PERFORMANCE MODELLING WITH TIME-AUGMENTED PETRI NETS

A *time-augmented Petri net* model [Coolahan & Roussopoulos, 1983] uses transitions to model the instantaneous event of the starting or stopping of a process, and places represent the condition of the process in execution, and so they are assigned a non-negative time value T_i (to place p_i).

There are several other timed Petri net models (e.g. the *extended timed Petri net* where transitions model processes with an assigned time representing the execution time of the process) but the time-augmented Petri net model will be used here.

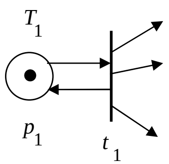
The following rules govern transition firing behaviour:

1. A token *only* becomes available to aid in enabling an output transition of place p_i after T_i time units have elapsed since p_i first received the token.
2. If one transition is enabled after a token becomes ready, then this transition *immediately* fires.
3. If multiple transitions become enabled when the token becomes ready, then one transition fires immediately (non-deterministic choice), and the remainder become disabled.

The Time-driven Petri net model

In this approach, a master timing mechanism, which controls repetitive activity cycles, is assumed, and is modelled by a PN construction with a periodic *driving cycle*. This PN construction has the following properties:

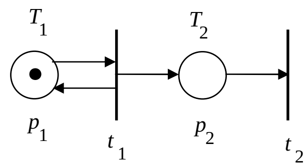
- $\mu(p_1)$ is reproduced with period T_1
- $I(t_1) = p_1$
- $p_1 \in O(t_1)$ and $|O(t_1)| > 1$
- $I(p_1) = O(p_1) = \{t_1\}$



The remainder of the PN model is formed by adding places and transitions so that:

1. Each place has a fixed positive execution time (when modelling a process) or zero execution time (when modelling an event).
2. The bipartite structure of the PN is preserved.
3. Every place and transition has a directed path from the driving cycle.
4. All paths terminate at transitions representing system outputs.

Example



The PN notion of *safeness* is violated without timing information, i.e. after the first firing of t_1, both t_1 and t_2 are enabled simultaneously $\rightarrow$ t_1 could fire again and safeness is violated.

To retain *safeness in the presence of time*, it is necessary to specify $T_2 \leq T_1$, i.e. provided transition t_2 fires before, or fires with, transition t_1, the PN is *safe*. In addition to safety, another concept is useful to construct the timing model of the system.

Relative Firing Frequency

The firing frequency of a transition with respect to the driving cycle plays an important role in subsequent net analysis.

Because of the restriction applied to PN construction in the above model, the ratio of *firing frequency* of a transition relative to that of the driving cycle, is inversely proportional to the token *interarrival time* at a place.

Define the following:

1. *Maximum Relative Firing Frequency (MRFF)*: the number of transition firings of a transition for each firing of the driving cycle (with firing priority going to the transition).
2. *Minimum Token Interarrival Time* (*MTIAT*): the shortest possible time between the arrivals of consecutive tokens at a place.

For each place p_j with input transition t_j, the following relation holds:

$$MTIAT(p_i) = \frac{T_1}{MRFF(t_j)}$$

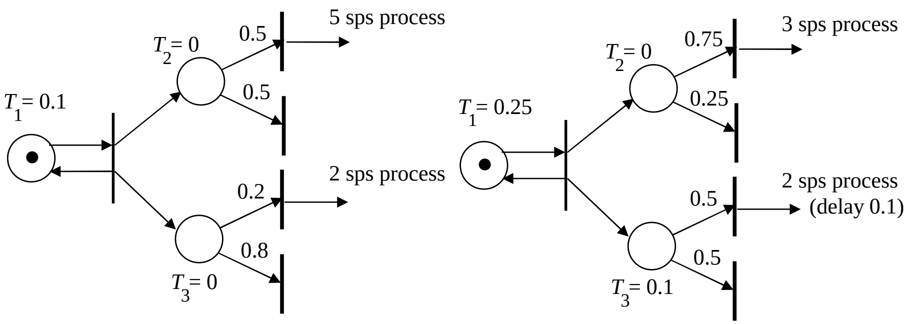
where T_1 is the driving cycle time.

The *MRFF* can be found directly from PN consistency tests, but additional information is required where a decision (or multiple-output) place is found, there are two decision classes:

- *predetermined* - distribution of output path frequency is known and controllable → express as a path ratio.
- *data-dependent* - how often each path is taken is not known as it is dependent on the data (and/or external parameters) → bound.

