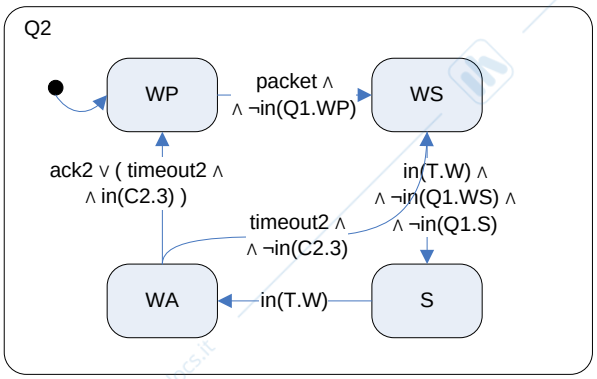
Si richiede di estendere lo statechart dato per cogliere due aspetti: 1) la possibilità che un pacchetto non venga consegnato correttamente e venga quindi ritrasmesso e 2) la presenza di due code concorrenti.

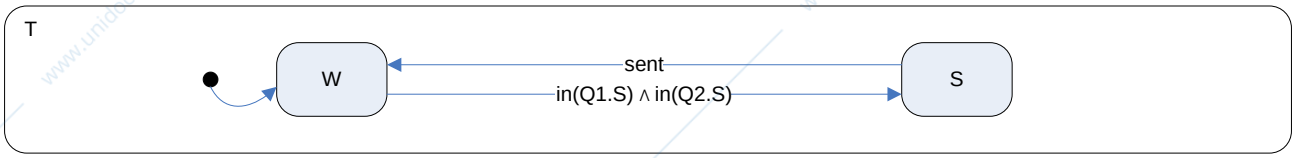
1) Una volta inviato un pacchetto tramite la coda *Q1* e non si abbia ricevuto il messaggio di conferma entro una scadenza prefissata, viene generato un evento *timeout1*. A fronte di *timeout1* il pacchetto viene ritrasmesso. Il numero massimo di ritrasmissioni è 2, dopodichè la coda gestisce un altro pacchetto. Si utilizzi uno stato contatore *C1* per tenere traccia del numero di trasmissioni di un pacchetto.
2) Si introduca una nuova coda *Q2*. Le due code funzionano in modo analogo in parallelo. Viene però data precedenza a *Q1*: se è disponile un pacchetto ed entrambe le code lo possono gestire viene preferita la coda *Q1*; analogamente se entrambe le due code sono in attesa di accedere al mezzo trasmissivo viene preferita la coda *Q1*.











Esercizio 1 (9 punti). Si consideri il problema di schedulare i processi aperiodici J_1-J_6 riportati in tabella qualora non sia possibile fare *preemption*:

	J_1	J_2	J_3	J_4	J_5	J_6
a_i	3	2	0	4	1	6
C_i	2	1	1	2	3	1
d_i	6	5	3	12	8	10

Si applichi l'algoritmo di Spring senza backtracking prima con euristica pari al tempo di arrivo e successivamente pari alla deadline e si riportino le schedulazioni trovate (non è necessario riportare l'albero di ricerca). Si dica se le due schedulazioni sono ammissibili. Si applichi poi l'algoritmo EDF al problema sopra e si riporti la soluzione trovata. Dire se questa è ammissibile e perché. Nel caso in cui non sia ammissibile, si segnalino le cause del perché EDF produce una schedulazione non ammissibile.

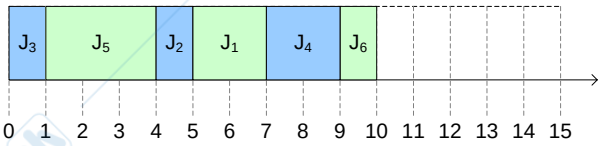
Si applichi poi l'algoritmo di Spring al problema sopra con euristica pari al tempo di arrivo utilizzando un numero qualsiasi di rami di backtracking e il Forward Checking. Se ne riporti l'albero di ricerca segnalando per ogni nodo: i domini, le euristiche, i rami generati.

Si consideri infine i processi periodici riportati nella tabella sotto, dove il costo computazionale del processo quattro è incognito:

	τ_1	τ_2	τ_3	τ_4
ϕ_i	0	0	0	0
C_i	2	3	1	?
T_i	8	12	5	20

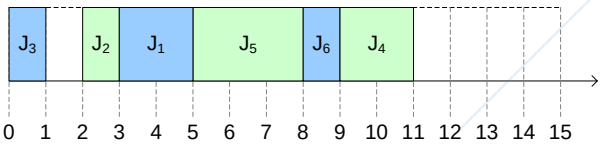
Dimensionare C_4 affinché il problema di schedulazione sopra sia schedulabile mediante RM. Si fornisca nell'ordine il massimo valore di C_4 per cui sia schedulabile tramite Lyu&Layland, il massimo valore di C_4 per cui sia schedulabile tramite Bini&Buttazzo, il massimo valore in assoluto di C_4 per cui sia schedulabile (si utilizzi la sezione critica).

