Parallel Simulation of Queueing Petri Nets

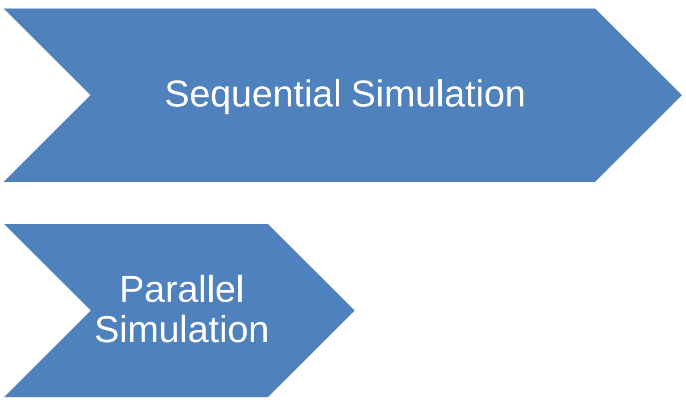
Jürgen Walter
Dept. of Computer Science, University of Würzburg

Symposium on Software Performance,
Nov 27th 2014, Stuttgart, Germany

Motivation

- Queueing Petri Nets are used for performance modelling and analysis

- Desire for performance prediction at run time

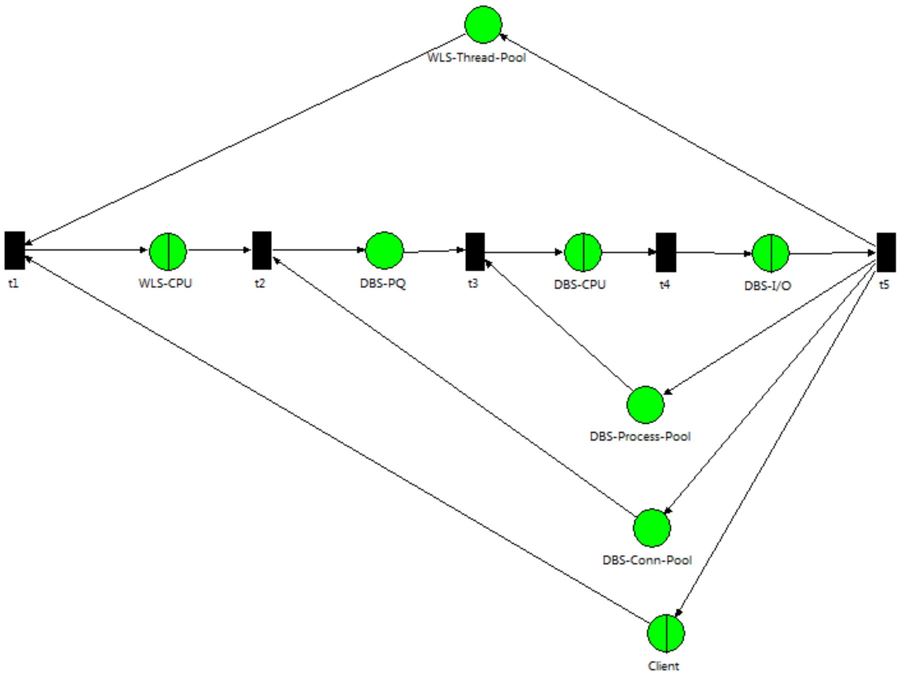


- Multi-core-processors are standard, but SimQPN is still sequential

Foundations QPNs

- Queueing Petri Nets (QPN)
  - Petri Nets (PN)
  - Queueing Networks (QN)

[Bause93a] [Bause93b]



- Model Parts
  - Places
  - Transitions
  - Token
  - Queues

Foundations Simulation

- Discrete Event Simulation
  - Scales better than Markov analysis [Kounev07]
  - Non-deterministic/ based on random seed

- Queueing Petri Net Modeling Environment (QPME)
  - SimQPN
    - Batch/means
    - Replication/deletion

http://tools.descartes/qpme

Foundations Concurrent Simulation

- Concurrent Simulation
  - Parallel Simulation
  - Distributed Simulation

Focus on parallel simualtion

- Logical Process (LP)