The predetermined decision class can be useful to allow a single driving cycle to synchronize several processes operating at different time rates:

Example



Subclasses of Time-Driven Systems

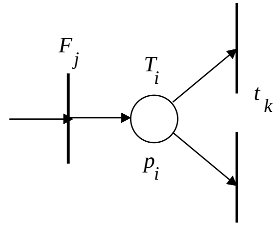
Although the analysis of arbitrary Petri nets is possible, the approach taken here is to restrict the form of constructions to defined subclasses to ease the subsequent analysis phase. Four subclasses are defined:

- *Asynchronous systems*: these are the basic components - they have no internal cycles and satisfy the restrictions:
  $|I(p_i)| = 1, |O(p_i)| \geq 1, \forall t_k \in O(p_i) : |I(t_k)| = 1$

  For safety in the presence of time we have:

  $$T_i \leq MTIAT(p_i) = T_1/F_j$$

  where F_j is the *MRFF* of the input transition to this *simple* place.

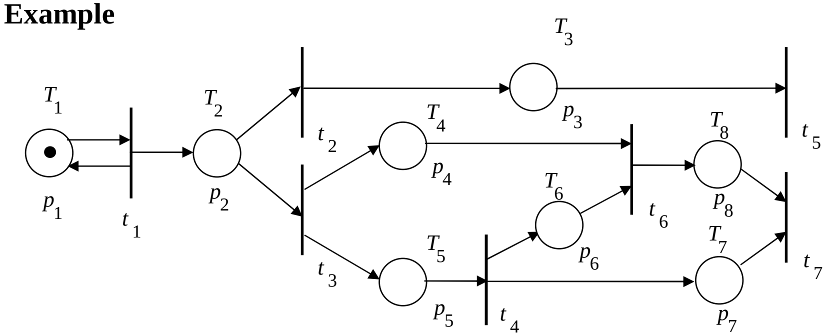


- *Synchronized systems*: as above but including 'synchronised parallel path constructions' not containing cycles. An additional timing constraint is imposed, which is specified as a *waiting time* at a final place for synchronization with a number of possible parallel paths.

  The additional constraint is:

  For any parallel path set, the sum of execution times and the waiting time at that place for any of the paths, *must not exceed* the *MTIAT* of that place.

Example



There are two overlapping synchronised parallel path constructions above. Note that p_4 above is not *safe in the presence of time* if the execution times for p_5 and p_6 exceeds the execution time of the driving cycle divided by the *MRFF* of t_3, i.e:

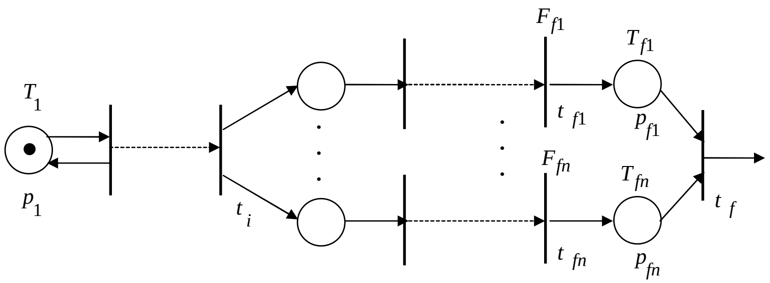
$$MTIAT(p_4) = T_1/MRFF(t_3) < T_5 + T_6 \Rightarrow p_4 \text{ is unsafe}$$

and similarly:

$$MTIAT(p_7) = T_1/MRFF(t_4) < T_6 + T_8 + W_6 \Rightarrow p_7 \text{ is unsafe}$$

where W_6 is the waiting time to synchronize at t_6 for p_6.

This constraint can be generalized :



So for each final place (p_{fi} where i =1, . . . , n), safety in the presence of time is achieved only if:

$$P_j - (P_i - T_{fi}) \leq T_1/F_{fi} \qquad \forall j = 1, . . . , n \quad i \neq j$$

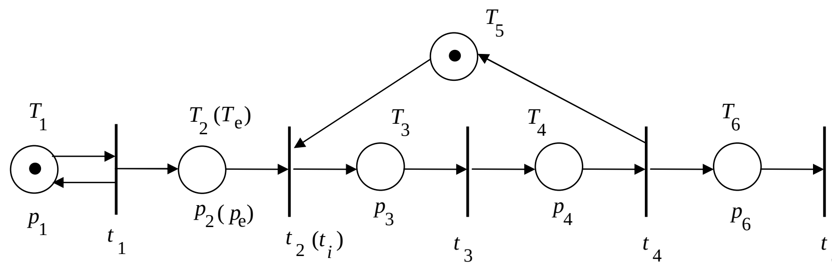
where: P_i is the total path time for path i (execution and waiting),
P_j is the total path time for path j (execution and waiting),

- *Independent cycle systems*: these systems allow all the previous constructions and include cyclic paths subject to the constraint:

  For any independent cycle, the cycle execution time (say from t_i to t_i, where t_i is the cycle transition after the entry place p_e) must *not* exceed the *MTIAT* of the entry place (p_e).