La schedulazione prodotta dall'algoritmo di Spring senza backtracking e con euristica pari al tempo di arrivo è:



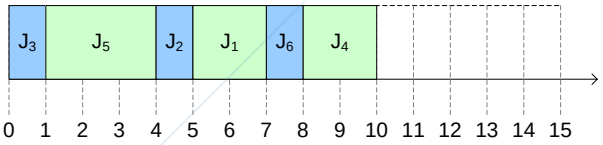
Questa schedulazione non è ammissibile in quanto il processo J_1 sfora la propria deadline.

La schedulazione prodotta dall'algoritmo di Spring senza backtracking e con euristica pari alla deadline:



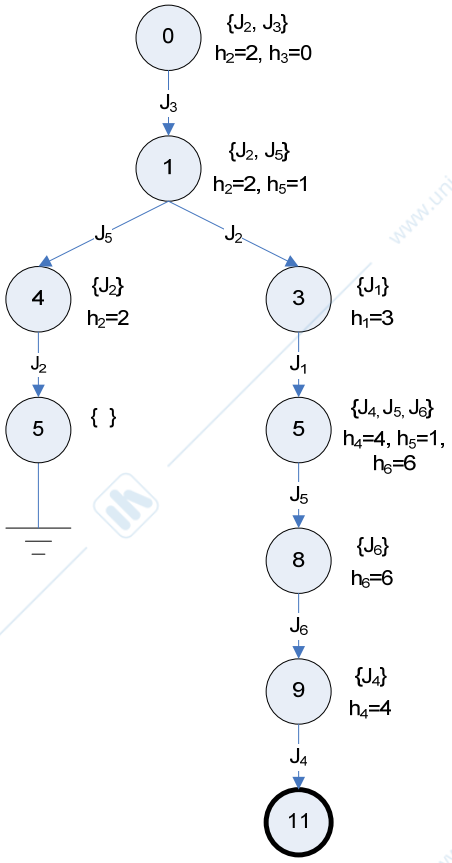
Questa schedulazione è ammissibile.

La schedulazione prodotta dall'algoritmo di EDF è:



Questa schedulazione non è ammissibile in quanto J_1 sfora la propria deadline. Questo è dovuto al fatto che non è possibile fare *preemption*. Sotto questa ipotesi EDF non è ottimale.

Riporto l'albero di ricerca quando l'euristica è data dal tempo di arrivo e sia possibile fare backtracking:



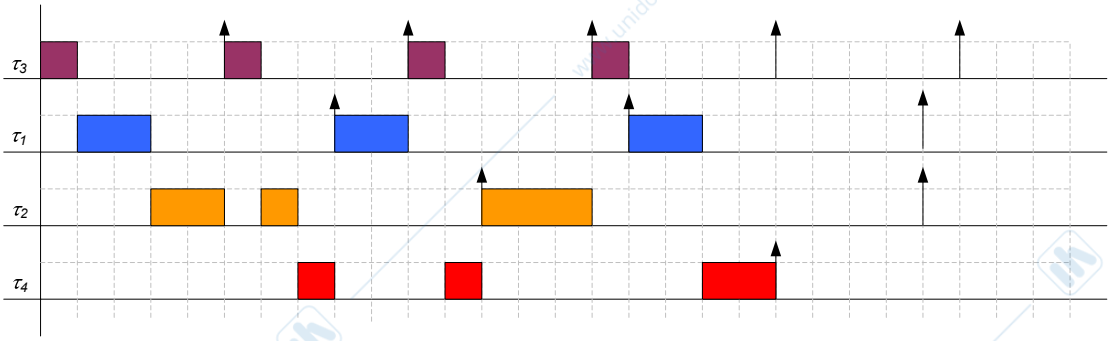
Il massimo valore per cui sia schedulabile tramite Lyu&Layland è calcolabile nel modo seguente:

$U_{lub} = 4(2^{0.25} - 1) = 0.7568$
$U = 1/4 + 1/4 + 1/5 + C_4/20 = 0.7 + C_4/20$
$U \leq U_{lub}$
$C_4 = 1$

Il massimo valore per cui sia schedulabile tramite Bini&Buttazzo è calcolabile nel modo seguente:

$(1/4 + 1)^2 \cdot (1/5 + 1) \cdot (C_4/20 + 1) \leq 2$
$C_4 = 1$

Il massimo valore in assoluto per cui sia schedulabile è possibile trovarlo mediante l'analisi della sezione critica.



La sezione critica mostra che il massimo valore è 4.

Esercizio 3 (6 punti). Rispondere alle seguenti domande nell'ambito della Schedulazione Real Time:

1. Descrivere in modo sintetico tre algoritmi per la schedulazione di processi misti.
2. Enunciare il Teorema di Tia-Liu-Shankar.

Esercizio 4 (6 punti). Si descriva la pila protocollare ISO/OSI specificandone sinteticamente ogni livello.