Parallel Simulation of Queueing Petri Nets

Jürgen Walter
Dept. of Computer Science, University of Würzburg

Symposium on Software Performance,
Nov 27th 2014, Stuttgart, Germany
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
Queueing Petri Nets (QPN)
- Petri Nets (PN)
- Queueing Networks (QN)

Model Parts
- Places
- Transitions
- Token
- Queues

[Bause93a] [Bause93b]
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
- Logical Process (LP)
- Synchronization
  - Conservative
  - Optimistic
- Lookahead

Focus on parallel simulation
Simulate subparts of simulation model
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

Queueing Networks
[Wagner89] [Lin90]

Concurrent Simulation Algorithms
[Fujimoto99] [Fujimoto00]

Parallel QPN Simulation

Petri Nets
[Chiola93], [Nketsa01], [Fang07] and [Jürgens97]

Motivation Foundations Related Work Approach Case Studies Conclusions

Jürgen Walter - Parallel Simulation of Queueing Petri Nets
How to Parallelize Simulation

APPROACH
Parallelization Levels

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
- 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
Spatial decomposition
Minimum Regions [Chiola93]
Merging Rules [Chiola93]
Parallel simulation works on a theoretical basis for every kind of model.

However:
- Event processing in few microseconds
- Synchronization overhead is too high for multiple models

Fujimoto:
- „Parallel Simulation: Will the field Survive?“
What works in Practice

- Closed workload models
  - Cannot be processed similar to a batch process

- 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
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

Jürgen Walter - Parallel Simulation of Queueing Petri Nets
Case Study: Artificial Model

Synchronization Interval Length

- Model: $6 \times$ [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?
<table>
<thead>
<tr>
<th>Reference</th>
<th>Citation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>Title and Details</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
</tbody>
</table>