From SAN to PEPA: a Technology Transfer

■

Leïla Kloul & Jane Hillston

LFCS

University of Edinburgh

Outline

1. Introduction
2. SAN Formalism
3. PEPA Formalism
4. Functional Dependencies
   - Modelling Flexibility
   - Model Size Reduction
   - Tensor Representation
   - Vector-Matrix Product
5. Conclusions

1. Introduction

Traditional performance models

- Synchronisation constraints
- System size constraint

$\rightarrow$ Complex models
$\rightarrow$ Exponential growth of the number of states

New formalisms

- Stochastic Petri Nets (SPN)
- Stochastic Automata Networks (SAN)
- Stochastic Process Algebra (SPA)

1. Introduction

State space explosion problem

- A compact representation of the generator matrix
  $\rightarrow$ to avoid the generation of the complete matrix (SAN, SPN)
- An aggregation or decomposition technique
  $\rightarrow$ to reduce the model size (SPN, PEPA)

Objective

- The functional dependencies in PEPA models
- The compact representation of the generator matrix in PEPA

2. SAN Formalism

- A Stochastic Automata Network (SAN) is a set of N finite state automata and s synchronization events.

  An automaton consists of

  – states,

  – local transitions: fixed or network-dependent rates (F_i),

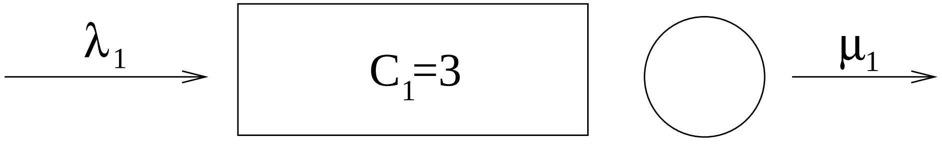
  – synchronized transitions $(S_{i,e})$.

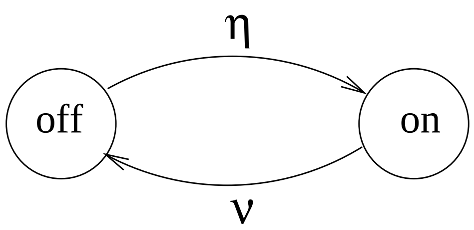
- The whole SAN is associated to a multidimensionnal continuous (discrete) time Markov chain.

$$Q = \bigoplus_{i=1}^{N} F_i + \sum_{e \in \varepsilon} \tau_e \left(\bigotimes_{i=1}^{N} S_{i,e} + \bigotimes_{i=1}^{N} \overline{S}_{i,e} \right)$$

2. SAN Formalism

Example 1:

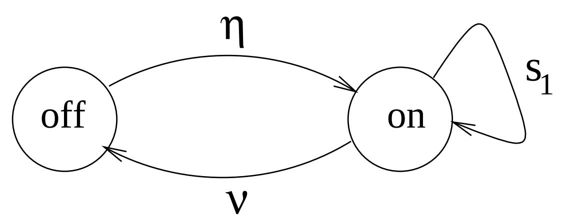




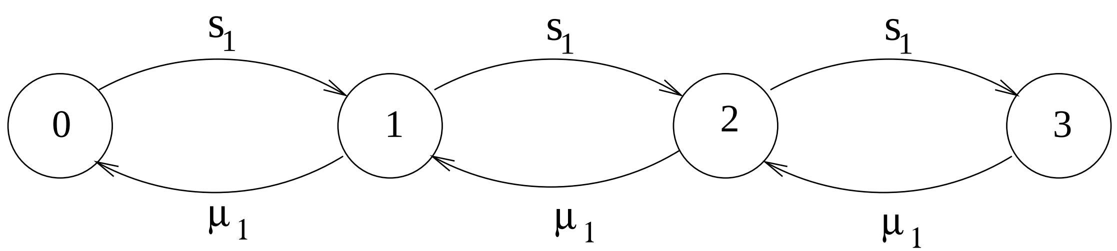
2. SAN Formalism

- **Automata Interaction** *by synchronizations*

A_0:



A_1:

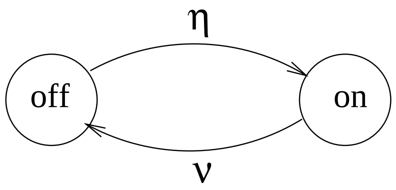


where S_1 rate is λ_1.

