UMR IRISA

Hiding communication delays in contention-free execution for SPM-based multi-core architectures

Benjamin Rouxel, Stefanos Skalistis, Steven Derrien, Isabelle Puaut

firstname.lastname@irisa.fr

Institut de Recherche en Informatique et Systèmes Aléatoires

Target Systems

Embedded real-time systems

Single application

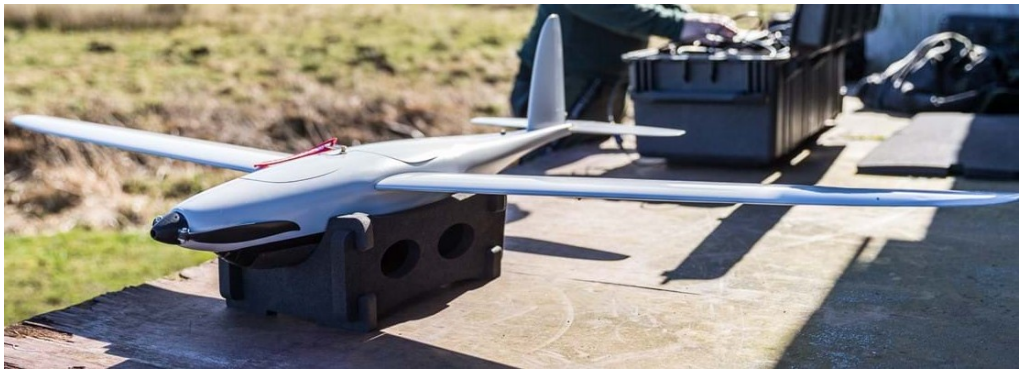
Set of tasks

Reliability

Safety

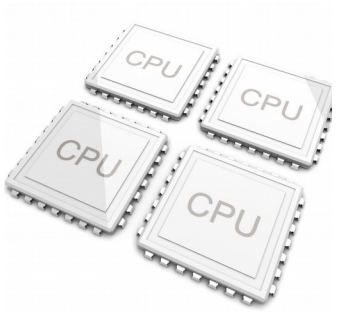
