Synthesis and Analysis of Product-form Petri Nets

Serge Haddad[(1)], Jean Mairesse[(2)], Hoang-Thach Nguyen[(2)]

(1) LSV CNRS & ENS Cachan & INRIA Saclay, (2) LIAFA CNRS & Université Paris 7

Petri Nets 2011
Mefosyloma seminar, the 4th May 2012

Plan

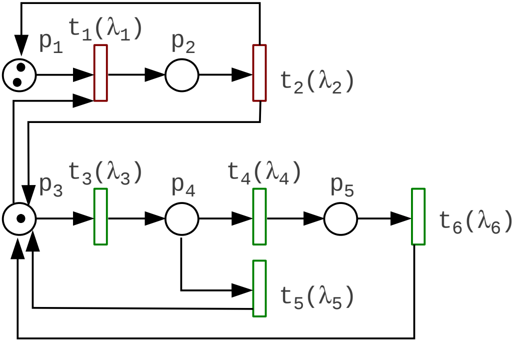
1. Context and motivation
2. Synthesis of product-form Petri nets
3. Complexity of product-form Petri net problems
4. A new subclass of product-form Petri nets

Plan

Stochastic Petri Net: Syntax

A stochastic Petri net (SPN) is defined by:

- a Petri net
- an exponential distribution per transition t with rate λ_t

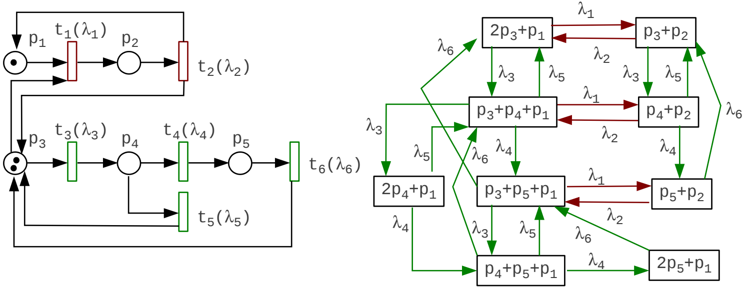


Stochastic Petri net: Semantic

The stochastic process associated with a SPN is:

- a continuous time Markov chain (CTMC)
- *isomorphic* to the reachability graph of the Petri net
- with infinitesimal generator

$$Q[m,m'] = \sum_{m \xrightarrow{t} m'} \lambda_t \text{ and } Q[m,m] = -\sum_{m' \neq m} Q[m,m']$$



Quantitative Analysis of SPN

Two kinds of analysis

- Transient behaviour
- Steady-state behaviour

Steady-state distribution

Assume that the marking process is ergodic.
(i.e. the reachability graph has a single terminal component)

Then the *steady-state distribution* is the unique probability distribution π_∞ that fulfills:

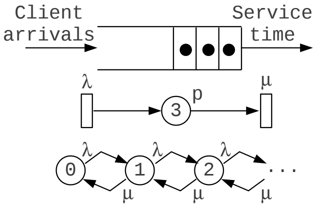
$$\pi_\infty \cdot Q = 0$$

Drawback: The numerical resolution suffers from the state space explosion.

Alternative: Look for models for which the steady-state distribution can be expressed by an analytical formula *(called product-form)*.

Product-form for Queues

A single queue

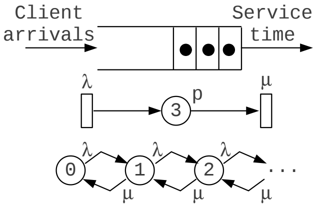


Let $\rho = \lambda/\mu$ be the utilisation factor.
If $\rho < 1$ then:

$$\pi_\infty(n) = \rho^n(1-\rho) \text{ where } n = m(p)$$

Product-form for Queues

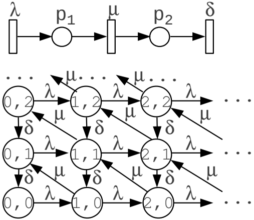
A single queue



Let $\rho = \lambda/\mu$ be the utilisation factor.
If $\rho < 1$ then:

