

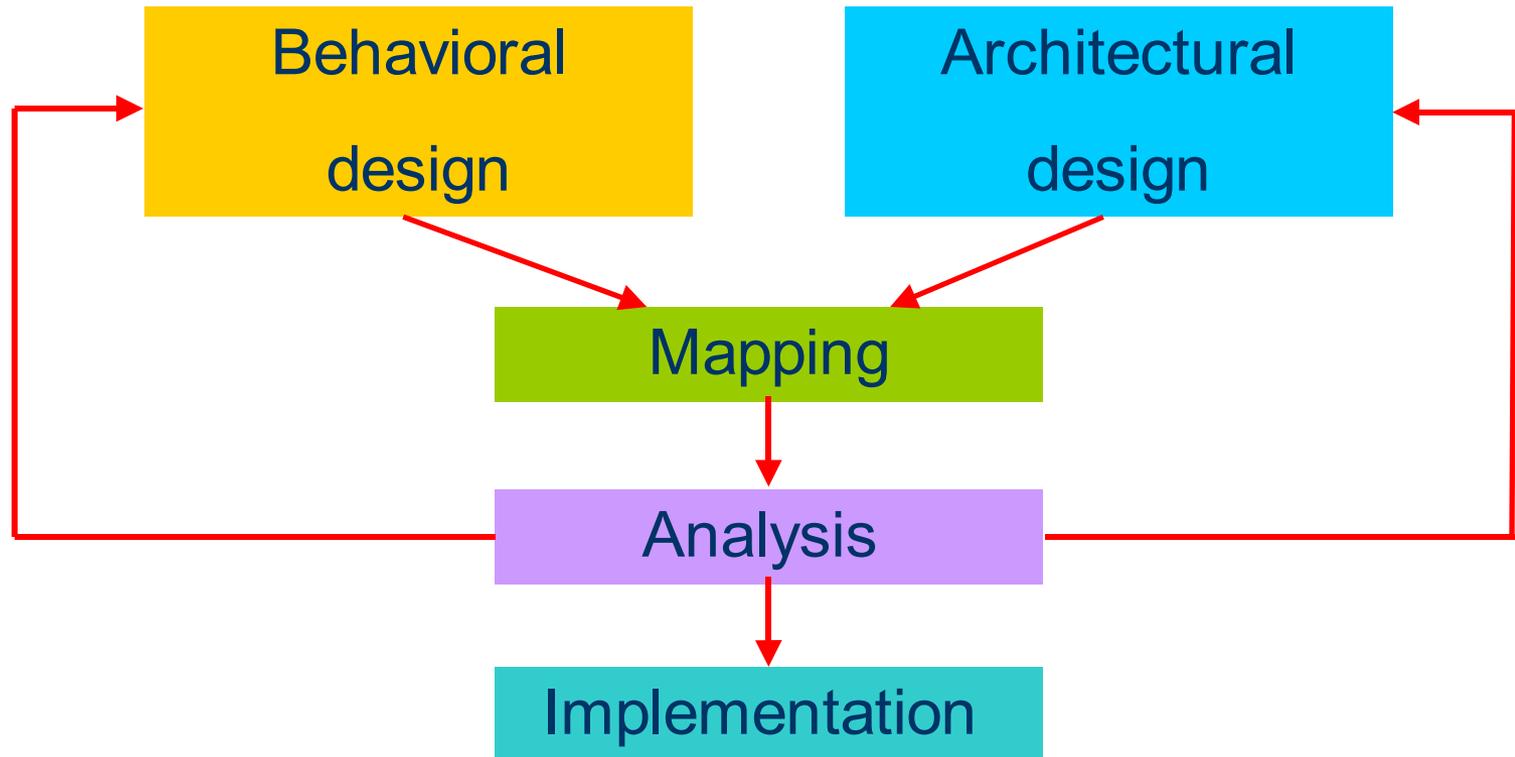
Process Partitioning of Dataflow Applications

