Scheduling with Optimized Communication for Time-Triggered Embedded Systems

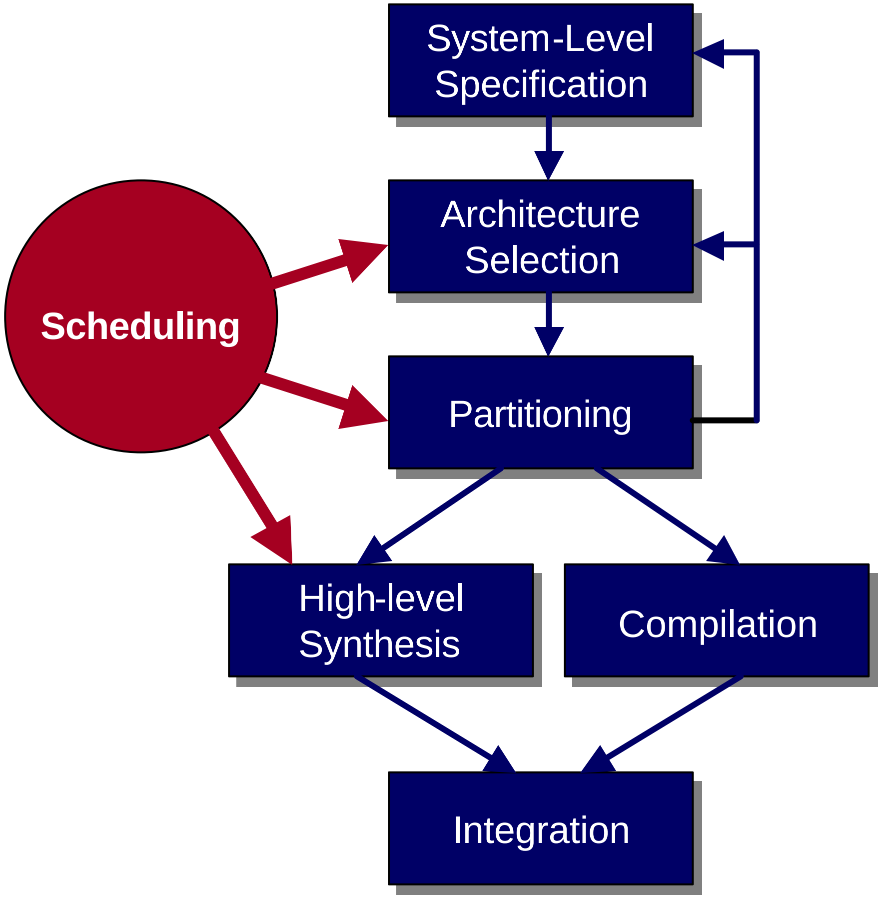
Paul Pop, Petru Eles and Zebo Peng

Dept. Of Computer and Information Science
Linkoping University
Sweden

Outline

- Motivation
- Conditional Process Graph
- System Architecture
- Problem Formulation
- Scheduling Example
- Scheduling Strategy
- Experimental Results
- Conclusions