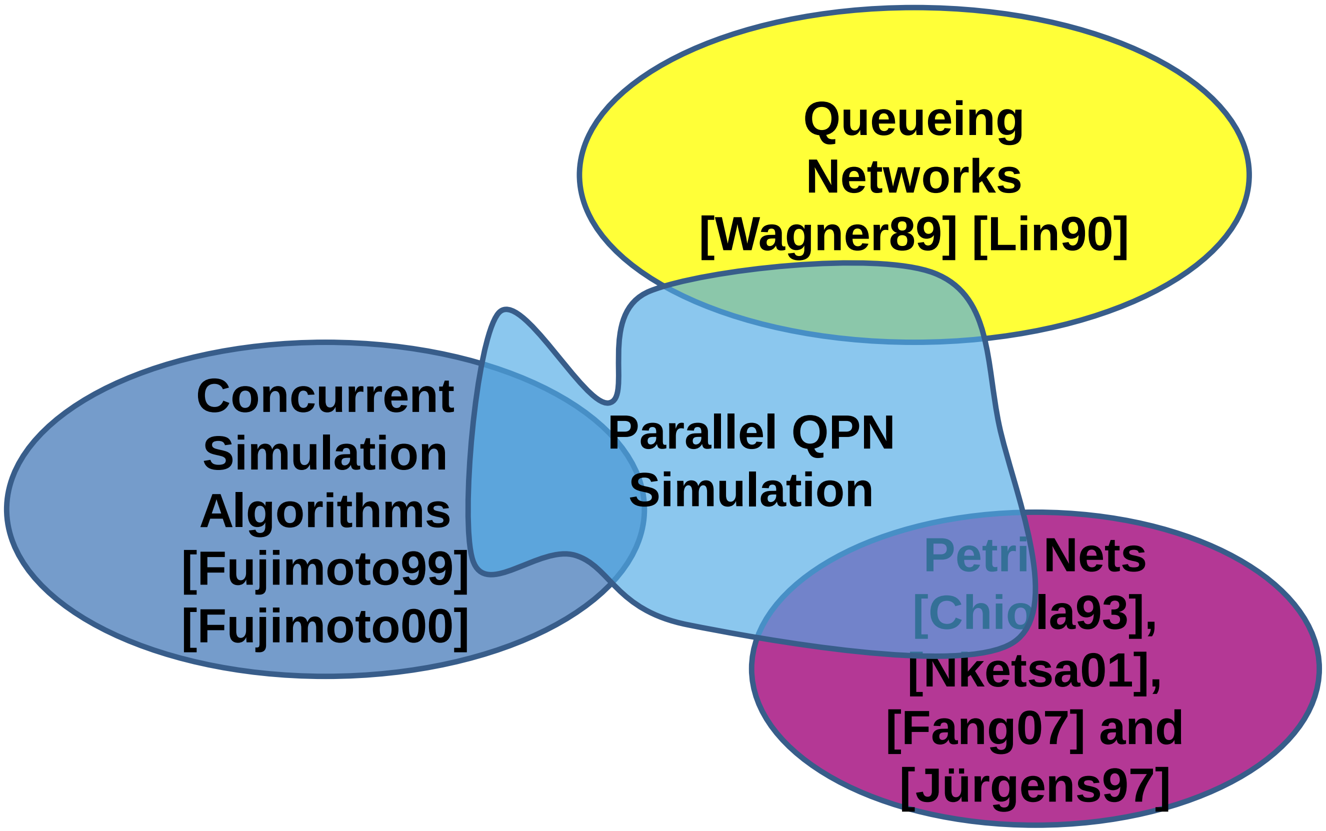
Simulate subparts of simulation model

- Synchronization
  - Conservative 
  - Optimistic 
- Lookahead

My Research in Short

- **Problem**:
  - Desire for increased QPN analysis speed
  - Sequential QPN simulation can not exploit multi core hardware
- **Idea**
  - Provide a parallel simulation engine for QPNs
- **Benefit**
  - Simulation runs faster
  - Improved applicability at runtime scenarios
- **Actions**
  - Identify suitable parallelization techniques
  - Implement these techniques
  - Evaluate the performance improvement

Related Work



How to Parallelize Simulation

APPROACH

Parallelization Levels

[Kaudel87]

Application Level

- Parallel execution of different simulation runs [Pawlinkowski94]

Functional Level

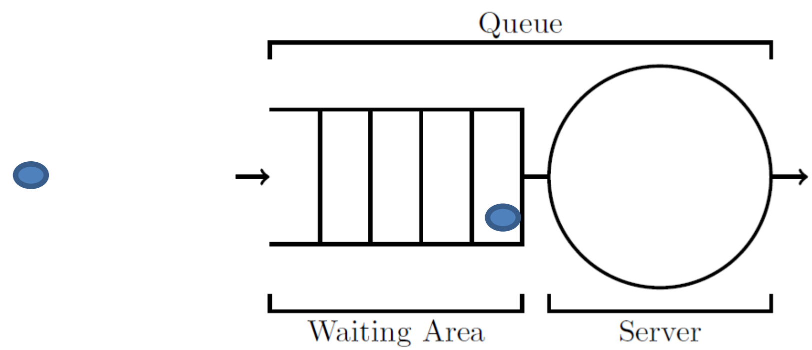
- Execution of helper functions (e.g. random number generation) parallel to simulation
- Existance of helper functions indicator for inefficient code [Jürgens97]

