

Politecnico di Milano



AA 2005 / 2006

Progetto RdP: Analisi dei tools di simulazione

Silvia Barbariga (675460)
Paolo Tagliaferri (675331)

Obbiettivi del progetto

Il progetto trattato in questa relazione mira alla sperimentazione di differenti tool presenti in rete, atti a simulare Reti di Petri.

Tra questi abbiamo analizzato i seguenti applicativi:

- Platform Independent Petri Net Editor 2 (PIPE2)
- Simulator for Petri net Representation for Embedded Systems (SimPRES)
- Petri net Tool (PeT)
- HPSim

I primi due programmi rientrano a far parte dell'elenco dei tool consigliati per il progetto, mentre gli altri due sono stati introdotti per ampliare l'analisi, rivolgendola a strumenti presenti in rete ma non sufficientemente considerati. In particolar modo sono stati scelti per confrontare le capacità fornite da strumenti di pari “*scope* simulativo” (PeT vs PIPE2, SimPRES vs HPSim).

I programmi sono in grado di rappresentare diverse tipologie di Reti di Petri: in particolar modo si distingue tra reti di petri temporizzate (trattate da SimPRES e HPSim) e reti di petri non temporizzate (trattate da PeT e PIPE2).

Inizialmente, per ogni programma, è stata effettuata un'analisi di insieme dell'ambiente di simulazione (controllando anche aspetti di usabilità dell'interfaccia, di correttezza formale e di stabilità e robustezza).

Successivamente sono stati costruiti i modelli proposti per il confronto delle capacità:

- Modello produttore/consumatore con buffer infinito;
- Modello produttore/consumatore con buffer finito;
- Modello produttore/consumatore con buffer finito e timeout sulla scrittura/lettura (dopo T istanti di indisponibilità del buffer per l'operazione, si scarta il dato)
- Modello di processi concorrenti che utilizzano risorse condivise (quindi con possibile Deadlock)

La realizzazione del terzo modello è stata possibile solo con HPSim e SimPRES, in quanto sono gli unici tool in grado di simulare reti temporizzate. In realtà anche PIPE2 prevede una generica modalità temporizzata per le transizioni (durante la simulazione in modalità GSPN), tuttavia non siamo stati in grado di capire come parametrizzare la simulazione in modo conveniente, né abbiamo trovato questa informazione sul manuale.

Platform Independent Petri Net Editor 2 (PIPE2)

Il tool PIPE2 (<http://pipe2.sf.net>) è un progetto open source sviluppato in linguaggio Java, con un'ottima documentazione in formato JavaDoc per gli sviluppatori. Per questi ultimi è prevista una facile integrazione con Eclipse o con altri strumenti di sviluppo tramite il repository CVS su cui è incentrato lo sviluppo di questo programma. La versione analizzata è la 2.0.

Una particolarità di questo tool è la conformità alla specifica PNML (Petri Net Markup Language – <http://www.informatik.hu-berlin.de/top/pnml/pnml.html>), che è un tentativo di standardizzare il formalismo di descrizione delle reti di petri tramite XML. Questo, unito alla portabilità ottenuta tramite la realizzazione modulare in Java, rende questo tool particolarmente versatile nonché facilmente interoperabile con altri applicativi aderenti allo standard PNML.

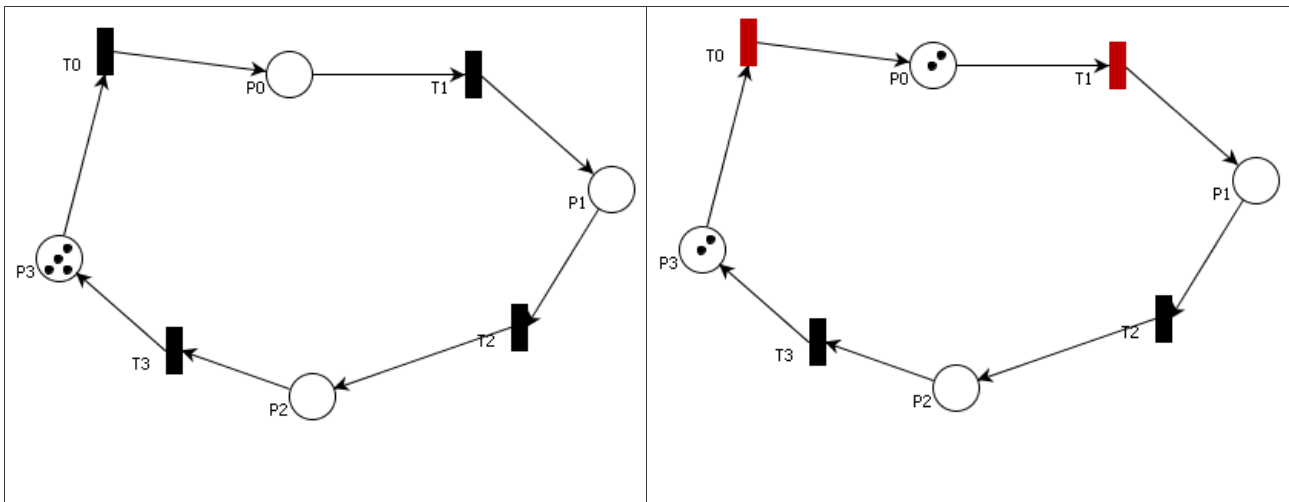
Capacità del tool

L'interfaccia è molto intuitiva e permette di disegnare diagrammi leggibili in modo semplice e rapido. E' possibile creare archi spezzati e/o curvi per evitare intersezioni poco chiare, ruotare le transizioni e inserire annotazioni testuali in ogni punto del diagramma. Tramite un apposito menù si può facilmente esportare il diagramma in formato grafico (PNG oppure PostScript), che possiamo stampare oppure includere in altri tipi di documento.

La simulazione di base prevede una modalità "animata" attivabile tramite apposito pulsante, che ci permette di osservare l'evolversi della rete. Possiamo eseguire un'animazione "step-by-step", andando a effettuare lo sparo di una transizione tra quelle abilitate, che vengono colorate in rosso. Alternativamente possiamo lasciare al programma la scelta (casuale) delle transizioni da attivare, anche in modalità continua (con un periodo di tempo selezionabile). E' sempre possibile arrestare la simulazione e tornare ai passi precedenti, per ispezionare la rete.

Il tool supporta le reti di Petri classiche e quelle stocastiche generalizzate.

Si è riscontrato un possibile baco per quanto riguarda l'evoluzione della rete. In molti casi infatti, i token sembrano procedere "appaiati" nonostante il peso degli archi coinvolti nello sparo della transizione siano unitari. Ad esempio, relativamente all'immagine seguente:



Si nota che nella foto a sinistra la transizione T0 risulta abilitata (ed impostata con peso sull'arco pari a 1). Eseguendo un passo della simulazione si vede che l'attivazione della transizione T0 genera 2 token nel posto P0 diversamente dal comportamento desiderato.

