An Approach to Incremental Design of Distributed Embedded Systems

Paul Pop, Petru Eles, Traian Pop, Zebo Peng

Department of Computer and Information Science
Linköpings universitet, Sweden

Outline

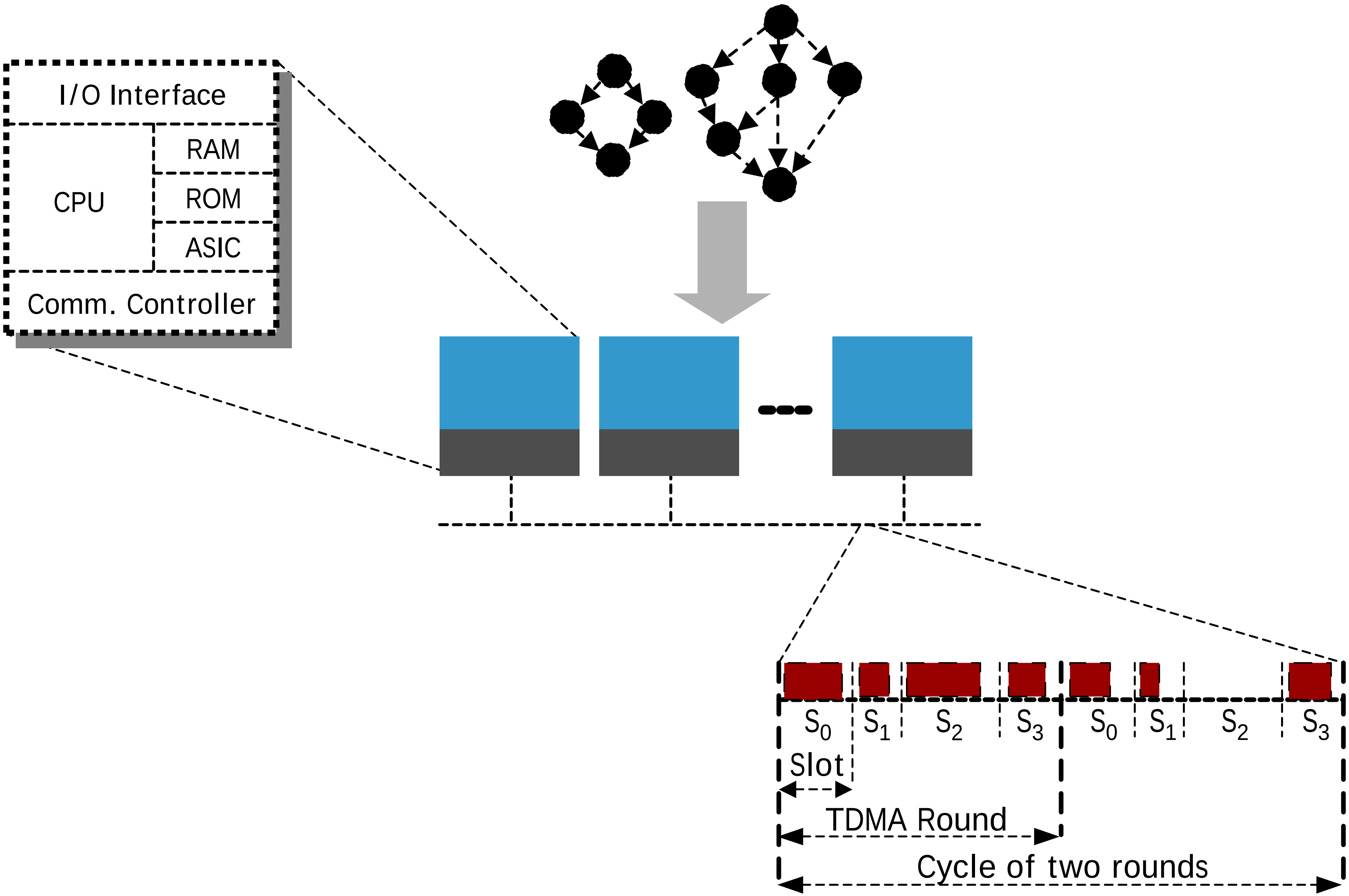
- Incremental design process
  - Mapping and scheduling
- Problem formulation
- Mapping strategy
- Experimental results
- Conclusions and future work