$$\pi_\infty(n) = \rho^n(1-\rho) \text{ where } n = m(p)$$

Queues in tandem

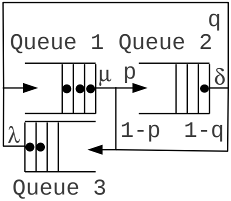


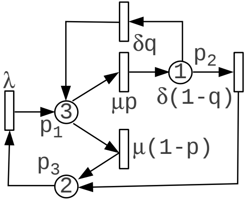
Let $\rho_1 = \lambda/\mu$ and $\rho_2 = \lambda/\delta$.
If $\rho_1 < 1$ and $\rho_2 < 1$ then:

$$\pi_\infty(n_1, n_2) = \rho_1^{n_1}(1-\rho_1)\rho_2^{n_2}(1-\rho_2)$$

where $n_i = m(p_i)$

Product-form for Closed Queueing Networks





Visit ratios of queues i: v_i (up to a constant)

Visit ratios fulfill: $v_1 = v_3 + qv_2$, $v_2 = pv_1$ and $v_3 = (1-p)v_1 + (1-q)v_2$

Thus: $v_1 = 1$, $v_2 = p$ and $v_3 = 1 - pq$.

Define $\rho_1 = \frac{v_1}{\mu}$, $\rho_2 = \frac{v_2}{\delta}$, $\rho_3 = \frac{v_3}{\lambda}$ and $n_i = m(p_i)$

- $\pi_\infty(n_1, n_2, n_3) = \frac{1}{G}\rho_1^{n_1}\rho_2^{n_2}\rho_3^{n_3}$ (with $n_1 + n_2 + n_3 = n$)
- where G is the *normalising constant*.

Computation of the Normalising Constant

G the normalising constant is computed by dynamic programming.

Let $G(k,s)$ be the normalising constant corresponding to k clients in the s first queues. Then:

- $\forall k \leq n \; G(k,0) = 0$,
- $\forall s \leq |P| \; G(0,s) = 1$,
- $\forall 0 < k \leq n \; \forall 0 < s \leq |P|$ *(decomposition w.r.t $n_s \left[\substack{>\\=}\right] 0$)*

$$G(k,s) = \sum_{n_1+\cdots+n_s=k} \prod_{0<i\leq s} \rho_i^{n_i}$$

$$= \rho_s \left(\sum_{n_1+\cdots+n_s=k-1} \prod_{0<i\leq s} \rho_i^{n_i} \right) + \sum_{n_1+\cdots+n_{s-1}=k} \prod_{0<i\leq s-1} \rho_i^{n_i}$$

$$= \rho_s G(k-1,s) + G(k,s-1)$$

Computation Time: $\Theta(np)$ versus $\Theta(n^{p-1})$

Specific Product-form for SPNs

First general product-form SPNs

- a structural subclass: Π-nets
- with an additional numerical condition on rates

Product Form Equilibrium Distributions
and a Convolution Algorithm for Stochastic Petri Nets
J.L. Coleman,W. Henderson, P.G. Taylor, Performance Evaluation, 1996

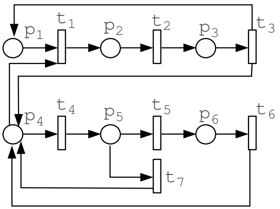
First purely structural product-form SPNs: Π^2-nets

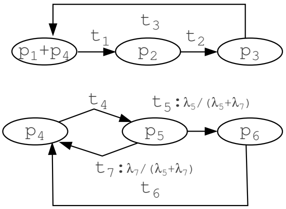
Product-Form and Stochastic Petri Nets: a Structural Approach
S. Haddad, P. Moreaux, M. Sereno, M. Silva, Performance Evaluation, 2005

Equivalence between Π^2-nets and some product-form biological systems

Deficiency Zero Petri Nets and Product Form
J. Mairesse, H-T. Nguyen, Fundamenta Informaticae, 2010

Π-nets: the Resource Graph



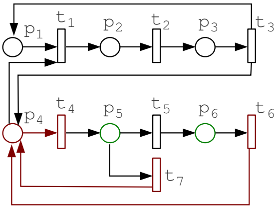


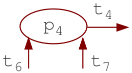
The resource graph

- The vertices are the input and the ouput bags of the transitions.
- Every transition of the net t yields a graph transition ${}^\bullet t \xrightarrow{t} t^\bullet$
- Client classes correspond to the connected components of the graph *(which can be also seen as DTMCs)*.

