Schedulability analysis for systems with data and control dependencies

Pop, Paul; Eles, Petru; Peng, Zebo

Published in:
Euromicro Conference on Real-Time Systems. Proceedings

Link to article, DOI:
10.1109/EMRTS.2000.854008

Publication date:
2000

Document Version
Publisher's PDF, also known as Version of record

Schedulability Analysis for Systems with Data and Control Dependencies

Paul Pop, Petru Eles, Zebo Peng

Department of Computer and Information Science
Linköpings universitet
Sweden

Outline

- Motivation
- System Model
- Problem Formulation
- Schedulability Analysis
- Experimental Results
- Conclusions

Motivation

Performance estimation:

- Based on schedulability analysis.

Schedulability analysis:

- Worst case response time of each process.
- Models in the literature:

    Independent processes;

    Data dependencies: *release jitter*, *offsets*, *phases*;

    Control dependencies: *modes*, *periods*, *recurring tasks*.

Characteristics and Message

Characteristics:

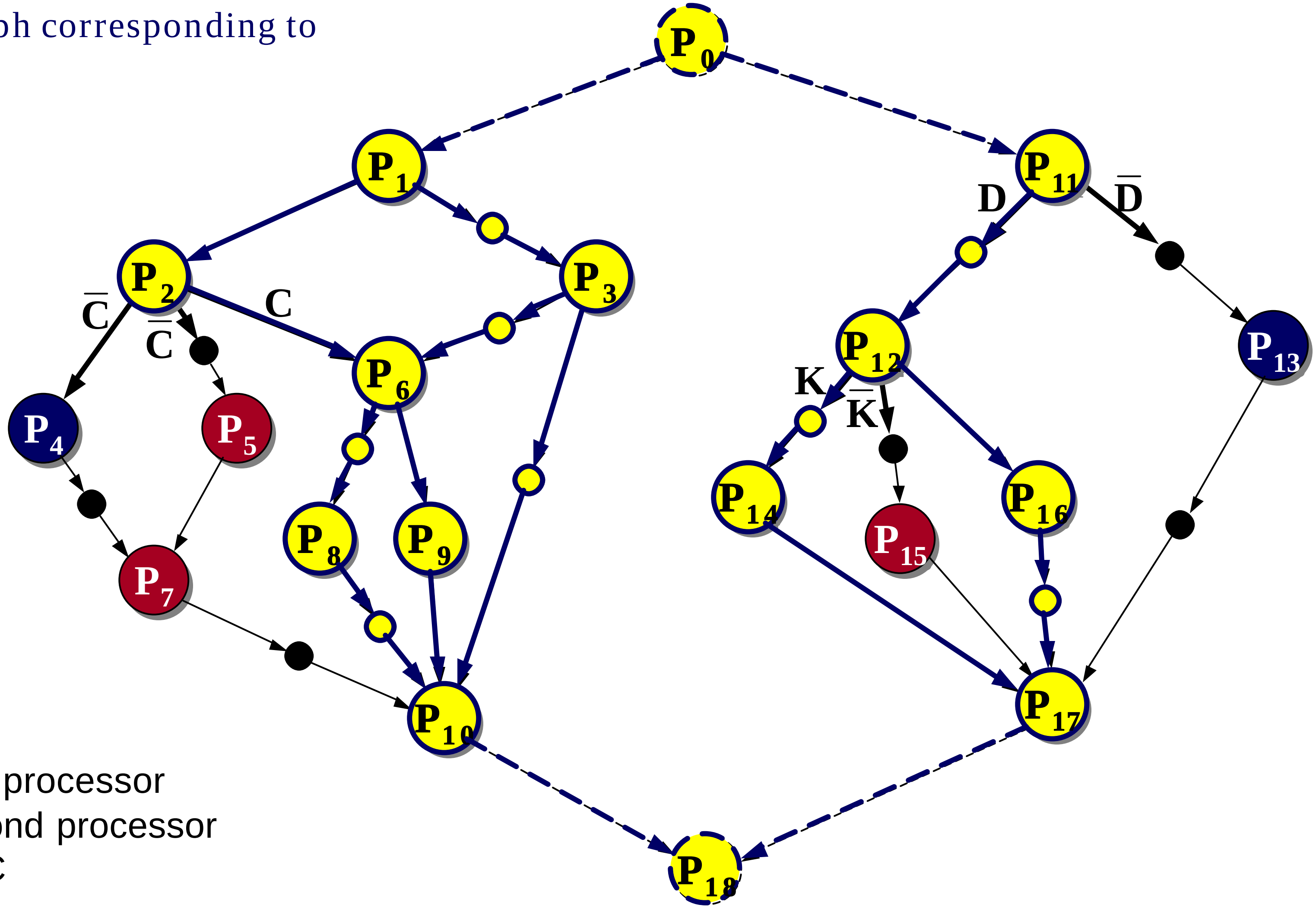
- Heterogeneous system architecture.
- Fixed priority preemptive scheduling.
- Systems with data and control dependencies.
- Tighter worst case delay estimations.

