Efficient Execution of Dependent Tasks on Many-Core Processors
Hamza Rihani, Claire Maiza, Matthieu Moy
Univ. Grenoble Alpes Verimag

RTSOPS 2016, July 5, 2016
○ Hard real-time systems
○ Dependent tasks statically scheduled, on a many-core processor
  ⚠️ Unpredictable delays due to shared resource interference
Context

- Hard real-time systems
- Dependent tasks statically scheduled, on a many-core processor

⚠️ Unpredictable delays due to shared resource interference
✔️ Use tightly estimated upper bounds on delays
○ Hard real-time systems
○ Dependent tasks statically scheduled, on a many-core processor
  ❄️ Unpredictable delays due to shared resource interference
  ✅ Use tightly estimated upper bounds on delays
  ✅ Connect existing approaches for an optimally efficient execution
Outline

1 Solved Problems
   - Code Generation
   - Task Mapping
   - WCRT Analysis

2 Toward a Solution

3 The Open Problem
Solved Problems

High-level Program → Code Generation → Timing models (static analysis) → Local WCRT Analysis

Tasks + Dependencies

Static Mapping/Scheduling

Mapping

Execution Order

Release Dates

Binary Generation

WCRT with Interferences → Tasks WCRT + WC Access

Probabilistic Models

Executable Binary
Solved Problems

High-level Program → Code Generation →
Tasks + Dependencies → Static Mapping/Scheduling

→ Mapping → Execution Order → Release Dates

Binary Generation → WCRT with Interferences

→ Executable Binary
Solved Problems: Code Generation

**Outputs**
- Task binaries
- Task dependency graph
- Execution models: (Pellizzoni et al.[6])
  - Single phase execution
  - acquisition, execution, replication phases
Solved Problems: Task Mapping/Scheduling

- Static Mapping/Scheduling
- WCRT with Interferences
- High-level Program
- Executable Binary

- Tasks
- Dependencies
- Mapping
- Execution Order
- Release Dates
- Binary Generation

- PE0: wcrt0 → τ0
- PE1: wcrt1 → τ1
- PE2: wcrt2 → τ2
- WCRT with Interferences

- (*)wcrt_x: safe WCRT

- Respect the dependency constraints
- Optimize the overall response time

Puffitsch et al. 2013 [7], Giannopoulou et al. 2013 [4], Walter et al. 2015 [8]
Solved Problems: WCRT Analysis

- High-level Program
- Code Generation
  - Tasks
  - Dependencies
  - Static Mapping/Scheduling
  - Mapping
  - Execution Order
  - Release Dates
- Binary Generation
- Executable Binary

WCRT with Interferences:

- Take the interference into account
- Update the release times

(*)$wcrt_x^+$: refined WCRT

The overall response time may not be optimal.
Solved Problems: WCRT Analysis

The overall response time may not be optimal

(*)\text{wcrt}^+_x$: refined WCRT

- Take the interference into account
- Update the release times
Toward a Solution

Provide new timing information to the mapping/Scheduling analysis
○ **Mapping/Scheduling:**
  ○ Taking into account new timing information
  ○ Co-schedule communications and computations (Melani et al. 2015 [5])
  ○ Clustering non-interfering tasks (Choi et al. 2016 [2])
Toward a Solution

- **Mapping/Scheduling:**
  - Taking into account new timing information
  - Co-schedule communications and computations (Melani et al. 2015 [5])
  - Clustering non-interfering tasks (Choi et al. 2016 [2])

- **WCRT Analysis:**
  - Trade-off: run-time/ pessimism
  - Altmeyer et al. 2015 [1], Dasari et al. 2015 [3]
Toward a Solution

- Mapping/Scheduling:
  - Taking into account new timing information
  - Co-schedule communications and computations (Melani et al. 2015 [5])
  - Clustering non-interfering tasks (Choi et al. 2016 [2])

- WCRT Analysis:
  - Trade-off: run-time/ pessimism

Altmeyer et al. 2015 [1], Dasari et al. 2015[3]

Fixed-point search algorithms
Iterate until an optimal solution is found
What about convergence?
Iterate until an optimal solution is found
What about convergence?

Suboptimal:
  - Compute several solutions, choose the best one
  - **How many iterations?**
The Open Problem

Iterate until an optimal solution is found

What about convergence?

Suboptimal:

- Compute several solutions, choose the best one
- **How many iterations?**

Multi/Many-core processors are a game changer in the interaction between WCRT analysis and task mapping/scheduling
Efficient Execution of Dependent Tasks on Many-Core Processors

Hamza Rihani, Claire Maiza, Matthieu Moy

Univ. Grenoble Alpes
Verimag
A generic and compositional framework for multicore response time analysis.

J. Choi, D. Kang, and S. Ha.
Conservative modeling of shared resource contention for dependent tasks in partitioned multi-core systems.

D. Dasari, V. Nelis, and B. Akesson.
A framework for memory contention analysis in multi-core platforms.
G. Giannopoulou, N. Stoimenov, P. Huang, and L. Thiele.
Scheduling of mixed-criticality applications on resource-sharing multicore systems.

A. Melani, M. Bertogna, V. Bonifaci, A. Marchetti-Spaccamela, and G. Buttazzo.
Memory-processor co-scheduling in fixed priority systems.

Worst case delay analysis for memory interference in multicore systems.
W. Puffitsch, E. Noulard, and C. Pagetti.
Mapping a multi-rate synchronous language to a many-core processor.

J. Walter and W. Nebel.
Energy–aware mapping and scheduling of large–scale macro data–flow applications.