Demand for high-performance

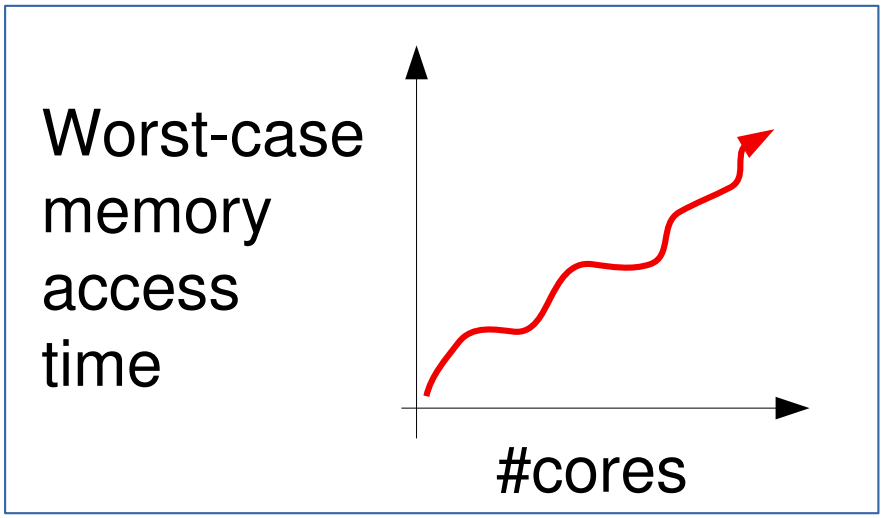
Multi-core architectures



Caveats in multi-/many-core usage

- **Resources sharing**
- **Data integrity**
- **Memory access bottleneck**



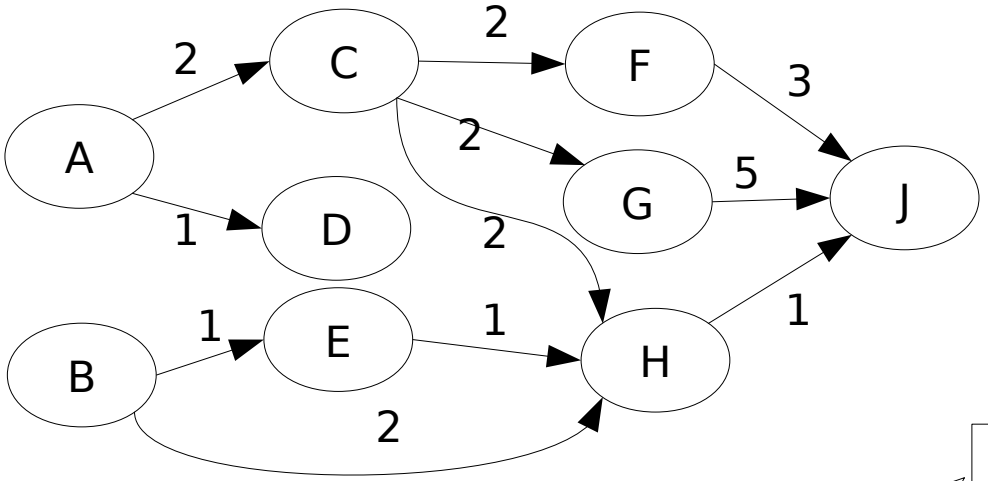


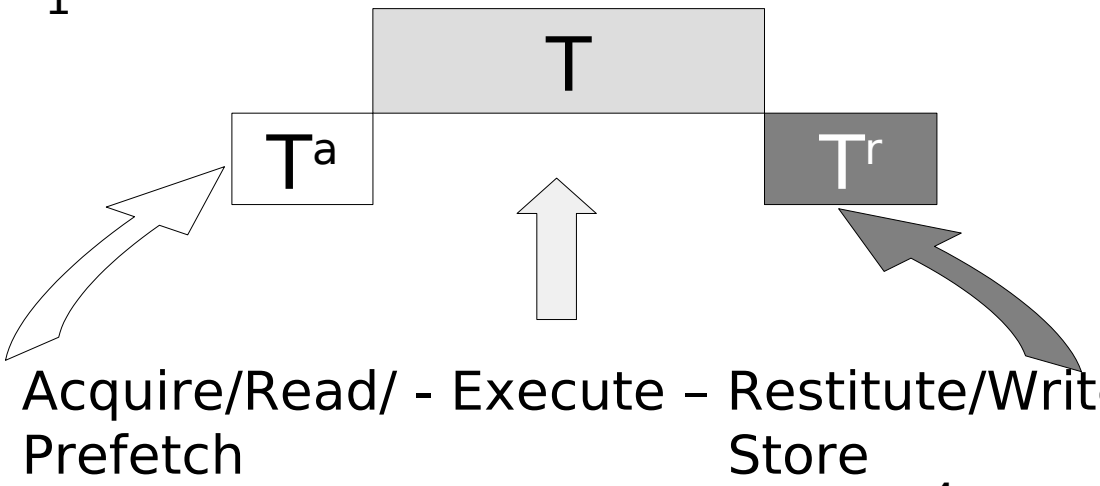
Let's try to avoid contention when scheduling off-line

Application & execution model

- **Directed Acyclic Graphs (DAG)**
- **Acquire Execution Restitution (AER) execution Model**