A Π-net is a net such that:
the connected components of the resource graph are strongly connected.

Π^2-nets: Witnesses





Vector $-p_5-p_6$ is a witness for bag p_4:

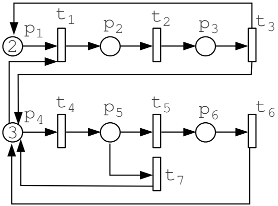
$(-p_5-p_6).W(t_6)=(-p_5-p_6).W(t_7)=1$
$(-p_5-p_6).W(t_4)=-1$
$(-p_5-p_6).W(t)=0$ for every other t

Witness for a bag b ($W(t)$ is the incidence of t)

- Let $In(b)$ (resp. $Out(b)$) the transitions with input (resp. output) b.
- Let v be a place vector, v is a *witness* for b if:
  - $\forall t \in In(b)\ v \cdot W(t) = -1$
  - $\forall t \in Out(b)\ v \cdot W(t) = 1$
  - $\forall t \notin In(b) \cup Out(b)\ v \cdot W(t) = 0$

A Π^2-net is a Π-net such that: every bag has a witness.

Steady-State Distribution of a Π^2-net



The reachability space:

m(p$_1$)+m(p$_2$)+m(p$_3$)=2

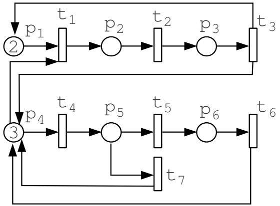
m(p$_2$)+m(p$_3$)+m(p$_4$)+m(p$_5$)+m(p$_6$)=3

Steady-state distribution

- Let $w(b)$ the witness for bag b.
- Compute the ratio visit of bags $v(b)$ on the resource graph.
- The output rate of a bag b is $\mu(b) = \sum_{t|\bullet t=b} \lambda_t$
- Then: $\pi_\infty(m) = \frac{1}{G} \prod_b \left(\frac{v(b)}{\mu(b)}\right)^{w(b)\cdot m}$

The product form can be rewritten as $\pi_\infty(m) = \frac{1}{G} \prod_{p \in P} \rho_p^{m(p)}$

The Normalising Constant of a Π^2-net



The reachability space:

m(p1)+m(p2)+m(p3)=2

m(p2)+m(p3)+m(p4)+m(p5)+m(p6)=3

The normalising constant can be efficiently computed if the reachability space is characterized by $Am = b$.

- $m_p = \min(\left\lfloor \frac{b[r]}{A[r,p]} \right\rfloor \mid A[r,p] > 0)$ is an upper bound of $m(p)$.
- $G(b, P) = \sum_{0 \leq i \leq m_p} \rho_p^i G(b_i, P \setminus \{p\})$ where $b_i[r] = b[r] - iA[r,p]$
- If $b = \vec{0}$ then $G(b, \emptyset) = 1$ else $G(b, \emptyset) = 0$

All known subclasses of Π^2-nets fulfilling this property are queueing systems!

Issues for Π^2-nets

Open problems

- Modelling with Π^2-nets
  Synthesis of Π^2-nets
- Qualitative analysis of Π^2-nets
  Complexity of standard problems (reachability, coverability, etc.)
- Quantitative analysis
  Handling the normalising constant without building the reachability graph

Our contributions

- Design of a sound and complete set of rules for synthetising every Π^2-net
- Characterization of complexity for standard problems
- Definition of a large subclass of Π^2-nets
  with an efficient computation of the normalising constant