Inoltre lo stesso comportamento si ripresenta quando due token distinti si “incontrano” in un posto durante l'evoluzione della rete.

Analisi supportate

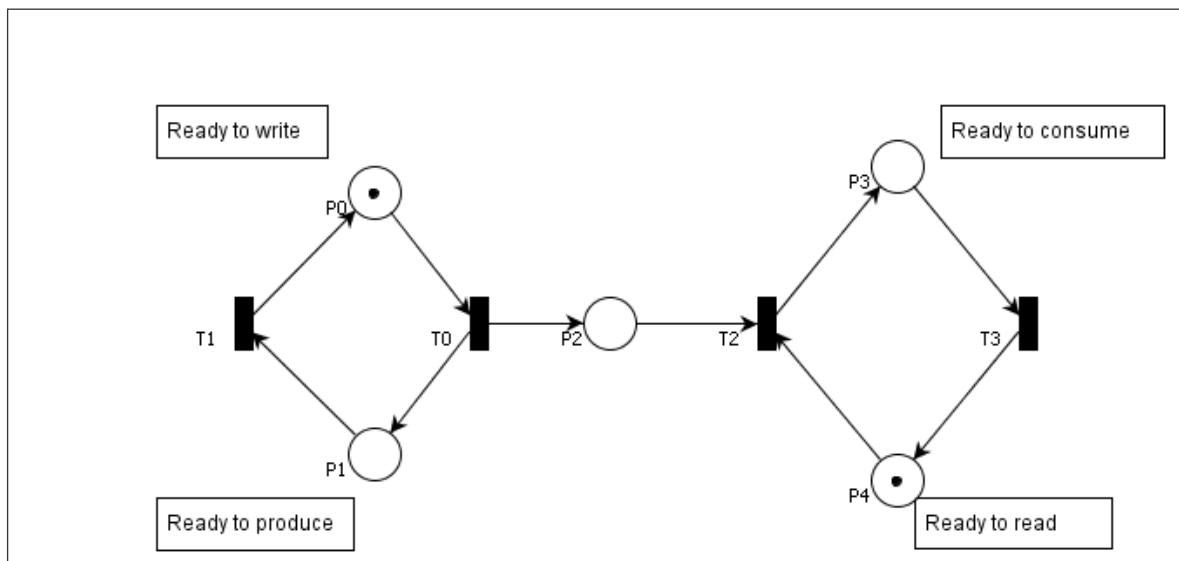
Il programma ha una struttura modulare, grazie alla quale è possibile aggiungere e rimuovere in modo semplice i componenti che effettuano analisi sulla rete. In particolare nella versione considerata sono presenti diversi moduli:

- **Classificazione:** il modulo esegue un'ispezione della rete e ne classifica le proprietà notevoli. Si è infatti in grado di stabilire se la nostra rete rappresenta una macchina a stati, un grafo marcato, una rete a scelta libera (anche estesa) oppure una rete semplice (anche estesa);
- **Confronto:** il modulo confronta due reti di petri e mostra all'utente le somiglianze, le corrispondenze e le differenze tra i modelli;
- **DNAmaca:** il modulo è in grado di interfacciarsi con un tool esterno (DNAmaca, un analizzatore di performance basato sull'analisi delle catene di Markov). Indicando le condizioni di partenza e le condizioni terminali, il tool genererà una distribuzione di probabilità relativa alle tempistiche coinvolte in questa operazione;
- **GSPN Stati stabili:** analisi relativa alle reti generalizzate stocastiche. Viene esplorato lo spazio degli stati, calcolato il numero medio di token per stato, la densità di probabilità dei token nei posti e il throughput medio delle transizioni temporizzate;

- **Invarianti:** questo modulo è in grado di cercare i P-invarianti ed i T-invarianti della rete esaminata, esaminarne la relativa copertura e quindi confermare o rifiutare le proprietà di vitalità e limitatezza della rete. Restituisce anche le equazioni correlate agli invarianti stessi;
- **Matrici d'incidenza e marcatura iniziale:** il modulo d'analisi estrae, data una marcatura, le matrici d'incidenza (in avanti, all'indietro e combinata) e mostra quali sono le transizioni abilitate e quali non lo sono;
- **Simulazione:** il modulo esegue un numero fissato di iterazioni di una simulazione e ricava il numero medio di token presenti in ogni posto, con un indice associato sulla confidenza del valore al 95%;
- **Analisi Spazio degli stati:** il modulo esegue un'analisi della rete e ricava le proprietà di limitatezza, sicurezza e deadlock per la rete considerata.

Comportamento del tool nei modelli proposti

Producer-Consumer, unbounded buffer

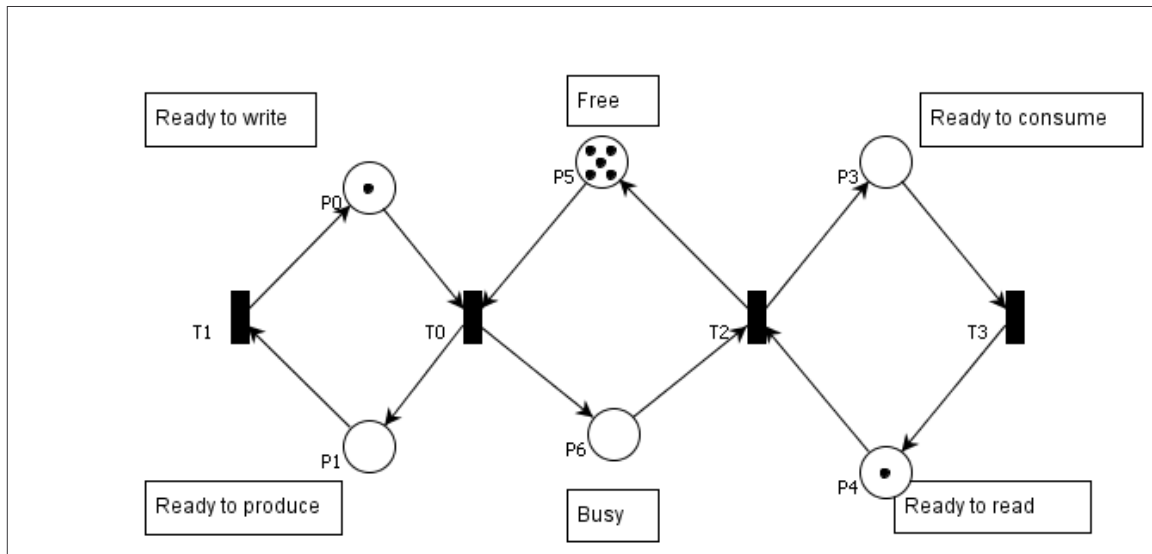


Abbiamo realizzato una simulazione di 500 abilitazioni casuali su 20 cicli, ottenendo questi risultati:

Petri net simulation results		
Place	Average number of tokens	95% confidence interval (+/-)
P0	0,48104	0,03417
P1	0,51896	0,03417
P2	5,54491	5,95572
P3	0,47505	0,03787
P4	0,52495	0,03787

Producer-Consumer, bounded buffer

La simulazione prevede la presenza di un buffer reale con 5 posizioni disponibili.