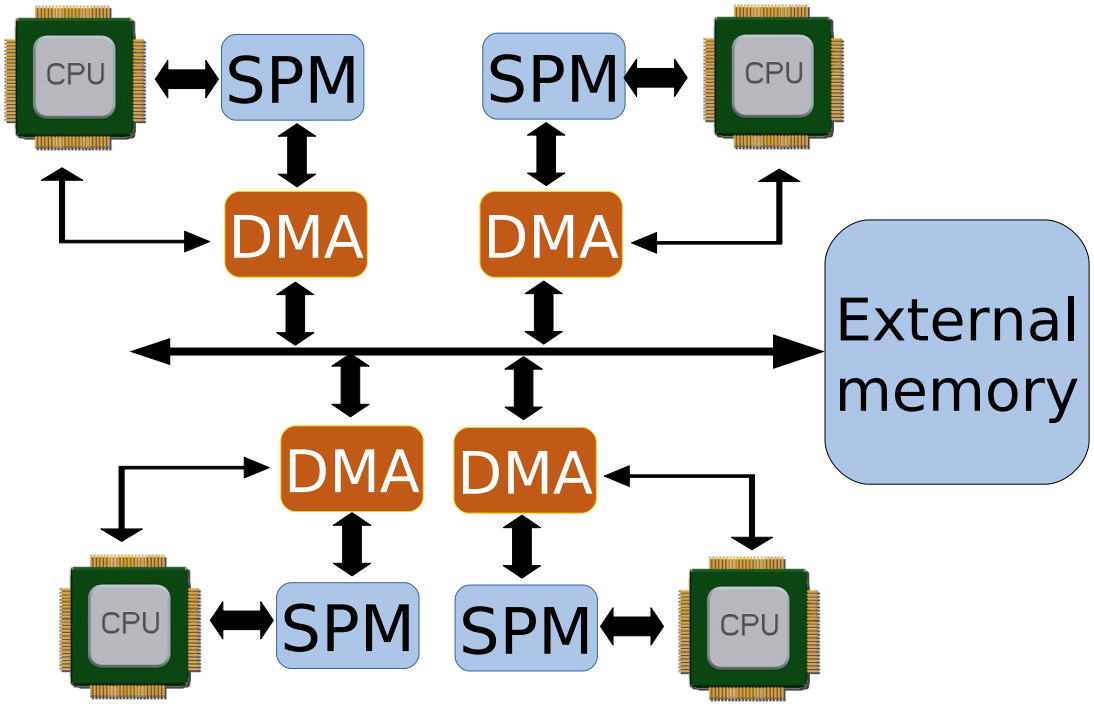
Maia'16



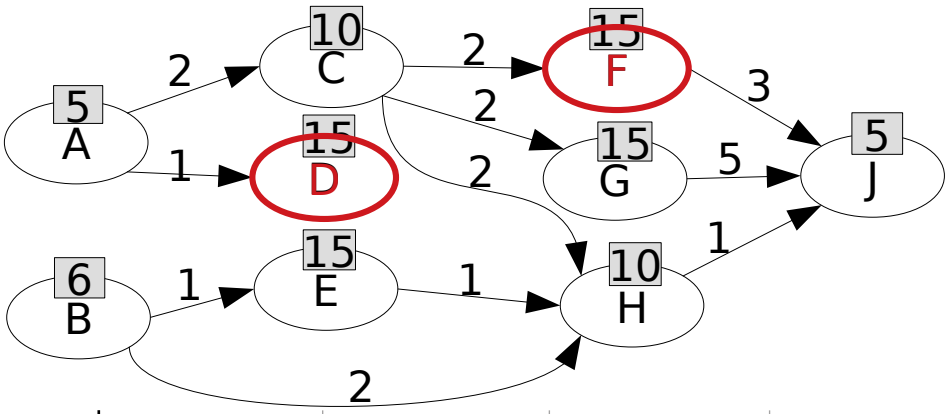


Multi-core architecture model

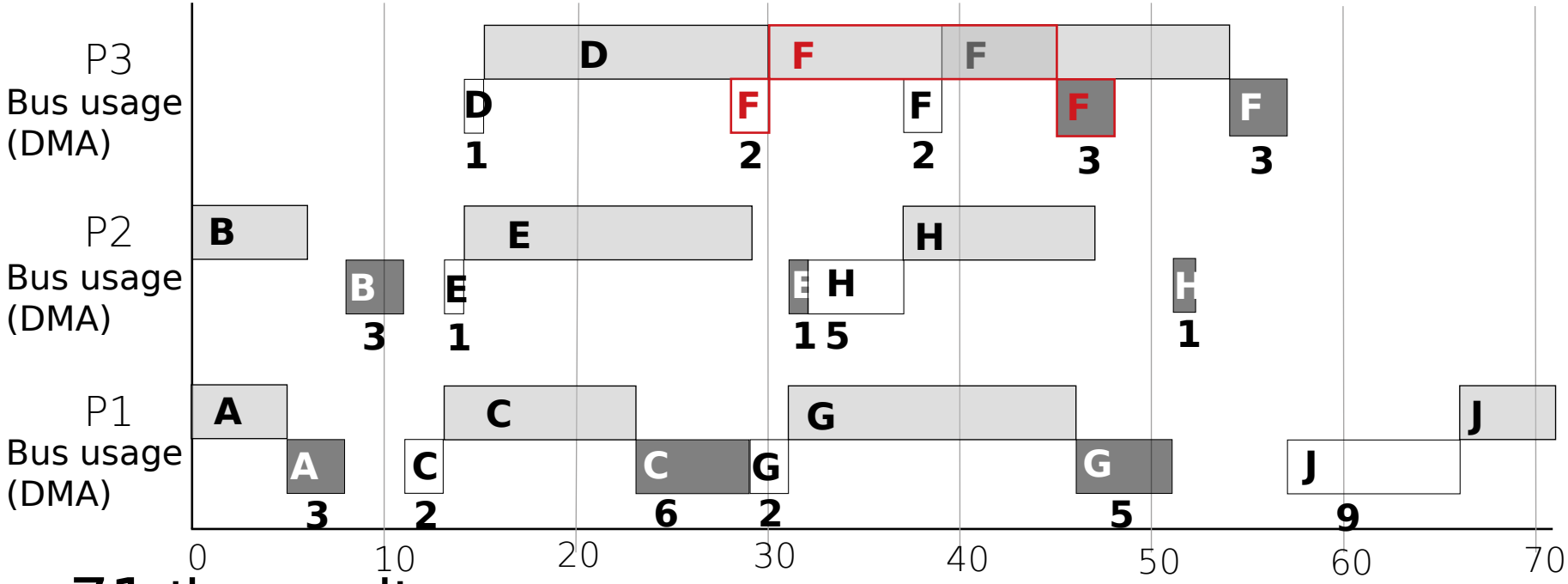
- ScratchPad Memory (SPM)
- Direct Memory Access engine (DMA)
- Dual-ported SPM
- Bus with FAIR round-robin arbitration