Introduction

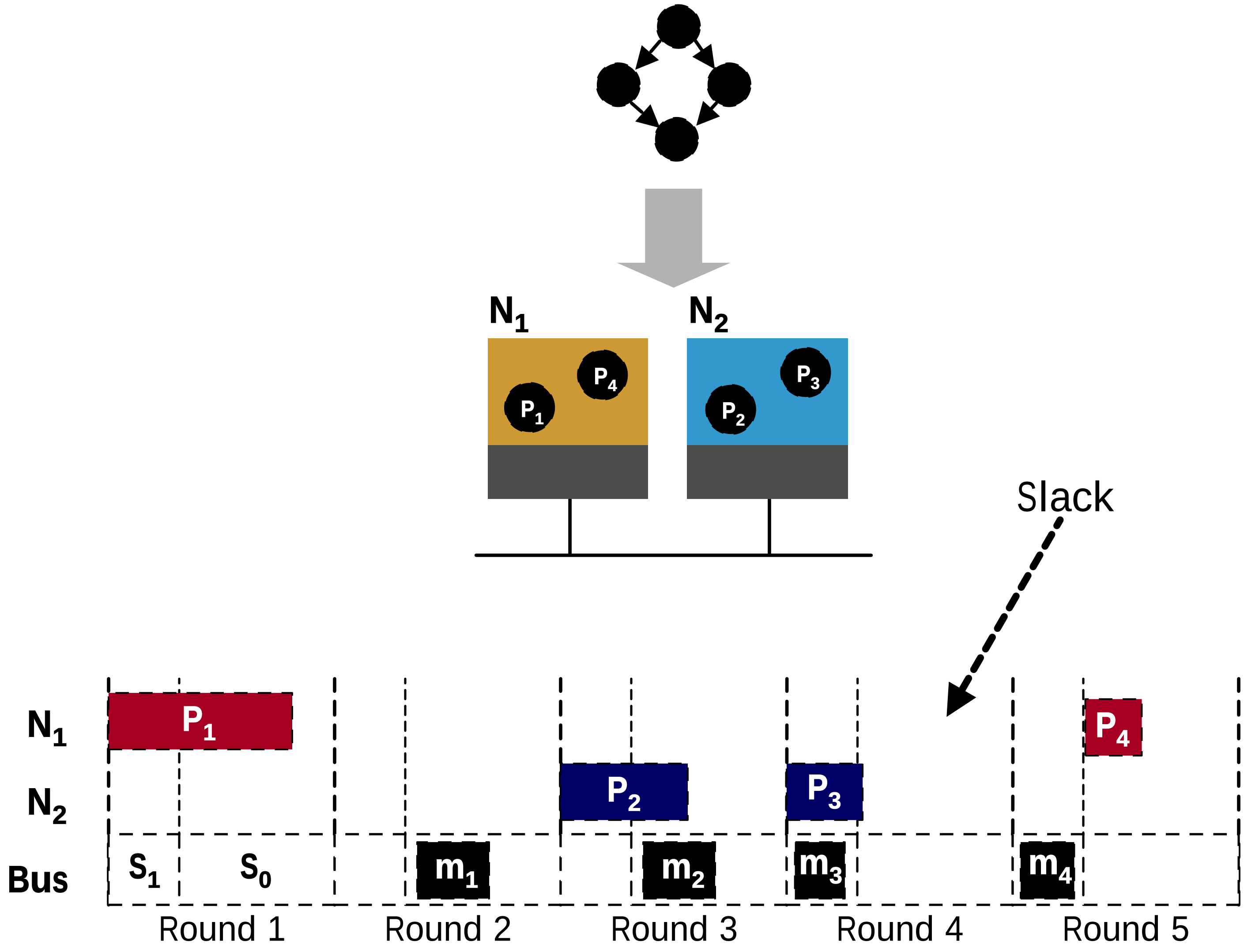
- **Characteristics:**
  - Incremental design process, engineering change;
  - Distributed real-time embedded systems; Heterogeneous architectures;
  - Static cyclic scheduling for processes and messages;
  - Communications using a time-division multiple-access (TDMA) scheme:

    H. Kopetz, G. Grünsteidl. TTP-A Protocol for Fault-Tolerant Real-Time Systems. IEEE Computer '94.
- **Contributions:**
  - Mapping and scheduling considered inside an incremental design process;
  - Two design criteria (and their metrics) that drive our mapping strategies to solutions supporting an incremental design process;
  - Two mapping algorithms.
- **Message:**
  - Engineering change can be successfully addressed at system level.