Abbiamo realizzato una simulazione di 500 abilitazioni casuali su 20 cicli, ottenendo questi risultati:

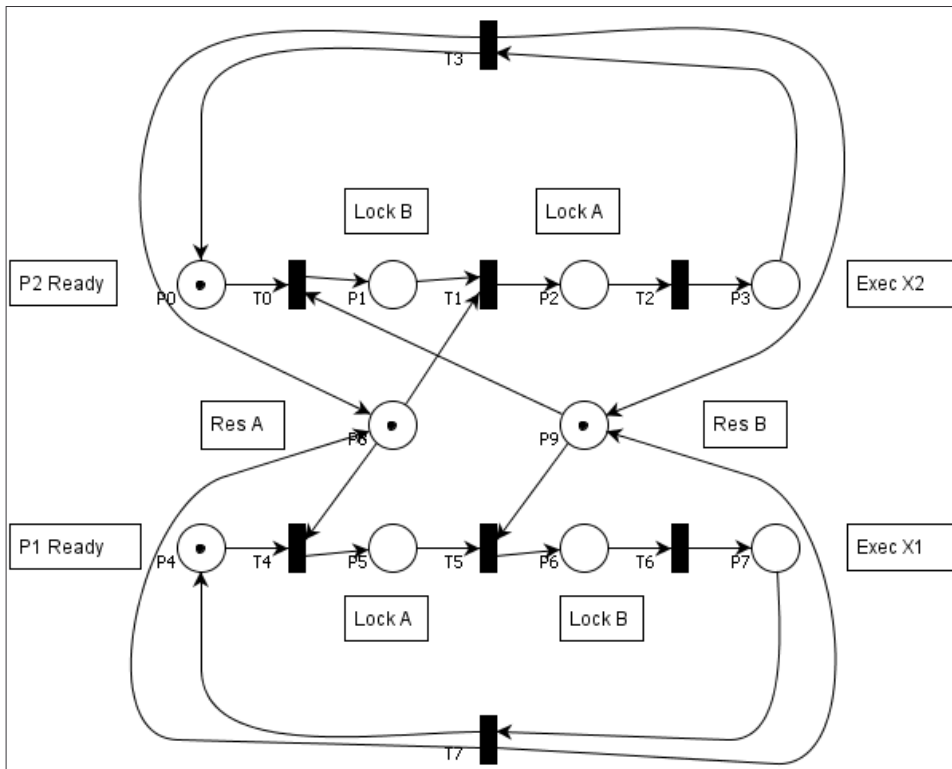
Place	Average number of tokens	95% confidence interval (+/-)
P0	0,46108	0,04901
P1	0,53892	0,04901
P3	0,46307	0,06123
P4	0,53693	0,06123
P5	3,45908	0,87405
P6	1,54092	0,87405

Producer-Consumer, bounded buffer (timeout)

Questo esercizio non è riproducibile con il presente tool perchè la temporizzazione non prevede l'impostazione di periodi specifici ma solamente l'inserimento di transizioni temporizzate generiche. Per cui non è possibile forzare il comportamento richiesto dall'applicazione.

Deadlock

Per analizzare la situazione di deadlock con processi concorrenti e locking di risorse condivise è stata disegnata la seguente rete. Si nota che le possibilità grafiche offerte dal tool utilizzato permettono di ottenere in modo semplice e veloce risultati gradevoli e facilmente comprensibili.



Utilizzando il modulo di analisi dello spazio degli stati si nota che il tool riconosce correttamente la possibilità di deadlock, indicando anche (a partire dalla marcatura iniziale) la sequenza minima di attivazioni delle transizioni che genera uno stato di deadlock.

Petri net state space analysis results	
Bounded	true
Safe	true
Deadlock	true
Shortest path to deadlock: T0 T4	

SimPRES

SimPRES è l'acronimo di Simulator for Petri net Representation for Embedded Systems, ed è un'applicazione Java (disponibile sia in versione standalone che in versione web-applet) in grado di simulare una rete di Petri estesa con le temporizzazioni. L'autore (<http://www.ida.liu.se/~luico/SimPRES/>) intende infatti modellare e verificare le proprietà formali dei sistemi embedded utilizzando reti di Petri modificate con informazioni aggiuntive associate ai token: dati e timestamp. Inoltre è possibile impostare funzioni e ritardi alle transizioni. SimPRES si limita a modellizzare e validare un sottoinsieme di queste reti (le reti di Petri temporizzate).

La versione da noi valutata è la standalone net-edition 0.2

Capacità del tool

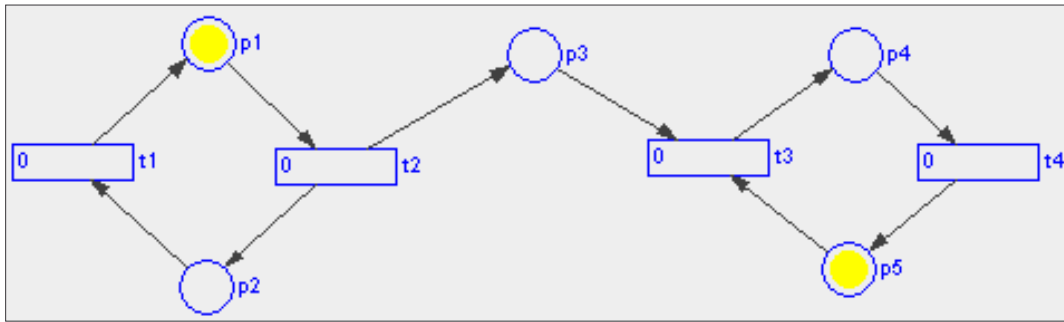
Rispetto a PIPE, l'interfaccia fornita dal tool è più "spartana". L'applicazione ci permette di disegnare la nostra rete e salvarla oppure caricare un file già esistente. I modelli vengono salvati in un file con formato proprietario. Il tool ci permette di inserire posti, archi e transizioni: queste ultime sono di tipo temporizzato e sono caratterizzate da ritardo minimo, ritardo massimo e policy di sparò. Le policies possibili sono quattro: distribuzione uniforme, distribuzione gaussiana, appena possibile o più tardi possibile. Gli archi possono avere esclusivamente peso unitario. Per valutare l'evoluzione della rete abbiamo a disposizione dei comandi di play, forward, pause e reset, tutti relativi ad un contatore globale che indica il tempo assoluto di riferimento.

Non sono disponibili particolari strumenti o moduli d'analisi: essa infatti è totalmente delegata all'utilizzatore, che può trarre le relative conclusioni ispezionando l'andamento temporale della rete posta in azione (*token game*).