Building a better contention-free schedule

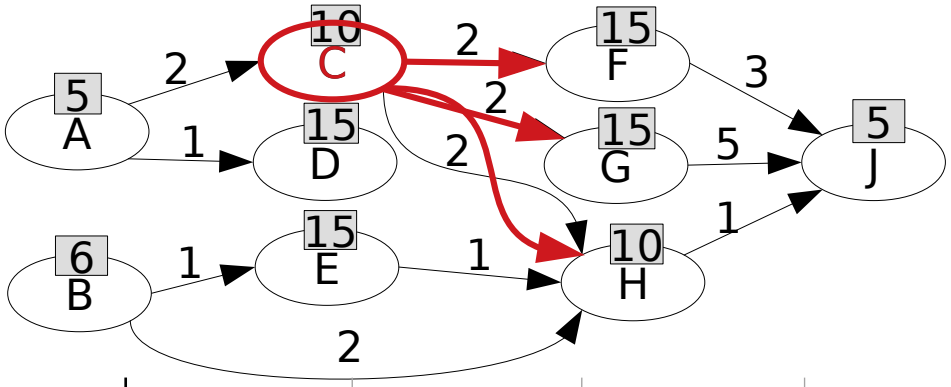


Hidden communications

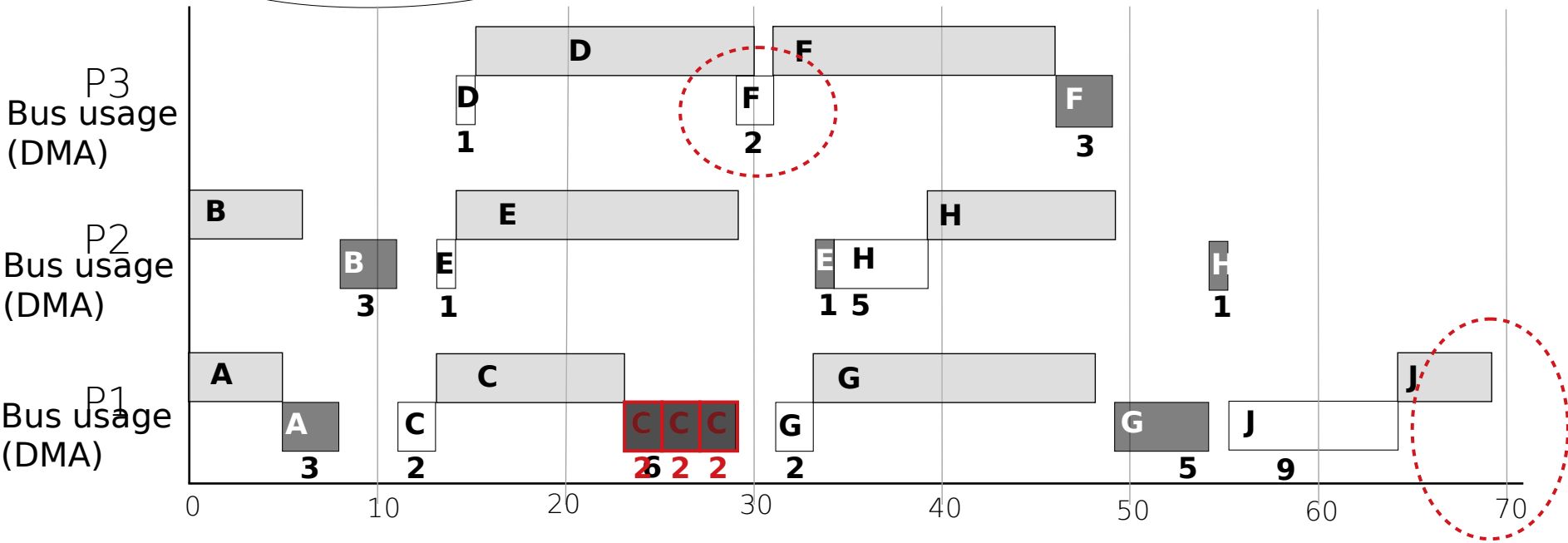


71 time units

Hiding communication

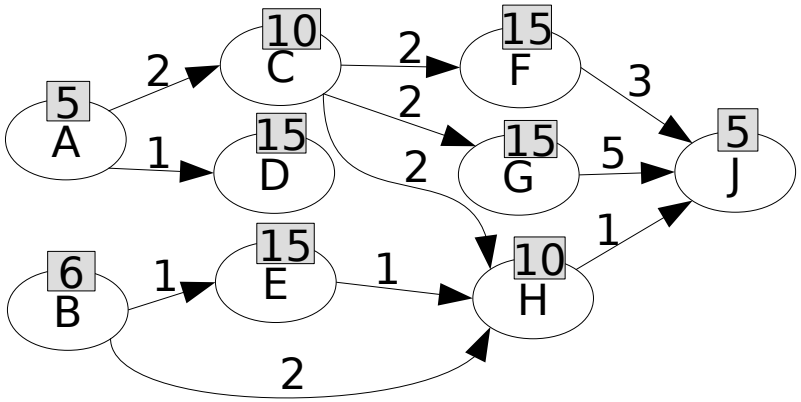


Fragmented communications

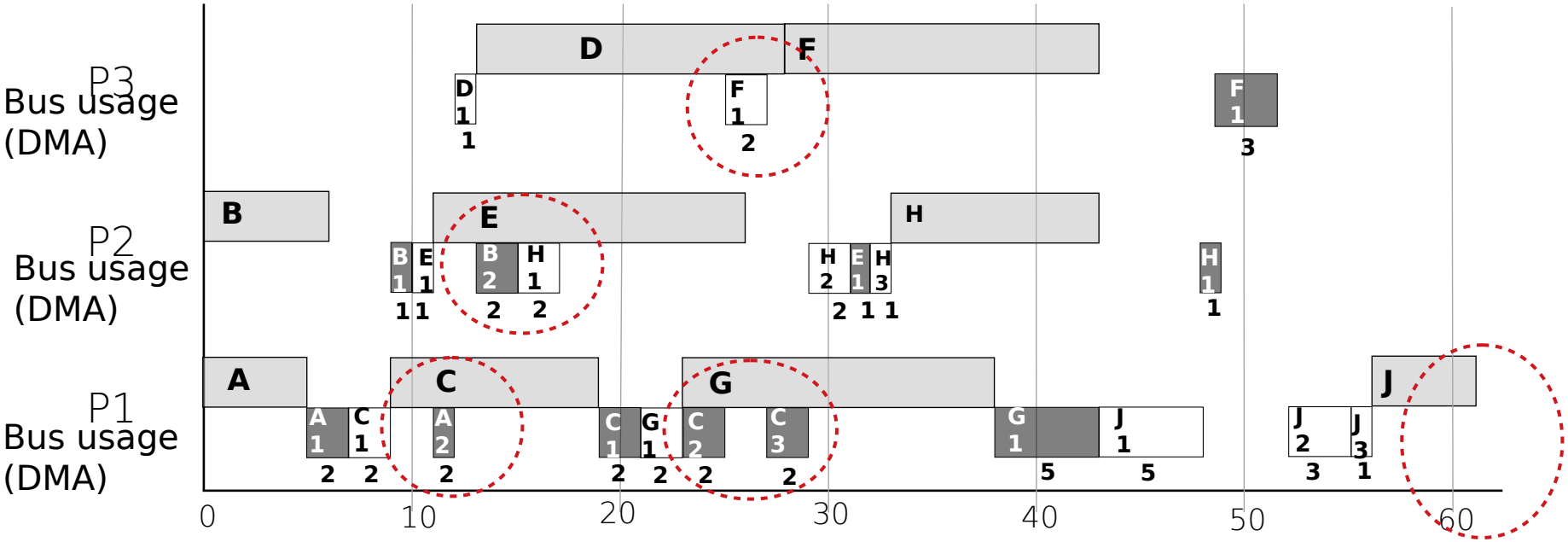


