Dynamic Slicing Techniques for Petri Nets

M. Llorens, J. Oliver, J. Silva, S. Tamarit, G. Vidal

Dep. of Information Systems and Computation (DSIC)
Technical University of Valencia (UPV)

4th International Workshop on
Programming Language Interference and Dependence
(PLID 2008)

Program Slicing

- Method for decomposing programs in order to extract parts of them, called program *slices*, which are of interest.
- Mark Weiser (1979).
- Program debugging $\Rightarrow$ For isolating the program statements that may contain a bug.
- In general, slicing extracts the statements that may affect some point of interest, referred to as *slicing criterion*.
- Classified into two classes:
  - *Static*: if the input of the program is unknown.
  - *Dynamic*: if a particular input for the program is provided.

Program Slicing

Example

```
(1)   read(n) ;
(2)   i := 1 ;
(3)   sum := 0 ;
(4)   product := 1 ;
(5)   while i <= n do
         begin
(6)         sum := sum + i ;
(7)         product := product * i ;
(8)         i := i + 1 ;
         end ;
(9)   write (sum) ;
(10)  write (product) ;
```

(a) Example program.

```
read(n) ;
i := 1 ;

product := 1 ;
while i <= n do
   begin

      product := product * i ;
      i := i + 1 ;
   end ;

write (product) ;
```

(b) Slice of this program w.r.t. the slicing criterion (10,product).

Petri Net

- Graphic, mathematical tool used to model and verify the behavior of systems that are concurrent, asynchronous, distributed, parallel, non-deterministic and/or stochastic.
- Verification $\Rightarrow$ Methods explore the state space, as the *reachability graph*.
- State explosion problem $\Rightarrow$ various approaches have been proposed to minimize the number of system states to be studied.

Static Slicing in PN

- C.K. Chang and H. Wang. A slicing algorithm of concurrency modelling based on Petri nets. In *Proc. of the 1986 Int. Conf. on Parallel Processing*, IEEE Computer Society Press, pp. 789-792, 1987.
- W.J. Lee, S.D. Cha, Y.R. Kwon, and H.N. Kim. A Slicing-based Approach to Enhance Petri Net Reachability Analysis. In *Journal of Research and Practice in Information Technology*, vol. 32(2), pp. 131-143, 2000.
- A. Rakow. Slicing Petri Nets. Technical Report, Department für Informatik, Carl von Ossietzky Universität, Oldenburg, 2007.
- A. Rakow. Slicing Petri Nets with an Application to Workflow Verification. In *Proc. of the 34th Conf. on Current Trends in Theory and Practice of Computer Science (SOFSEM 2008)* , Springer LNCS 4910, pp. 436-447, 2008.

Algorithm of Rakow'07

- The slicing criterion for $\mathcal{N} = (P, T, F)$ is $\langle Q \rangle$ where $Q \subseteq P$ is a set of places.
- It computes all the parts of the Petri net which could transmit tokens to the slicing criterion Q.
  1. The marking of a place p depends on its input and output transitions,
  2. a transition may only be fired if it is enabled, and
  3. the enabling of a transition depends on the marking of its input places.
- The slice $\mathcal{N}' = (P', T', F')$ is a subnet of $\mathcal{N}$ that includes all input places of all transitions connected to any place p in P', starting with $Q \subseteq P'$.
- Limitation $\Rightarrow$ Big slices due to no initial marking nor firing sequence were considered.

Dynamic Slicing in PN

- *Proposal 1*: It extends the slicing criterion in Rakow'07 in order to also consider an initial marking.
- *Proposal 2*: It reduces the size of the computed slice by only considering a particular execution, a sequence of transition firings.