Example

Suppose in the diagram below that the computation in the cycle $p_3 \rightarrow p_4 \rightarrow p_5$ represents a running computation of the standard deviation (say) of the last n values in a database. Process p_3 accumulates the statistics of the new data point with the last (n-1) data points, process p_4 computes the new standard deviation and process p_5 removes the nth oldest point's statistics from the running accumulation (and then allows t_2 to accept a new input value). The output is stored by process p_6:



In the above example, if:

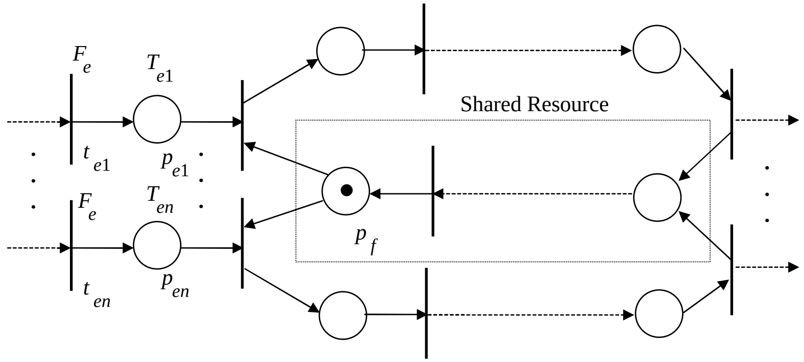
$$MTIAT(p_2) < T_3 + T_4 + T_5$$

then p_2 is unsafe in the presence of time.

This can be generalized to safeness in the presence of time requiring that $MTIAT(p_e) \geq T_c$ where T_c is the cycle time of all places in the independent cycle. The entry place (p_e) must also satisfy the condition for safeness of a simple place, i.e. $MTIAT(p_e) \geq T_e$

- *Shared resource systems*: these systems are modelled by the sharing of a path in otherwise non-intersecting independent cycles (as just defined). The shared path is the common shared resource, which is modelled as at least an entry place to the shared path and a final place in the shared path (which also represents the control mechanism for mutual exclusion of the resource).

Example



The token in the final place of the shared resource can only enable *one* output transition at a time. The constraint that is imposed is:

Only one token at a time can take control of the shared resource to prevent token 'starvation'. The *MTIAT* for each entry place (p_{ej}) is assumed to be the same (and hence the *MRFF* is the same), i.e:

$$T_{ej} + \sum_{k=1, k \neq j}^{n} T_{ck} \leq MTIAT(p_{ej}) = T_1/F_e \qquad \forall j = 1, \ldots, n$$

where T_{ej} is the execution time of place p_{ej}
and T_{ck} is the execution time of all places in the cycle of the shared resource construction cycle activated by p_{ej}.

Net Construction Methodology

The basis for the construction of an analysable PN model is to use these basic building blocks in the construction of the system model. The PN produced in this manner should be *consistent*, i.e. have a bounded number of tokens to correspond to any real system of interest.

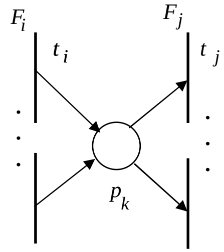
To evaluate the consistency of the PN model, a variable representing the *MRFF* is assigned to each transition, and an input-output balance is applied at every place in the system.

The procedure is:

a) Assign an F_i to each transition t_i

b) For each place p_k form the balance equation:

$$\sum_{i=1}^{n} F_i = \sum_{j=1}^{m} F_j$$



where $t_i \in I(p_k)$, $t_j \in O(p_j)$, $|I(p_k)| = n$, $|O(p_k)| = m$

c) Starting with places which are outputs of the driving cycle transition, the following operations are performed:
   i) For a single output transition place, find the *MRFF* by solving the place's equation for F_j as a function of F_1.
   ii) For a multiple output transition place the decision ratios can either be predetermined or data dependent:
      - for predetermined decisions - the *MRFF* of each output transition is given by: $F_j = R_j \sum_{i=1}^{n} F_i$
      - for data-dependent decisions - the MRFF of each output transition is given be: $F_j = \sum_{i=1}^{n} F_i$ (so that a 'worst-case' analysis results)

d) Each place is evaluated as above, its output transitions are marked as solved, and the process iterates until no place remains unsolved.

Once the *MRFF*'s have been evaluated by the above method, the safeness properties can be applied to obtain the PN time constraints.