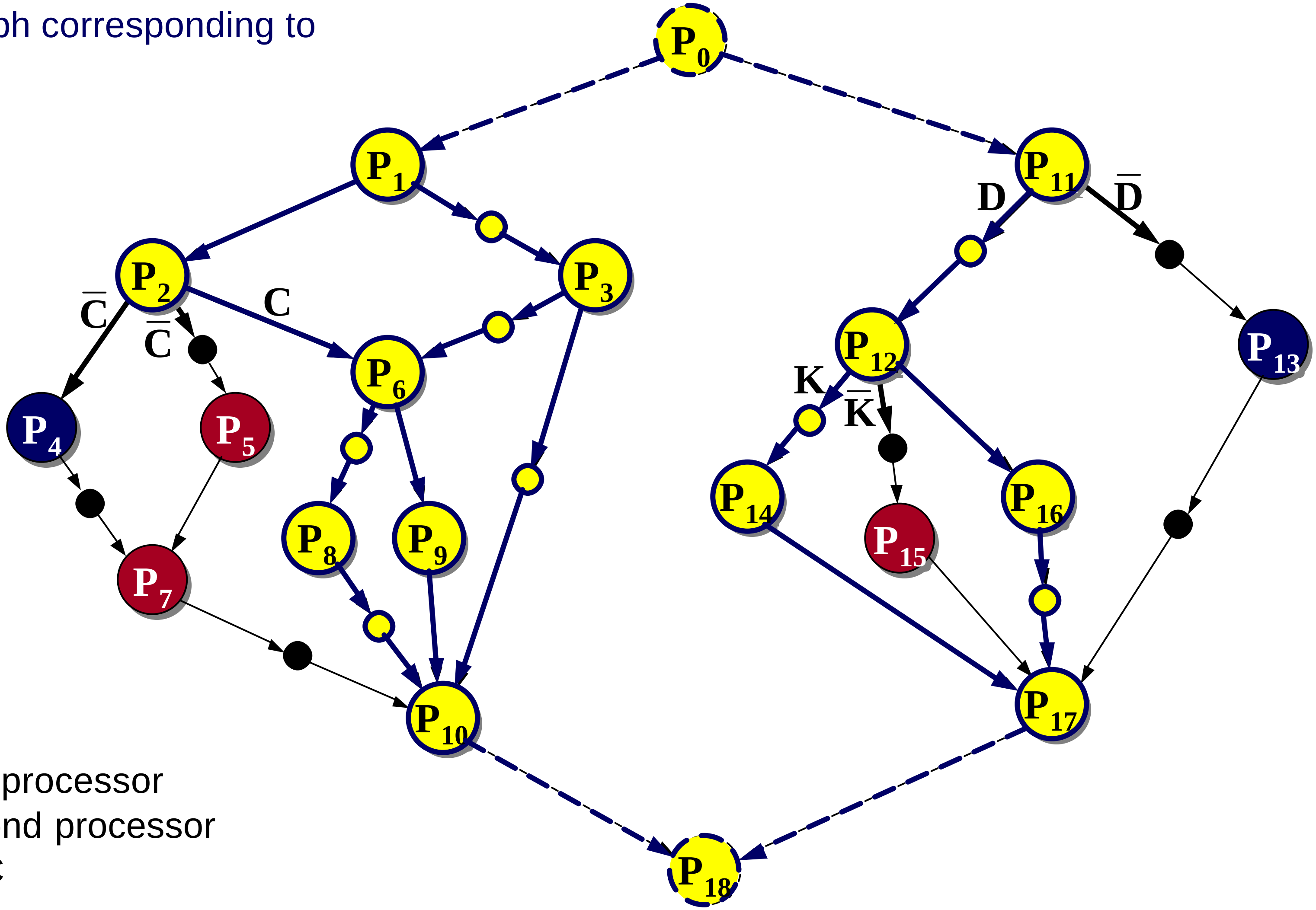
Motivation



- System model captures both the flow of data and that of control.
- Heterogeneous system architecture: nodes connected by a broadcast communication channel .
- Communication of conditions and messages considered for a time-triggered protocol (TTP) implementation.
- Improved schedule quality by considering the characteristics of TTP and the overheads of the real-time kernel.
- Scheduling algorithms proposed can be used both for performance estimation and for system synthesis.