“Classic” Mapping and Scheduling



“Classic” Mapping and Scheduling Example

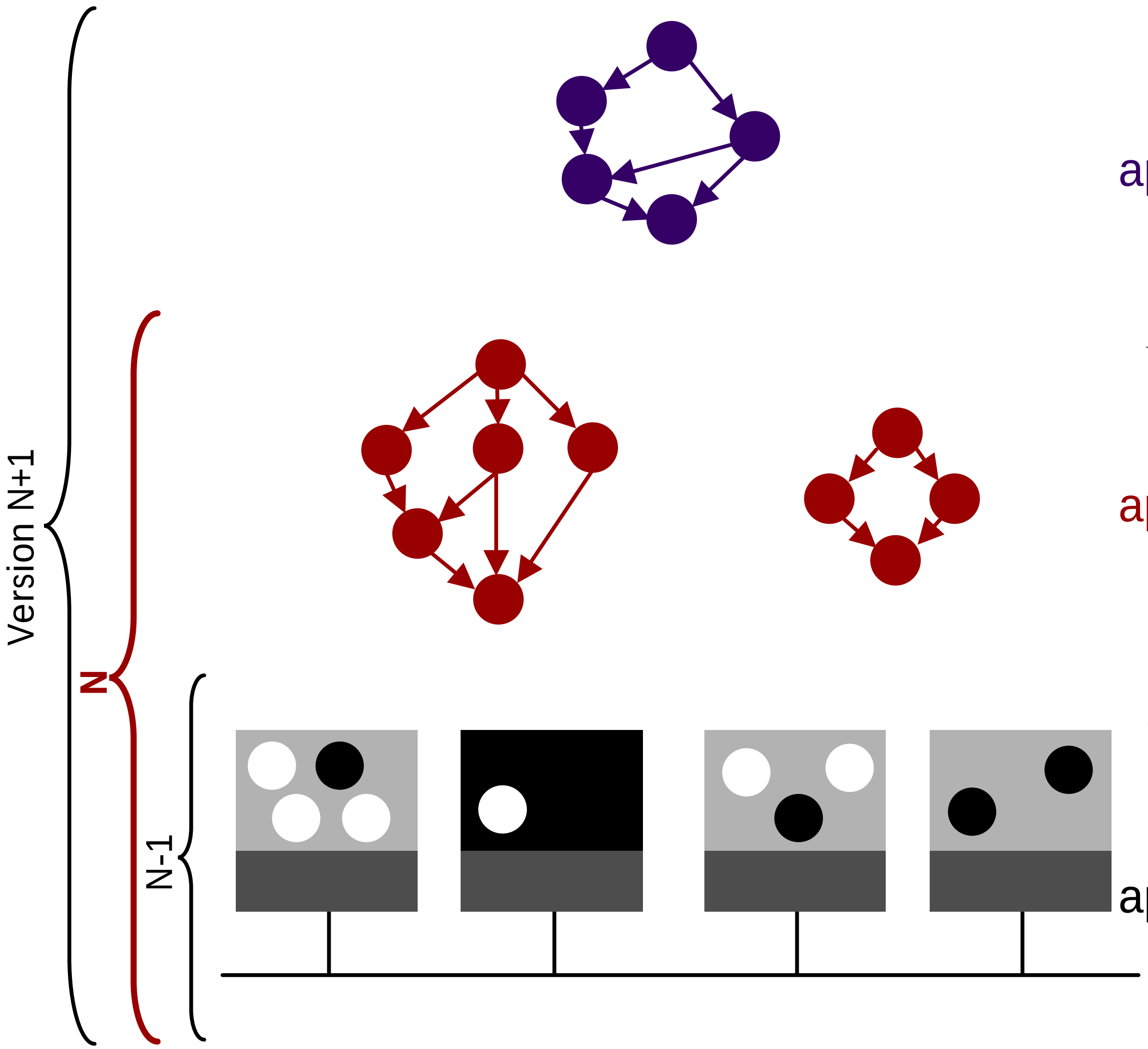


Incremental Design Process

- Start from an already existing system with applications:
  - In practice, very uncommon to start from scratch.

- Implement new functionality on this system (increment):
  - As few as possible modifications of the existing applications, to reduce design and testing time;
  - Plan for the next **increment**:
    It should be easy to add functionality in the future.

Mapping and Scheduling



Future applications

Do not exist yet at Version N!

