Composition of product-form Generalized Stochastic Petri Nets: a modular approach

Simonetta Balsamo and Andrea Marin

Università Ca' Foscari di Venezia
Dipartimento di Informatica
Italy

October 2009

Markov process: steady state analysis

- Steady-state analysis: analysis of the system (if possible) when $t \to \infty$
- Γ: state space
- q_{ij}: transition rate from states i to j, $i \neq j$, $i, j \in \Gamma$. Let $\mathbf{Q} = [q_{ij}]$ with $q_{ii} = -\sum_{j \neq i} q_{ij}$
- $\pi(i)$: probability of observing state i when $t \to \infty$ (limiting distribution), $\boldsymbol{\pi} = [\pi(i)]$

Theorem (Stationary distribution)

If the CTMC is ergodic the limiting distribution is unique and independent of the initial state. The stationary distribution is given by:

$$\boldsymbol{\pi}\mathbf{Q} = \mathbf{0} \ \wedge \ \boldsymbol{\pi}\mathbf{1} = 1$$

Generalized Stochastic Petri Nets (GSPN)

- Petri net based stochastic models
- Very expressive
    - PNs with inhibitor arcs are Turing-complete
- High modelling power
    - Allow for both immediate and timed transitions
    - Timed transitions have exponentially distributed delays
    - Allow for marking dependent semantics for the firing rates or conflict resolutions
    - Immediate transitions may use priority
- Well-defined underlying stochastic process
- The Semi-Markov process underlying a net can be reduced to a Markov process

Problems in the analysis of GSPNs

Problems. . .

- Determining the set of all the reachable states of the model (without inhibitor arcs) is an EXPSPACE problem
- Even small models may have huge state spaces (possibly infinite)
- Even when the number of states is finite, solving the GBE system may be computationally infeasible!

Solutions. . .

- using approximations (e.g. fluid approximations)
- using *Divide et Impera* approach
    - Analysis of small models and then compose them!

Compositionality and steady state analysis: product-form

- Consider model S consisting of sub-models $S_1, \dots, S_N$
- Let $m = (m_1, \dots, m_N)$ be a state of model S and $\pi(m)$ its steady state probability
- S is in product-form with respect to $S_1, \dots, S_N$ if:

$$\pi(m) \propto g_1(m_1) \cdot g_2(m_2) \cdots g_N(m_N)$$

  where $g_i(m_i)$ is the steady state probability distribution of S_i appropriately parametrised
- The cardinality of the state space of S is proportional to the product of the state space cardinalities of its sub-models $\Rightarrow$ product-form models can be studied more efficiently!

Identifying product-form solutions

We base our result on two main theoretical results on product-forms:

- The Markov implies Markov property ($M \Rightarrow M$)
- The Reversed Compound Agent Theorem (RCAT)

Peculiarities...

- They consider each sub-model in isolation
- These results are not specific for GSPNs
- They are not structural
- Computationally expensive to check conditions for sub-models with large (infinite) state spaces

Goal

The goal of this work is to define a framework in which the modeller. . .

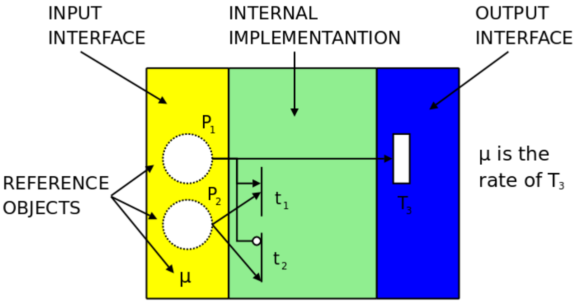
- has a library of models that are known to be in product-form under a set of conditions
- composes the GSPN models in a simple graphical way
- has not sophisticated skills in solving Markov processes

By using the theoretical results, it is possible to. . .

- efficiently decide if the conditions for the product-form are satisfied
- compute the stationary distributions by solving the model traffic equations

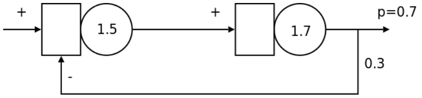
GSPN modules

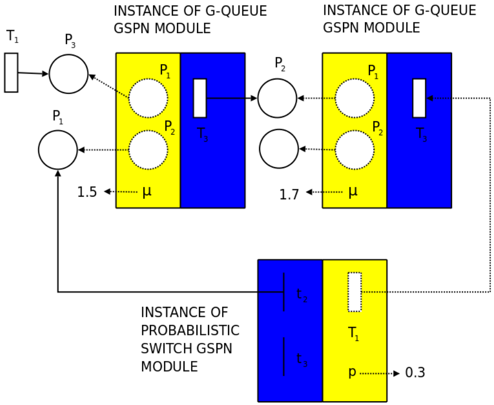
What is a GSPN module? (Kindler et al. approach)



- Scope of the names is local
- Input objects must be instantiated
- Places, transitions, and symbols can be imported/exported

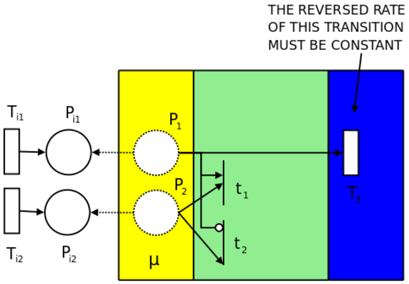
Example of GSPN module usage




INSTANCE OF G-QUEUE GSPN MODULE
INSTANCE OF G-QUEUE GSPN MODULE
T_1
P_3
P_1
P_2
T_3
P_1
1.5
μ
P_2
P_1
P_2
T_3
1.7
μ
INSTANCE OF PROBABILISTIC SWITCH GSPN MODULE
t_2
t_3
T_1
p
0.3