Bartolini Cesare
ISTI – CNR



ERICSSON 

Embedded System Design



Embedded System Design

Ideal

- Comprehensive software tools
- Automated testing
- Optimized systems

Current practice

- Manual operation
- Long time
- Arbitrary choices
- Scarce optimization

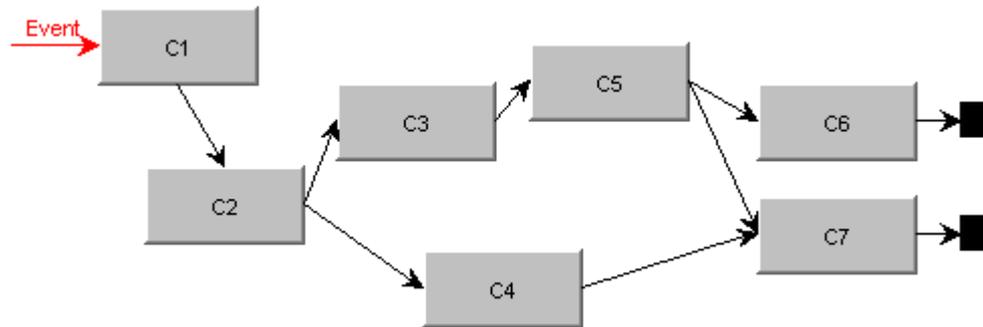
Goals of This Research

- Define a simple model of execution
 - To express the behavior of the system
- Generate task partitioning
 - Assign functions to tasks
- Assign deadlines/priorities for scheduling
- Optimize the task set
- HW/SW co-design

Behavioral Design

- Dataflow Application
- Directed Acyclic Graph (DAG)
- Minimal unit is function, not task
- Requirements:
 - Path deadlines
 - Minimum interarrival times of external events
 - WCETs are not required for modeling but only for analysis

A Sample Application



Mapping Functionality to Tasks

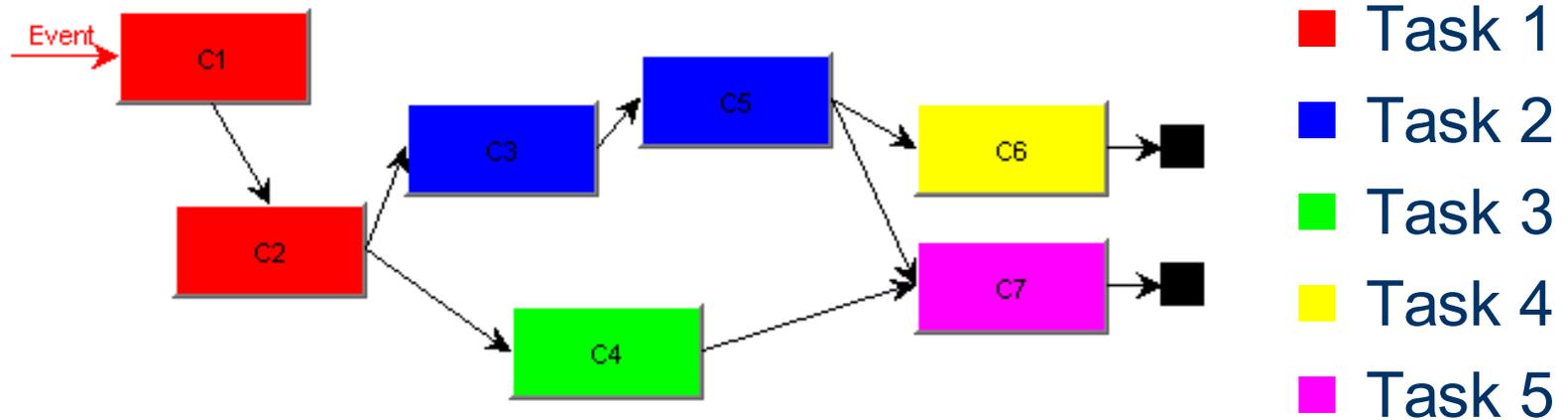
- We propose 4 different algorithms
 - Early Activation (EA)
 - Late Activation (LA)
 - Rising Deadline (RD)
 - Joined Rising Deadline (JRD)

A preemptive EDF scheduler is required

Early Activation

- Simple algorithm
- Only end-to-end constraints are required
- Generates many small tasks
- Activations are *before* execution
- Little algorithm overhead
- Easy implementation, but not very efficient