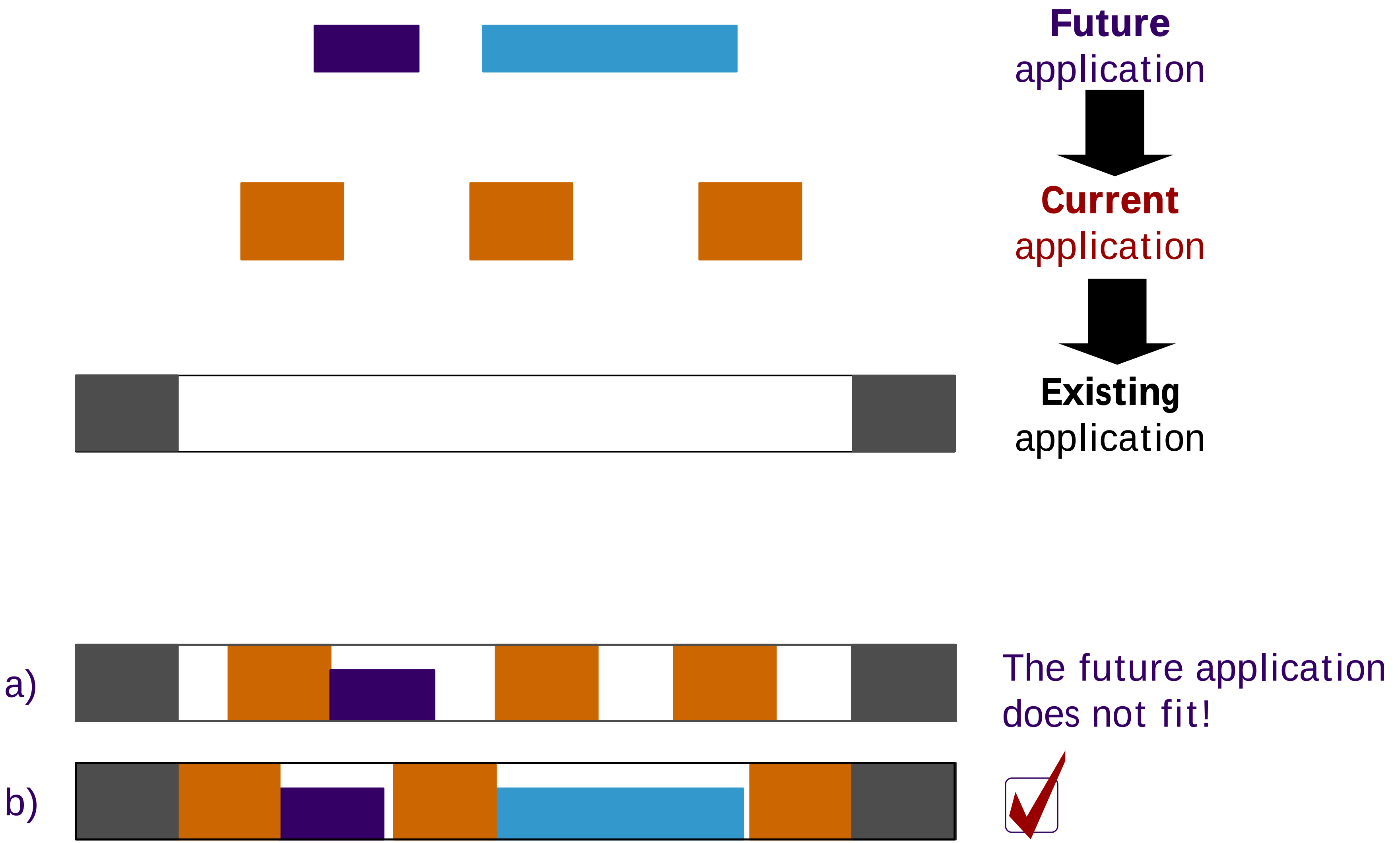
Current applications

Map and schedule so that the future applications will have a chance to fit.

Existing applications

No modifications are performed to the existing applications.

Mapping and Scheduling Example



Problem Formulation

Input

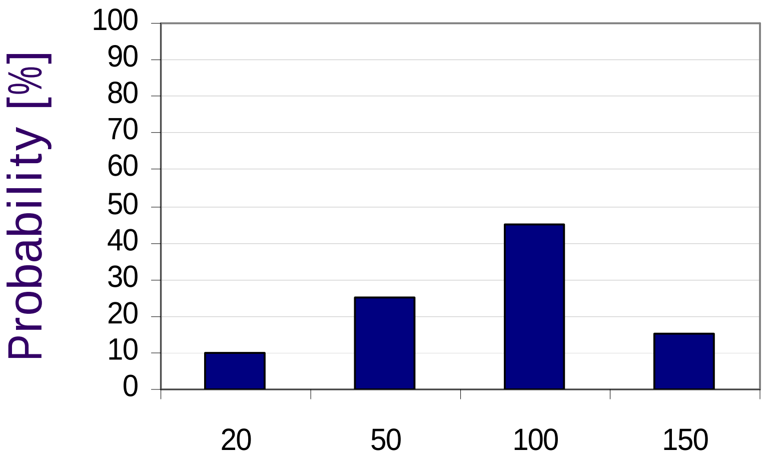
- A set of *existing* applications modelled using process graphs;
- A *current* application to be mapped modelled using process graphs;
- Each process graph in the application has its own *period* and *deadline*;
- Each process has a *potential set of nodes* to be mapped to and a *WCET*;
- Certain information about *future* applications (next slide);
- The system architecture is given.