Plan

Three Sound and Complete Synthesis Rules for Π^2-nets

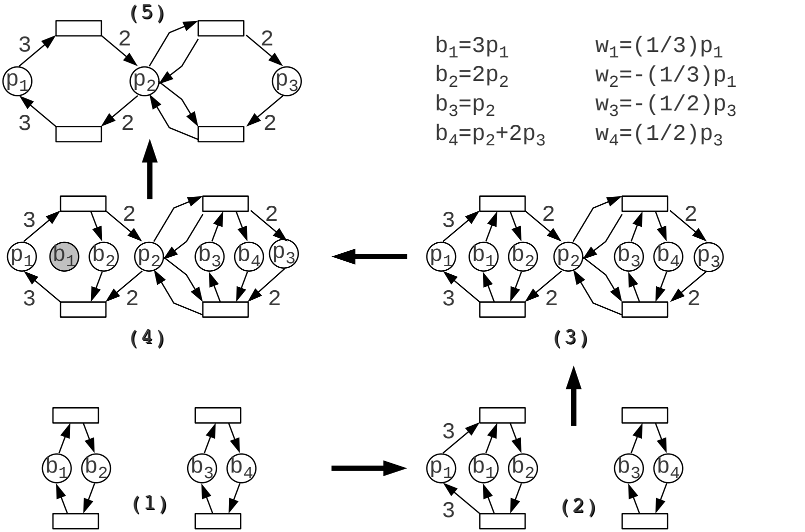
First rule. Add a disjoint strongly connected state machine.

Second rule. Delete an isolated place.

Third rule. Substitute to an input/output bag b the bag $b + kp$ $(b(p) + k \geq 0)$ with the following requirements:

- $b + kp$ is not already an input/output bag
- Every bag has a witness whose support does not contain p *(decidable in polynomial time)*

Synthesis process example



$$b_1=3p_1 \qquad w_1=(1/3)p_1$$
$$b_2=2p_2 \qquad w_2=-(1/3)p_1$$
$$b_3=p_2 \qquad w_3=-(1/2)p_3$$
$$b_4=p_2+2p_3 \qquad w_4=(1/2)p_3$$

Plan

Complexity results

In safe Petri nets

Previous results

- The reachability and liveness problems for safe Π-nets are PSPACE-complete.
- The reachability and liveness problems for safe Π^2-nets are NP-hard.

Our results

The reachability and liveness problems for safe Π^2-nets are PSPACE-complete.

Complexity results

In safe Petri nets

Previous results

- The reachability and liveness problems for safe Π-nets are PSPACE-complete.
- The reachability and liveness problems for safe Π^2-nets are NP-hard.

Our results

The reachability and liveness problems for safe Π^2-nets are PSPACE-complete.

In general Petri nets

Previous results

The reachability and coverability problems for Π-nets are EXPSPACE-complete.

Our results

The coverability problem for Π^2-nets is EXPSPACE-complete.

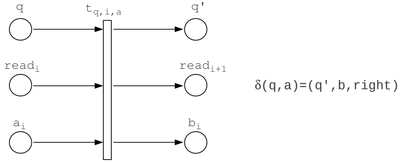
The Reachability Problem for Safe Petri nets

Reachability is in PSPACE.

Incrementally guess a firing sequence of length less than $2^{|P|}$

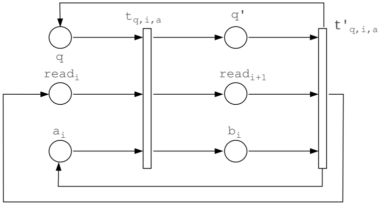
Reachability is PSPACE-hard. Given a Turing machine $\mathcal{M} = (Q, \Sigma, q_0, q_f, \delta)$ operating in space $p(n)$ and a word w of length n, build a net net with:

- Places q for $q \in Q$, places $read_i$ for $1 \leq i \leq p(n)$ and places a_i for $a \in \Sigma$ and $1 \leq i \leq p(n)$;
- Transitions $t_{q,i,a}$ for $q \in \Sigma$, $1 \leq i \leq p(n)$ and $a \in \Sigma$ whose forward incidence depend on δ;
- The initial marking is $q_0 + \sum_{i=1}^{n} w_i + \sum_{i=n+1}^{p(n)} blank_i + read_1$ and the final marking is $q_f + \sum_{i=1}^{p(n)} blank_i + read_1$.