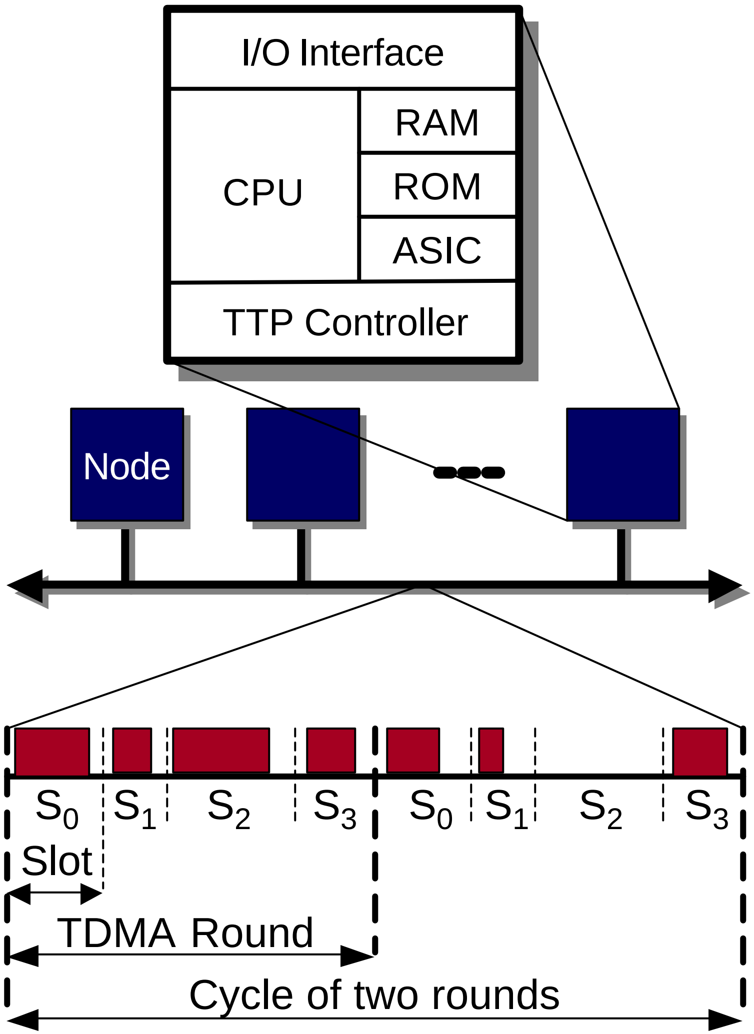
Conditional Process Graph

Subgraph corresponding to $D \wedge C \wedge K$



- First processor
- Second processor
- ASIC

Hardware Architecture



- Safety-critical distributed embedded systems.
- Nodes connected by a broadcast communication channel.
- Nodes consisting of: TTP controller, CPU, RAM, ROM, I/O interface, (maybe) ASIC.
- Communication between nodes is based on the time-triggered protocol.
- Buss access scheme: time-division multiple-access (TDMA).
- Schedule table located in each TTP controller: message descriptor list (MEDL).

Software Architecture

- Real-Time Kernel running on the CPU in each node.
- There is a local schedule table in each kernel that contains all the information needed to take decisions on activation of processes and transmission of messages.
- Time-Triggered System: no interrupts except the timer interrupt.
- The worst case administrative overheads (WCAO) of the system calls are known:

U_t	WCAO of the timer interrupt routine
δ_{PA}	process activation overhead
δ_S	overhead for sending a message on the same node
δ_{KS}	overhead for sending a message between nodes
δ_{KR}	overhead for receiving a message from another node

Problem Formulation

Input

- Safety-critical application with several operating modes.
- Each operating mode is modelled by a conditional process graph.
- The system architecture and mapping of processes to nodes are given.
- The worst case delay of a process is known:

$$T_{P_i} = (d_{PA} + t_{P_i} + q_{C_1} + q_{C_2})$$

$$q_{C_1} = \sum_{i=1}^{N_{out}^{local}(P_i)} d_{S_i} \qquad q_{C_2} = \sum_{i=1}^{N_{out}^{remote}(P_i)} d_{KS_i} + \sum_{i=1}^{N_{in}^{remote}(P_i)} d_{KR_i}$$