Output

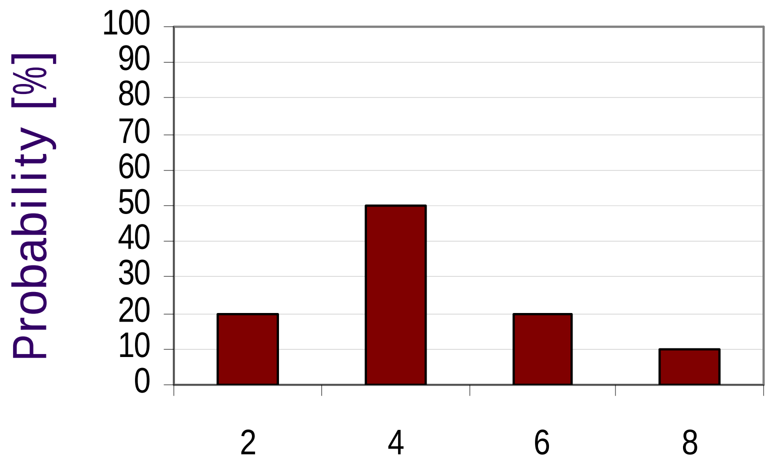
- A **mapping and scheduling of the *current* application**, so that:
  Requirement a: constraints of the *current* application are satisfied
  and no modifications are performed to the *existing* applications;
  Requirement b: new *future* applications can be mapped
  on the resulted system.

Characterizing Future Applications

For a family of future applications we know:



Typical process WCET [time units]



Typical message size [bytes]

The most demanding future application:

- Smallest expected period T_{min}
- Expected necessary processor time t_{need} inside T_{min}
- Expected necessary bandwidth b_{need} inside T_{min}

Mapping and Scheduling Strategy

Mapping and scheduling of the *current* application, so that:

- **Requirement a)**

  Constraints of the *current* application are satisfied and

  no modifications are performed to the *existing* applications.

  - Initial Mapping (IM) constructs an initial mapping with a valid schedule;

    starting point: Heterogeneous Critical Path (HCP) algorithm from

    P.B. Jorgensen, J. Madsen. Critical Path Driven Cosynthesis for Heterogeneous Target Architectures. CODES'97

- **Requirement b)**

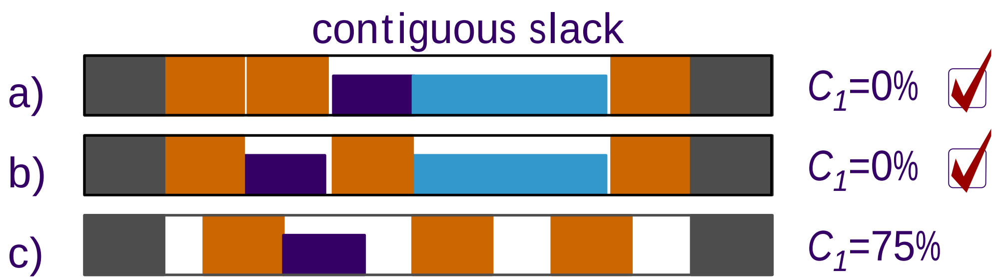
  New *future* applications can be mapped on the resulted system.

  - Design criteria reflect the degree to which a design meets the requirement b);
  - Design metrics quantify the degree to which the criteria are met;
  - Heuristics to improve the design metrics.

Mapping and Scheduling: First Criterion

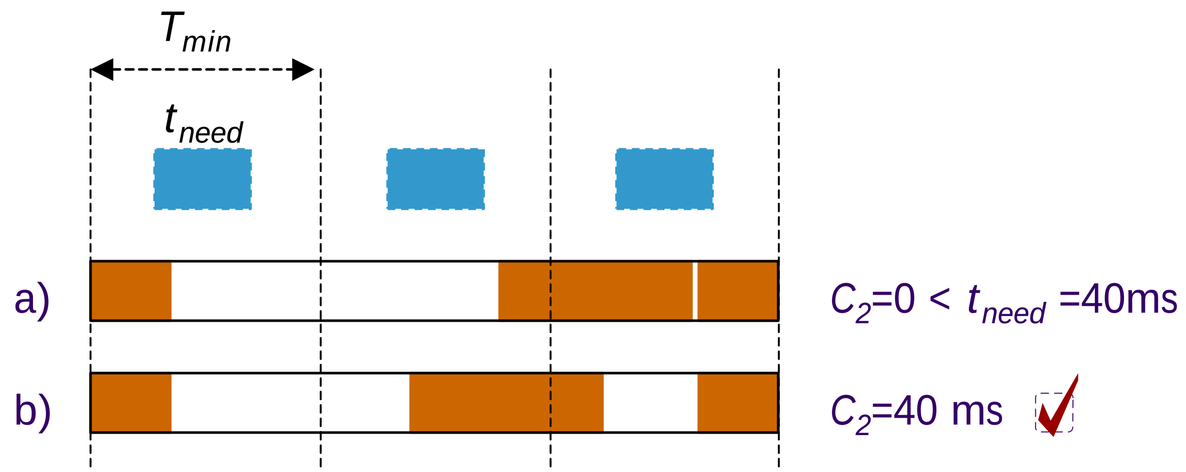
- First design criterion: slack sizes
  - How well the slack sizes of the *current* design alternative accommodate a family of *future* applications that are characterized as outlined before;
  - Tries to cluster the available slack: the best slack would be a contiguous slack.

