How to Deal with State Space Explosion

Some Thoughts about the Analysis of Large Markov Models for Dependability Analysis

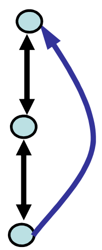
Peter Buchholz
Informatik IV
TU Dortmund
Germany

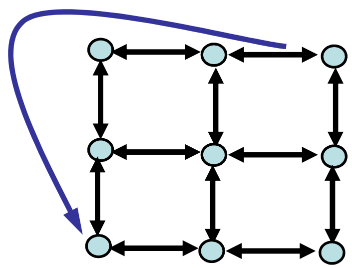
- Systems are built from connected components
- Problem of state space explosion

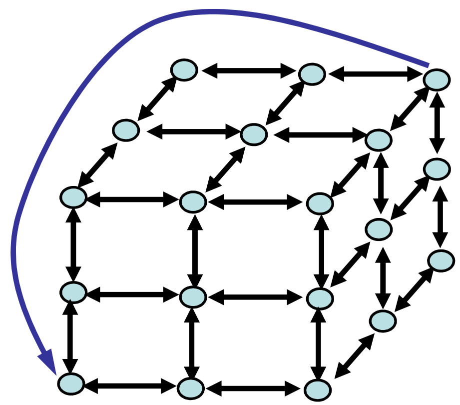
  J components with n_j states each ($j = 1,\ldots,J$)

  result in $O(\Pi\, n_j)$ states
- For J components we obtain a J-dimensional transition system

Functional Analysis – Model Checking can deal with state space explosion!







Representation of multi-dimensional structures at a state level

$\Rightarrow$ Tensor-Algebra

implicit encoding of transition relation using products

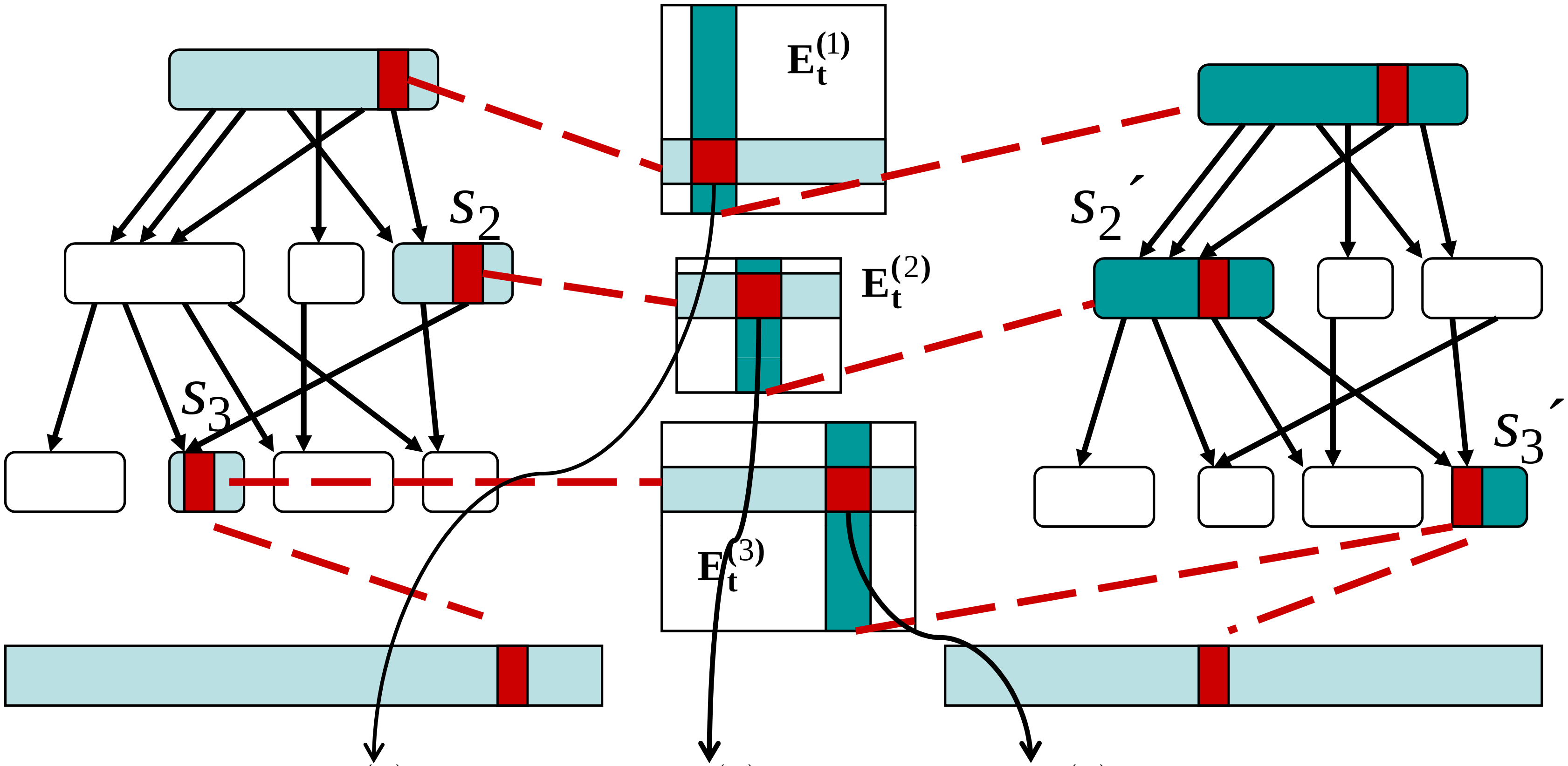
Basic representation of a transition matrix