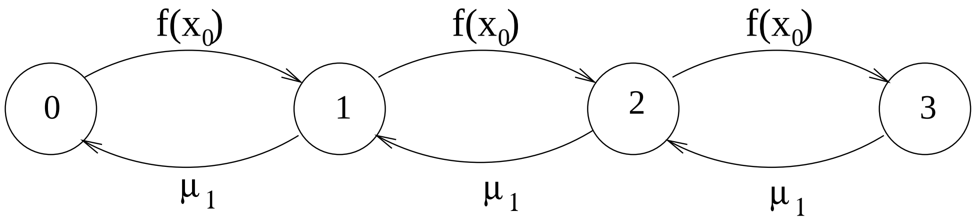
2. SAN Formalism

- **Automata Interaction** *by function*

A_0:



A_1:



$$\text{where } f(x_0) = \begin{cases} 1 & \text{if } \ x_0 = on \\ 0 & \text{otherwise} \end{cases}$$

3. PEPA Formalism

System = set of *components* which interact between them.
They engage, either singly or multiply, in *activities*.

- An activity **a** is caracterised by:
  - an *action type*: $\alpha \in \mathcal{A}$
  - a *duration* which is a r.v. exponentially distribued: $1/r$
- The set of activities is defined as $\mathcal{A}ct \subseteq \mathcal{A} \times R^+$ where

$$R^+ = \{r | r > 0; r \in R\} \cup \{\top\}$$

3. PEPA Formalism

- **Interaction in PEPA** *by cooperation*

$$
\begin{aligned}
Buffer_0 &\stackrel{def}{=} (in, \top).Buffer_1 \\
Buffer_1 &\stackrel{def}{=} (in, \top).Buffer_2 + (service, \mu).Buffer_0 \\
Buffer_2 &\stackrel{def}{=} (in, \top).Buffer_3 + (service, \mu).Buffer_1 \\
Buffer_3 &\stackrel{def}{=} (service, \mu).Buffer_2
\end{aligned}
$$

$$
\begin{aligned}
Arrival_{on} &\stackrel{def}{=} (in, \lambda).Arrival_{on} + (off, \nu).Arrival_{off} \\
Arrival_{off} &\stackrel{def}{=} (on, \eta).Arrival_{on}
\end{aligned}
$$

$$
System \stackrel{def}{=} Buffer_0 \underset{\{\mathbf{in}\}}{\bowtie} Arrival_{on}
$$

4. Functional Dependencies

- **One component**
  - the rate value depends on the component state
    => the rate is a positive number and can never be zero
- **Several components**
  - an activity to be performed by a component depends on the current state of one or more other components
  - the rate value depends on the components current state
    => the rate is a positive number and can be zero

The set of activities $\mathcal{A}ct$ is now defined as $\mathcal{A}ct \subseteq \mathcal{A} \times R^*$ where

$$R^* = \{r | r \geq 0; r \in R\} \cup \{\top\}$$

4.1. Modelling Flexibility

- **Component interaction** *by cooperation*

$$
\begin{aligned}
Buffer_0 &\stackrel{def}{=} (in, \top).Buffer_1 \\
Buffer_1 &\stackrel{def}{=} (in, \top).Buffer_2 + (service, \mu).Buffer_0 \\
Buffer_2 &\stackrel{def}{=} (in, \top).Buffer_3 + (service, \mu).Buffer_1 \\
Buffer_3 &\stackrel{def}{=} (service, \mu).Buffer_2
\end{aligned}
$$

$$
\begin{aligned}
Arrival_{on} &\stackrel{def}{=} (in, \lambda).Arrival_{on} + (off, \nu).Arrival_{off} \\
Arrival_{off} &\stackrel{def}{=} (on, \eta).Arrival_{on}
\end{aligned}
$$

$$
System \stackrel{def}{=} Buffer_0 \underset{\{\mathbf{in}\}}{\bowtie} Arrival_{on}
$$

4.1. Modelling Flexibility

- **Component interaction** *by functions*

$$
\begin{aligned}
Buffer_0 &\stackrel{def}{=} (in, \lambda \times f).Buffer_1 \\
Buffer_1 &\stackrel{def}{=} (in, \lambda \times f).Buffer_2 + (service, \mu).Buffer_0 \\
Buffer_2 &\stackrel{def}{=} (in, \lambda \times f).Buffer_3 + (service, \mu).Buffer_1 \\
Buffer_3 &\stackrel{def}{=} (service, \mu).Buffer_2
\end{aligned}
$$

$$
\begin{aligned}
Arrival_{on} &\stackrel{def}{=} (off, \nu).Arrival_{off} \\
Arrival_{off} &\stackrel{def}{=} (on, \eta).Arrival_{on}
\end{aligned}
$$

where $f(i) = \begin{cases} 1 & \text{if } i = on \\ 0 & \text{otherwise} \end{cases}$

$$System \stackrel{def}{=} Buffer_0 || Arrival_{on}$$

4.2. Model Size Reduction

Example 2: The resource sharing system

- N processors
- M resources with $M \leq N$
- Different rates: λ_i, μ_i, $1 \leq i \leq N$