- Design metrics for the first design criterion
  - C_1^P for processes, C_1^m for messages;
  - How much of the largest *future* application (contiguous slack), *cannot* be mapped on the *current* design alternative;
  - *Bin-packing algorithm* using the *best-fit policy*: processes as objects to be packed, and the slack as containers.



Mapping and Scheduling: Second Criterion

- Second design criterion: slack distribution
  - How well the slack of the *current* design alternative is distributed in time to accommodate a family of *future* applications;
  - Tries to distribute the slack so that we periodically (T_{min}) have enough necessary processor time t_{need} and bandwith b_{need} for the most demanding future application.
- Design metrics for the second design criterion
  - C_2^P for processes, C_2^m for messages;
  - C_2^P is the sum of minimum *periodic* slack inside a T_{min} period on each processor.



Mapping and Scheduling Strategy, Cont.

- Two steps:
  - Initial mapping and scheduling (IM) produces a valid solution
  - Starting from a valid solution, heuristics to minimize the objective function:

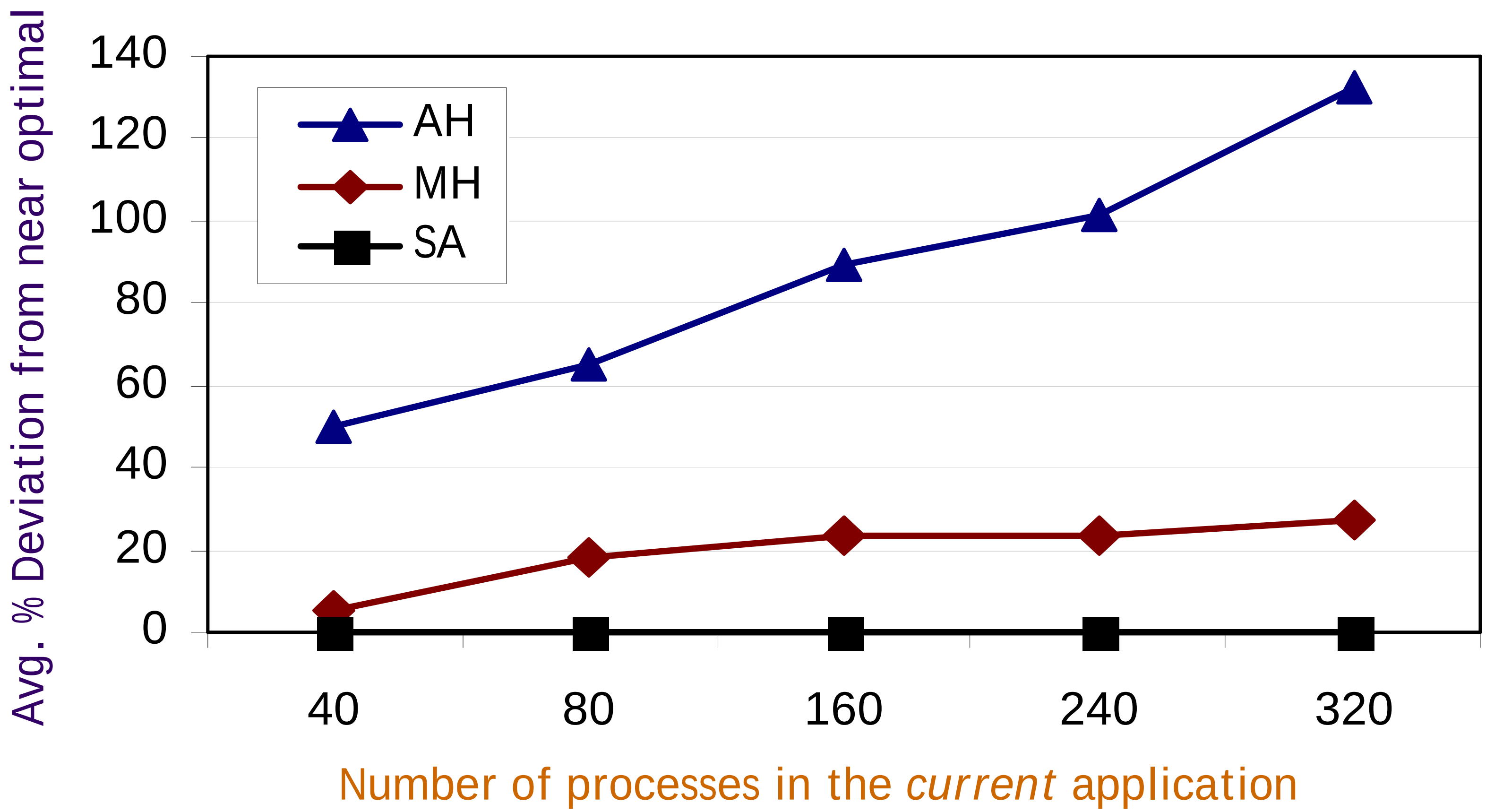
$$C = w_1^P(C_1^P) + w_1^m(C_1^m) + w_2^P \max(0, t_{need} - C_2^P) + w_2^m \max(0, b_{need} - C_2^m)$$

