A Simulation Methodology for Worst-Case Response Time Estimation of Distributed Real-Time Systems

Soheil Samii, Sergiu Rafiliu, Petru Eles, Zebo Peng
Embedded Systems Laboratory
Department of Computer and Information Science
Linköpings universitet
Sweden
Outline

• Motivation and background
• Simulation environment
• Example
• Solution overview
• Experiments
• Conclusions
Motivation and background

- Real-time systems: timing characteristics are of interest
- In this paper: worst-case response times (WCRTs) of the processes in the applications
- Analytical methods
  - Pessimistic upper bounds
  - May lead to overdesigned systems and underutilized resources
  - Available only for restricted application models and execution platforms (e.g., communication protocols)
Motivation and background

• Simulation-based approach
  • Practical when no analysis is available
  • Not pessimistic (but optimistic lower bounds)
    – Avoid overdesign
    – Complements analysis
    – Validation of timing analysis w.r.t. pessimism

How to drive the simulator towards WCRT?

BCRT

WCRT

Response time

Lower bound on WCRT

Pessimism

Maximum pessimism

Upper bound on WCRT

Soheil Samii, Sergiu Rafiliu, Petru Eles, Zebo Peng
Simulation environment

Application model
Execution platform

Code

Functional output

Simulator kernel

Specification of possible execution times

Mapped to

Timing properties

Soheil Samii, Sergiu Rafiliu, Petru Eles, Zebo Peng
• Jobs are released at certain moments in time
  • Periodic release
• A job has an execution time
  • Execution time in $[BCET, WCET]$
Response time

Response time = $t_{\text{finish}} - t_{\text{release}}$

- Response time of a job (of a process)
  - Its execution time
  - Execution of higher-priority jobs
  - Time to wait for messages (communication delay)
Observations

- The number of execution scenarios is huge
  - Most of them do not lead to the WCRT
- The scenario where all jobs execute for their WCET does not necessarily produce the WCRT
Example

- $C_{P_1}=10$, $C_{P_2}$ in $[25, 35]$, $C_{P_3}=10$, $C_{P_4}=30$ (execution times)
- P4 has lowest priority
- Instantaneous communication

How to produce the scenario that results in the WCRT of a process?

Scheduling anomaly
Solution overview

• Choose between all points in [BCET, WCET]
• Intelligently reduce the execution time candidates to a discrete set

Execution-time space

Reduced execution-time space

Reduced space cannot be simulated in affordable time
Solution overview

• How to explore the reduced execution-time space to reach a good solution?

1. Execution-time space reduction
2. Execution-time space exploration
Execution-time space reduction

• Corner-case reduction (CC)
  • For each job, choose either the BCET or the WCET
  • Intuition and experiments: extreme cases produce usually large response times

![Graph showing BCET and WCET for execution time j]
Execution-time space reduction

- Improved corner-case reduction (ICC)
  - Find additional points (related to scheduling anomalies)
  - Analysis by Racu and Ernst (RTAS’06)

\[
\text{WCRT}_i \quad \xrightarrow{\text{Scheduling anomaly}} \quad \text{Execution time } j
\]

Point of interest in simulation
Execution-time space exploration

• How do we choose job execution times at a given point during simulation?

• Random exploration
  • Initial space of execution times
    – Choose randomly
  • Corner-cases (CC) and improved corner-cases (ICC)
    – Randomly
    – Intuition and experiments: more towards WCET

Soheil Samii, Sergiu Rafiliu, Petru Eles, Zebo Peng
Execution-time space exploration

- Optimization problem
  - Cost function: The response time of a process
    - Given by the simulator
  - Variables: job execution times
    - Execution-time generator
- Genetic algorithm-based exploration
  - Developed for CC and ICC
## Summary of approaches

<table>
<thead>
<tr>
<th></th>
<th>Initial</th>
<th>CC</th>
<th>ICC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>R-Initial</td>
<td>R-CC</td>
<td>R-ICC</td>
</tr>
<tr>
<td>GA</td>
<td>-</td>
<td>GA-CC</td>
<td>GA-ICC</td>
</tr>
</tbody>
</table>
Experiments: System architecture

Processes
- Execution time in [BCET, WCET]
- Jobs released periodically
- Priority-based scheduling

Messages
- CAN: message priorities
- FlexRay: TDMA + dynamic segment

Soheil Samii, Sergiu Rafiliu, Petru Eles, Zebo Peng
Experiments

• Compare the approaches with respect to producing large response times
  • Generated applications with varying timing characteristics and varying data dependency structures

• Reference point: in-house analysis tool (WCRT is unknown)
  • Ratio = $R_{\text{sim}} / R_{\text{analysis}}$

• For each approach:
  • Average ratio
  • Number of times the approach found the best solution (among all approaches)
## Experiments

<table>
<thead>
<tr>
<th>Approach</th>
<th>Average ratio [%]</th>
<th>Frequency [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-Initial</td>
<td>77.6</td>
<td>0</td>
</tr>
<tr>
<td>R-CC</td>
<td>87.3</td>
<td>30</td>
</tr>
<tr>
<td>R-ICC</td>
<td>87.4</td>
<td>32.9</td>
</tr>
<tr>
<td>GA-CC</td>
<td>88.0</td>
<td>41.4</td>
</tr>
<tr>
<td>GA-ICC</td>
<td>90.5</td>
<td>97.1</td>
</tr>
<tr>
<td>Only WCET</td>
<td>83.7</td>
<td>0</td>
</tr>
</tbody>
</table>

- All approaches have run for the same amount of time
  - Up to 10 minutes
  - On average: 100 seconds
Experiments: Pessimism estimation

- Maximum pessimism = (R_analysis – R_sim) / R_sim
- CAN- and FlexRay-based systems
Pessimism estimation - CAN

Soheil Samii, Sergiu Rafiliu, Petru Eles, Zebo Peng
Conclusions

- Simulation methodology for WCRT estimation of distributed real-time systems
  - Reduce the space of execution times
  - Efficient exploration strategy
- Useful approach:
  - No analysis tool available
  - Avoid overdesign when deadline misses can be tolerated
  - Validate a timing analysis
A Simulation Methodology for Worst-Case Response Time Estimation of Distributed Real-Time Systems

Soheil Samii, Sergiu Rafiliu, Petru Eles, Zebo Peng
Embedded Systems Laboratory
Department of Computer and Information Science
Linköpings universitet
Sweden
Case study

- Automotive cruise-controller application: 28 processes mapped to 5 computation nodes
- Analyzed 2 processes that produce the control data

- CAN implementation
  - 35.2% and 8.5% pessimism
- FlexRay implementation
  - 39.6% and 6.7% pessimism

- Pessimism relatively small $\Rightarrow$ the implementations are tight and cost efficient