Durante le simulazioni si è notato un possibile bug nella gestione delle transizioni (specialmente nel caso di producer-consumer bounded buffer con timeout): infatti più volte è capitato (soprattutto aumentando la velocità di simulazione) che il modello "si bloccasse" e che una particolare transizione cominciasse a scattare in loop infinito, facendo esplodere il numero di token presenti. Resettando il modello (e incrociando le dita) è comunque possibile riprovare la simulazione per ottenere risultati sensati.

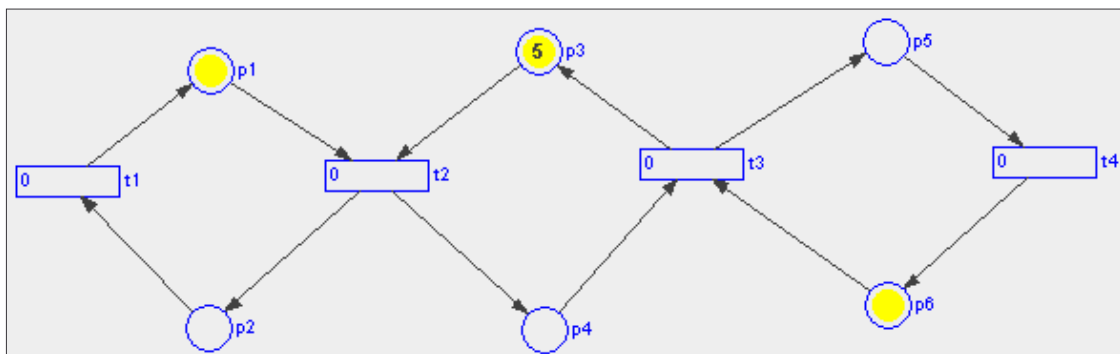
Comportamento del tool nei modelli proposti

Producer-Consumer, unbounded buffer



Per rimuovere la temporizzazione, abbiamo posto tutte le transizioni con ritardo minimo e massimo uguali e pari a zero. In questo modo il simulatore opera nel modo classico senza utilizzare le temporizzazioni (difatti il global clock non viene aggiornato).

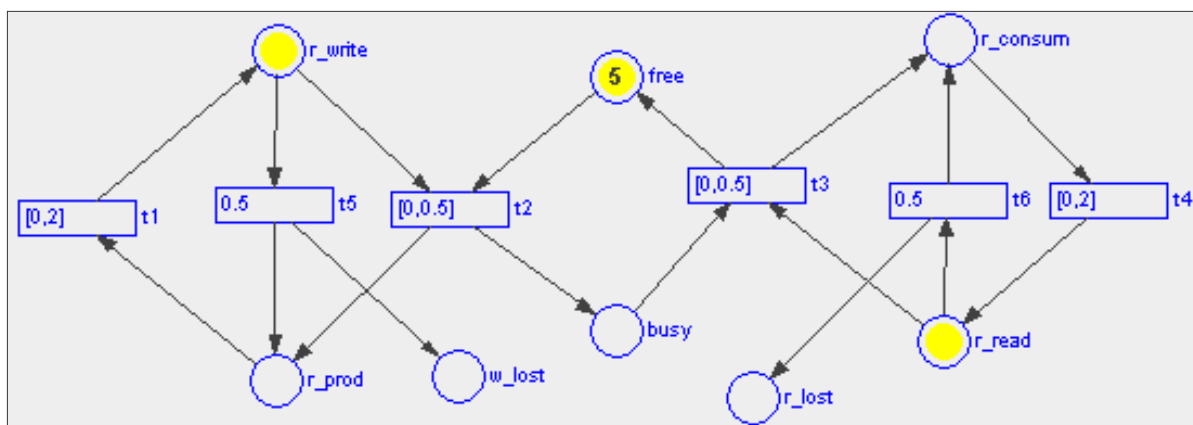
Producer-Consumer, bounded buffer



Anche in questo caso è stato annullato il comportamento temporizzato imponendo a 0 i ritardi delle transizioni. Il buffer reale introdotto è sempre a 5 posizioni disponibili come nella simulazione con il precedente tool.

Producer-Consumer, bounded buffer (timeout)

Il tool si presta ad una preparazione ottimale di questo esperimento, essendo progettato in modo specifico per simulare le reti di Petri temporizzate:



Riguardo alla figura soprastante possiamo fare le seguenti considerazioni:

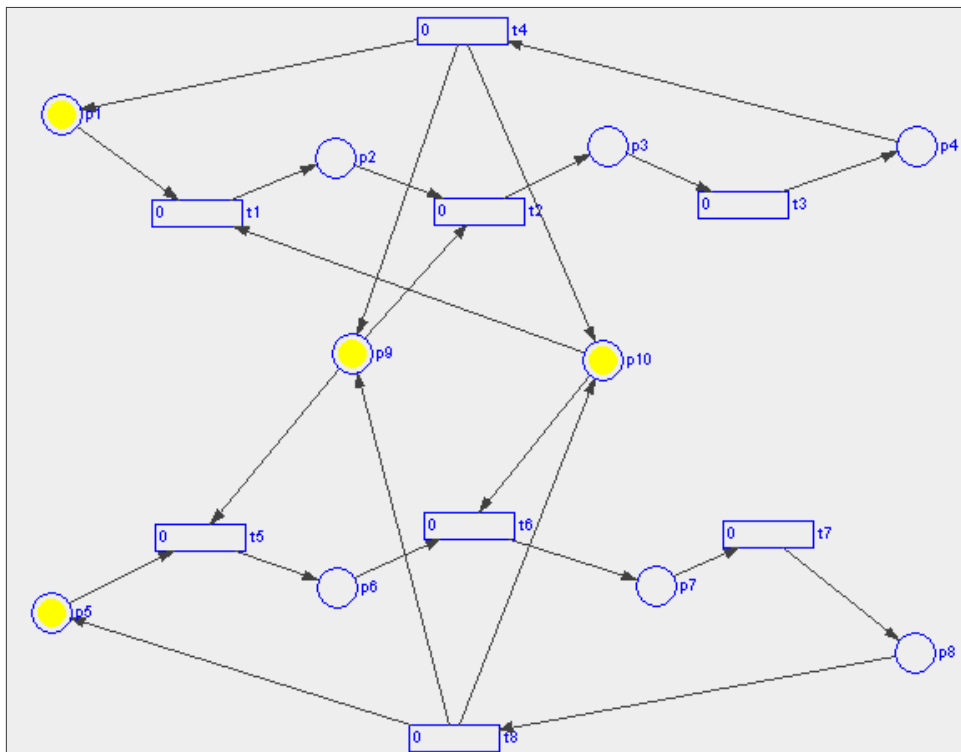
- Rispetto alla rete non temporizzata senza perdite abbiamo aggiunto due transizioni e due posti, rispettivamente T5, T6, w_lost e r_lost
- Avendo impostato un periodo limite pari a 0.5 unità temporali, le transizioni di interazione con il buffer (T2 e T3) si attiveranno il prima possibile, con ritardo minimo di 0 e massimo di 0.5
- Vicendevolmente, le transizioni T5 e T6, che rappresentano gli eventi di perdita rispettivamente sul produttore e sul consumatore, saranno abilitate con politica "*As soon as possible*" con ritardo esattamente pari al periodo limite di 0.5 unità temporali
- **w_lost** e **r_lost** sono posti "fittizi" che mostrano graficamente gli eventi di perdita determinati dalla capacità limitata del buffer.
- Le transizioni T1 e T4 rappresentano la "capacità di processing" del produttore e del consumatore. E' stata impostata una funzione con distribuzione uniforme su un intervallo arbitrario tra 0 e 2 unità temporali. Questo intervallo può essere modificato per modellare differenti situazioni come si vedrà in seguito