Message:

- The pessimism of the analysis can be drastically reduced by considering the conditions during the analysis.

Conditional Process Graph

Subgraph corresponding to
D∧C∧K



- First processor
- Second processor
- ASIC

Problem Formulation

Input

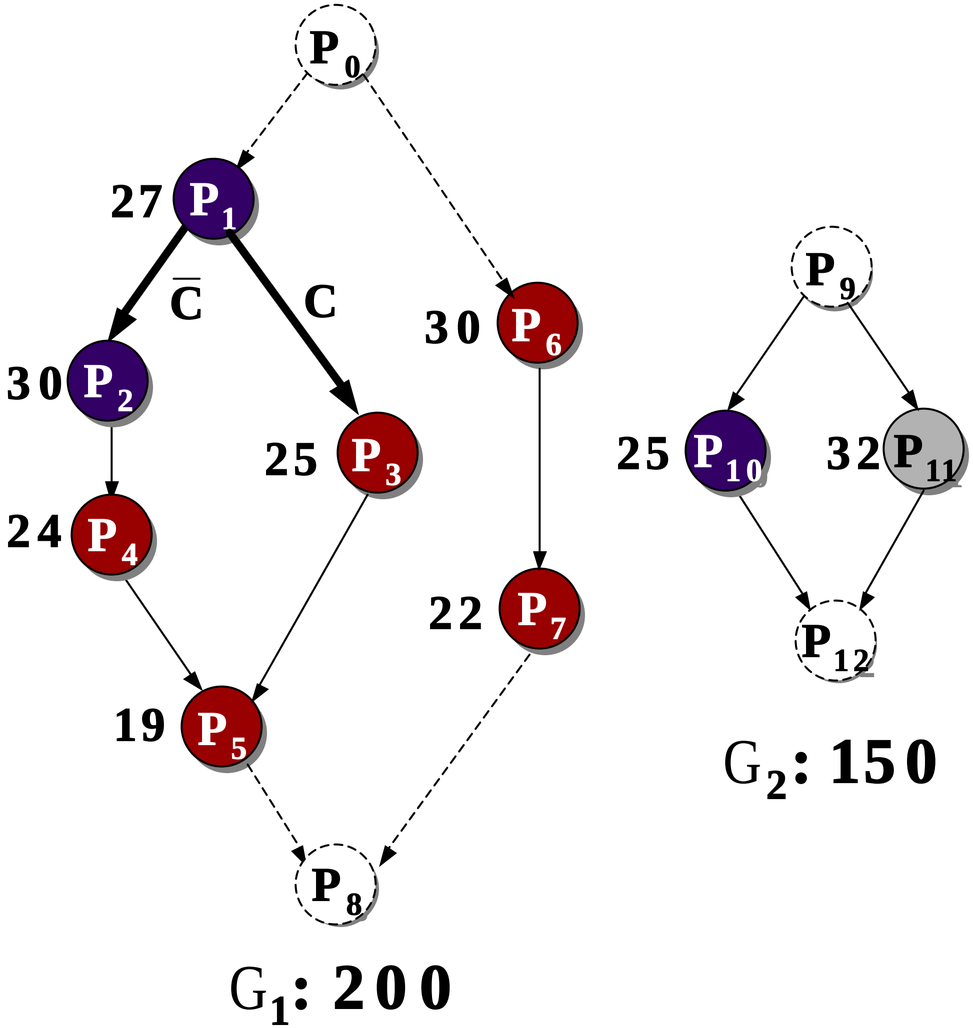
- An application modelled as a set of conditional process graphs (CPG).
- Each CPG in the application has its own independent period.
- Each process has an execution time, a deadline, and a priority.
- The system architecture and mapping of processes are given.

