Update on AADLInspector and Cheddar: new interface and multiprocessors analysis


* Ellidiss Technologies
+ Lab-STICC UMR CNRS 6285/UBO
$ ReDCAD Laboratory, University of Sfax
Summary

1. AADLInspector & Cheddar

2. AADLInspector 1.6 : new features

3. Multiprocessor analysis features
   1. Features in AADLInspector
   2. Current research activities outcomes (in Cheddar but not yet in AADLInspector)
AADLInspector
Model Processing Framework

AADL model

Cheddar

Marzhin

www.ellidiss.com
Cheddar
Scheduling analysis framework

CheddarADL

Simulator

Schedulability tests
From AADLInspector to Cheddar

AADLInspector

Industrialization
Tool packaging
Commercial support (Ellidiss)

R&D,
collaborative projects,
prototyping
(UBO + Ellidiss + others)

Cheddar

Research activities (Lab-STICC/UBO)
Summary

1. AADLInspector & Cheddar

2. AADLInspector 1.6 : new features

3. Multiprocessor analysis features
   1. Features in AADLInspector
   2. Current research activities outcomes (in Cheddar but not yet in AADLInspector)
AI 1.6 Features

- **Imports XML/XMI models:**
  - generic transformation process for ECore based models using LMP
  - existing prototypes for UML/MARTE, SysML, Capella, …

- **AADL model processing:**
  - turnkey embedded tools:
    - Cheddar (scheduling analysis)
    - Marzhin (event based simulation)
    - Ocarina (AADL compliancy analysis, code generation)
  - customizable plugins using the LMP AADL toolbox:
    - AADL parser (aadlrev)
    - AADL processing libraries (instance model, legality rules, …)

- **AADL projects manager:**
  - core 2.2 + annex sub-languages EMV1, EMV2, BA 2.0
  - interface with other AADL editors (Osate, Stood, …) and github
  - hierarchical

- **Improved simulation interface**
AADL Projects manager

- **Hierarchical project structure:**
  - AADL environment (libraries, property sets)
  - Sharable sub-projects
  - Simulation scenarios
  - Documentation sections
New presentation of the analysis results

Cheddar Theoretical Tests

Cheddar Simulation

Marzhin Simulation

<table>
<thead>
<tr>
<th>Deadline</th>
<th>Computed</th>
<th>Max Cheddar</th>
<th>Max Marzhin</th>
<th>Avg Cheddar</th>
<th>Avg Marzhin</th>
<th>Min Cheddar</th>
<th>Min Marzhin</th>
</tr>
</thead>
<tbody>
<tr>
<td>bus_scheduling</td>
<td>10</td>
<td>2.000000</td>
<td>2</td>
<td>2</td>
<td>2.00</td>
<td>2.00</td>
<td>2</td>
</tr>
<tr>
<td>data_distributio</td>
<td>10</td>
<td>4.000000</td>
<td>8</td>
<td>8</td>
<td>4.10</td>
<td>4.16</td>
<td>4</td>
</tr>
<tr>
<td>control_task</td>
<td>20</td>
<td>6.000000</td>
<td>10</td>
<td>10</td>
<td>6.20</td>
<td>6.32</td>
<td>6</td>
</tr>
<tr>
<td>radio_task</td>
<td>20</td>
<td>8.000000</td>
<td>16</td>
<td>16</td>
<td>8.40</td>
<td>8.64</td>
<td>8</td>
</tr>
<tr>
<td>camera_task</td>
<td>20</td>
<td>10.000000</td>
<td>18</td>
<td>18</td>
<td>10.40</td>
<td>10.64</td>
<td>10</td>
</tr>
<tr>
<td>mesure_task</td>
<td>400</td>
<td>18.000000</td>
<td>18</td>
<td>18</td>
<td>18.00</td>
<td>18.00</td>
<td>18</td>
</tr>
<tr>
<td>meteo_task</td>
<td>400</td>
<td>38.000000</td>
<td>26</td>
<td>26</td>
<td>26.00</td>
<td>26.00</td>
<td>26</td>
</tr>
</tbody>
</table>
New features to interact during simulation

- Scenarii
- Process I/O
- Event bases simulation
- Threads
- Response Time
- Simulation
- Threads
- Response Time
1. AADL Inspector & Cheddar

2. AADL Inspector 1.6: new features

3. Multiprocessor analysis features
   1. Features in AADL Inspector
   2. Current research activities (in Cheddar but not yet in AADL Inspector)
Multiprocessor with AADLInspector & Cheddar

- **SMART project and later:**
  - Define typical distributed/multiprocessor architectures AADLInspector should support
  - How to model such distributed/multiprocessor architectures with AADL
  - Choose or design scheduling analysis features for those patterns
  - Prototype in Cheddar, to be made available in AADLInspector

- **Focus on:**
  - Classical multiprocessor scheduling algorithms: partitioned vs global scheduling algorithms
  - Shared resources between processing units, e.g. cache, memory bus, NoC
Multiprocessor with AADLInspector & Cheddar

Partitioned scheduling: first assign offline each task on a processing unit; each processing unit schedules its own task set.
- No migration. Both online and off-line.