Osservando la simulazione abbiamo riscontrato un comportamento simile a quello atteso: l'occorrenza delle perdite è proporzionata alla dimensione del buffer, ma anche dalle capacità di processing di produttore e consumatore. Mantenendo costanti gli intervalli di T1 e T4 si simula infatti una situazione dove il produttore opera con ritmo simile a quello del consumatore.

E' possibile sbilanciare questa situazione per osservarne gli effetti: ad esempio si può pensare ad un produttore che genera oggetti ad un ritmo più lento rispetto alle richieste del consumatore. Questo è possibile aumentando l'intervallo della distribuzione uniforme su T1 (ad esempio, ponendola tra 0 e 4) e lasciando inalterato quello su T4. In questo caso, come atteso, si osservano un gran numero di perdite in lettura.

Viceversa è possibile pensare ad un produttore con frequenza maggiore della disponibilità di processing del consumatore (simmetricamente a quanto visto in precedenza): in questo caso si osserva che, una volta esauriti rapidamente i posti liberi nel buffer, il produttore comincia a scartare eventi di scrittura. Inoltre per simulare una condizione "a regime" possiamo inserire dei token anche nel posto *busy* della rete (si immagina quindi di osservare la situazione in un momento in cui sono già presenti in partenza dati nel buffer): in questo caso si nota che gli eventi di perdita sono alquanto limitati, se i ritmi di produttore e consumatore sono equilibrati tra di loro.

Deadlock



Le transizioni sono state poste in modalità “immediata” (ritardi pari a zero) per simulare la rete senza temporizzazioni.

Come si può vedere la chiarezza grafica di SIMPres non è all'altezza del tool precedentemente analizzato (PIPE2). Abbiamo dovuto disporre posti e transizioni in modo più confusionario per rendere leggibili le loro etichette. La mancanza degli archi ricurvi ci ha obbligato a intersecare le parti del diagramma. Tuttavia la simulazione ha dato gli stessi risultati dell'esperimento precedente: ripetendo svariate volte l'esecuzione delle operazioni, prima o poi si è verificata la condizione di blocco dovuta al conflitto di risorse.

PeT

PeT (Petri net Tool) è un applicativo sviluppato in Java da un gruppo di docenti e studenti del Politecnico di Milano, guidati dal professor Luca Ferrarini (<http://www.elet.polimi.it/upload/ferrarin/pet.html>). Questo tool, completamente free e in continua evoluzione, permette di analizzare a priori le reti che si progettano e di identificare subito eventuali problemi. Il tool consiste in un editor grafico per progettare Reti di Petri, più un insieme di plugins per una efficiente analisi strutturale della rete.

Capacità del tool

Le analisi possibili sono parecchie e sono espandibili grazie al sistema di plugin.

Le potenzialità di questo tool sono innumerevoli e permettono di fornire in modo esaustivo le proprietà notevoli delle reti modellate.

Tra i plugin più significativi riportiamo:

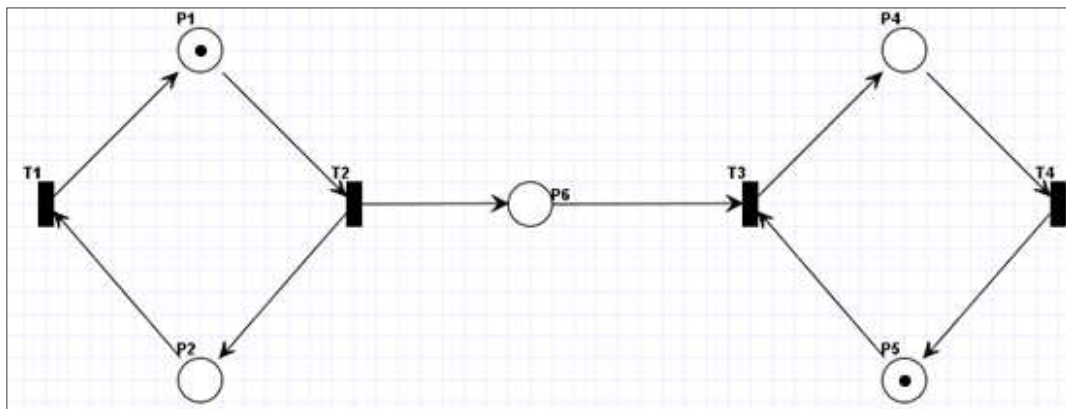
- Calcolo della rappresentazione matriciale della rete;
- Analisi comportamentale della rete (proprietà di limitatezza, copertura, “liveness”, reversibilità e calcolo dell'albero di raggiungibilità)
- Classificazione della rete (scelta libera, macchina a stati, grafo marcato);
- Calcolo del supervisore basato su invarianti;
- Controllo sui sifoni e analisi strutturale di trappole e invarianti

Inoltre è possibile generare una rappresentazione grafica di una rete di petri inserendo i dati relativi alle matrici di incidenza e a quelle di input/output.

I risultati dell'analisi possono essere salvati, ed in seguito esportati come immagini bitmap, come file per MatLab (file.m per la matrice delle equazioni della rete), oppure possono essere visualizzati sullo schermo.