The Reachability Problem for Safe Π-nets

In order to get a safe Π-net, one adds reverse transitions.



The reduction is still valid.

However generally the net is not a Π^2-net.

The Reachability Problem for Safe Π^2-nets

PSPACE-Hardness is proved by reduction of the QBF satisfiability problem of:

$$\varphi \equiv \forall x_n \exists y_n \forall x_{n-1} \exists y_{n-1} \ldots \forall x_1 \exists y_1 \psi \text{ (with } \psi \text{ in CNF)}$$

Principle of the reduction.

- A subnet modelling two $n+3$-bit (from 0 to $n+2$) counters to enumerate all tuples $(x_n, x_{n-1}, \ldots, x_1)$.
- For all $1 \leq i \leq n$, a subnet enabling to choose values for $(y_i, \ldots, y_1)$ each time x_i is changed.
- A subnet checking the truth of ψ including additional synchronizations with the previous subnets.

The bit 0 forces the checking of ψ for every tuple $(x_n, x_{n-1}, \ldots, x_1)$.

The bit $n+1$ is used for the initial guess of $(y_n, \ldots, y_1)$.

The bit $n+2$ is used for resetting $(y_n, \ldots, y_1)$
in order to get a reachability problem.

A Π^2-net Modelling a Counter

$P = \{p_0, \ldots, p_{k-1}, q_0, \ldots, q_{k-1}\}$

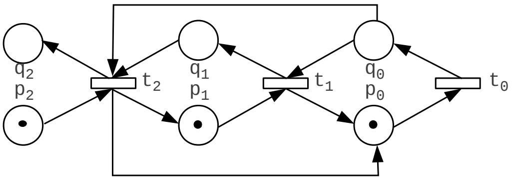
$T = \{t_0, \ldots, t_{k-1}, t_0^-, \ldots, t_{k-1}^-\}$

For all $0 \leq i < k$, ${}^\bullet t_i = p_i + \sum_{j<i} q_j$ and $t_i^\bullet = q_i + \sum_{j<i} p_j$

For all $0 \leq i < k$, t_i^- is the reverse transition of t_i

For all $0 \leq i < k$, $m_0(p_i) = 1$ and $m_0(q_i) = 0$

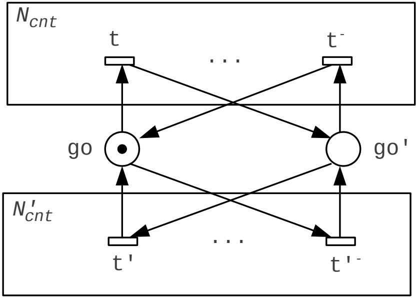
A 3-bit counter (without the reverse transitions)



The bag $p_i + \sum_{j<i} q_j$ (resp. $q_i + \sum_{j<i} p_j$) has for witness:

$$p_i + \sum_{j>i} 2^{j-i-1} p_j \text{ (resp. } q_i + \sum_{j>i} 2^{j-i-1} q_j)$$

Synchronizing Two Counters



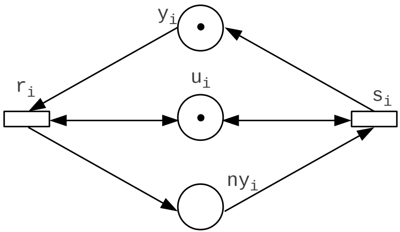
Places go and go' synchronize the two counters in such a way that:
either $count' = count$ or $count' = count - 1$.

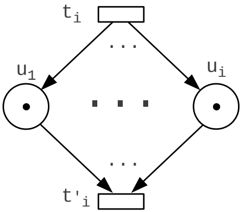
The bags are enlarged with place go or go'.

The witnesses are unchanged.

Handling the Existential Variables

Between the firing of t_i and t'_i $(i \geq 1)$ which updates $x_i, \ldots, x_1$, the following subnet allows to update $y_i, \ldots, y_1$.