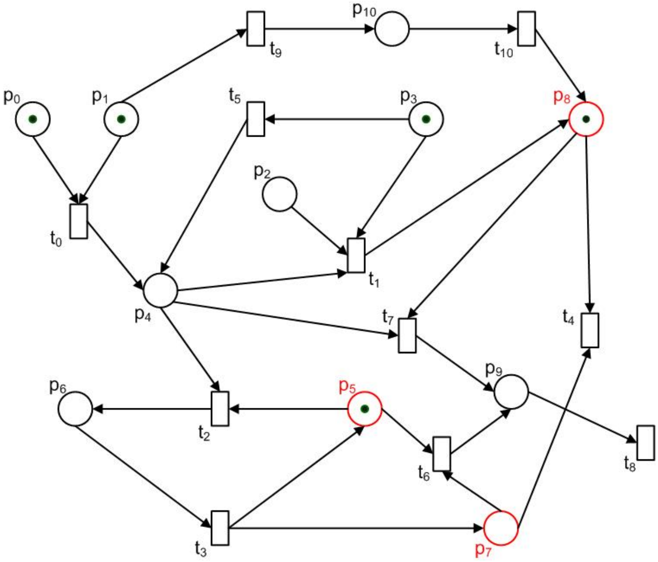
Motivation

- Dynamic $\Rightarrow$ Initial marking $\Rightarrow$ Debugging
- In a particular trace of a marked Petri net, an erroneous state is reached.
- Extracting the set of places and transitions that may erroneously contribute tokens to the places of interest.

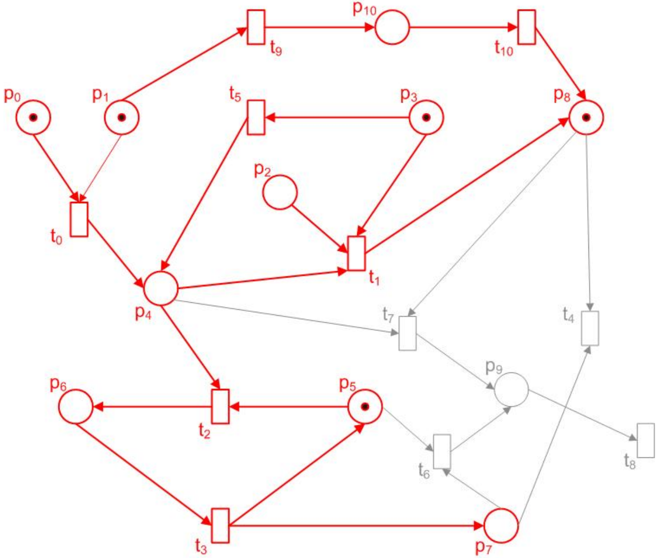
Intuitive idea

- The slicing criterion for $\mathcal{N} = (P, T, F)$ is a pair $\langle M_0, Q \rangle$ where M_0 is an initial marking for $\mathcal{N}$ and $Q \subseteq P$ is a set of places.
- The slice $\mathcal{N}' = (P', T', F')$ is a subnet with those places and transitions of $\mathcal{N}$ which can contribute to change the marking of Q in any execution starting in M_0.
- How?
  1. We first compute the possible paths which lead to the slicing criterion (*backward slice*).
  2. We also compute the paths that may be followed by the tokens of the initial marking (*forward slice*).
  3. The result is the intersection of backward and forward slices.

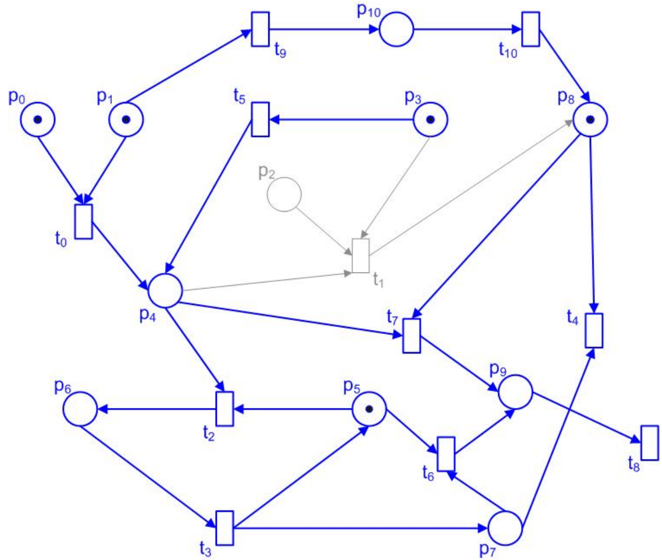
- Initial PN $(\mathcal{N}, M_0)$ and slicing criterion $\langle M_0, \{p_5, p_7, p_8\}\rangle$



