Methods and algorithms for the performance analysis of large systems

Brigitte Plateau

Eiad Sulaiman

Ihab Sbeity

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

Institut National Polytechnique de Grenoble

INRIA INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

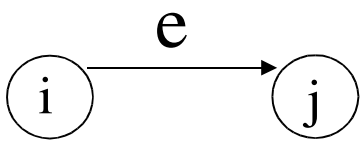
GRENOBLE 1 UNIVERSITÉ JOSEPH FOURIER SCIENCES,TECHNOLOGIE,MÉDECINE

Motivations

- Continuous Time Markov Chains (CTMC): performance analysis of dynamic systems.
- Huge CTMCs: compact and structured model. State space explosion.
- Computing stationary and transient solutions: sophisticated algorithms and data structure are needed.

SAN

- Change of the global state (combination of the local states of each automata):
  - local or synchronising event, transition rate λ.
- Local event:
  - Change of the local state of only one automaton.

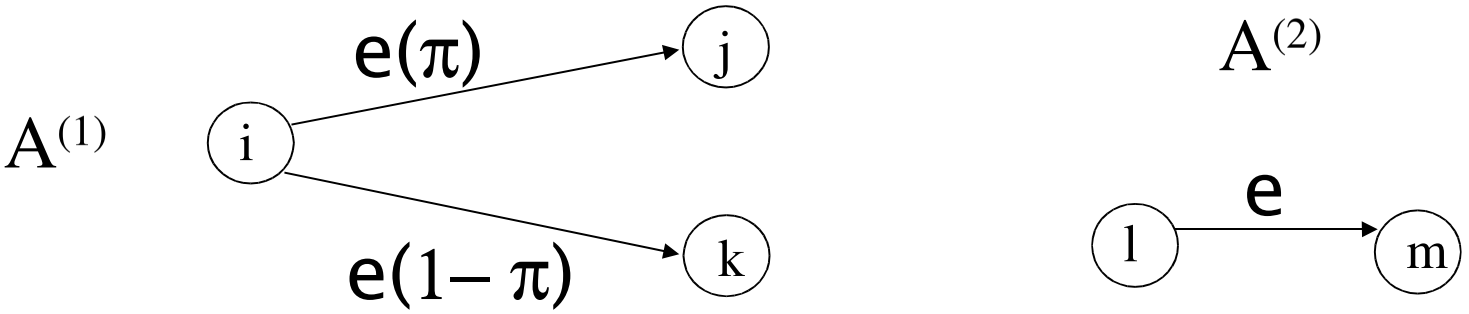


- Synchronising event:
  - Simultaneous change of the local state of more than one automaton.

SAN

- Probability of occurrence

  when the occurrence of the same event from a given departure state can lead to different arrival states.



SAN

- Other way of interacting: functional rates.
  - The transition rate of an event can be a function of the global state of the SAN.
  - Can apply to local or synchronising events.

SAN

descriptor

- SAN: modular description and manipulation. Generator of the Markov chain = descriptor Q.

$$Q = D + \sum_{i=1}^{N} {}_g Q_l^{(i)} + \sum_{e=1}^{E} \prod_{i=1}^{N} {}_g Q_e^{(i)} = \sum_{j=1}^{N+2E} \left[\prod_{i=1}^{N} {}_g Q_j^{(i)} \right]$$