Comportamento del tool nei modelli proposti

Producer-Consumer, unbounded buffer

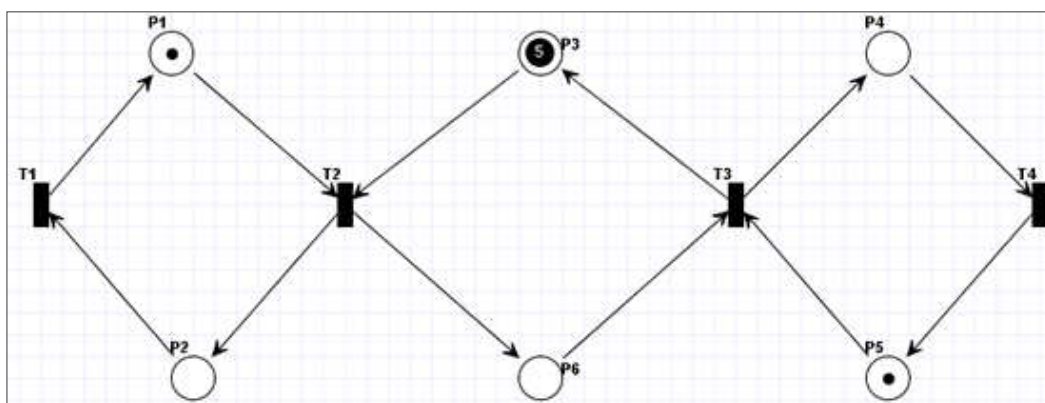


Eseguendo un controllo sulle caratteristiche generali della rete, il tool ha fornito le seguenti risposte:

- **Bounded** No
- **Reversible** Yes
- **Live** Yes
- **Strictly Conservative** No

I seguenti risultati sono in linea con quanto ci si aspetta

Producer-Consumer, bounded buffer



A differenza della rete precedente, il controllo sulle caratteristiche ci mostra le differenti proprietà di questo buffer così modellato:

- **Bounded** Yes
- **Reversible** Yes

- **Live** Yes
- **Strictly Conservative** Yes

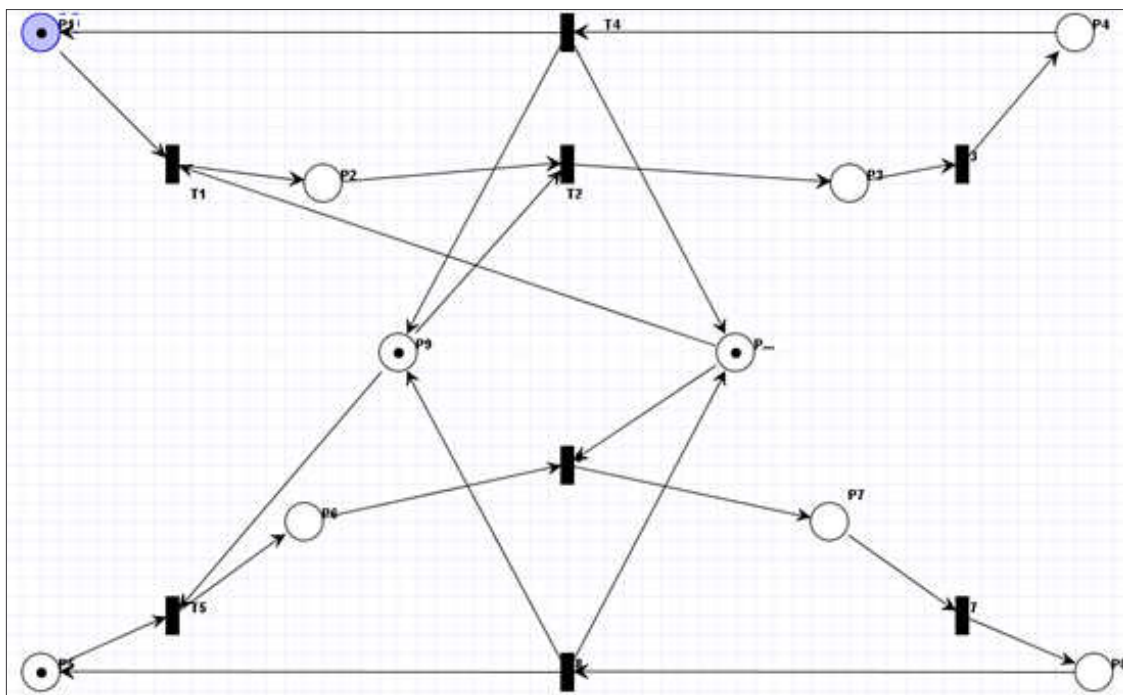
In questo caso abbiamo riscontrato un'anomalia con i risultati forniti dal plugin di analisi strutturale: esso infatti afferma che la rete è conservativa ma non strettamente. In questo caso è evidente che la rete è strettamente conservativa quindi il risultato fornito dal plugin è errato.

In più, abbiamo provato a simulare una rete semplice con due posti e due transizioni (catena circolare) e un token nel primo posto. Questa rete banale è sicuramente strettamente conservativa (infatti il numero di token rimane costante in qualsiasi stato la rete si trovi). Anche in questo caso il plugin di analisi strutturale ha fornito un risultato errato (mentre quello di analisi riassuntiva delle proprietà fornisce un risultato corretto)

Producer-Consumer, bounded buffer (timeout)

Con questo tool, come nel caso di PIPE2, non è possibile modellare la temporizzazione richiesta da questo esperimento, che non viene quindi realizzato.

Deadlock



Utilizzando l'analisi riassuntiva come nel caso precedente, si ottengono i seguenti risultati:

- **Bounded Yes**
- **Reversible Yes**
- **Live No**
- **Strictly Conservative Yes**

HPSim

HPSim, reperibile su <http://www.winpesim.de/> , è un tool gratuito che permette di realizzare le seguenti tipologie di reti:

- Place/Transition Nets
- Stochastic Petri Nets
- Petri Nets with Time

Il tool è disponibile solamente su piattaforme Windows in quanto è realizzato con Microsoft Visual Studio.

Capacità del tool

Questo tool è ottimo per il primo approccio a questi argomenti, permette di creare immediatamente reti e di eseguirne in modo anche gradevole la simulazione. Il limite è che, oltre all'editor grafico ed alla simulazione del *Token Game*, non permette un'analisi maggiore delle caratteristiche delle Reti di Petri. Infatti il menu simulation permette solamente una simulazione (anche temporizzata o step-by-step) ma nulla più.

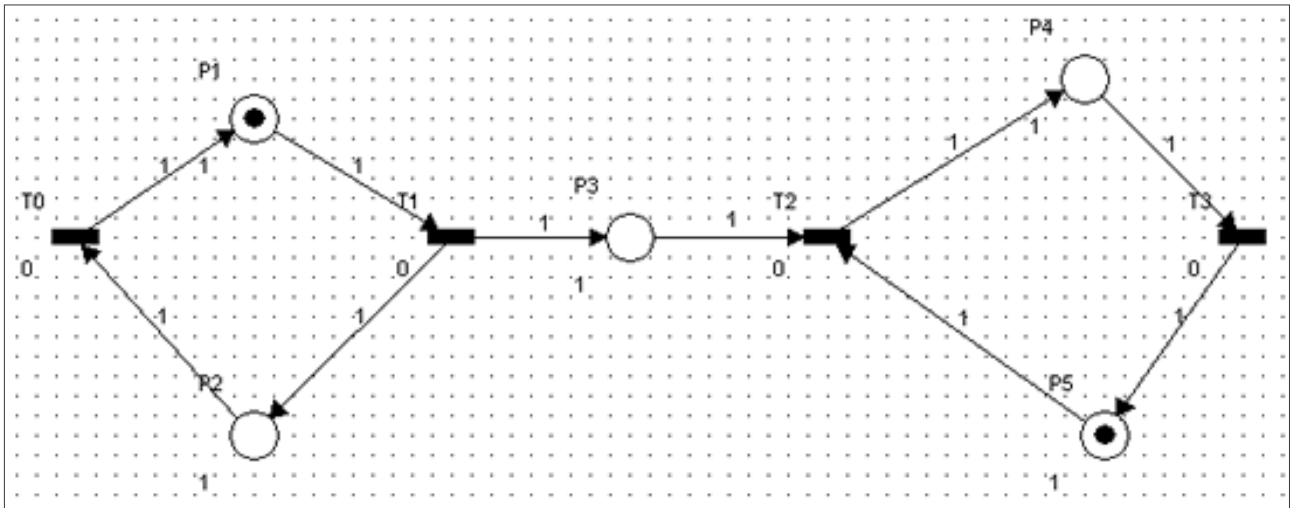
In modalità simulazione in verde vengono evidenziati le transizioni attivate, mentre in giallo quelle abilitate; dopo il passaggio su una transizione attivata (verde) si ha lo stato finale e quindi la nuova marcatura, mentre a monte della transizione abbiamo la marcatura relativa allo stato precedente.