Output

- Schedulability analysis for systems modelled as a set of conditional process graphs (both data and control dependencies).

- Fixed priority preemptive scheduling.
- Communication of messages not considered, but can be easily added.

Example



CPG	Worst Case Delays	
	No conditions	Conditions
G_1	120	100
G_2	82	82

Task Graphs with Data Dependencies

- K. Tindell: Adding Time-Offsets to Schedulabilty Analysis, Research Report

  Offset: fixed interval in time between the arrival of sets of tasks.

  Can reduce the pessimism of the schedulability analysis.

  Drawback: how to derive the offsets?

- T. Yen, W. Wolf: Performance Estimation for Real-Time Distributed Embedded Systems, IEEE Transactions On Parallel and Distributed Systems

  Phase (similar concept to offsets).

  Advantage: gives a framework to derive the phases.

Schedulability Analysis for Task Graphs

```
DelayEstimate(task graph G, system S)
      for each pair (P_i, P_j) in G
            maxsep[P_i, P_j]=∞
      end for
      step = 0
      repeat
            LatestTimes(G)
            EarliestTimes(G)
            for each P_i ∈ G
                  MaxSeparations(P_i)
            end for
      until maxsep is not changed or step < limit
      return the worst case delay δ_G of the graph G
end DelayEstimate
```

LatestTimes(G): worst case response times and upper bounds for the offsets

EarliestTimes(G): lower bounds for the offsets

MaxSeparations(P_i): maximum separation: maxsep[P_i, P_j]=0 if the execution of the two processes never overlaps

Schedulability Analysis for CPGs, 1

Two extreme solutions:

- Ignoring Conditions (IC)

  Ignore control dependencies and apply the schedulability analysis for the (unconditional) task graphs.

- Brute Force Algorithm (BF)

  Apply the schedulability analysis after each of the CPGs in the application have been decomposed in their constituent unconditional subgraphs.

Schedulability Analysis for CPGs, 2

In between solutions:

- Conditions Separation (CS)

  Similar to *Ignoring Conditions* but uses the knowledge about the conditions in order to update the **`maxsep`** table:

  `maxsep[`$\mathbf{P_i, P_j}$**`] = 0`** if $\mathbf{P_i}$ and $\mathbf{P_j}$ are on different conditional paths.

- Relaxed Tightness Analysis (two variants: RT1, RT2)

  Similar to the *Brute Force Algorithm*, but tries to reduce the execution time by removing the iterative tightening loop (relaxed tightness) in the **`DelayEstimation`** function.