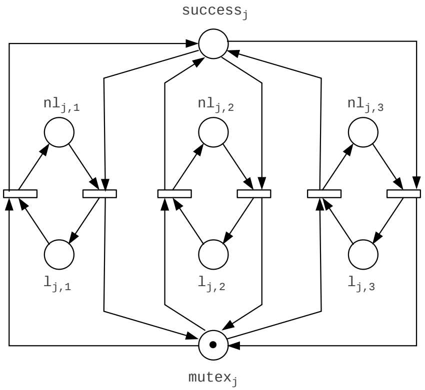


The bags of the counter subnet are enlarged with places u_i
and their witnesses remain the same.

The new bag $y_i + u_i$ (resp. $ny_i + u_i$) has for witness y_i (resp. ny_i).

Checking a Clause

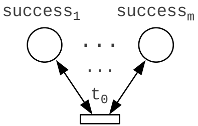
It consists to choose literal $l_{j,k}$ that proves the truth of clause C_j.



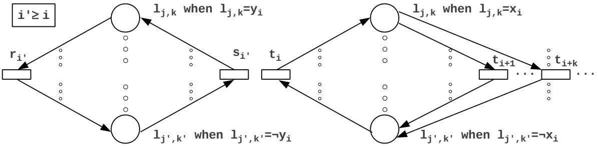
The bag $mutex_j + \ell_{j,k}$ (resp. $success_j + n\ell_{j,k}$)
has for witness $-n\ell_{j,k}$ (resp. $n\ell_{j,k}$).

Synchronizing the Clauses and the Literals

In order to set bit 0
of the first counter (transition t_0)
to 1, all clauses must be proved.



The literals are synchronized with the variables
(the initial marking is chosen accordingly).

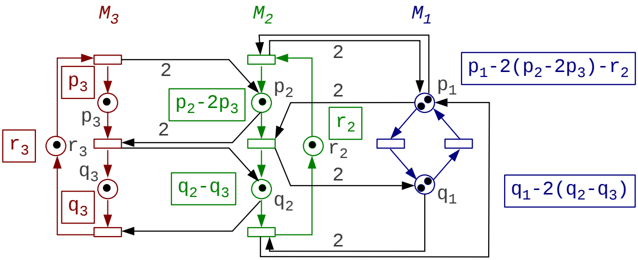


The bags are enlarged but the witnesses are unchanged.

When literal $l_{j,k}$ proves clause C_j (place $nl_{j,k}$ marked) the value of the corresponding variable cannot be changed.

Plan

Ordered Π-nets

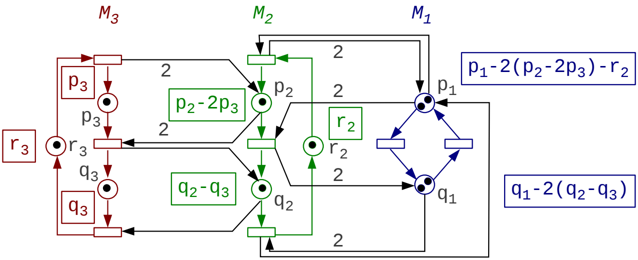


An ordered Π-net is a set of strongly connected state machines $\mathcal{M}_1, \ldots, \mathcal{M}_n$ with additional edges where:

- transitions of $\mathcal{M}_{i+1}$ are only connected to $P_{i+1} \cup P_i$ *(with at least an edge between $\mathcal{M}_{i+1}$ and $\mathcal{M}_i$)*.
- $\mathcal{M}_{i+1} \cup \mathcal{M}_i$ is a Π-net with resource graph isomorphic to the disjoint unions ot the two machines

There exists a witness for every bag which can be computed inductively from higher to lower levels. So ordered Π-nets are Π^2-nets.

Linear Invariants of Ordered Π-nets



In ordered Π-nets, there is a basis of linear invariants

- whose cardinality is n, the number of state machines,
- obtained by summing the witnesses per state machine.

