Verifica Di Programmi Concorrenti Formalismi: esercizio con le reti di Petri - Consegna 2

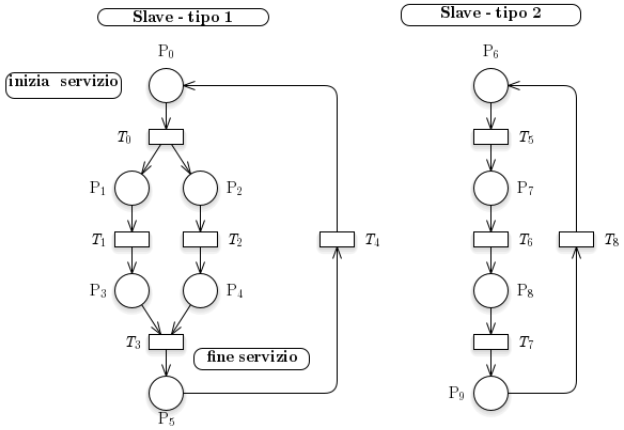
Studente: Andrea Senese 811634

Marzo 2019

Esercizio

Si modelli un sistema master slave nelle seguenti versioni:

1. Rete A: M master identici e S slave di tipo 1 identici. Il modello deve essere parametrico in M e S;
2. Rete B: M master identici e due slave, uno di tipo 1 e uno di tipo 2. Ad ogni ciclo il master sceglie con scelta libera "free choice" (cioè indipendentemente dalla disponibilità degli slave) di quale dei due slave servirsi;
3. Rete C: due master distinti (seppur di uguale struttura). Un master di tipo 1 e uno di tipo 2. Ad ogni ciclo il master sceglie in modo indipendente di quale dei due slave servirsi;
4. Rete D: due master distinti (seppur di uguale struttura) con uno slave associato al singolo master (il primo master usa sempre lo slave di tipo 1 e il secondo usa sempre quello di tipo 2).



Algorithm 1 Master - struttura tipo:

```
1: procedure MASTER{
2:     loop{
3:         operazione locale;
4:         richiedi servizio allo slave;
5:         elaborazione risultato servizio;
6:     }
7: }
```

Nota: L'operazione "richiedi servizio" è bloccante, nel senso che non si può passare a elaborare il risultato del servizio sino a che non si è ottenuta una risposta dallo slave.La comunicazione avviene tramite buffer. Nei casi in cui il master effettui una scelta fra gli slave, la scelta deve essere fatta dal master in totale autonomia (per esempio non deve dipendere da quale slave sia libero).

Richieste:

1. Disegnare la rete e usare la funzionalità di Token game (freccia verde nella barra principale) per mettere a punto il modello;
2. Costruire il grafo di raggiungibilità con GreatSPN. Per le reti parametriche sperimentare con diverse marcature;
3. Applicare le tecniche di analisi strutturale (P- e T- invarianti);
4. Applicare le tecniche di riduzione alla rete A;
5. Costruire il decision diagram per la rete D, sperimentando diverse assegnazioni dei posti ai livelli del decision diagram, osservando come cambia il numero di nodi del decision diagram.

Relazione:

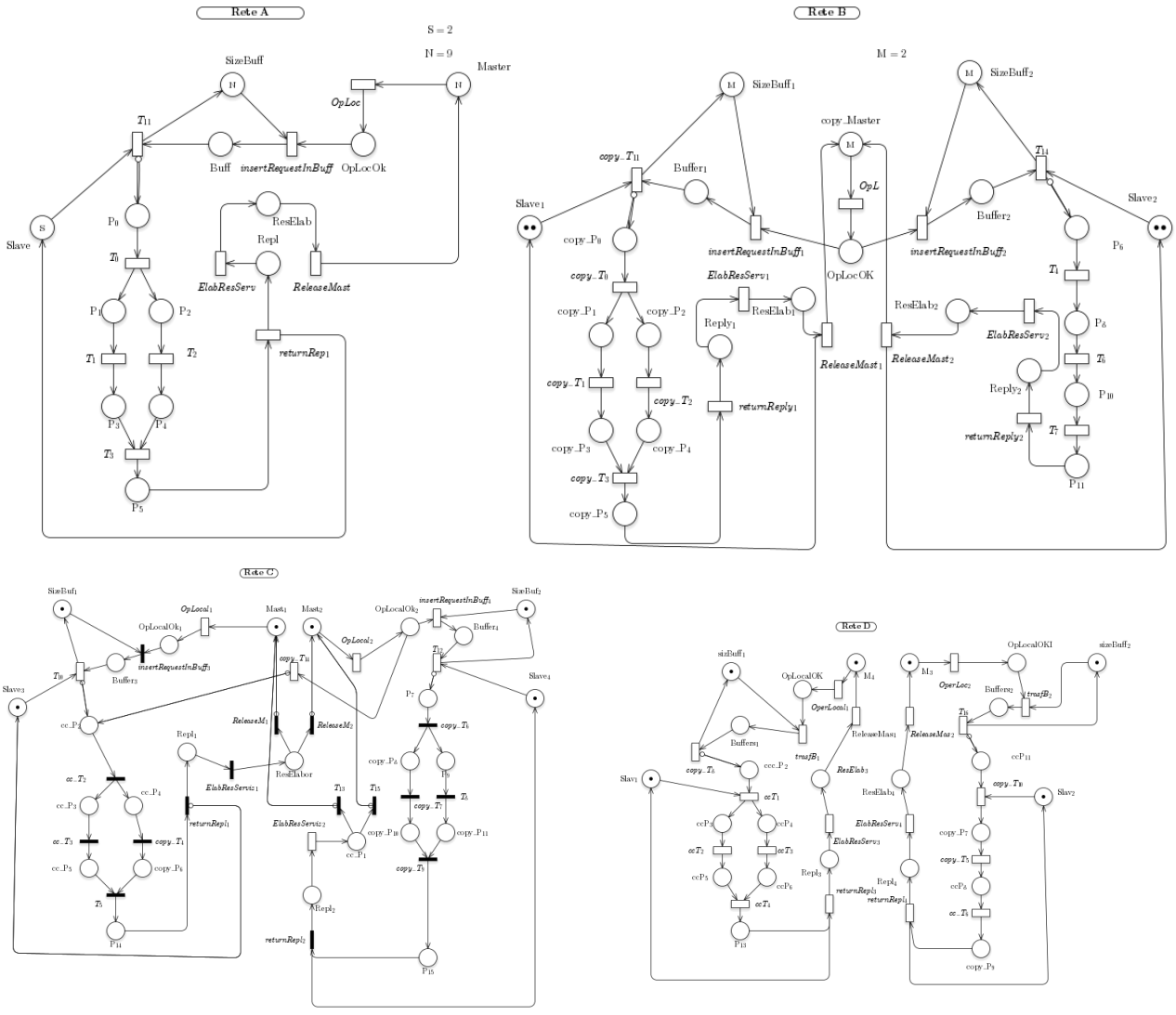
1. Commentare i singoli modelli, e confrontare fra loro le dimensioni dei grafi di raggiungilità (nodi e archi);
2. Sperimentate la dipendenza delle dimensioni del reachability graph dalla marcatura iniziale, provando ad aumentare il numero di master e di slave delle reti parametriche. Sino a che numero di master e di slave potete arrivare con la memoria a vostra disposizione (indicare quanta) e tempi di esecuzione che non superino i 10 minuti?
3. In quali casi siamo certi che il join avvenga fra due sottoprocessi creati dalla stessa fork? E, in caso di risposta negativa, come dovremmo cambiare il modello e/o la specifica dell'esercizio per fare in modo che questo succeda?

Relazione

Andrea Senese
Matricola : 811634

Marzo 2020

1 Modelli



2 Descrizione

Da quanto si è potuto osservare durante la sperimentazione, il modello più appropriato è la rete D in quanto ho trovato meno difficoltà nel rappresentarlo e inoltre a parer mio sembra più opportuno avere i due master separati in quanto migliora nella leggibilità della rete e per quanto riguarda le dimensioni il modello di rete D infatti è quello che ha il reachability graph minore. A prim'occhio, guardando la rete sembra che la rete A sia la più piccola ma generando il reachability graph ho ricavato le seguenti informazioni:

- Rete A : Total Marking: 80 out of 9955
- Rete B : Total Marking: 80 out of 168
- Rete C : Total Marking: 80 out of 183
- Rete D : Totale Marking: 80 out of 99

e da queste informazioni si evince che si ottiene un esplosione di stati nella Rete A. Quindi quanto considerato e detto, il miglior modo è di avere due master distinti dove ognuno di loro utilizza un determinato slave (ossia la rete D). Sperimentando la dipendenza delle dimensioni del RG dalla marcatura iniziale, provando ad aumentare il numero di master e di slave nella rete A ho tratto le seguenti conclusioni: ossia ho raggiunto un massimo di 18 token nel master e 18 token negli Slave. Tutto ciò è stato fatto impostando un limite bound di tempo (un massimo di 10 min) ed ho tratto che:

$$Time(RGAlg) > 10\ min \tag{1}$$

e quindi con un massimo di 15 token riesco a rientrare nella fascia di tempo imposta. Per quanto riguarda il join, nel caso dello slave di tipo 1 è possibile accertarci che il join avvenga tra due sottoprocessi creati dalla stessa fork mentre nello slave di tipo 2 no perché lavora lo stesso processo senza forkare. Guardando in generale, ad esempio, se guardiamo il master possiamo notare che non è possibile identificare il lavoro richiesto. Come stiamo vedendo nella parte delle Well-formed Petri Nets per poter identificare si ha bisogno di introdurre i colori per poter catturare le eventuali simmetrie del sistema.