$$\mathbf{Q} = \sum_{t=1}^{2TT} \left(\bigotimes_{j=1}^{J} \mathbf{E}_t^{(j)} \right)$$ where $\mathbf{E}_t^{(j)}$ is $n_j \times n_j$

Implicit assumption everything is reachable (often not realistic)

$\Rightarrow$ more sophisticated coding is necessary

Implict coding via Matrix Diagrams (or similar data structures)



$$\mathbf{E}_t^{(1)}[K_1, L_1] \bigotimes \mathbf{E}_t^{(2)}[K_2, L_2] \bigotimes \mathbf{E}_t^{(3)}[K_3, L_3]$$

Each level describes one component

Hierarchical representation (micro states per macro state)

$$\begin{pmatrix} \mathbf{Q}[0,0] & \ldots. & \mathbf{Q}[0,N] \\ . & . & . \\ \mathbf{Q}[N,0] & \ldots. & \mathbf{Q}[N,N] \end{pmatrix}$$

$$\mathbf{E}_t^{(j)} = \begin{pmatrix} \mathbf{E}_t^{(j)}[0,0] & \cdots & \mathbf{E}_t^{(j)}[0,N_j] \\ \vdots & \ddots & \vdots \\ \mathbf{E}_t^{(j)}[N_j,0] & \cdots & \mathbf{E}_t^{(j)}[N_j,N_j] \end{pmatrix}$$

$$\mathbf{Q}[K,L] = \sum_{t=1}^{2TT} \left(\bigotimes_{j=1}^{J} \mathbf{E}_t^{(j)}[K_j, L_j] \right)$$

Each submatrix as a sum of tensor products

(appropriate data structures are available)

Where are we?

- Compact representations of huge state spaces (10^{30} and beyond) and transition relation
- Efficient generation of state space and transition relation
- Often efficient analysis of functional properties
- But what about non-functional properties? (dependability, performance, safety(!?), …)

- Solution of the transient or stationary probability distribution requires storage of $O(\Pi\, n_j)$ states
  - Can be done for 10^7-10^8 states but not beyond
  - Alternative compact storage schemes for probability vectors (e.g, MDDs, PDGs) failed

How to obtain results for really large models?

Brute force:

- Simulation
- Disk based numerical solution

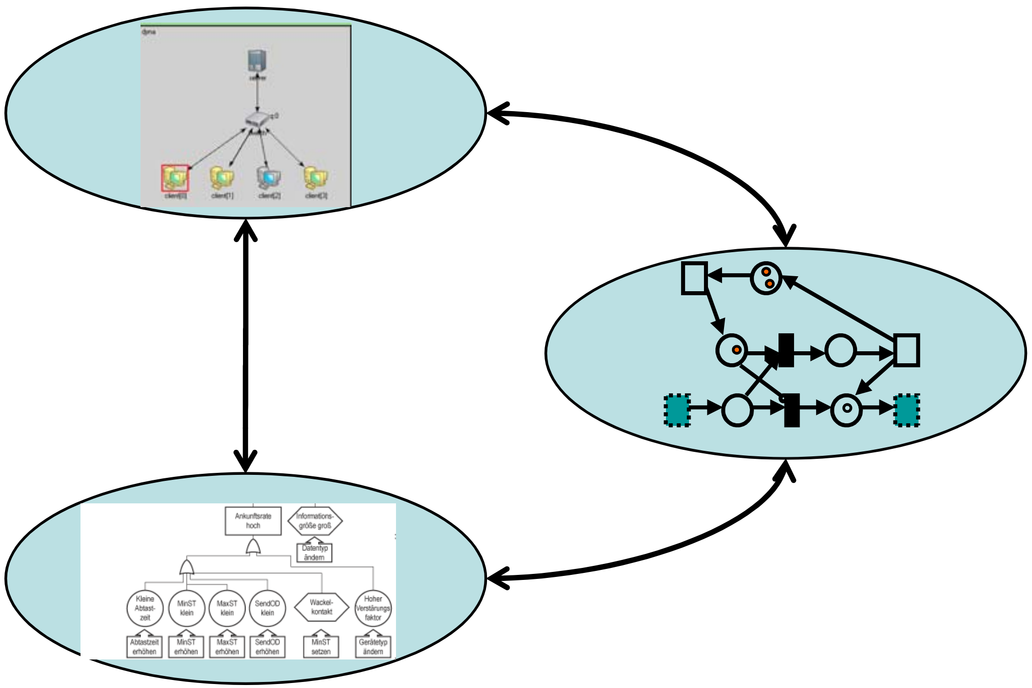
Bounding:

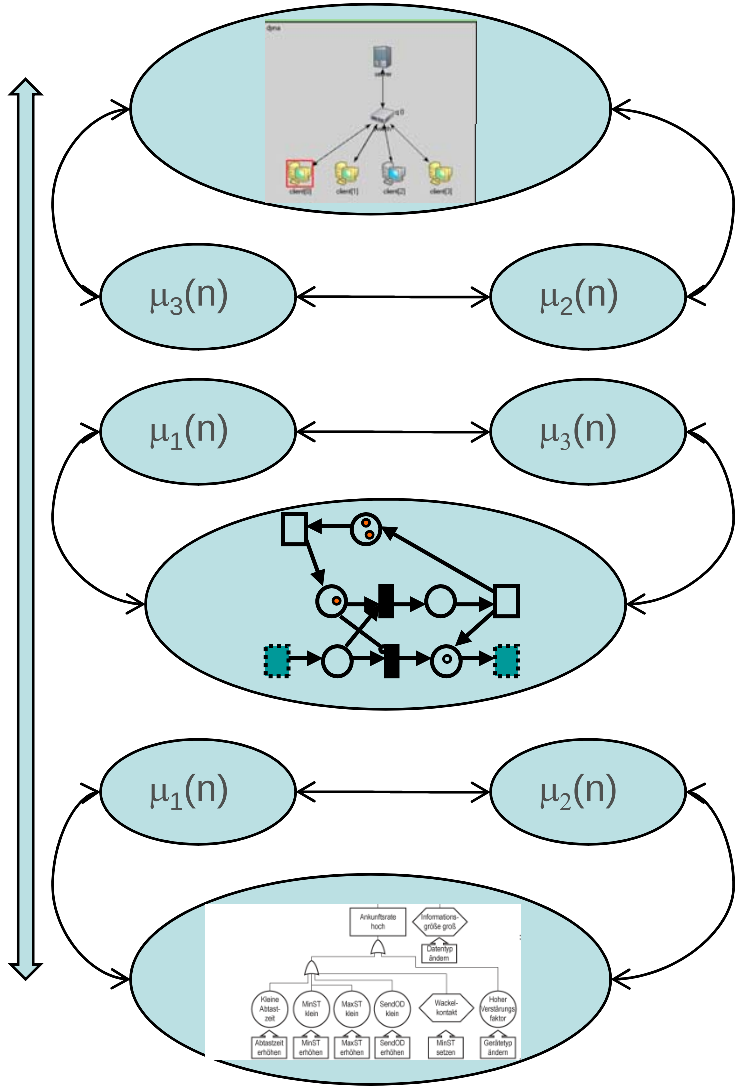
- Aggregation
- Path based

Approximation:

- **Decomposition & fixed point**
- **Residual norm minimization**

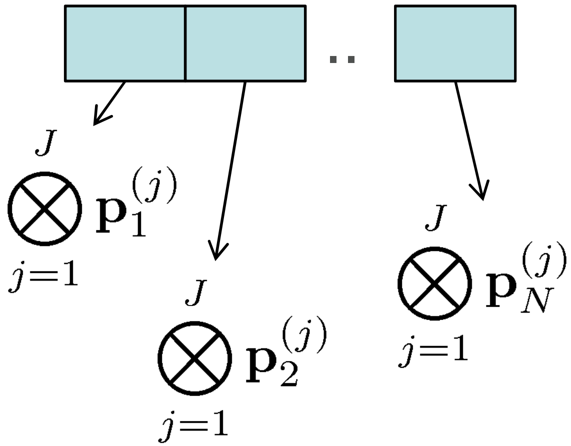
Basic idea of fixed point approaches





Vector representation in the algorithms

Macro states



Corresponding

to the matrix

representation

$$\begin{pmatrix} \mathbf{Q}[0,0] & \ldots. & \mathbf{Q}[0,N] \\ . & & . \\ \mathbf{Q}[N,0] & .. & \mathbf{Q}[N,N] \end{pmatrix}$$

$$\mathbf{Q}[K,L] = \sum_{t=1}^{2TT} \left(\bigotimes_{j=1}^{J} \mathbf{E}_t^{(j)}[K_j, L_j] \right)$$

- **Can be used for stationary and transient (!) analysis**
- Effort for J components with n_j states and d_j non-zeros per row

  Memory $O(\Sigma d_j n_j)$ versus $O(\Pi d_j n_j)$ for exact methods

  Time:

Transient analysis:

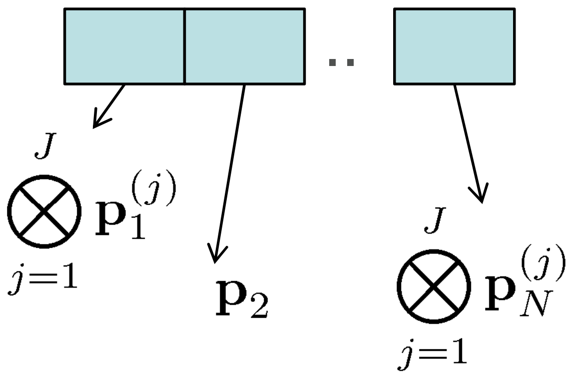
- $O(\alpha t \Sigma d_j n_j)$ versus $O(\alpha t \Pi d_j n_j)$ where t is the time bound and α is a bound for the transition rates

Stationary analysis:

- $O(it_1 \Sigma d_j n_j)$ versus $O(it_2 \Pi d_j n_j)$ where usually $it_1 > it_2$ but $O(it_1) = O(it_2)$

- **But this is an approximate method! What about the error?**
- Usually errors are small (often below 1%) but this not a guarantee!
- Two possibilities to reduce and control the error
  - Partial expansion of the vector
  - Residual computation and multiple Kronecker products

Expand parts of the vector



- Easy to integrate in a solution approaches (stationary and transient)
- Effort grows but usually remains less than 1/N of the exact computation

Which parts should be expanded and how to use the results?

- Use parts where the probability and/or the reward is high
- Compare solutions resulting from different expansions

Computation of residuals (for stationary analysis)

- Let π be the exact solution and **q** be an approximation, then

  $\pi\mathbf{Q} = \mathbf{0}$ and $\mathbf{qQ} = \mathbf{r}$

  $||\mathbf{r}||_2$ is an indicator for the error

- The residual is computed with one vector matrix product

  vector + sparse matrix $O(\Pi d_j n_j)$

  vector + Kronecker matrix $O(\Sigma d_j \Pi n_j)$ (with larger constants)

  Kronecker vector + Kronecker matrix $O(\Sigma d_j n_j)$

Can be done!

How to improve solutions?

- Reduce the resiudal $$\min_{\mathbf{p}^{(1)},\ldots,\mathbf{p}^{(J)}} \left(\left\| \bigotimes_{j=1}^{J} \mathbf{p}^{(j)} \mathbf{Q} \right\|_2 \right)$$

  non-linear optimization problem

  approximation algorithm solves iteratively quadratic problems

- Extension by using more than one vector per component

$$\min_{\mathbf{p}^{(1),1},\ldots,\mathbf{p}^{(J),K}} \left(\left\| \sum_{k=1}^{K} \left(\bigotimes_{j=1}^{J} \mathbf{p}^{(j),k} \right) \mathbf{Q} \right\|_2 \right)$$

- Computation of minimum is still a challenge but first results are promising

E.g. tandem queuing system

- 3.2 million states
- Fixed point approach solves systems of size 20
  errors: mean population < 1%, single population < 5%,
  joint distribution factor of 7, residual 2.51e-4
- Product form with K=50 solves quadr. opt. problem of size 1000
  errors for all results with an error of less than 1%
  residual 6.62e-6, overall 5000 elements to store!