Process Partitioning With EA



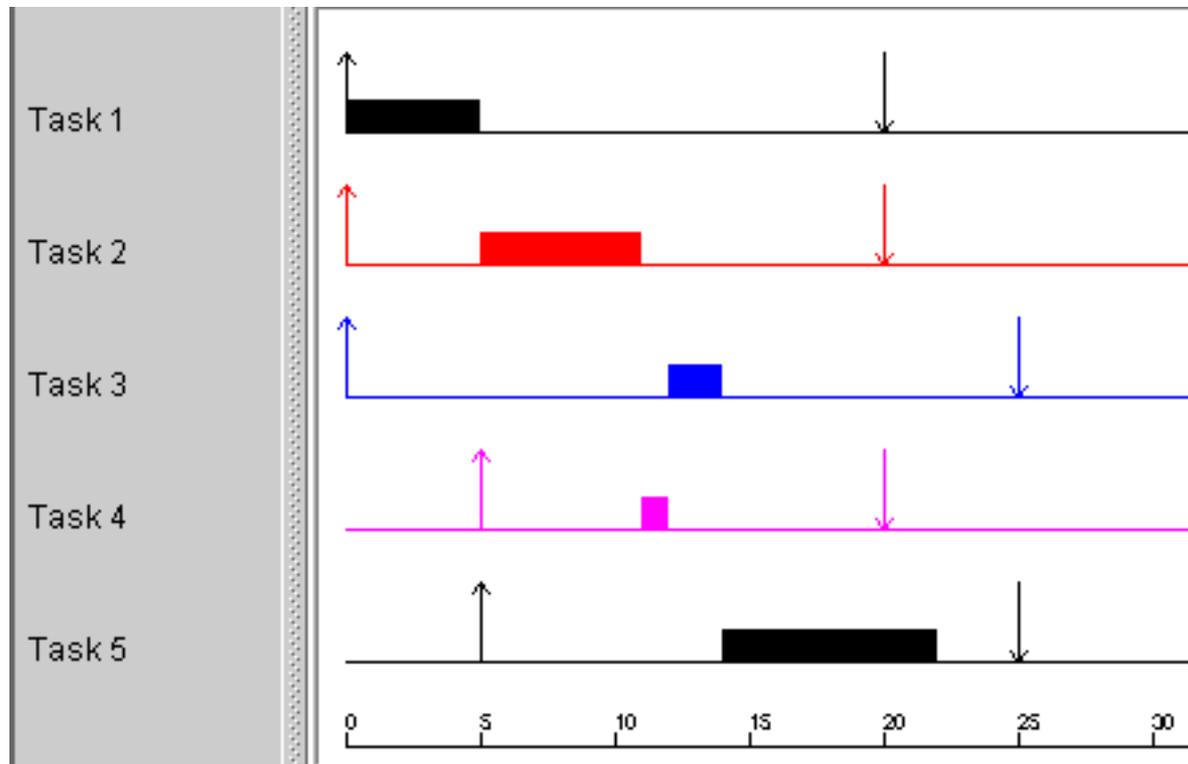
Deadline Assignment With EA

For task $p_i \in P$:

$$d_i = \min_j \{ \delta_j \mid p_i \in P_j \}$$

$$D_i = \min_j \{ \Delta_j \mid p_i \in P_j \} - \sum_k w_k, p_k \in P \wedge p_k < p_i$$

EA Scheduling Chart



EA Issues

Advantages

- Few preemptions
- Schedulability test available
- A fixed priority scheduler could be used

Drawbacks

- All tasks in a path have the same deadline
- Tasks are always activated
- The scheduler must be modified
- HW-mapped blocks need special treatment
- Generates a large task set

Late Activation

- Similar to EA
- Same requirements as EA
- Activation are *after* execution
- Generates the same task set as EA
- Usually produces the same schedule as EA
- Slight efficiency improvement