~~71~~ 69 time units

Hidden & Fragmented communications



Gain : 20%

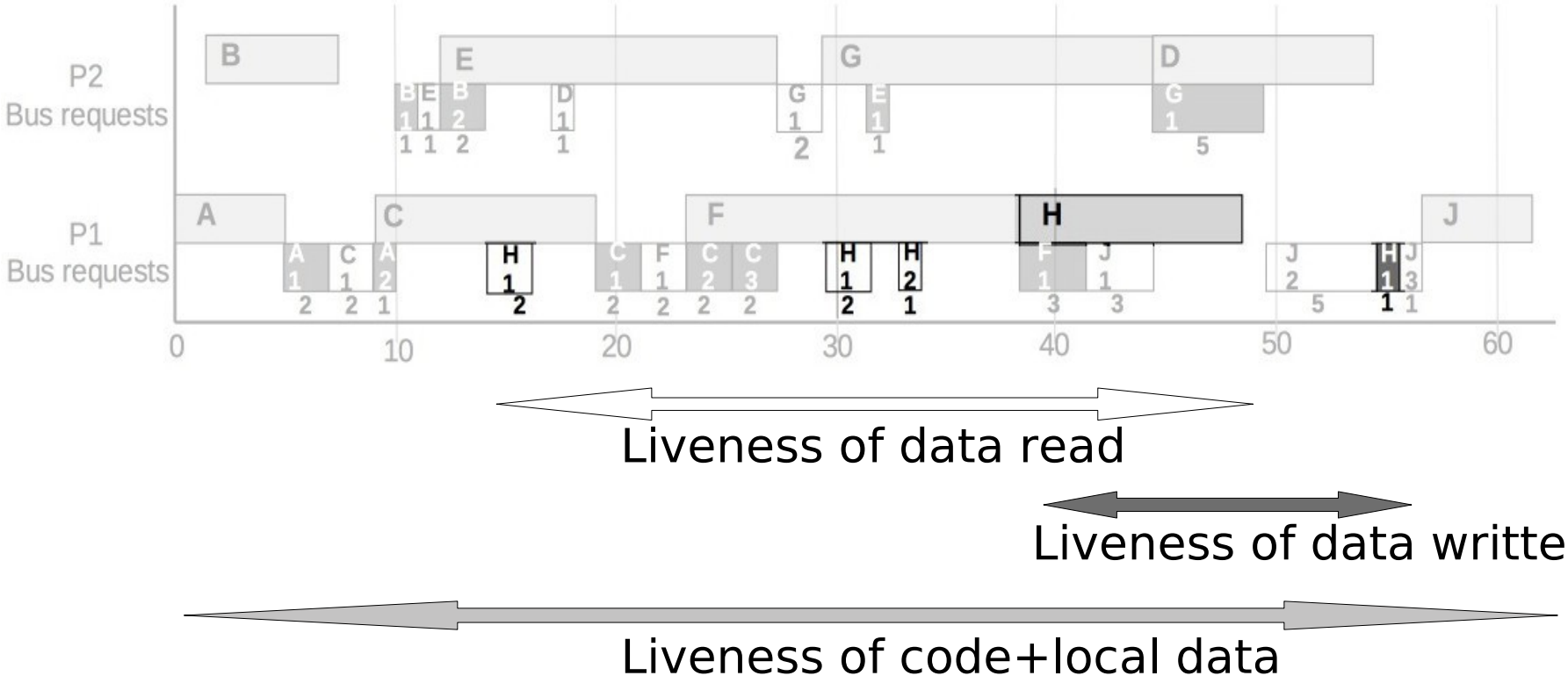


~~71~~ ~~69~~ 61 time units

SPM mapping scheme

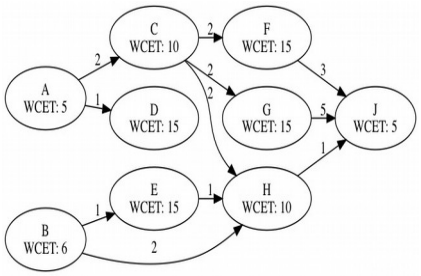
- **Split SPM into regions**
- **Map communication phases to regions**
- **Ensure data integrity**