Event Level

- Parallel execution of one simulation run
  - Lookahead
  - Decomposition into Logical Processes
  - Synchronization

Lookahead

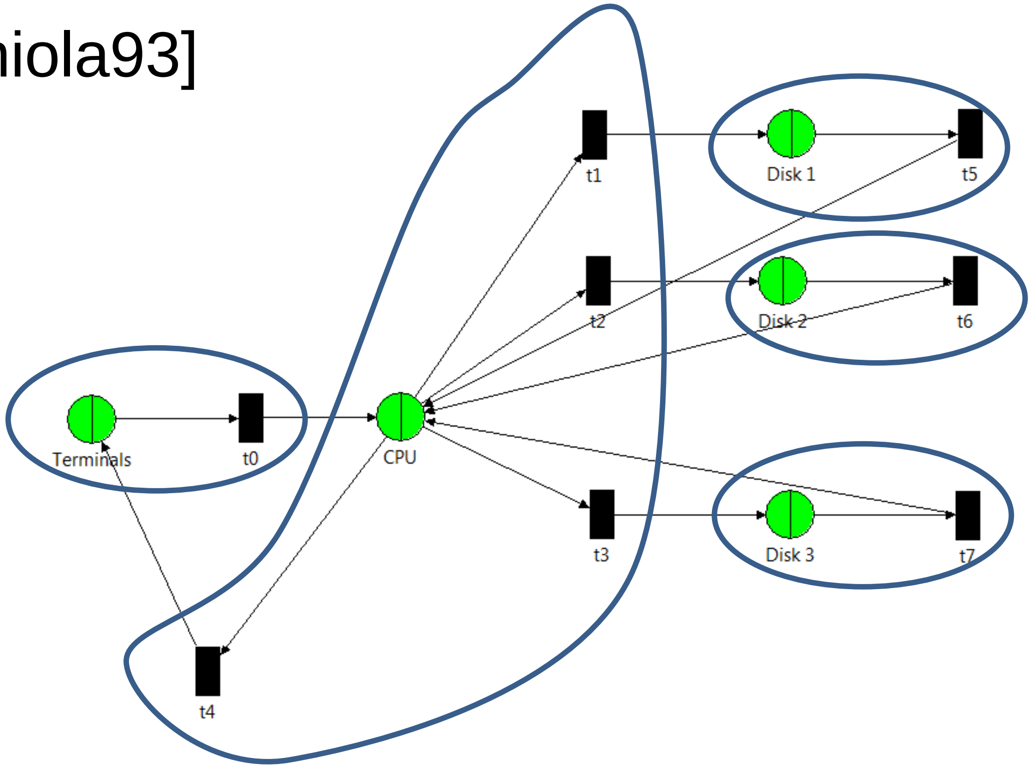
- Token emittance hard to predict for several queueing strategies



- Solution: Presampling of scheduling times [Wagner89]
  - Limit number of tokens
  - Lower bound on service time distribution

Decomposition

- Spatial decomposition
- Minimum Regions [Chiola93]
- Merging Rules [Chiola93]

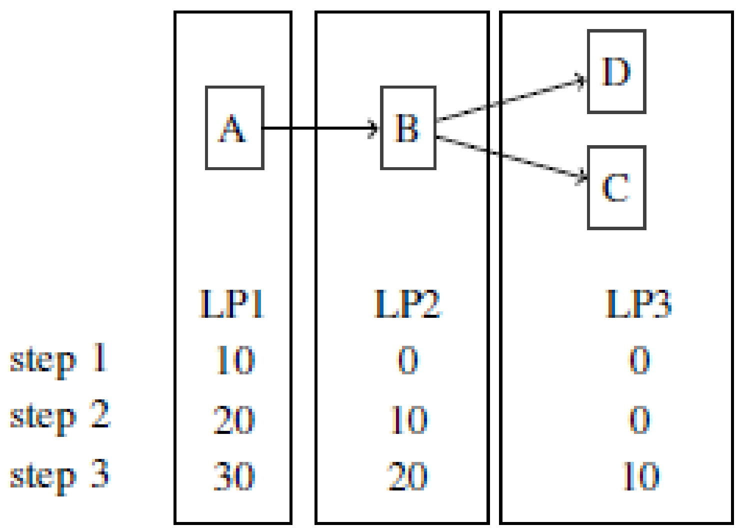


Theory versus Practice

- Parallel simulation works on a theoretical basis for every kind of model
- However:
  - Event processing in few microseconds
  - Synchronization overhead is to high for multiple models
- Fujimoto:
  - „Parallel Simulation: Will the field Survive?“