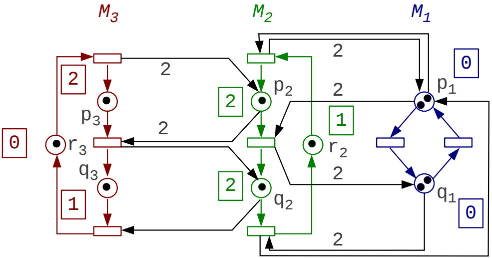
Example $(m(P_i) = \sum_{p \in P_i} m(p))$

$m(P_3) = 3$

$m(P_2) - (2m(p_3) + m(q_3)) = -3$

$m(P_1) - ((2m(p_2) + 2m(q_2) + m(r_2)) - (4m(p_3) + 2m(q_3))) = 10$

Π^3-nets: Potentials

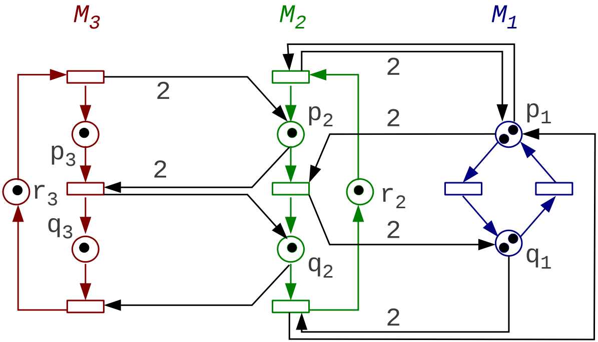


Potential of a place $p \in \mathcal{M}_{i+1}$

Let $t \in T_{i+1}$ such that $p \in t^\bullet$, then the potential of p is the number of tokens produced by t in $\mathcal{M}_i$ (i.e. $|t^\bullet \cap P_i|$).

A Π^3-net is an ordered Π-net where transitions of $\mathcal{M}_{i+1}$ are only connected to places of $\mathcal{M}_i$ with maximal potential.

Reachability in Π^3-nets: Spurious Markings



The set of markings fulfilling the invariants is a *superset* of reachable markings.

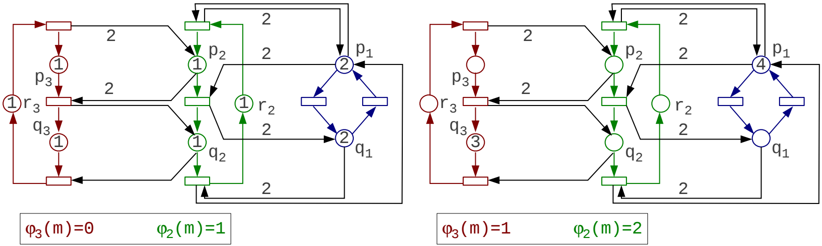
In nets, a *spurious* marking is an unreachable marking fulfilling the invariants.

Π^3-nets admit spurious markings (e.g. $m = 3q_3 + 4p_1$).

Liveness: a Polynomial Time Characterization (1)

A net is *live* if for all reachable marking m and transition t,
there exists a marking m_t reachable from m and where t is fireable.

The i-*minimal marked potential* of a marking m, $\varphi_i(m)$,
is the minimal potential of marked places of P_i.



Liveness: a Polynomial Time Characterization (2)

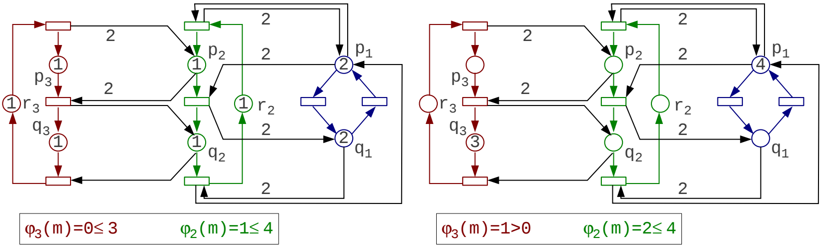
Theorem

Let $\mathcal{N}$ be a Π^3-net. Then a marking m is live if and only if:

$$\mathcal{M}_n \text{ is marked (i.e. } m(P_n) > 0\text{) and } \forall 1 < i \leq n \; m(P_{i-1}) \geq \varphi_i(m)$$

(Only if) $m(P_n) = 0 \Rightarrow \mathcal{M}_n$ dead. $\exists 1 < i \leq n \; m(P_{i-1}) < \varphi_i(m) \Rightarrow \mathcal{M}_i$ dead.