Global scheduling: choose the next task to run on any available processing unit (or preempt if all busy).
- With migration. Fully on-line.
Multiprocessor analysis features

- **AADLInspector 1.6**:  
  - Partitioned scheduling only  
  - Available partitioning policies: Best fit, First Fit, Next Fit, GT, SF

- **Cheddar 3.x only (not in AI yet)**:  
  - Global scheduling: any uniprocessor policies + specific policies such as EDZL, LLREF, Pfair, … *(finished now)*  
  - Shared resources on multiprocessor architectures: Cache, NoC, memory *(on going)*  
  - Partitioning optimization approaches based on PAES *(on going)*
In fixed priority preemptive scheduling context, tasks can preempt and evict data of other tasks in the cache.

Cache related preemption delay (CRPD): additional time to refill the cache with the cache blocks evicted by the preemption.

Problem statement:
- CRPD is high, non-negligible preemption cost. It can present up to 44% of the WCET of a task (Pellizzoni et al., 2007)
- No fixed priority assignment algorithm takes CRPD into account.
Cache/CRPD-Aware Priority Assignment Algorithm

- Extend Audsley’s priority assignment algorithm (Audsley, 1995) to take into account CRPD.
- CRPD-aware priority assignment algorithms (CPA) that assign priority to tasks and verify theirs schedulability.
- 5 algorithms with different levels of schedulability efficiency (1) and complexity (2,3).
- Implemented into Cheddar

<table>
<thead>
<tr>
<th></th>
<th>CPA-PT-Simplified</th>
<th>CPA-PT</th>
<th>CPA-Tree</th>
<th>Exhaustive Search</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>0.65</td>
<td>0.72</td>
<td>0.80</td>
<td>0.87</td>
</tr>
<tr>
<td>(2)</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
<td></td>
</tr>
<tr>
<td>(3)</td>
<td>100 tasks</td>
<td>30 tasks</td>
<td>10 tasks</td>
<td></td>
</tr>
</tbody>
</table>
Cache-Aware Scheduling Simulator

- **Problem Statement:**
  - Various parameters need to be taken into account in scheduling analysis of systems with cache.
  - Lack of tool addressing all parameters in the state-of-the-art work.
  - Theoretical issues (feasibility interval, sustainability)

<table>
<thead>
<tr>
<th>Scheduling analysis for systems with cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache Configuration</td>
</tr>
<tr>
<td>Memory Layout</td>
</tr>
</tbody>
</table>

- **Approach:**
  - Extend Cheddar component modeling related cache entities.
  - Extend Cheddar scheduling simulator.
Networks-on-Chip Aware Scheduling Analysis

- **Context:**
  - Networks-on-Chip (NoC)
  - Communication infrastructure based on links and routers that interconnect PU or memory unit, providing packet-based data transfer.

- **Problem statement:**
  - Relationships between thread models and communication models
  - Various AADL thread communication design patterns
  - Various NoC designs
  - Today: AADL data port & 4x4 Wormhole XY NoC

- How to model both thread and communication models in order to enforce schedulability?
Networks-on-Chip Aware Scheduling Analysis

- **Dual Task and Flow Model (DTFM):**
  - Computes the flow model from a task model, task mapping and precedence constraints
  - Identify/compute delays induced by the NoC architecture and perform scheduling analysis

- **DTFM Implemented into Cheddar**

- **DTFM is evaluated with a multiscale toolset**
  composed of a tick accurate real-time scheduling tool (Cheddar) and a cycle accurate SystemC NoC simulator (SHOC).
Partitioning methods, multi-objective optimization

Problem Statement:
- Mapping functions to software architectures (i.e. RTOS tasks)
- Conflicting objective functions, e.g. number of preemptions vs laxity
  - Tradeoffs between large number of candidate software architectures

Contributions:
- Formulation based on PAES (Pareto Archived Evolution Strategy) to explore possible functions to tasks assignments
- Implementation into Cheddar, both sequential and parallel implementation
- Uniprocessor only right now
Partitioning methods, multi-objective optimization

**Functional Specification**

**Initial architectural model**

**Initial current solution:** one task for each function

**Set of periodic independent Tasks**

$F_1$, $F_2$, ..., $F_n$

**Set of periodic independent atomic functions**

$F_1$, $F_2$, ..., $F_n$

**Schedulability Analysis**

By simulation with the *Cheddar* tool

**Pareto Archived Evolution Strategy (PAES) Formulation**

**Best tradeoff architectures (schedulable task sets)**

Not schedulable

Schedulable
Conclusion

- **New features in AADLInspector & Cheddar: about multiprocessors analysis:**
  1. Typical multiprocessor architectures (SMART project)
  2. Classical multiprocessor scheduling algorithms: partitioning vs global scheduling algorithms
  3. New analysis features when (hardware) shared resources between computing units

- **AADL models handled during those activities**
  - Bindings and Implemented_As ??
  - Need a white paper

- **Questions ?**