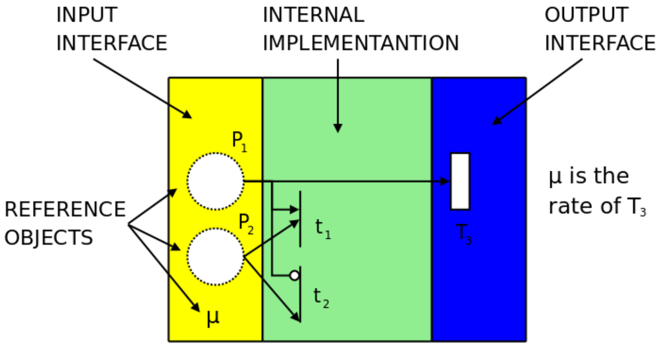
Embedding product-form information in module description

Consider an *isolated instance* of a module with independent Poisson token arrivals to each input place:



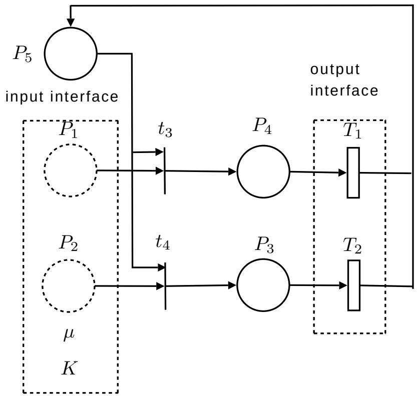
- Rates of T_{i1} and T_{i2} are unknowns: $\mathcal{I} = \{\chi_{t1}, \chi_{t2}\}$
- $\mathcal{V} = \{\mu\}$ is the set of known parameters
- 3 Functions:
  - $f_{RCAT}(\mathcal{I}, \mathcal{V})$ = true if the module satisfies RCAT product-form for a given parametrisation
  - Similarly $f_{M \Rightarrow M}(\mathcal{I}, \mathcal{V})$
  - $K_3(\mathcal{I}, \mathcal{V})$ is the reversed rate of T_3

Example of G-queue



- $f_{RCAT}(\mathcal{I}, \mathcal{V}) = \textbf{true}$
- $f_{M \Rightarrow M}(\mathcal{I}, \mathcal{V}) = (\chi_{t_2} == 0)$
- $K_3(\mathcal{I}, \mathcal{V}) = \frac{\chi_{t_1}}{\chi_{t_2} + \mu}\mu$

Example of module for shared bus contention



- K: number of initial tokens in P_5
- μ: firing rate of T_1 and T_2

Module definition:

$f_{RCAT}(\mathcal{I}, \mathcal{V}) = (K == 1)$
$f_{M \Rightarrow M}(\mathcal{I}, \mathcal{V}) = \textbf{true}$
$K_1(\mathcal{I}, \mathcal{V}) = \chi_{t_1}$
$K_2(\mathcal{I}, \mathcal{V}) = \chi_{t_2}$

Allowed module instance connections

Formally...

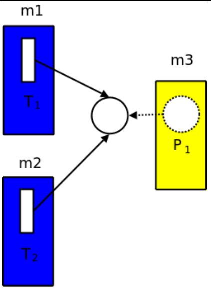
- If T_i is an input transition then it is associated with just one output transition of another instance or a transition with null input and output vector
- Each place of the net is associated with one input place and an output transition which is not associated with an input transition can have an outgoing arc to just one place.
- All arcs connecting the instances have weight 1

Informally...

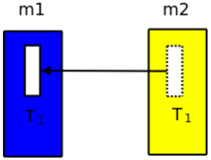
- Only pairwise cooperation is allowed among module instances
  - Fork and join constructs can still be modelled but within a module
- No batch token movements are allowed between modules
- Probabilistic module connections can be easily introduced

Automatic generation of the traffic equations

The solution of the traffic equations gives us the rates of the transitions needed for isolating the modules



$$m3.\chi_{t_1} = m1.K_1(m1.\mathcal{I}, m1.\mathcal{V}) + m2.K_2(m2.\mathcal{I}, m2.\mathcal{V})$$



$$m2.\chi_{t_1} = m1.K_1(m1.\mathcal{I}, m1.\mathcal{V})$$

where $\mathcal{I}$ is the set of the unknown input rates of an isolated instance, $\mathcal{V}$ an instance parametrisazation, χ_{t_i} the unknown rates of the systems

Deciding product-form

- Once the traffic equations are solved product-form property of the network of module instances can be decided
- Basically the condition is:

Product-form condition

$$(\forall \text{ istances } \ mi \quad f_{RCAT}(mi.\mathcal{I}, mi.\mathcal{V}) = \textbf{true}) \vee$$
$$(\forall \text{ istances } mi \quad f_{M\Rightarrow M}(mi.\mathcal{I}, mi.\mathcal{V}) = \textbf{true})$$

Conclusion

- We propose a framework that integrates the modularity concepts introduced for Petri Nets in order to allow for a modular analysis
- It is possible to mix different formalisms in product-form in a unique model
  - Note that GSPNs are very exprissive
  - Every CTMC can be modelled by a GSPN
  - Often compact models can be obtained
- The modeller do not need specific skills about product-form model resolution
  - In this presentation I did not formally introduce RCAT or $M \Rightarrow M$

Open problems and future works

- Non-linear traffic equations may be derived applying RCAT
  - How to solve the system efficiently?
- For closed systems the computation of the normalizing constant is required to obtain the performance measures
  - Any efficient algorithm can be formulated?
  - Possibility to generalize Convolution or MVA algorithms?
- How to integrate this framework with existing tools?

Thanks for the attention. . . Questions?