(If, sketch) In any $\mathcal{M}_i$, a token can be moved from one place to another place without moving the other tokens in $\bigcup_{j \geq i} \mathcal{M}_j$.



Reachability: a Polynomial Time Characterization

Theorem

Let $(\mathcal{N}, m_0)$ be a live Π^3-net. Then a marking m is reachable if and only if:
it fulfills the invariants and $(\mathcal{N}, m)$ is live

Sketch of Proof

Let m^* be a marking fulfilling the invariants
and where marked places of any $\mathcal{M}_i$ have maximal potential.

Then m^* is reachable from m_0 (using the proof of liveness).

Then any live m fulfilling the invariants is reachable from m_0
(using *weak reversibility* of Π-nets).

Recurrence Equations for the Normalising Constant (1)

Places of $P_i = \{p_{i1}, \ldots, p_{ik_i}\}$ are ordered by increasing potential.

v_i is the flow vector defining the linear invariant related to $\mathcal{M}_i$.

Let $E(i, j, c_1, \ldots, c_i)$ be the set of markings such that:

- $\mathcal{M}_n, \ldots, \mathcal{M}_{i+1}$ are unmarked and places $p_{i(j+1)}, \ldots, p_{ik_i}$ are unmarked.
- $\forall 1 \leq s \leq i \; v_s \cdot m = c_s$
- $c_i > 0$
- m is live in $\mathcal{M}_1 \cup \ldots \cup \mathcal{M}_i$

Then:

- If $a < c_i$ and $j \geq 2$ then $E(i, j, c_1, \ldots, c_i) \cap \{m | m(p_{ij}) = a\}$
  $= \{m + a p_{ij} \mid m \in E(i, j-1, c_1 - a v_1(p_{ij}), \ldots, c_i - a v_i(p_{ij}))\}$
- If $i > 1$ and $c_{i-1} > c_i v_{i-1}(p_{ij})$ then $E(i, j, c_1, \ldots, c_i) \cap \{m | m(p_{ij}) = c_i\}$
  $= \{m + c_i p_{ij} \mid m \in E(i-1, k_{i-1}, c_1 - c_i v_1(p_{ij}), \ldots, c_{i-1} - c_i v_{i-1}(p_{ij}))\}$
- If $i > 1$ and $c_{i-1} \leq c_i v_{i-1}(p_{ij})$ then $E(i, j, c_1, \ldots, c_i) \cap \{m | m(p_{ij}) = c_i\} = \emptyset$

Observation: This result is based on a behavioural analysis.

Recurrence Equations for the Normalising Constant (2)

Let us consider:

- the "relative" product-form distribution as $\pi_r(m) = \prod u_{ij}^{m(p_{ij})}$
- when $c_i > 0$, the normalising constant

$$G(i, j, c_1, \ldots, c_i) = \sum_{m \in E(i,j,c_1,\ldots,c_i)} \pi_r(m)$$

- when $c_i \leq 0$, $G(i, j, c_1, \ldots, c_i) = 0$

Then (for $i \geq 2$ and $j \geq 2$):

$$G(i, j, c_1, \ldots, c_i) = \sum_{\nu=0}^{c_i - 1} u_{ij}^{\nu} G(i, j-1, c_1 - \nu v_1(p_{ij}), \ldots, c_i - \nu)$$

$$+ u_{ij}^{c_i} G(i-1, k_{i-1}, c_1 - c_i v_1(p_{ij}), \ldots, c_{i-1} - c_i v_{i-1}(p_{ij}))$$

Observation:
The c_j's may be negative but their absolute value is bounded by some constant.

Conclusion and Perspectives

Contributions

- Synthesis rules for Π^2-nets
- Characterization of the complexity of qualitative problems for Π^2-nets
- Introduction of Π^3-nets with efficient qualitative and quantitative analysis

Perspectives

- Developing or extending a tool (like GreatSPN)
- Extending the set of rules and specializing them for Π^3-nets
- Proving EXPSPACE-completeness for reachability of Π^2-nets
- Handling unbounded nets