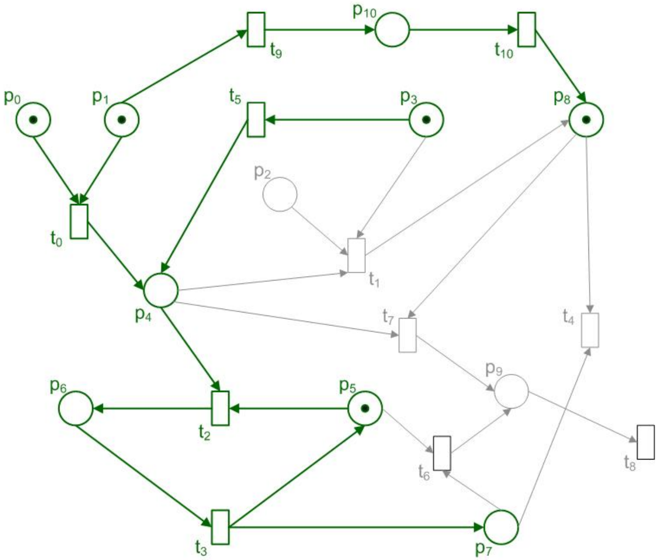
- Backward slice $(\mathcal{N}_1, M_0|_{P_1})$



- Forward slice $(\mathcal{N}_2, M_0|_{P_2})$

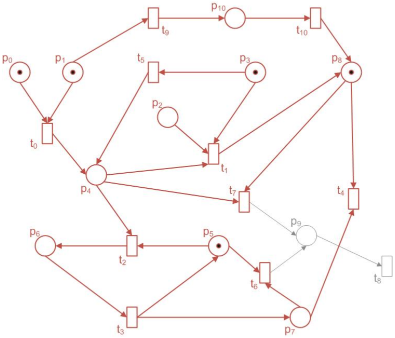


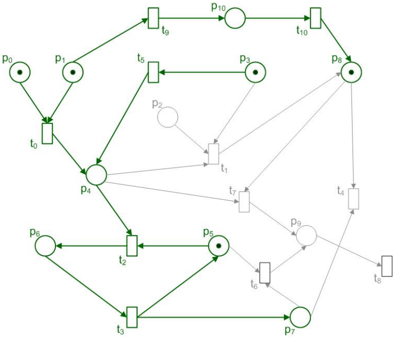
- Intersection: Final result of Algorithm 1



Comparison

- Rakow's slice — Our slice
- $\langle \{p_5, p_7, p_8\} \rangle$ — $\langle M_0, \{p_5, p_7, p_8\} \rangle$





Comparison

- Rakow's algorithm computes all the parts of the Petri net which could transmit tokens to $\langle Q \rangle$.
- We compute all the parts of the Petri net which could transmit tokens to Q from M_0.
- Our technique is more general than Rakow's technique $\Rightarrow$ the Rakow's slice w.r.t. $\langle Q \rangle$ is the same as our slice w.r.t. $\langle M_0, Q \rangle$ if $M_0(p) > 0$ for all $p \in P$.
- But it keeps its simplicity and efficiency.
- Its cost is bounded by the number of transitions T of the original Petri net, $\mathcal{O}(2T)$.

Motivation

- We refine the notion of slicing criterion including the firing sequence that represents the erroneous simulation.
- The slice is more precise than the one produced by previous proposal.