A differenza degli altri tool abbiamo ulteriori possibilità offerte per la modellizzazione: infatti si possono utilizzare:

- archi inibitori
- posti con capacità finita
- funzioni di distribuzione temporale per le transizioni

Comportamento del tool nei modelli proposti

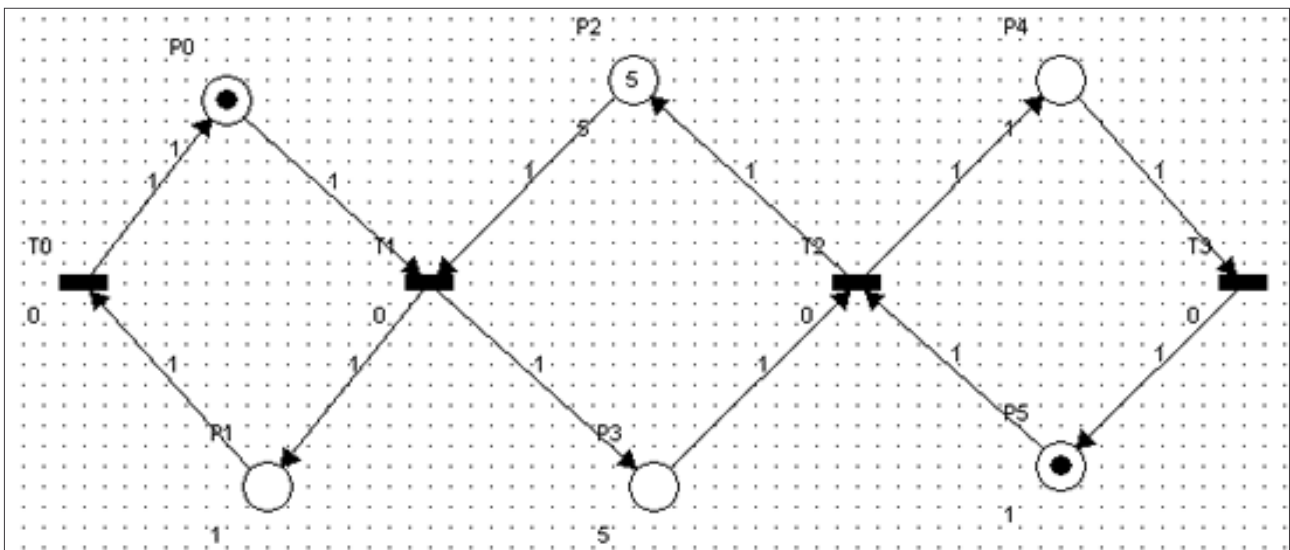
Producer-Consumer, unbounded buffer



Come è stato già detto in precedenza il tool non supporta alcuna analisi comportamentale della rete: tuttavia il comportamento visualizzato è conforme alle aspettative. La simulazione ci ha mostrato che ad ogni step vengono sparate tutte le transizioni abilitate (e non solamente una a caso tra esse)

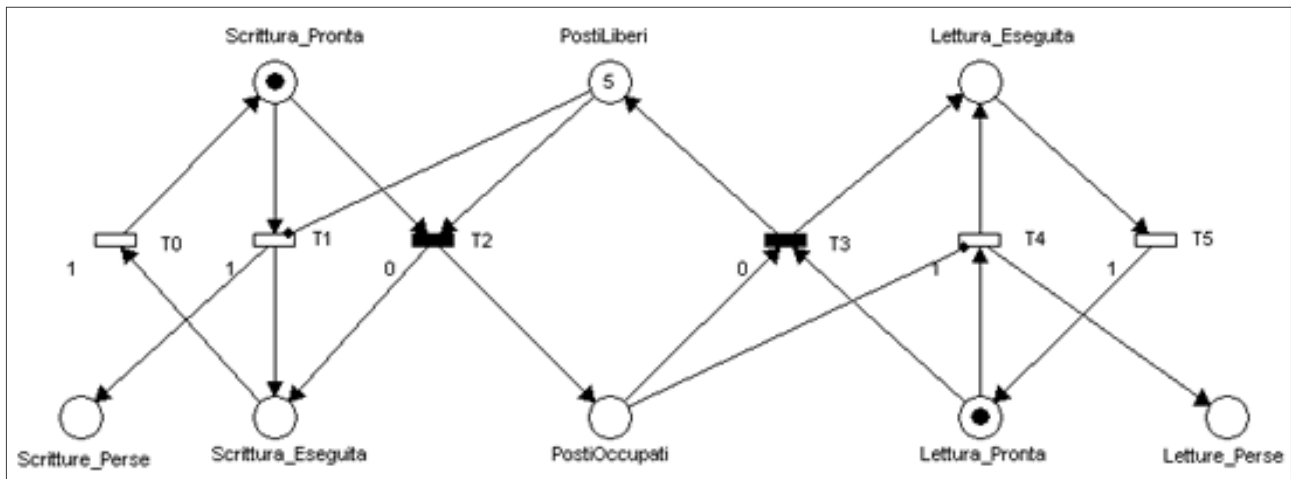
Per riprodurre un comportamento non deterministico, abbiamo impostato le funzioni di distribuzione delle transizioni come esponenziali. In questo modo non si ha sempre lo stesso ordine di sparo ad ogni ciclo (cosa che avviene lasciando le transizioni con sparo immediato)

Producer-Consumer, bounded buffer



Anche in questo esperimento il comportamento mostrato dal modello è conforme a quanto ci si aspetta. Essendo il tool più specifico per reti temporizzate non si possono effettuare analisi di rilievo su un modello non temporizzato come il presente.

Producer-Consumer, bounded buffer (timeout)



Questo esperimento mostra le potenzialità di questo tool, in quanto esso è specificatamente progettato per simulare reti di petri temporizzate. A differenza di Simpres, questo tool permette la creazione di file di output contenente l'andamento temporale delle marcature. Il formato .cvs semplifica l'importazione dei risultati in un foglio di calcolo oppure in Matlab.