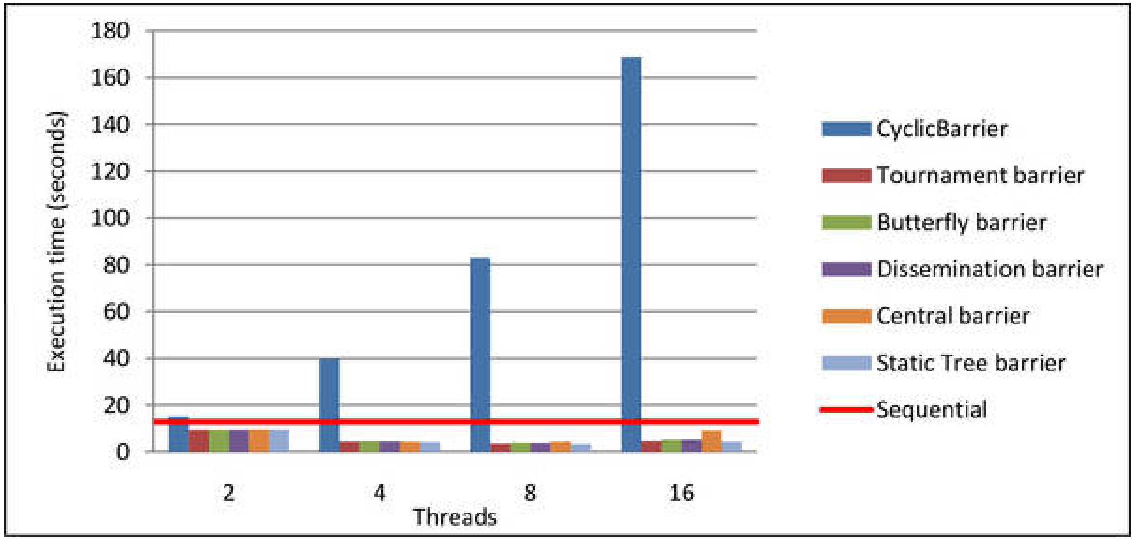
What works in Practice

- Closed workload models
- Open workload models
  - Can be processed similar to a batch process
  - Technical Solution: Virtual time steps
  - Consequence: Conservative simulation to reduce overheads



Synchronization

- Java SE Barriers perform bad on small time slices
- Barrier synchronization in Java [Ball03]
  - Active Wait
  - Hierarchical Barriers



Barrier synchronization available at:
http://net.cs.uni-bonn.de/wg/cs/applications/jbarrier/