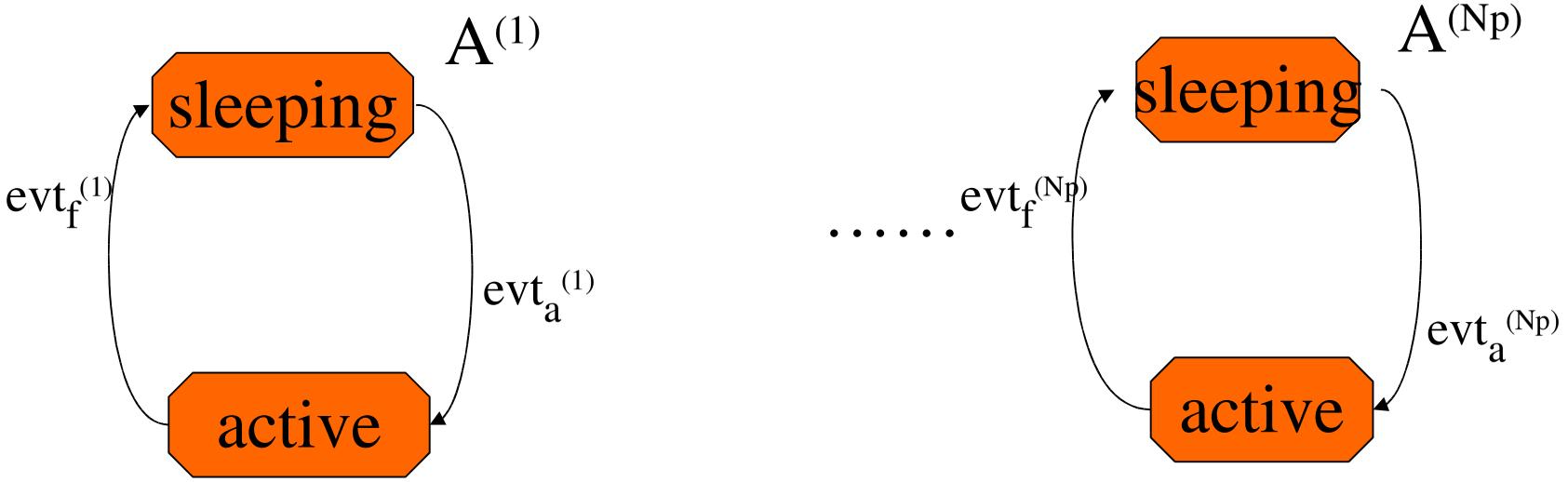
  - Tensor sum: local part.
  - Tensor product: synchronising part - E sync events.

- Use of generalised tensor algebra (functional matrices) [Plateau, Fernandes ...].

SAN

study example - mutex1

- Np processes, Nr resources - model with functions



For i=1..Np, $evt_a^{(i)}$ is a local event « acquiring a resource », rate $\lambda_i f(A^{(1)},\ldots,A^{(Np)})$.

$f(A^{(1)},\ldots,A^{(Np)})$ is a function of the number of automata in state active.

If this number is greater than or equal to Nr, f equals 0, otherwise 1.

For i=1..Np, $evt_f^{(i)}$ is a local event « freeing a resource », rate μ_i .

SAN

example - mutex1 - descriptor

- For i=1..Np, define $Q_l^{(i)} = \begin{pmatrix} -{}_i f & {}_i f \\ \mu_i & -\mu_i \end{pmatrix}$

- Then the descriptor (algebraic formulation of the transition matrix) is:

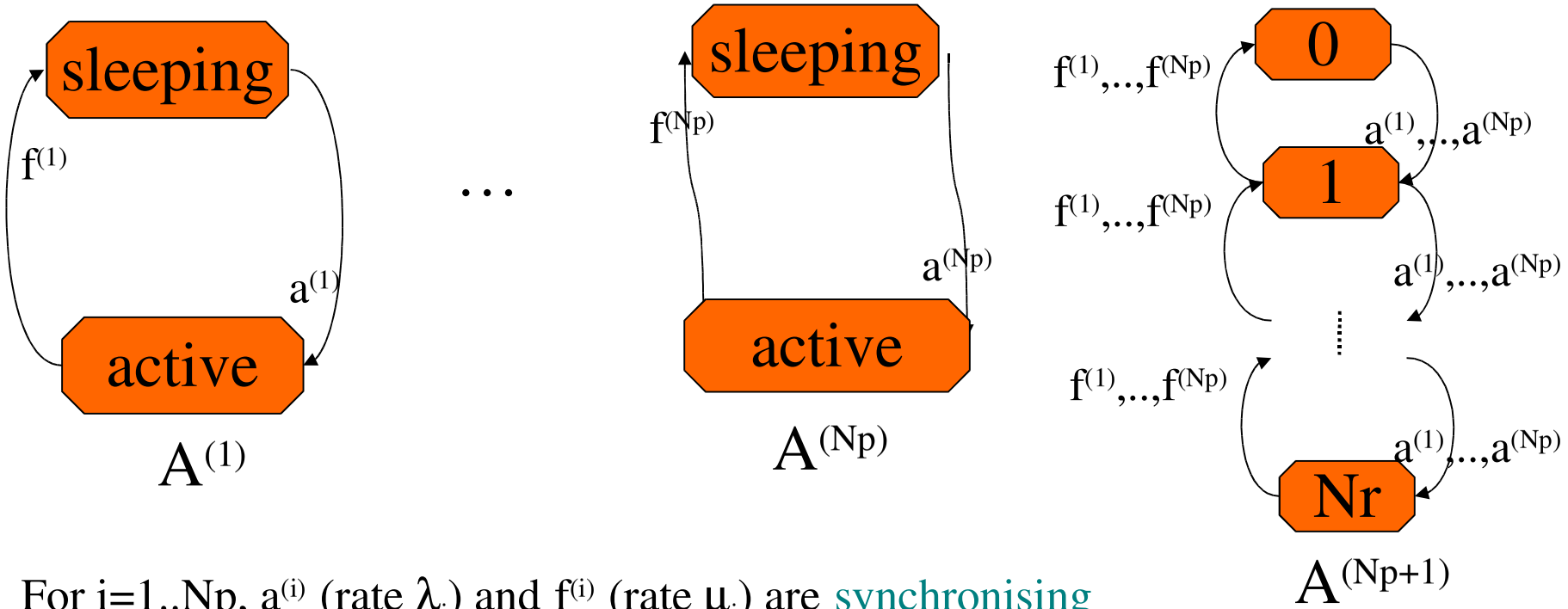
$$Q = Q_l^{(1)} \oplus_g Q_l^{(2)} \oplus_g \ldots \oplus_g Q_l^{(Np)}$$