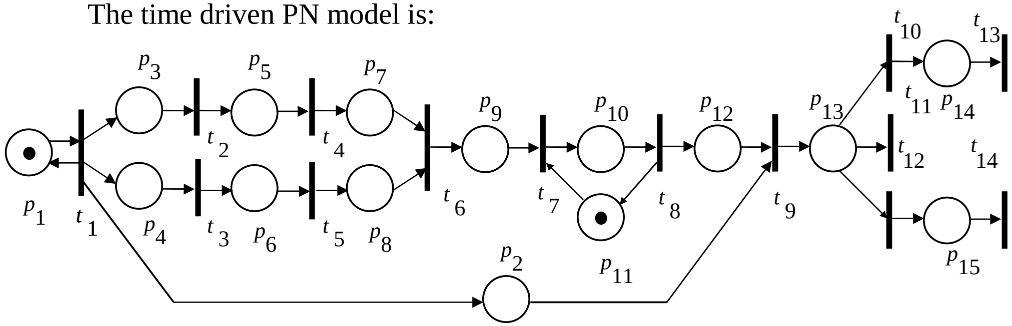
Example Application

A small laboratory process control system maintains a salt water bath to test oceanographic instruments at a constant operator setable salinity. At regular intervals, a temperature and conductivity sensor are sampled by two identical A/D converters. The samples are range checked, then converted to temperature and conductivity. A salinity value is calculated from these two values, which are fed into a several sample mean and standard deviation calculation. Old data points are removed from the accumulation, and the new mean and standard deviation are recorded.

The mean value is compared to the operator input value, and if it is within specified acceptable limits, no action is taken. Otherwise:

- if the salinity is too low, a small quantity of water is drained from the bath while a saline solution is added.
- if the salinity is too high, a small quantity of water is drained and fresh water is added.

The time driven PN model is:



With the following mapping of processes to places:

p_1 - master timing process	p_2 - sample operator value
p_3 - sample temperature A/D	p_4 - sample conductivity A/D
p_5 - temperature range check	p_6 - conductivity range check
p_7 - unit conversion	p_8 - unit conversion
p_9 - salinity computation	p_{10} - mean and standard deviation
p_{11} - old data point removal	p_{12} - record new mean + std dev
p_{13} - compare mean to selection	p_{14} - saline solution injector
p_{15} - fresh water injector	

To determine if the time constraints can be met, the PN model is checked for safeness in the presence of time:

1. Determine all the transition *MRFF*'s:
   - Let F_1 =1 time unit, and apply the consistency balance to all places: $F_9 = 1, F_2 = 1, F_3 = 1 \rightarrow F_4 = 1, F_5 = 1 \rightarrow F_6 = 1$ $\rightarrow F_7 = 1 \rightarrow F_8 = 1 \rightarrow F_9 = 1$ which is consistent.
   - Also $F_{10} + F_{11} + F_{12} = F_9 = 1$ which is a data dependent decision so select worst case with: $F_{10} = F_{11} = F_{12} = F_{13} = F_{14} = 1$ so all *MRFF*'s are set to 1.
2. The safeness criteria is applied:
   - p_9 is an entry place for an independent cycle of p_{10} and p_{11}, i.e.

   $T_c \leq T_1/F_e \rightarrow T_{10} + T_{11} \leq T_1/F_6 = T_1$ as $F_6 = 1$

   and $T_e \leq T_1/F_e \rightarrow T_9 \leq T_1$

   - p_7 and p_8 are final places in a parallel synchronised path p_3, p_5, p_7 and p_4, p_6, p_8 i.e.

   $$P_j - (P_i - T_{fi}) \leq T_1/F_{fi} \qquad \forall j = 1, 2, i \neq j$$
   $$\rightarrow T_4 + T_6 + T_8 - (T_3 + T_5) \leq T_1$$
   $$T_3 + T_5 + T_7 - (T_4 + T_6) \leq T_1$$

   - p_2 and p_{12} are final places in parallel synchronised paths p_2 and $p_3, p_5, p_7, p_9, p_{10}, p_{12}$; and p_2 and $p_4, p_6, p_8, p_9, p_{10}, p_{12}$;

   $$\rightarrow T_2 - (T_3 + T_5 + T_7 + T_9 + T_{10}) \leq T_1$$
   $$T_3 + T_5 + T_7 + T_9 + T_{10} + T_{12} \leq T_1$$
   $$T_2 - (T_4 + T_6 + T_8 + T_9 + T_{10}) \leq T_1$$
   $$T_4 + T_6 + T_8 + T_9 + T_{10} + T_{12} \leq T_1$$

   - $p_3, p_4, p_5, p_6, p_{10}, p_{13}, p_{14}, p_{15}$ are all simple places as they satisfy $|I(p_i)| = 1, |O(p_i)| \geq 1, \forall t_k \in O(p_i) : |I(t_k)| = 1$

   $$\rightarrow T_i \leq MTIAT(p_i) = T_1/F_j$$

   i.e. $T_3 \leq T_1, T_4 \leq T_1, T_5 \leq T_1, T_6 \leq T_1, T_{10} \leq T_1, T_{13} \leq T_1,$ $T_{14} \leq T_1, T_{15} \leq T_1$

Provided all the above constraints can be met, the system specification is satisfiable.