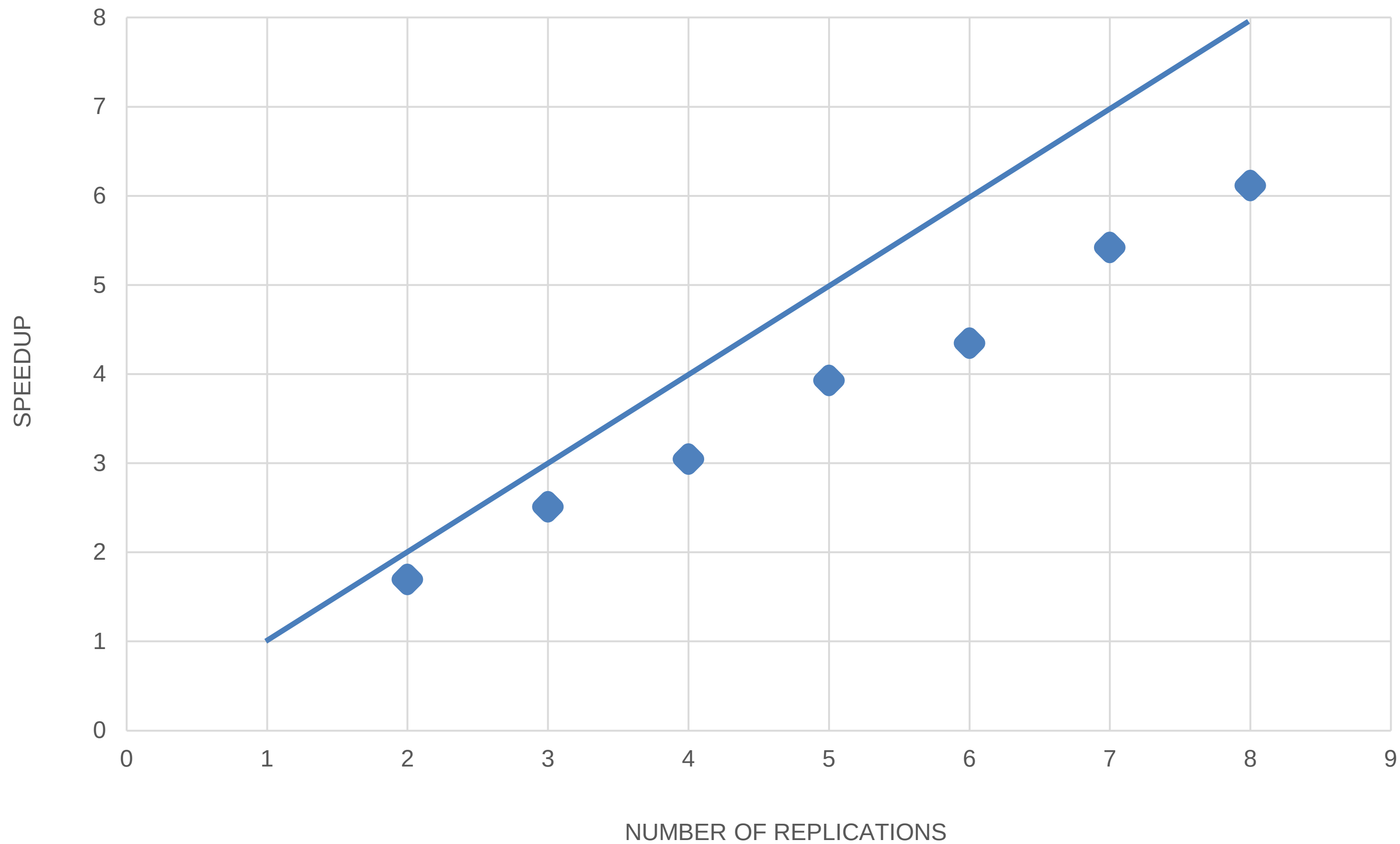
Contributions

- QPN decomposition
  - Applicability of existing Petri Net rules
  - Introduction of own merging rules
- QPN lookahead improvement by the use of queueing network best practices
- Implementation of parallel SimQPN version
  - Application level
  - Event level