- If the N automata were independent (i.e. the function f constant), the descriptor would be the same, with classical tensor sum.

SAN

example - mutex2

- Np processes, Nr resources - synchronising events



For i=1..Np, $a^{(i)}$ (rate λ_i) and $f^{(i)}$ (rate μ_i) are synchronising events, implying automata $A^{(i)}$ and $A^{(Np+1)}$.

SAN

comparison mutex1/ mutex2

- Product state space:

$$|\hat{S}_1| = 2^{Np} \; and \; |\hat{S}_2| = 2^{Np} * (Np + 1)$$

- Reachable state space size: $|S_1|$ and $|S_2|$

  equal to the number of solutions of

$$\sum_{i=1}^{Np} x^{(i)} \leq Nr \;, \;\; with \;\; x^{(i)} = 0 \;\; or \;\; 1$$

- Descriptor: Mutex1 - (Np-1) tensor sums.
- Mutex2 - $2(Np)^2$ tensor products of very sparse matrices.

On-going work

... and items of interest

Modelling large computer systems
formalisms

- Queuing networks
- Petri nets
- Process algebra
- Automata networks
- others:
  - UML
  - Mobius
  - combinations of PA and PN, PA and SAN?

Modelling large computer systems

storing S

- Reachability function
  - Access to the reachable states in constant time.
  - Example: PEPS.
- Multi-level approaches
  - Decision diagrams [Ciardo, Miner].
  - Efficient generation and storage of S.

Modelling large computer systems
lumping of models

reduce the complexity of the Markov chain, using replications of components and exact lumping.

- Aggregation from state space analysis
- Aggregation from the high level formalism:
  - [Siegle], [Buchholz]
  - [Benoit]: SANs with replicas and tensor structure of the aggre
  - extension and combination

Modelling large computer systems
numerical algorithms

- Context: iterative methods
- tensor: work on a product state space $\hat{S}$
- Probability vectors $\hat{}$ « extended » (the size of $\hat{S}$)
- Q: descriptor of the SAN $Q = \sum_{j=1}^{N+2E} \left[\bigotimes_{i=1}^{N} Q_j^{(i)} \right]$

- sparse implementation, in parallel?

PEPS to developp

- All these algorithms are implemented in PEPS

  (Performance Evaluation of Parallel Systems)
  - SAN editor
  - Describes easily SANs with replicas, and replicated states with identical behavior (queuing networks).
  - Computes and tests the descriptor.
  - Implements the basic multiplications, with optimizations for the function evaluation (automata reordering, ...).
  - Various iterative methods (power, Jacobi, GMRES).
  - Allows to output sparse format (HBF).
  - Compute performance indices.