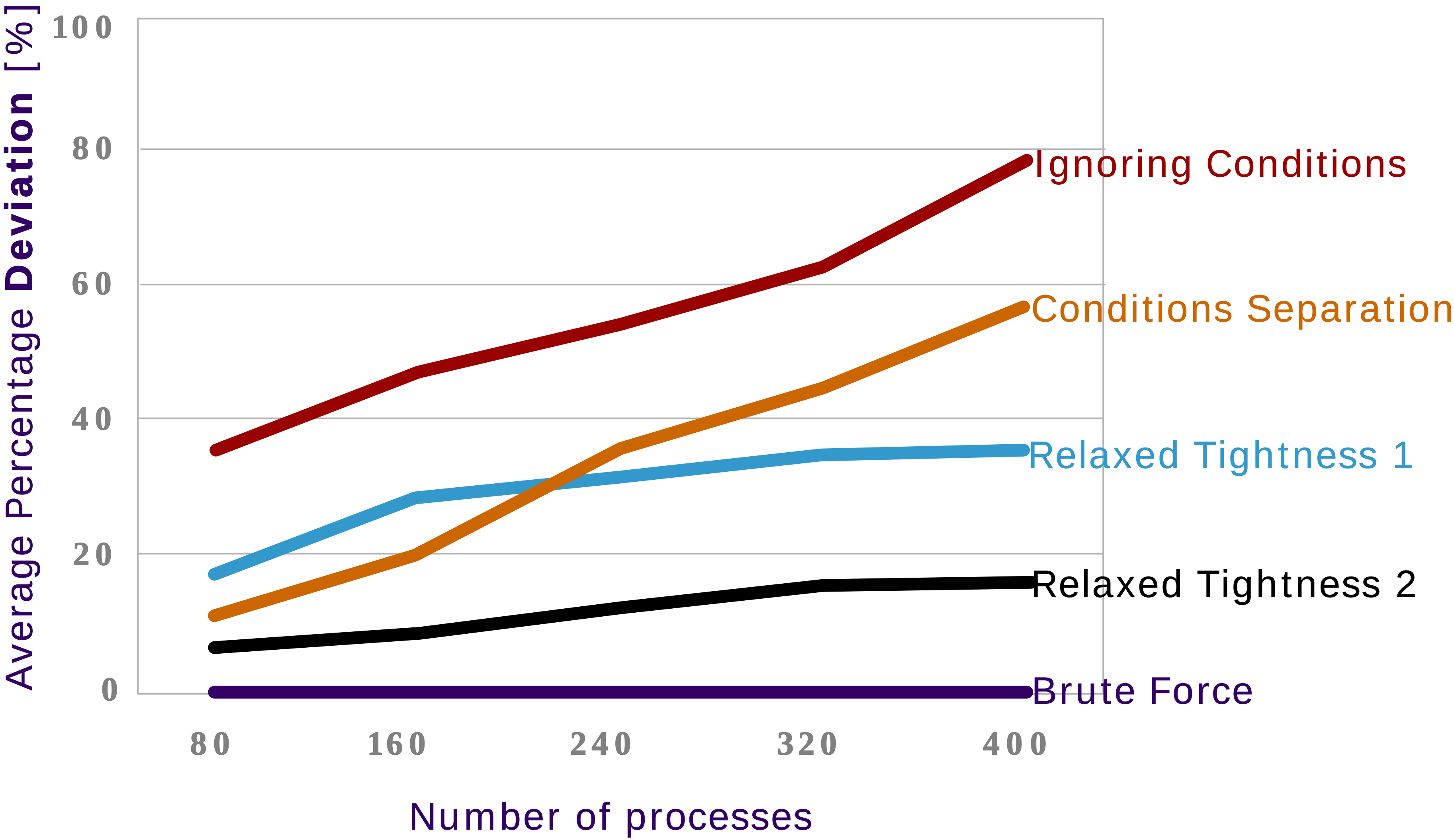
Experiments Setup

- Number of Graphs: 150
  30 for each dimension of 80, 160, 240, 320, 400 nodes;
  2, 4, 6, 8, 10 conditions.
- Graphs Structure:
  Random and regular (trees, groups of chains).
- Architecture:
  2, 4, 6, 8, 10 nodes.
- Mapping:
  40 processes / node; random and using simple heuristics.