- Three heuristics:
  - Ad-Hoc approach (AH), little support for incremental design.
  - Simulated Annealing (SA), near optimal value for *C*.
  - Mapping Heuristic (MH):
    - Iteratively performs *design transformations* that improve the design;
    - Examines only transformations with the *highest potential* to improve the design;
    - Design transformations:

      moving a process to a different slack on the same or different processor,

      moving a message to a different slack on the bus.

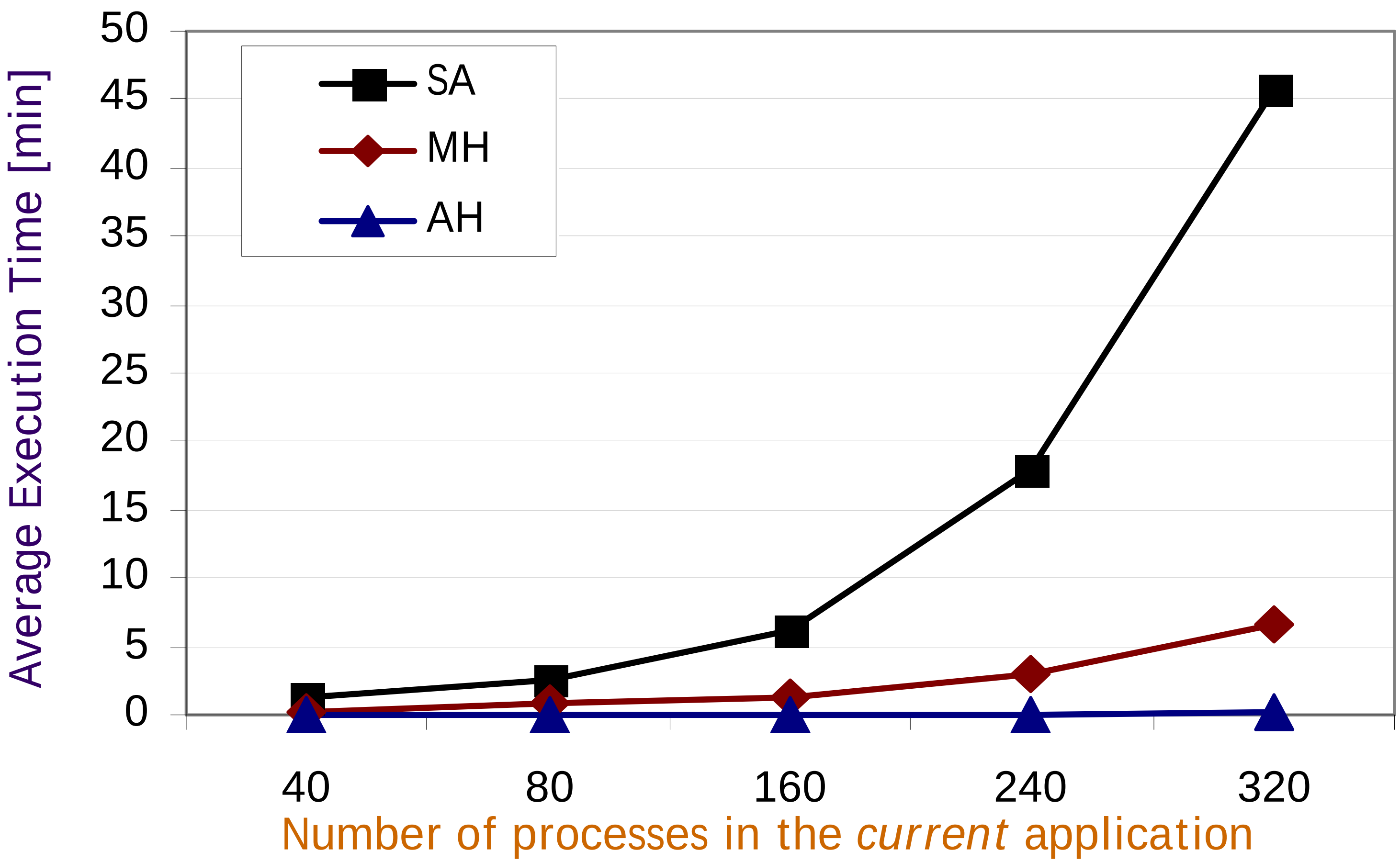
How does the **quality** (cost function) of the mapping heuristic (MH) compare to the ad-hoc approach (AH) and the simulated annealing (SA)?



existing applications: 400

Experimental Results, Cont.