Kim'2014

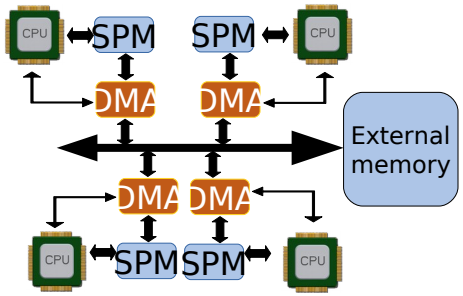


Hiding communication delays

Application graph



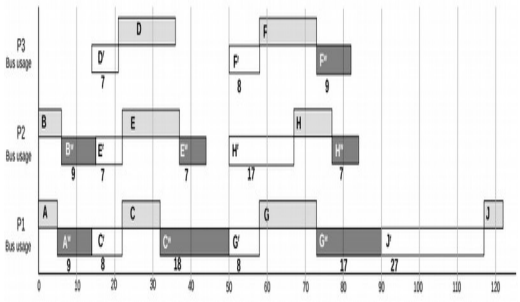
Multi-core architecture



Scheduling strategy

- **List scheduling**
- **Integer Linear Programming (ILP)**

Contention-free off-line schedule



- **Static**
- **Partitioned**
- **Time-triggered**
- **Non-preemptive**

Scheduling strategy: Forward List Scheduling

- **Statically sort tasks**
- **Try to map 1 task the earliest on each core while ensuring a proper schedule**
  - Communication phases can not overlap in time
  - Execution phases do not overlap in time on the same core
  - Precedencies between phases and tasks
- **Communication phases mapping to a SPM region**
  - Try to reuse an allocated region according to lifetime
  - If not found, create a new one
  - If not enough space, then throw Unschedulable
- **Keep the mapping/scheduling with minimal makespan**
- **Start again with an other tasks**

Reminder

- **Integer Linear Programming**
  - Objective function, maximise or minimise
  - Set of variables, and linear inequalities (constraints)
  - Optimal results
  - Non-ambiguous formulation

Experiments :
Heuristic results degradation vs ILP results

- **Task-graph Generator For Free (TGFF)** *Dick'1998*
- **Synthetic benchmarks, wide range of topologies**
- **Parameters:**
  - Number of graphs: 50
  - Number of tasks: [5 ; 69]
  - Number of cores: {2, 4, 8, 12}
  - SPM size: {4KB, 2MB}
  - Round-robin fair time T_{slot}: [1 ; 10]
  - Timeout: 11h

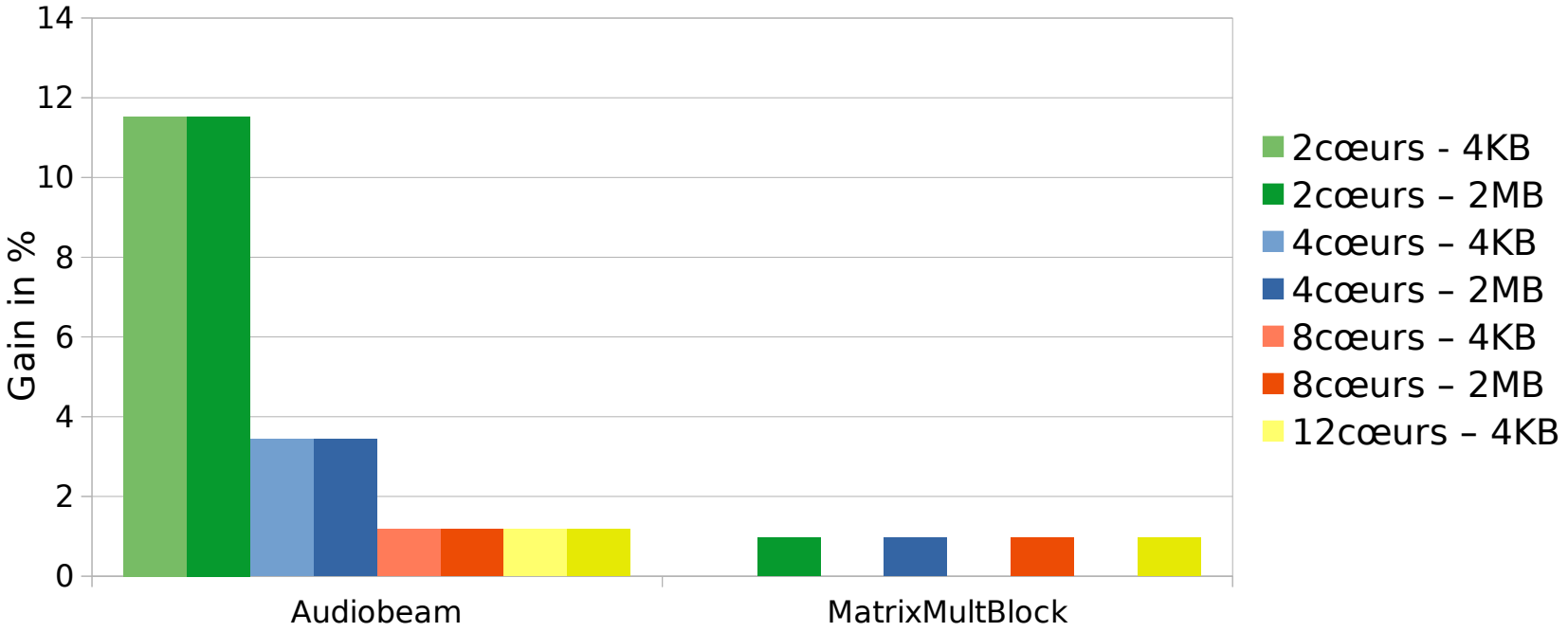
- **Degradation**
  - Minimal: 0 %
  - Average: 3 %
  - Maximal: 20 %