Evaluation

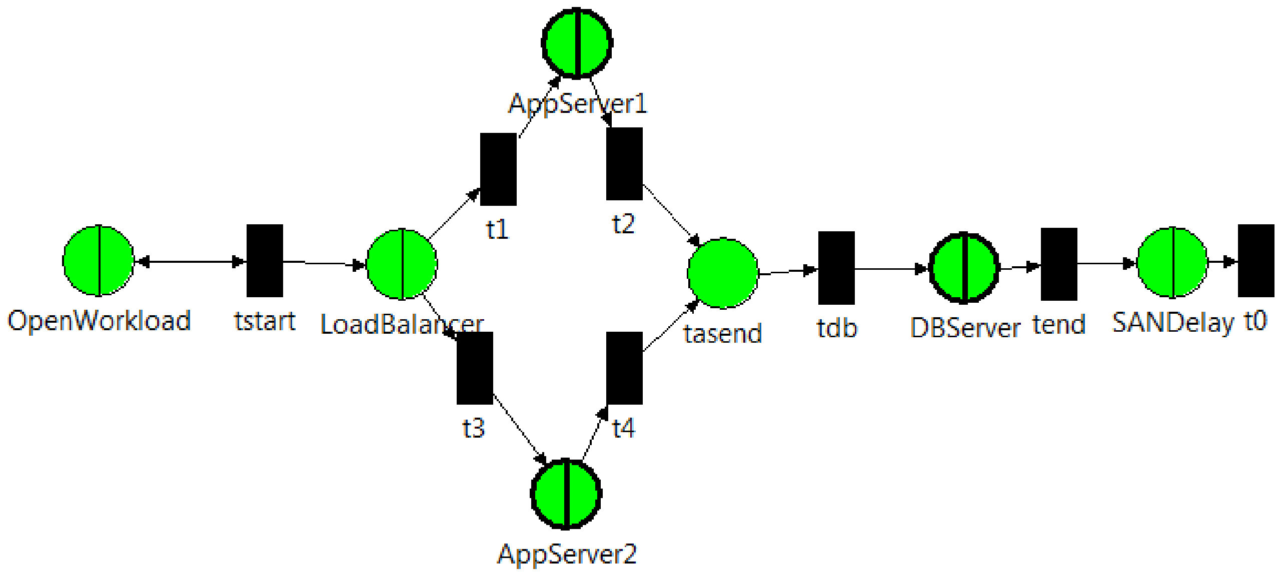
CASE STUDIES

Case Study: Application Level



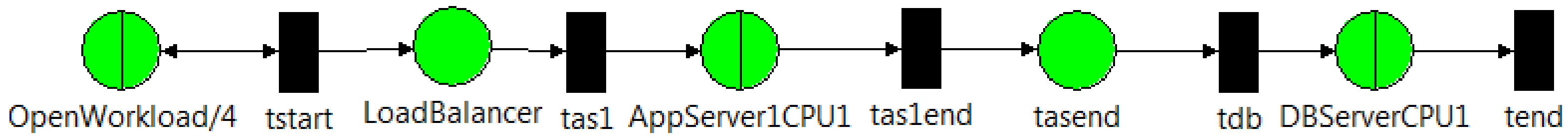
- Similar curve for all tested models

Case Study: Small Model



- Model provided by a big cloud provider
- Even more reduced …

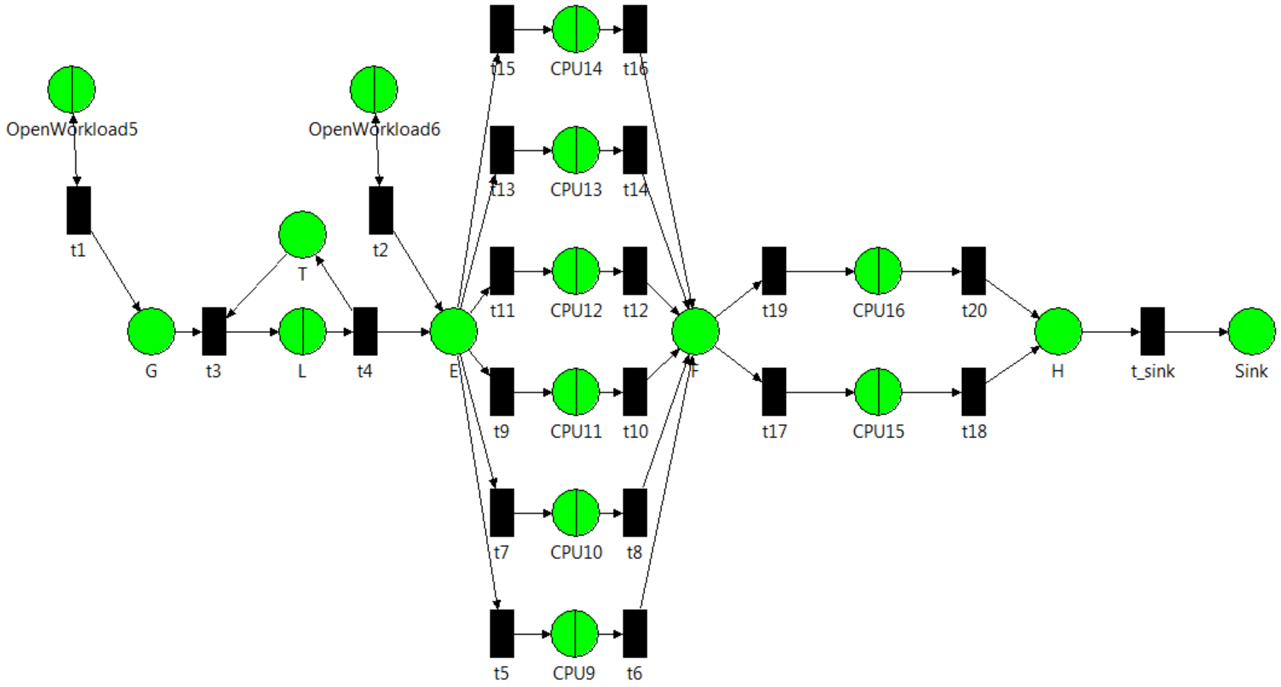
Case Study: Small Model



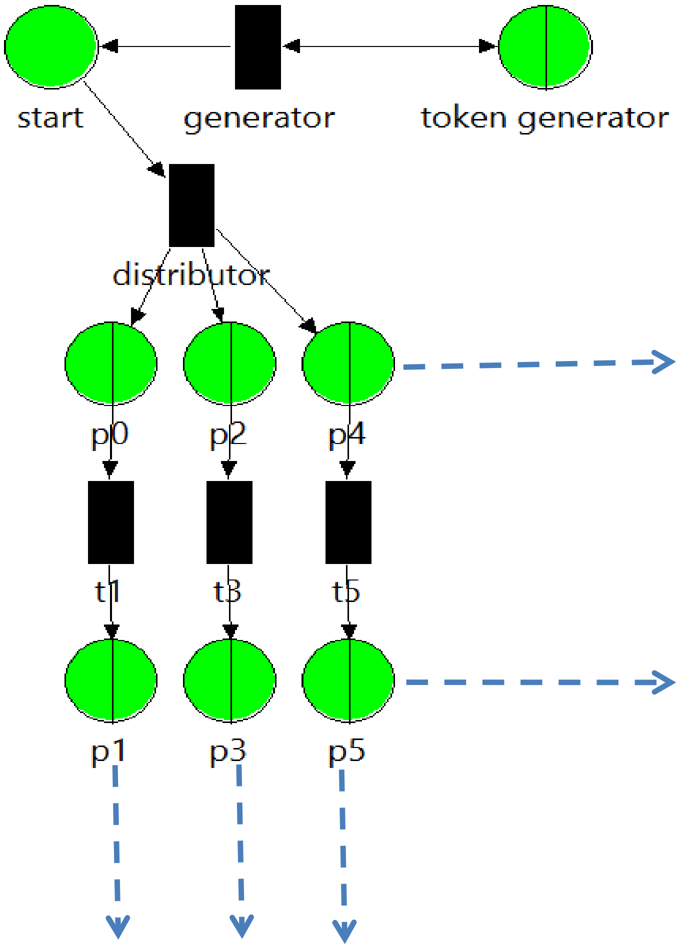