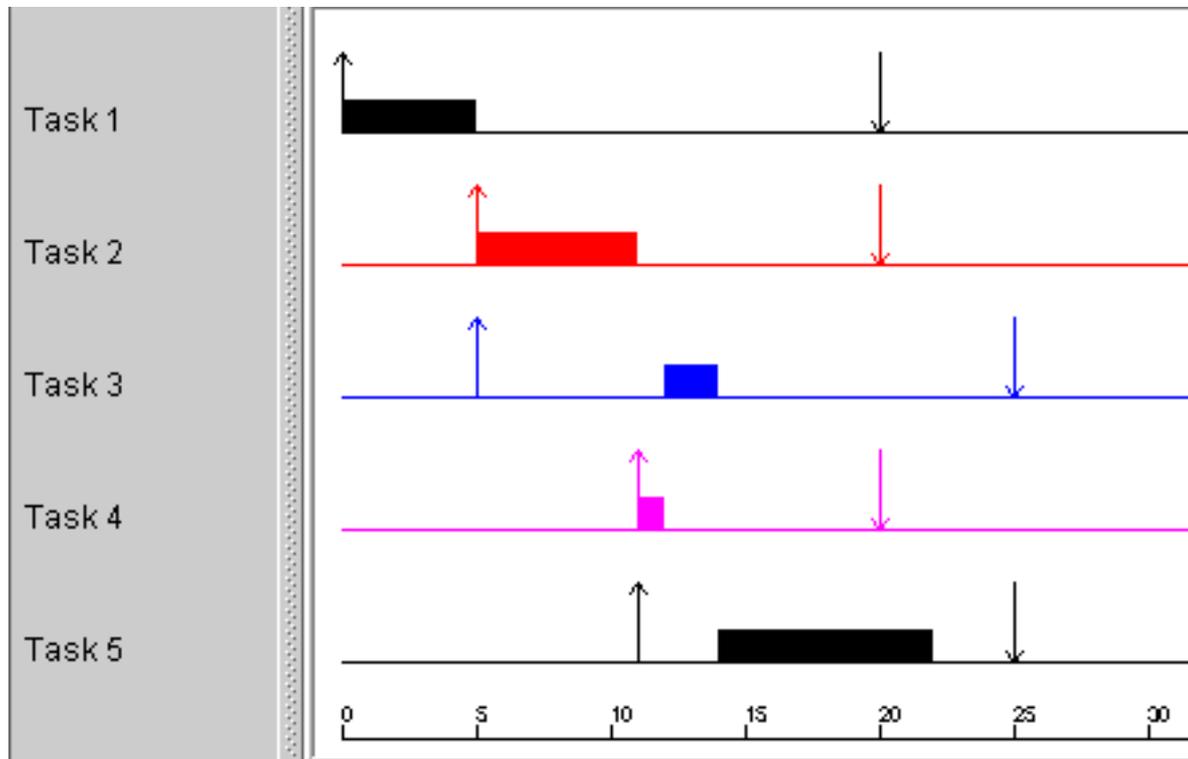
Deadline Assignment With LA

For task $p_i \in P$:

$$d_i = \min_j \{ \delta_j \mid p_i \in P_j \}$$

$$D_i = \min_j \{ \Delta_j \mid p_i \in P_j \} - \sum_k (w_k + c_k), p_k \in P \wedge p_k < p_i$$

LA Scheduling Chart



LA Issues

Advantages

- Few preemptions
- Tasks are activated only as required
- Worst-case analysis is the same as EA
- Normal treatment for HW-mapped blocks
- Can work with fixed priorities
- Variant: *signaled activation*

Drawbacks

- Deadline assignment is no better than EA
- Large number of tasks created

Rising Deadline

- Requires knowledge or esteem of WCETs
- The activation protocol is the same as in late activation
- The task set is the same as in previous algorithms
- Deadline assignment is greatly improved
- Schedule is generally different