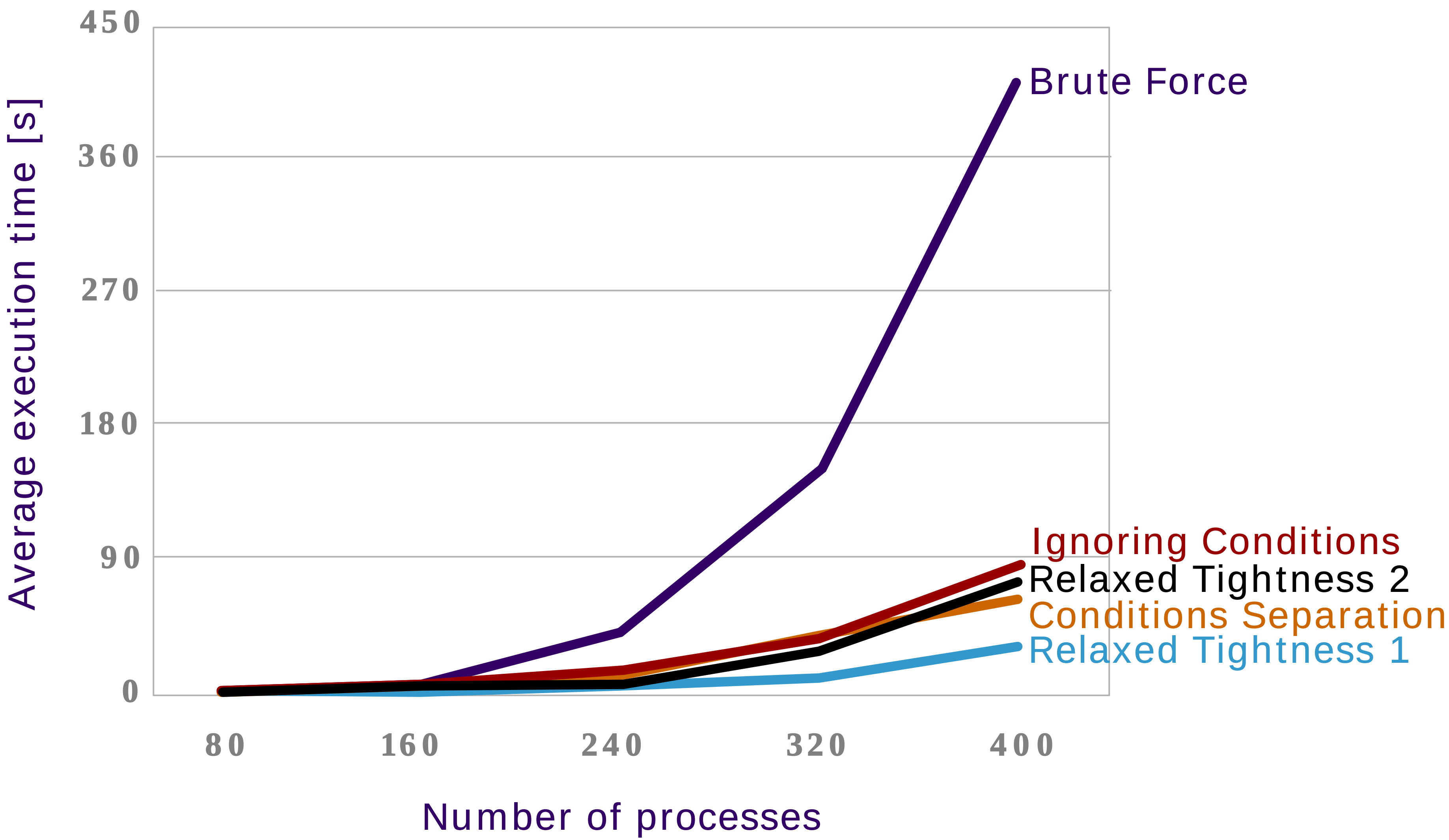
- Cost function: **degree of schedulability**