Il modello da noi proposto è simile a quello realizzato in Simpres. La struttura è praticamente identica, tranne che per l'aggiunta di archi inibitori per impedire lo sparo delle transizioni di timeout quando sono disponibili posti scrivibili nel buffer oppure elementi da leggere.

Le transizioni sono state tipizzate in questo modo:

- **T2 , T3:** sono transizioni immediate, e sparano non appena abilitate
- **T1, T4:** sono le transizioni di timeout, e hanno caratteristica deterministica con ritardo pari a un tempo t .
- **T0, T5:** sono le transizioni che generano nuove potenziali letture e scritture. Per modellare la casualità di questi eventi, esse sono state impostate con distribuzione esponenziale di media 1: questo per provare un modello leggermente differente rispetto a quello utilizzato in SimPRES, dove si è usata la distribuzione uniforme).

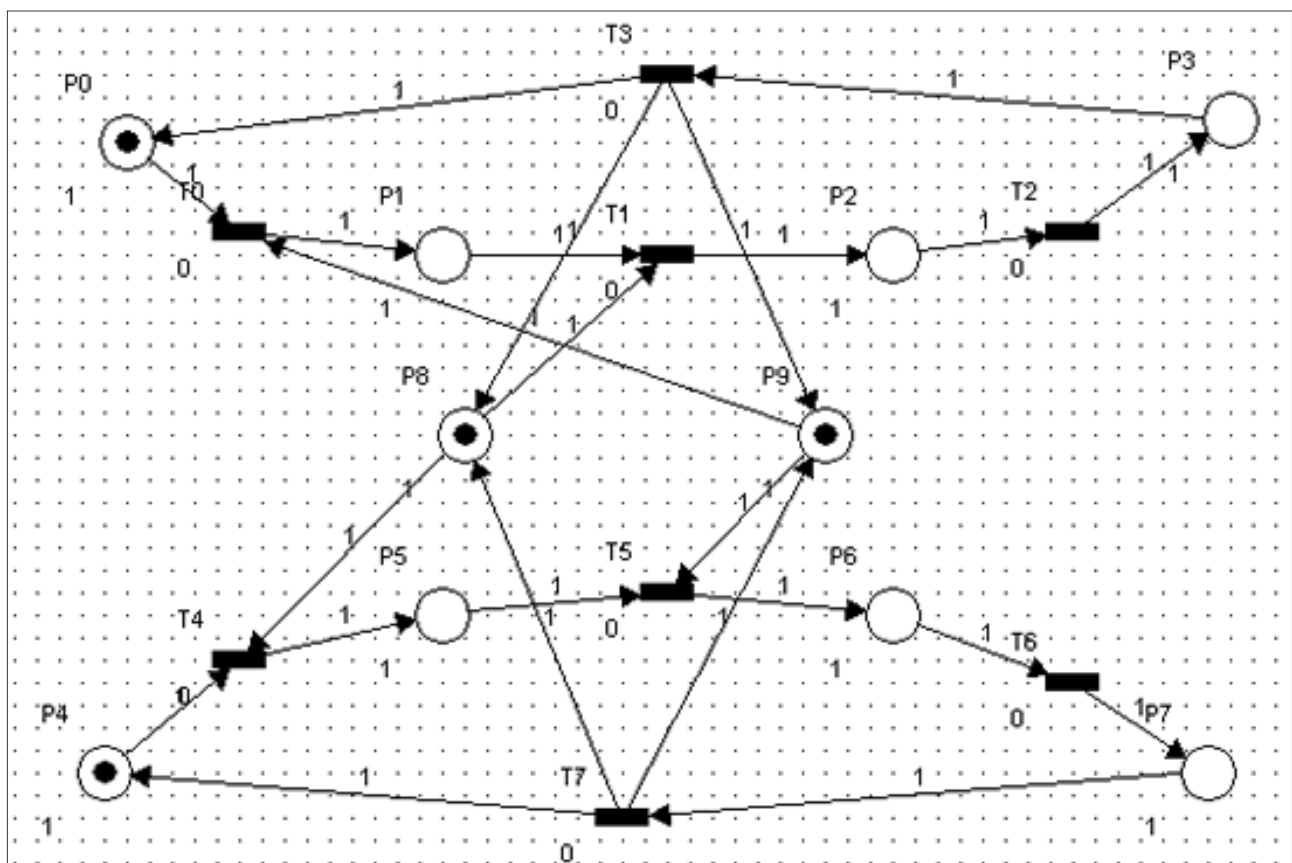
Gli esperimenti sono stati realizzati su un tempo di simulazione pari a 300ms e tempo di step pari a 1 ms. Nella sottostante tabella vengono riportati i risultati raggiunti con le parametrizzazioni utilizzate:

Eventi di perdita con ritardo di timeout costante (t=1ms) e capacità buffer variabile

<i>Dimensione Buffer</i>	<i>Scritture perse / ms</i>	<i>Letture perse / ms</i>
5	0.08	0.14
10	0.03	0.09
20	0.007	0.07

Come si può notare all'aumentare delle dimensioni del buffer gli eventi di perdita diminuiscono. Il fatto che le perdite in lettura siano più consistenti è dovuto alla fase di inizializzazione del buffer, che inizialmente è vuoto e quindi non offre molti elementi da leggere.

Deadlock



Lasciando le transizioni in modalità “immediata”, la simulazione notifica che avviene una condizione di errore (precisamente di Deadlock) allo step 2, ovvero quando sono scattate in sequenza le transizioni T0 e T4.

Per introdurre casualità e rendere quindi più realistica la simulazione, è possibile settare le transizioni con distribuzione di probabilità esponenziale di media k . In questo caso si nota che, dopo un numero variabile di cicli, prima o poi i due cicli di lavorazione si ritrovano nella condizione di deadlock precedentemente riportata.

Conclusioni

In generale si è osservato che i tool rappresentanti reti di Petri non temporizzate hanno a disposizione un più vasto numero di plugin di analisi e di classificazione. Al contrario, i tool che rappresentano una classe di reti più ampia (comprendente anche quelle temporizzate e/o stocastiche) sono generalmente meno ricchi di strumenti di analisi: infatti si incentrano maggiormente sull'evoluzione del token game e sull'ispezione diretta della rete in un dato punto della sua storia.

Inoltre si è constatato che nella maggior parte dei casi i tool salvano la rappresentazione della rete in un formato proprietario: questo ha obbligato a riprogettare i modelli per ogni tool utilizzato. E' quindi evidente che la mancanza di standardizzazione nei formalismi di descrizione causa gravi difficoltà quando si rende necessario utilizzare diversi applicativi nell'ambito di un qualsiasi progetto. Da questo punto di vista si menziona il tool PIPE2 che utilizza uno standard in formato XML (PNML) in grado di migliorare notevolmente la portabilità dei modelli.