Intuitive idea

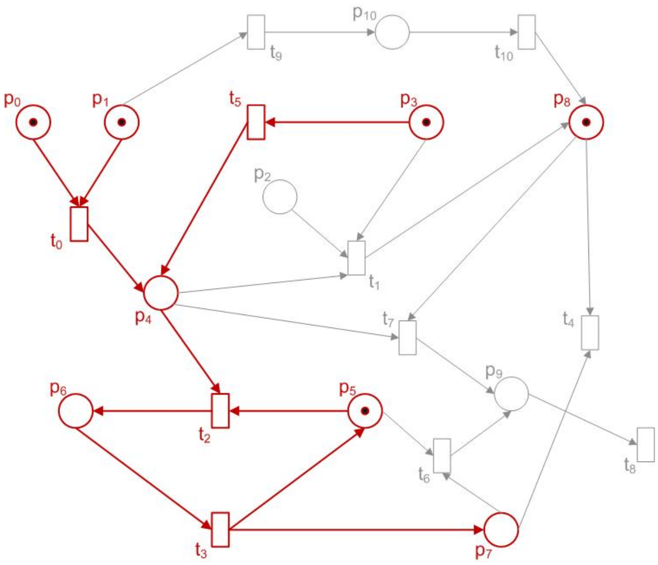
- The slicing criterion for $\mathcal{N} = (P, T, F)$ is $\langle M_0, \sigma, Q \rangle$.
- The slice $\mathcal{N}' = (P', T', F')$ is a subnet with those places and transitions of $\mathcal{N}$ which are necessary to move tokens to the places in Q, following $\sigma = t_1 t_2 \ldots t_n$ $(M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \ldots M_{i-1} \xrightarrow{t_i} M_i \ldots \xrightarrow{t_n} M_n)$.
- How?
  - Function slice(M_n, Q) is called, where M_n is a marking reachable from M_0 through σ.
  - $\mathrm{slice}(M_i, W) =$

$$\begin{cases} W & \text{if } i = 0 \\ \mathrm{slice}(M_{i-1}, W) & \text{if } \forall p \in W.\ M_{i-1}(p) \geq M_i(p),\ i > 0 \\ \{t_i\} \cup \mathrm{slice}(M_{i-1}, W \cup {}^{\bullet}t_i) & \text{if } \exists p \in W.\ M_{i-1}(p) < M_i(p),\ i > 0 \end{cases}$$

- Slicing criterion $\langle M_0, \sigma = t_5 t_2 t_3 t_0 t_2 t_3, \{p_5, p_7, p_8\}\rangle$

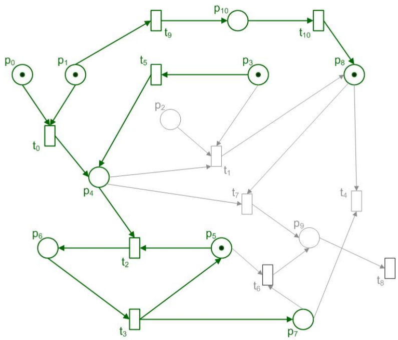
	p_0	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
M_0	1	1	0	1	0	1	0	0	1	0	0
t_5	1	1	0	0	1	1	0	0	1	0	0
t_2	1	1	0	0	0	0	1	0	1	0	0
t_3	1	1	0	0	0	1	0	1	1	0	0
t_0	0	0	0	0	1	1	0	1	1	0	0
t_2	0	0	0	0	0	0	1	1	1	0	0
t_3	0	0	0	0	0	1	0	2	1	0	0

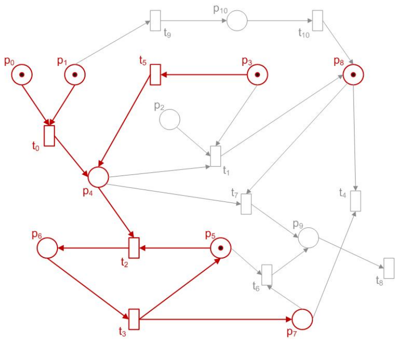
- Slice result of Algorithm 2



Comparison

- First slice — Second slice
- $\langle M_0, \{p_5, p_7, p_8\}\rangle$ — $\langle M_0, \sigma = t_5 t_2 t_3 t_0 t_2 t_3, \{p_5, p_7, p_8\}\rangle$





Conclusions

- We have introduced two different techniques for dynamic slicing of Petri nets.
- To the best of our knowledge, this is the first approach to dynamic slicing for Petri nets.
- The first proposal takes into account the Petri net and an initial marking, but produces a slice w.r.t. any possibly firing sequence.
- The second proposal further reduces the computed slice by fixing a particular firing sequence.
- In general, our slices are smaller than previous (static) approaches where no initial marking nor firing sequence were considered.

Future work

- We plan to carry on an experimental evaluation of our slicing techniques in order to test its viability in practice.
- We also find it useful to extend our slicing techniques to other kind of Petri nets (e.g., coloured Petri nets [Jensen97] and marked-controlled reconfigurable nets [Llorens and Oliver04]).