- Model provided by a big cloud provider
- Average speedup 1,91

Case Study: SPECj App Server

- Decomposition with heuristics into four logical processes
- Speedup of 2,45 but we expect decomposition not to be optimal



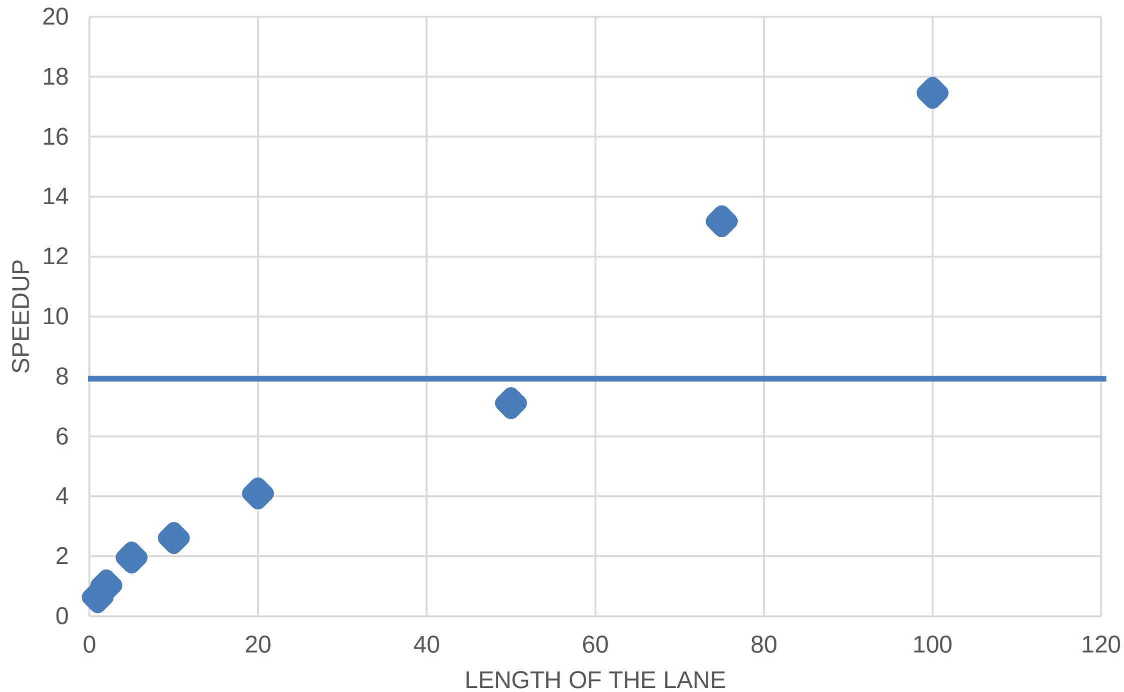
Case Study: Artificial Model



Model Choice

- Speedup heavily depends on model characteristics
- Use of a generated model
- Example shows 3x2 model