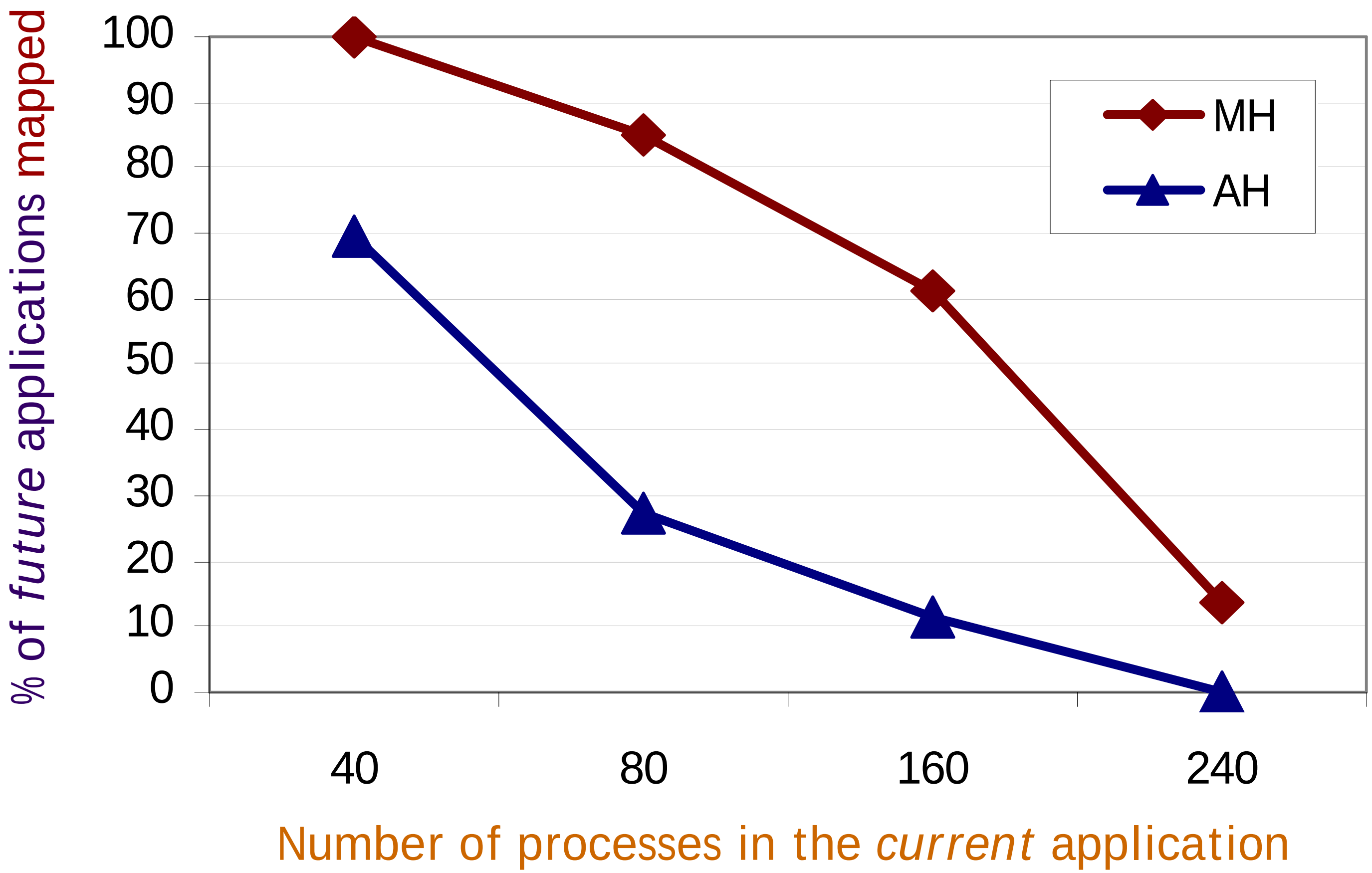
How does the **runtime** of the mapping heuristic (MH) compare to the ad-hoc approach (AH) and the simulated annealing (SA)?



existing applications: 400

Experimental Results, Cont.

Are the mapping strategies proposed facilitating the implementation of future applications?



existing applications: 400, future application: 80

Conclusions and Future Work

- **Conclusions:**
  - Mapping and scheduling considered inside an incremental design process;
  - Two design criteria (and their metrics) that drive our mapping strategies to solutions supporting an incremental design process;
  - Iterative improvement mapping heuristic.

- **CODES 2001:**
  - Allow modifications to the existing applications:
    - How to capture the modification cost (engineering changes);
    - How to decide which applications should be modified;
    - Modification cost should be minimized.