Other Perspectives

- Graphical interface in Java for PEPS: more conviviality, easy to use [Mathieu Le Coz]
- Discrete-time SANs: we are working on a new formalism; develop new numerical methods.
- Compute bounds
- Simulation ?

Annexe (1)

tensor product

Product $A \otimes B =$

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \otimes \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix}$$

$=$

$$\left(\begin{array}{ccc|ccc} a_{11}b_{11} & a_{11}b_{12} & a_{11}b_{13} & a_{12}b_{11} & a_{12}b_{12} & a_{12}b_{13} \\ a_{11}b_{21} & a_{11}b_{22} & a_{11}b_{23} & a_{12}b_{21} & a_{12}b_{22} & a_{12}b_{23} \\ a_{11}b_{31} & a_{11}b_{32} & a_{11}b_{33} & a_{12}b_{31} & a_{12}b_{32} & a_{12}b_{33} \\ \hline a_{21}b_{11} & a_{21}b_{12} & a_{21}b_{13} & a_{22}b_{11} & a_{22}b_{12} & a_{22}b_{13} \\ a_{21}b_{21} & a_{21}b_{22} & a_{21}b_{23} & a_{22}b_{21} & a_{22}b_{22} & a_{22}b_{23} \\ a_{21}b_{31} & a_{21}b_{32} & a_{21}b_{33} & a_{22}b_{31} & a_{22}b_{32} & a_{22}b_{33} \end{array}\right)$$

Annexe (2)

tensor sum

$$\text{Sum } A \oplus B = A \otimes Idn_B + Idn_A \otimes B$$

$$= \left(\begin{array}{ccc|ccc} a_{11}+b_{11} & b_{12} & b_{13} & a_{12} & & \\ b_{21} & a_{11}+b_{22} & b_{23} & & a_{12} & \\ b_{31} & b_{32} & a_{11}+b_{33} & & & a_{12} \\ \hline a_{21} & & & a_{22}+b_{11} & b_{12} & b_{13} \\ & a_{21} & & b_{21} & a_{22}+b_{22} & b_{23} \\ & & a_{21} & b_{31} & b_{32} & a_{22}+b_{33} \end{array}\right)$$

Annexe (3)

generalised tensor product

$$\begin{pmatrix} a_{11}(k) & a_{12}(k) \\ a_{21}(k) & a_{22}(k) \end{pmatrix} \otimes_g \begin{pmatrix} b_{11}(h) & b_{12}(h) & b_{13}(h) \\ b_{21}(h) & b_{22}(h) & b_{23}(h) \\ b_{31}(h) & b_{32}(h) & b_{33}(h) \end{pmatrix}$$

$$\left(\begin{array}{ccc|ccc}
a_{11}(1)b_{11}(1) & a_{11}(1)\,b_{12}(1) & a_{11}(1)\,b_{13}(1) & a_{12}(1)b_{11}(1) & a_{12}(1)\,b_{12}(1) & a_{12}(1)\,b_{13}(1) \\
a_{11}(2)(1) & a_{11}(2)(1) & a_{11}(2)(1) & a_{12}(2)b_{21}(1) & a_{12}(2)b_{22}(1) & a_{12}(2)b_{23}(1) \\
a_{11}(3)b_{31}(1) & a_{11}(3)b_{32}(1) & a_{11}(3)b_{33}(1) & a_{12}(3)b_{31}(1) & a_{12}(3)b_{32}(1) & a_{12}(3)b_{33}(1) \\
\hline
a_{21}(1)b_{11}(2) & a_{21}(1)\,b_{12}(2) & a_{21}(1)\,b_{13}(2) & a_{22}(1)b_{11}(2) & a_{22}(1)\,b_{12}(2) & a_{22}(1)\,b_{13}(2) \\
a_{21}(2)(2) & a_{21}(2)(2) & a_{21}(2)(2) & a_{22}(2)(2) & a_{22}(2)(2) & a_{22}(2)(2) \\
a_{21}(3)b_{31}(2) & a_{21}(3)b_{32}(2) & a_{21}(3)b_{33}(2) & a_{22}(3)b_{31}(2) & a_{22}(3)b_{32}(2) & a_{22}(3)b_{33}(2)
\end{array}\right)$$