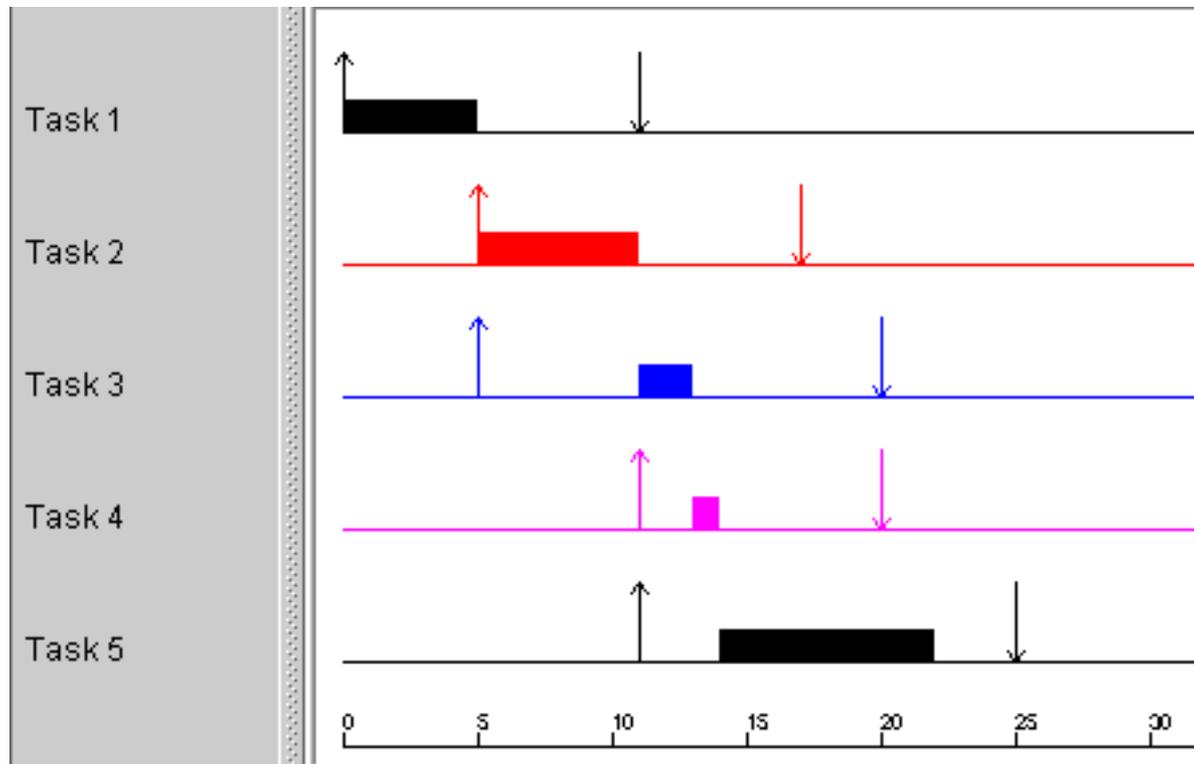
Deadline Assignment With RD

For task $p_i \in P$:

$$B_i = \min_j \{ \Delta_j - WCET_{P_j}(p_i) \}$$

$$D_i = B_i - \sum_k (w_k + c_k), p_k \in P \wedge p_k < p_i$$

RD Scheduling Chart



RD Issues

Advantages

- Every task has its own deadline
- Tasks are activated only if needed
- Base deadlines can be computed offline
- Fits easily in a codesign environment
- Can work in fixed priorities