- **PEPA Model** *without functions*
  - Components $Processor_0^{(i)}$, $1 \leq i \leq N$
  - Component $NumberR_0$

4.2. Model Size Reduction

$$
\begin{aligned}
Processor_0^{(i)} &\stackrel{def}{=} (use_i, \lambda_i).Processor_1^{(i)} \\
Processor_1^{(i)} &\stackrel{def}{=} (free_i, \mu_i).Processor_0^{(i)}
\end{aligned}
$$

$$
\begin{aligned}
NumberR_0 &\stackrel{def}{=} \sum_{i=1}^{N}(use_i, \top).NumberR_1 \\
NumberR_1 &\stackrel{def}{=} \sum_{i=1}^{N}(use_i, \top).NumberR_2 + \sum_{i=1}^{N}(free_i, \top).NumberR_0 \\
\ldots &\quad \ldots \\
NumberR_M &\stackrel{def}{=} \sum_{i=1}^{N}(free_i, \top).NumberR_{M-1}
\end{aligned}
$$

$$
System \stackrel{def}{=} \left(Processor_0^{(1)} || \ldots || Processor_0^{(N)}\right) \underset{K}{\bowtie} NumberR_0
$$

where $K = \{use_1, \ldots, use_N, free_1, \ldots, free_N\}$

4.2. Model Size Reduction

- **PEPA Model** *with functions*

$$
\begin{aligned}
Processor_0^{(i)} &\stackrel{def}{=} (use_i, \lambda_i \times f(x_1, \ldots, x_N)).Processor_1^{(i)} \\
Processor_1^{(i)} &\stackrel{def}{=} (free_i, \mu_i).Processor_0^{(i)}
\end{aligned}
$$

$$
\text{where } f(x_1, \ldots, x_N) = \begin{cases} 1 & \text{if } \sum_{j=1}^{N} x_j < M \quad x_j \in \{0, 1\} \\ 0 & \text{otherwise} \end{cases}
$$

$$
System \stackrel{def}{=} Processor_0^{(1)} || \ldots || Processor_0^{(N)}
$$

4.2. Model Size Reduction

- **PEPA Model** *without functions*
  - $N + 1$ components
  - $2^N(M + 1)$ states
- **PEPA Model** *with functions*
  - N components
  - 2^N states

4.3. Tensor Representation

Global Generator Matrix

$$Q = \bigoplus_{i=1}^{N} R_i + \sum_{\alpha \in \mathcal{Z}} r_\alpha \left(\bigotimes_{i=1}^{N} P_{i,\alpha} - \bigotimes_{i=1}^{N} \overline{P}_{i,\alpha} \right)$$

where

- R_i is the transition matrix of component C_i relating solely to its individual activities.
- $P_{i,\alpha}$ is the probability transition matrix of component C_i due to activity of type α. Its elements' values are between 0 and 1.
- $\overline{P}_{i,\alpha}$ is a matrix representing the normalization associated with the shared activity α in component C_i.
- r_α is the minimum of the functional rates of action type α over all components C_i, $i = 1 \ldots N$.

4.4. The Vector-Matrix Product

- **Global Generator Matrix**

$$Q = \sum_{k=1}^{2\mathcal{Z}+N} \bigotimes_{i=1}^{N} Q_{k,i}$$

- **Markov Chain Solution**

$$x\,Q = \sum_{k=1}^{2|\mathcal{Z}|+N} x \bigotimes_{i=1}^{N} Q_{k,i} = 0$$

where

- x is a vector of length $\Pi_{i=1}^{N} T_i$
- T_i is the size of component C_i

4.4. The Vector-Matrix Product

- If the matrices contain only constant values

$$Cost = \Pi_{i=1}^{N} T_i \times \sum_{i=1}^{N} T_i$$

- If the matrices contains functional rates, but there is no cycle in the functional dependency graph

$$Cost = \Pi_{i=1}^{N} T_i \times \sum_{i=1}^{N} T_i$$

- If there is a cycle in the functional dependency graph,

$$Cost = \left(\Pi_{i=1}^{N} T_i\right) \left(\Pi_{i=1}^{t} T_i\right) \left(\sum_{i=t+1}^{N} T_i \right)$$

where t is the number of automata involved in the cycle.

5. Conclusions

- Introduction of functional dependencies in PEPA
  - modelling flexibility
  - model size reduction in some cases
  - direct tensoriel representation of the Markov chain

- In the future . . .
  - implement and incorporate our approach to PEPA Workbench
  - investigate the solving techniques which exploit the Kronecker representation