Experiments : Hidden & fragmented gain on STR2RTS

- **Benchmark suite STR2RTS** *Rouxel'2017*
- **Real test-cases, streaming applications**
- **Parameters:**
  - Number of used cases: 18
  - Number of tasks: [7,340]
  - Number of cores: {2, 4, 8, 12}
  - SPM size: {4KB, 2MB}
  - Round-robin fair time T_{slot}: [1 ; 10]

Experiments : Hidden & fragmented gain on STR2RTS

- **Base : non-hidden and non-fragmented**

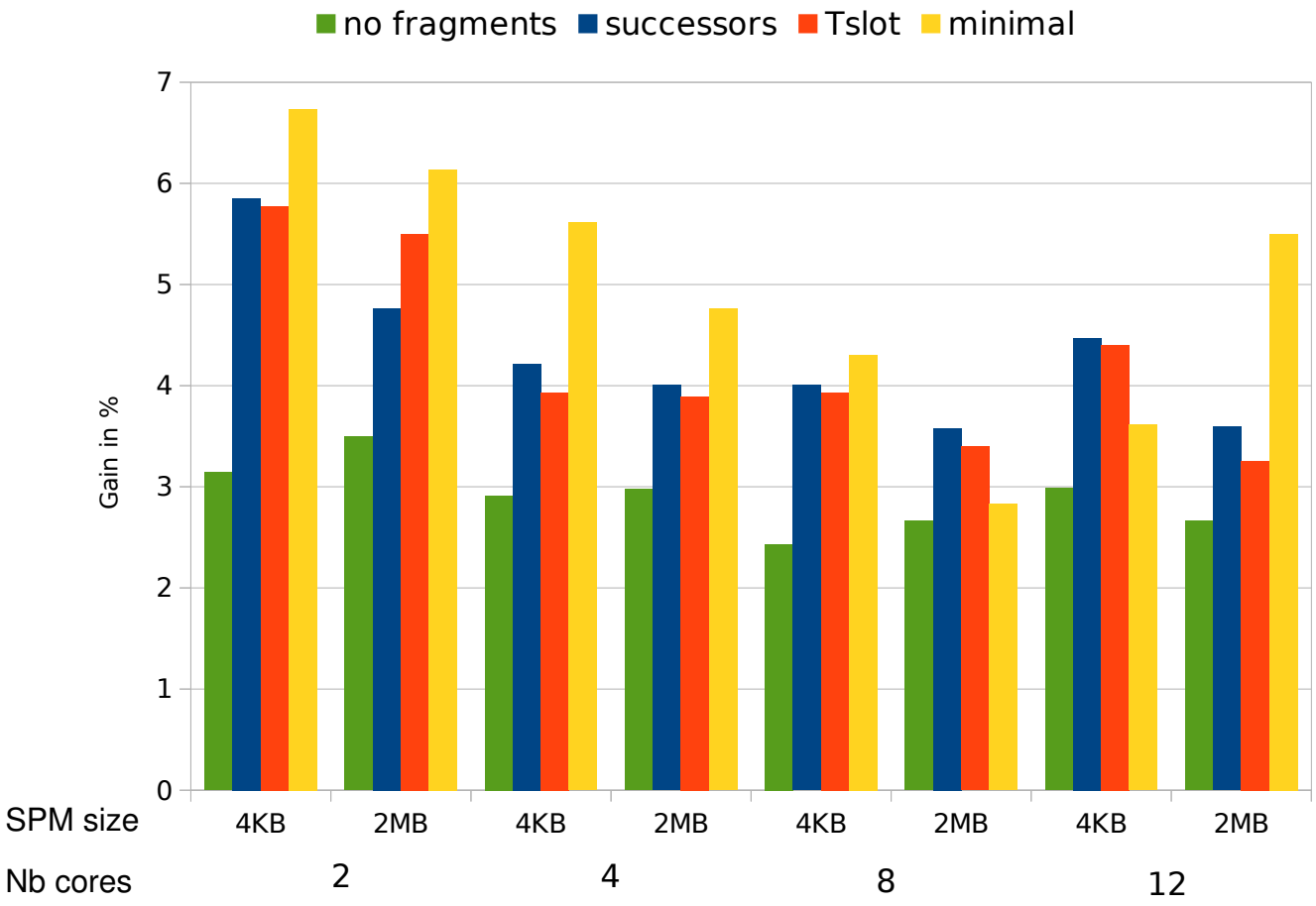


Average over all benchmarks 4%

Experiments : Hidden & fragmented gain on STR2RTS

- **Base: non-hidden and non-fragmented**
- **STR2RTS**
- **Fragmentation scheme:**
  - No fragmentation, hidden only
  - Successors communications
  - Round-Robin FAIR time configuration T_{slot}
  - Minimal size of exchanged data

Experiments : Hidden & fragmented gain on STR2RTS



Experiments : Schedule implementation

Platform

- **Kalray MPPA 256 Bostan**
- **1 cluster, 16 cores**