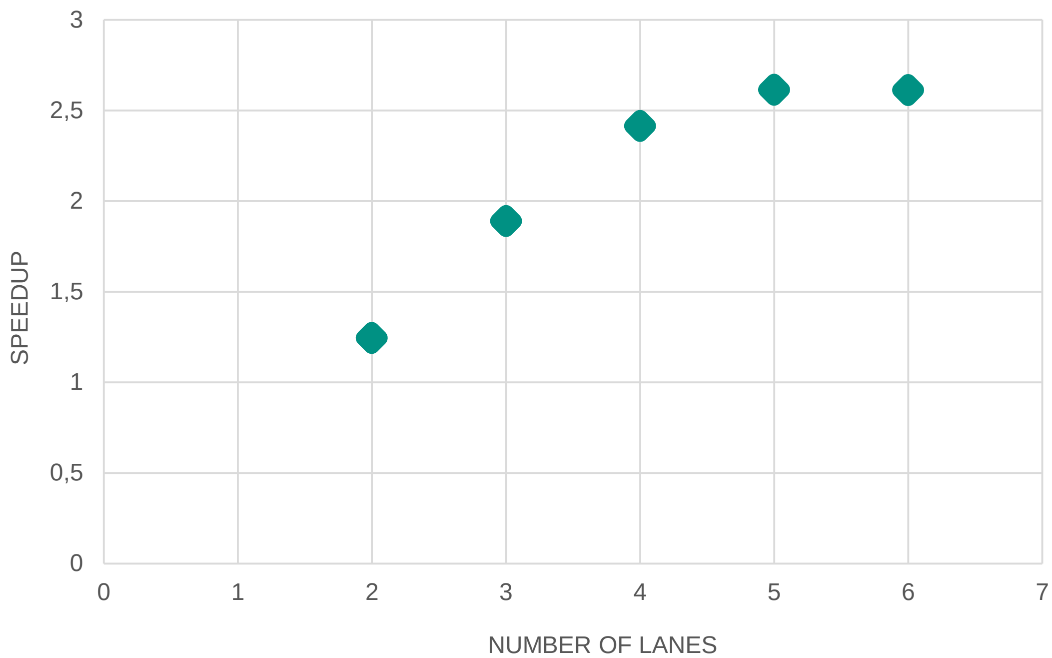
Case Study: Artificial Model



Synchronization Interval Length

- Model: 6 x [length of the lane]
- Less synchronization, higher speedup
- Speedup depends on model

Case Study: Artificial Model



Barrier Contention

- Model: [number of lanes] x 10
- More LPs, more contention for the barrier

Summary

- Actions
  - Survey of techniques
  - Parallel simulation engine
    - Event level
    - Application level

- Benefits
  - Parallel simulation runs faster than sequential.
  - SimQPN is applicable to more scenarios.

- Future Work
  - Automate decomposition
  - Apply to more case studies

Thank you for your attention!
Questions?

References

Bause93a	F. Bause, “Queueing petri nets-a formalism for the combined qualitative and quantitative analysis of systems," in Petri Nets and Performance Models, 1993. Proceedings., 5th International Workshop on, oct 1993, pp. 14-23.
Bause93b	F. Bause, “Qn + pn = qpn - combining queueing networks and petri nets," Department of CS, University of Dortmund, Germany, Technical Report No.461,1993
Lin1990	Y.-B. Lin and E. Lazowska, “Exploiting lookahead in parallel simulation," Parallel and Distributed Systems, IEEE Transactions on, vol. 1, no. 4, pp. 457-469, 1990.
Kounev07	Samuel Kounev and Alejandro Buchmann. *Petri Net, Theory and Application*, chapter On the Use of Queueing Petri Nets for Modeling and Performance Analysis of Distributed Systems. Advanced Robotic Systems International, I-Tech Education and Publishing, Vienna, Austria, 2 2007.
Fujimoto99	R. Fujimoto, “Parallel and distributed simulation," in Simulation Conference Proceedings, 1999 Winter, vol. 1, 1999, pp. 122 - 131 vol.1.

References

Fujimoto00	R. Fujimoto, “Parallel and Distribution Simulation Systems”, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 2000.
Chiola93	G. Chiola and A. Ferscha, “Distributed simulation of timed petri nets: Exploiting the net structure to obtain eciency," in Application and Theory of Petri Nets 1993, ser. Lecture Notes in Computer Science, M. Ajmone Marsan, Ed. Springer Berlin Heidelberg, 1993, vol. 691, pp. 146-165.
Nketsa01	A. Nketsa and N. B. Khalifa, “Timed petri nets and prediction to improve the chandy-misra conservative-distributed simulation," Applied Mathematics and Computation, vol. 120, no. 1-3, pp. 235 - 254, 2001
Fang07	X. Fang, Z. Xu, and Z. Yin, “Distributed processing based on timed petri nets," in Proceedings of the Third International Conference on Natural Computation - Volume 05, ser. ICNC '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 287-291.