$$Cost\ function = \sum_{i=1}^{n} \left(D_{\Gamma_i} - d_{\Gamma_i} \right)$$

Experimental Results

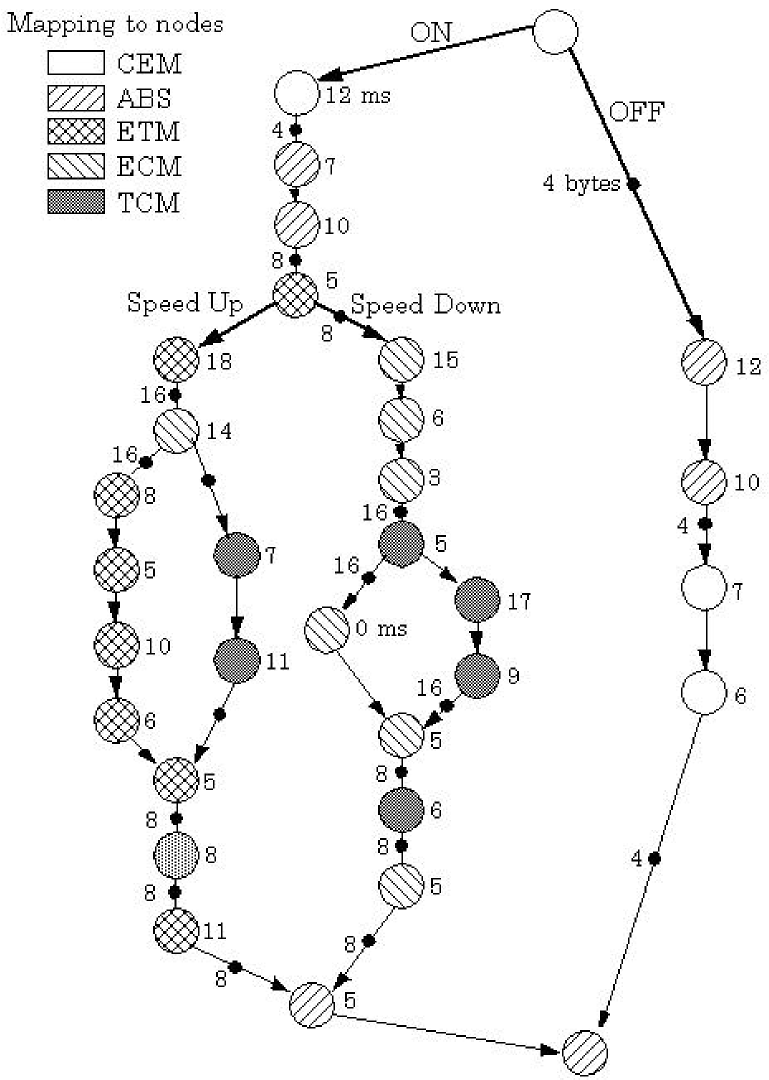


Experimental Results (Cont.)



Real Life Example

- **Vehicle cruise controller.**
- Modelled with a CPG of 32 processes and two conditions.
- Mapped on 5 nodes: CEM, ABS, ETM, ECM, TCM.



- Deadline 130:
  - Ignoring Conditions: 138 ms
  - Conditions Separation: 132 ms
  - Relaxed Tightness 1, 2: **124** ms
  - Brute Force: **124** ms

Conclusions

- Schedulability analysis for hard real-time systems with control and data dependencies.
- The systems are modelled using conditional process graphs that are able to capture both the flow of data and that of control.
- Distributed architectures, fixed priority scheduling policy.
- Five approaches to the schedulability analysis of such systems.
- Extensive experiments and a real-life example show that: **considering the conditions during the analysis the pessimism of the analysis can be significantly reduced.**