Output

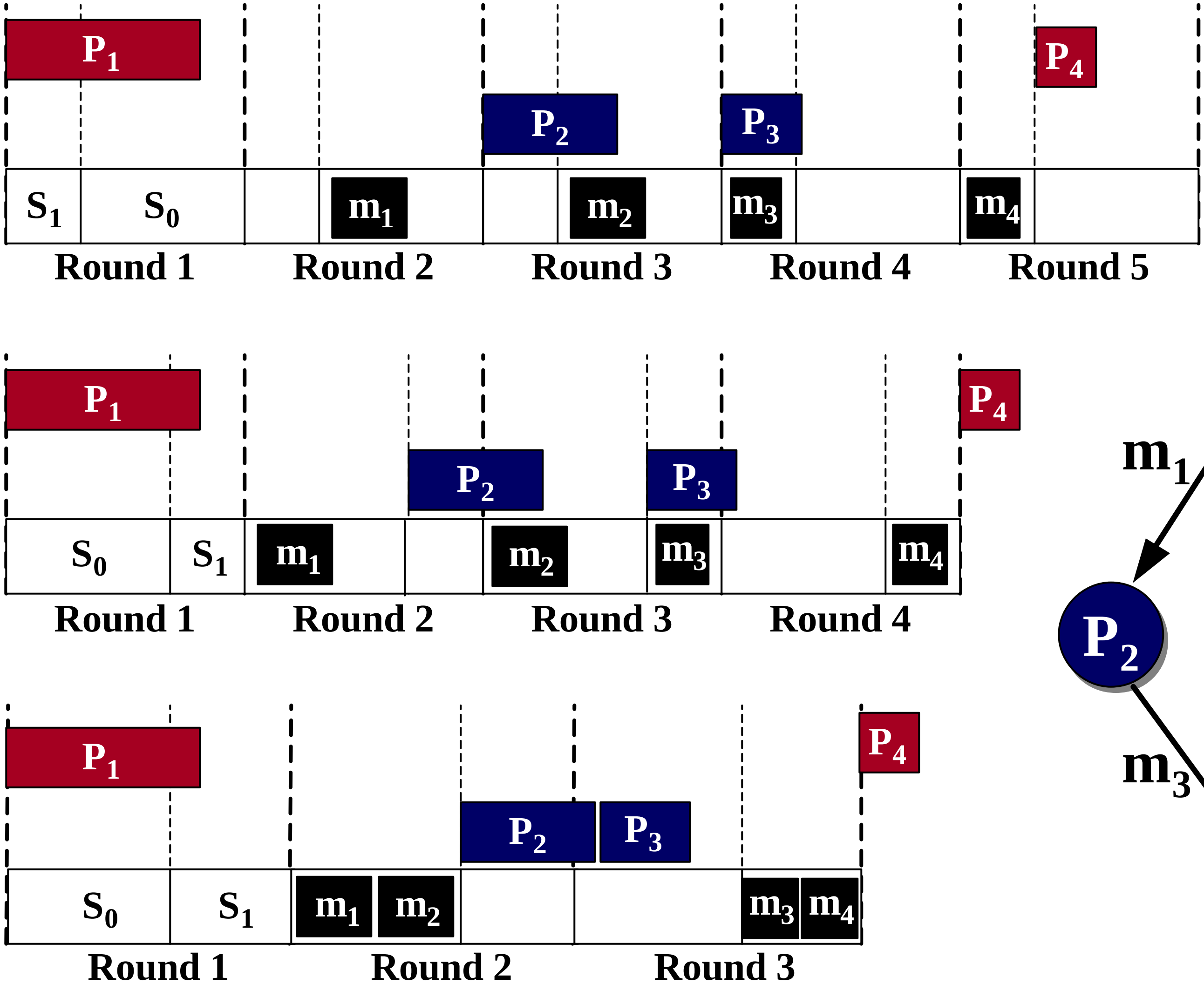
- Local schedule tables for each node and the MEDL for the TTP controllers.
- Delay on the system execution time for each operating mode, so that this delay is as small as possible.