- **16 privatised memory banks**
- **8 buses, fix priority arbitration**

Limitations

- **Only 8 cores**
- **+1 core as software DMA**
- **Measured WCET & communication latencies**

Gain over non-fragmented communications

- **Successors: 36 %**
- **T_{slot}: 22 %**
- **Minimal: 12 %**

Conclusion

Summary

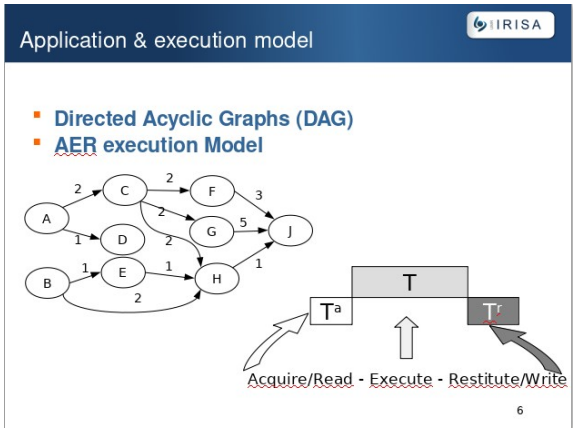
- **Tightening schedule makespan by hiding and fragmenting communications**
- **ILP & heuristic scheduling strategies**
- **Schedule implementation**

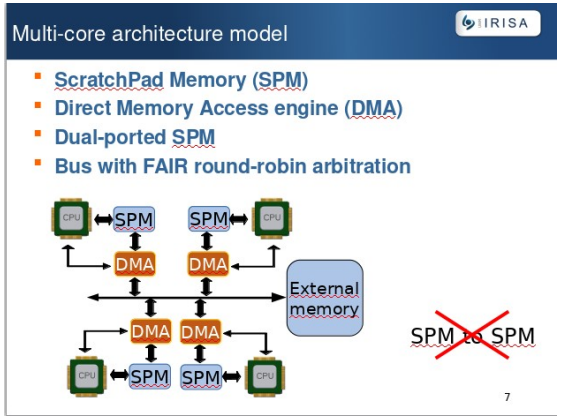
Results

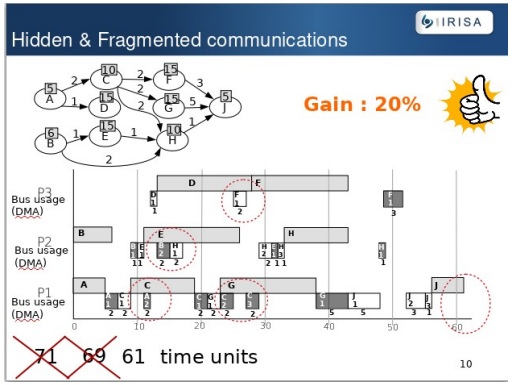
- **Average gain of 8% with real test-cases**
- **Finer fragments are not the best**

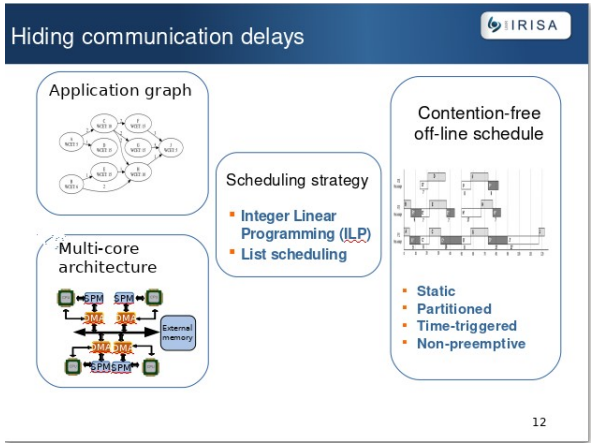
Future work

- **Allowing SPM-to-SPM communication**
- **NoC extension**
- **Improving schedule implementation**









https://gitlab.inria.fr/brouxel/methane

https://gitlab.inria.fr/brouxel/STR2RTS