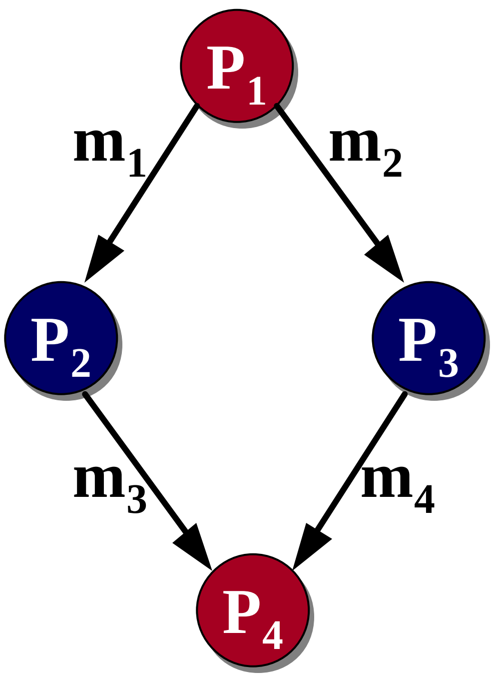
Scheduling Example

24 ms

22 ms

20 ms



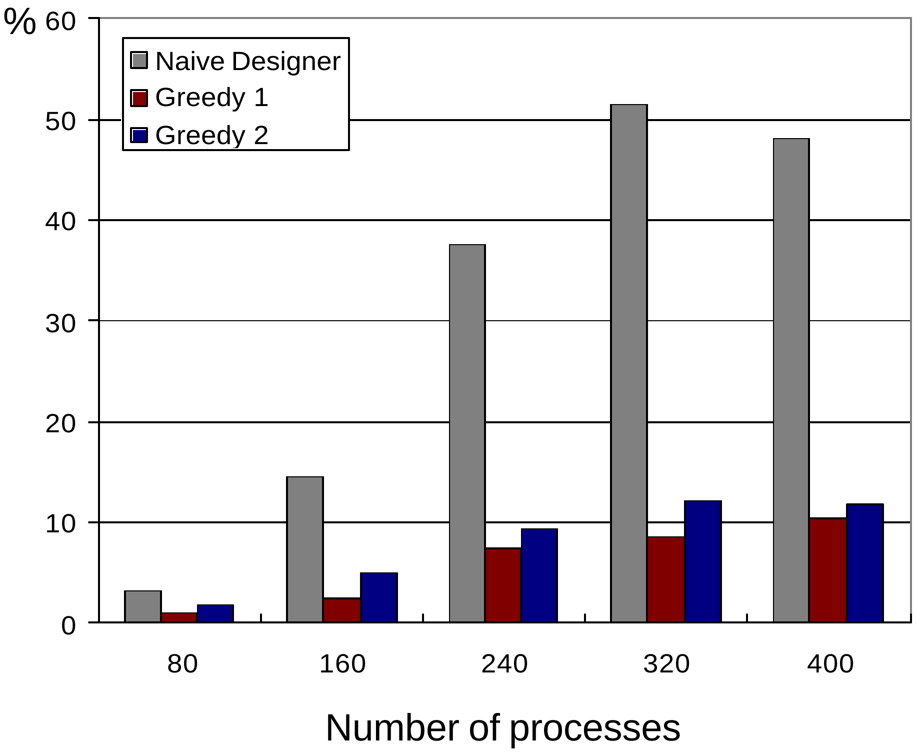


Scheduling Strategy

- Eles et al. “Scheduling of Conditional Process Graphs for the Synthesis of Embedded Systems”, DATE’98
- Previous work extended to handle scheduling of messages within TTP for a given TDMA configuration: **`schedule_message`**.
- Sequence and lengths of the slots in a TDMA round are determined to reduce the delay.
- Two approaches: Greedy heuristic, Simulated Annealing (SA).
- Two variants: Greedy 1 tries all possible slot lengths, Greedy 2 uses feedback from the **`schedule_message`** function.
- SA parameters are set to guarantee finding near-optimal solutions in a reasonable time.

Experimental Results

Average percentage deviations
from the lengths of near-optimal schedules



- The Greedy Approach is producing accurate results in a very short time (few seconds for graphs with 400 processes).
- Greedy 1 performs slightly better than Greedy 2, but it is a bit slower.
- SA finds near-optimal results in a reasonable time (few minutes for graphs with 80 processes and 275 minutes for graphs with 400 processes).
- A real-life example implementing a vehicle cruise controller validated our approach.

Conclusions

- An approach to process scheduling for the synthesis of safety-critical distributed embedded systems.
- Process level representation which captures both data flow and the flow of control.
- Communication of data and conditions based on TTP.
- Communication has been optimized through packaging of messages into slots with a properly selected order and lengths.
- Improved schedule quality by considering the overheads of the real-time kernel and of the communication protocol.
- Evaluation based on experiments using a large number of graphs generated for experimental purpose as well as real-life examples.