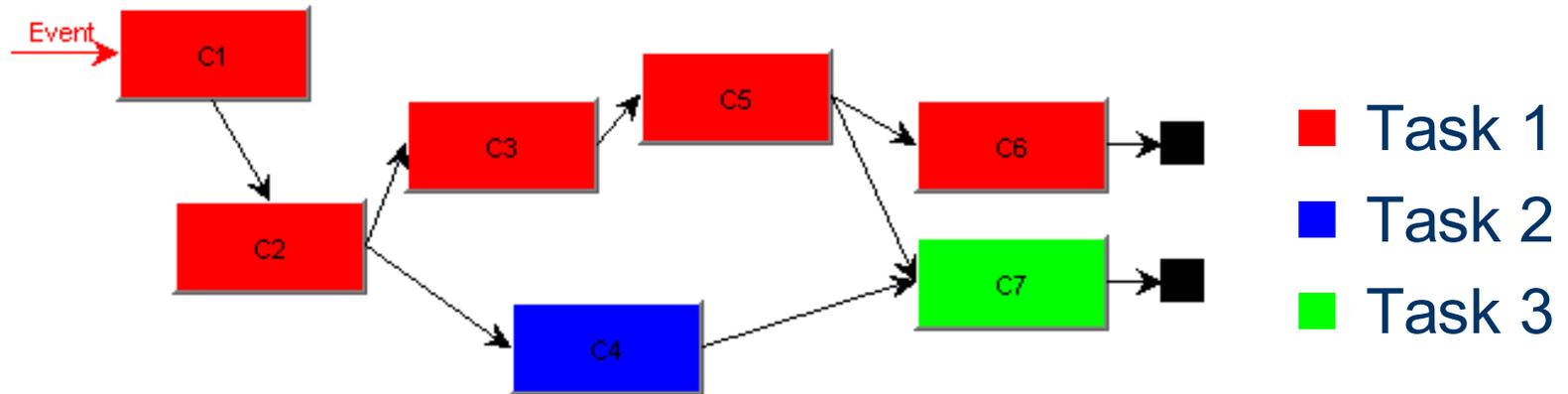
Drawbacks

- Introduces greater overhead
- No schedulability analysis currently available

Joined Rising Deadline

- Derived from the RD algorithm
- Requires knowledge or esteem of WCETs
- Uses the signaled activation semantics
- Generates a smaller set of larger tasks
- Better use of resources
- Greater algorithm overhead

Process Partitioning With JRD



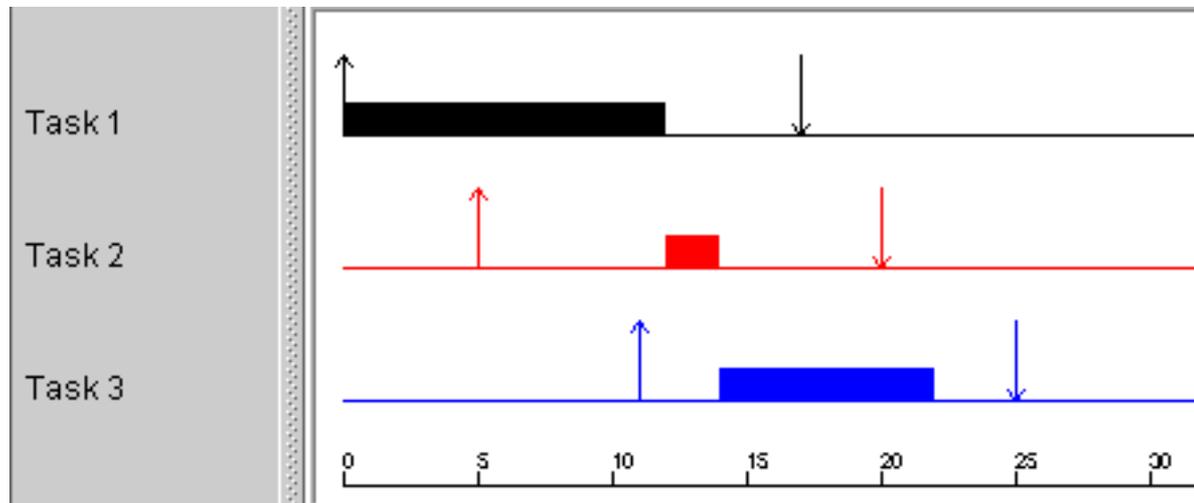
Deadline Assignment With JRD

For task $p_i \in P$:

$$B_i = \min_j \{ \Delta_j - WCET_{P_j}(p_i) \}$$

$$D_i = B_i - \sum_k (w_k + c_k(t_k)), p_k \in P \wedge p_k < p_i$$

JRD Scheduling Chart



JRD Issues

Advantages

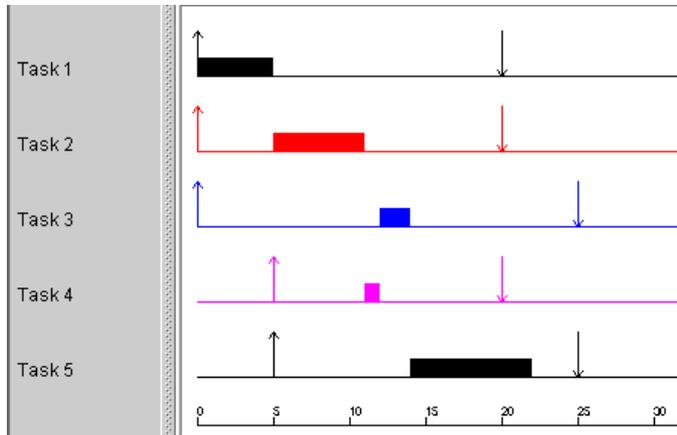
- Better efficiency, since less tasks must be scheduled and less memory is used
- A good implementation creates no more overhead than RD
- Can probably work on fixed priorities

Drawbacks

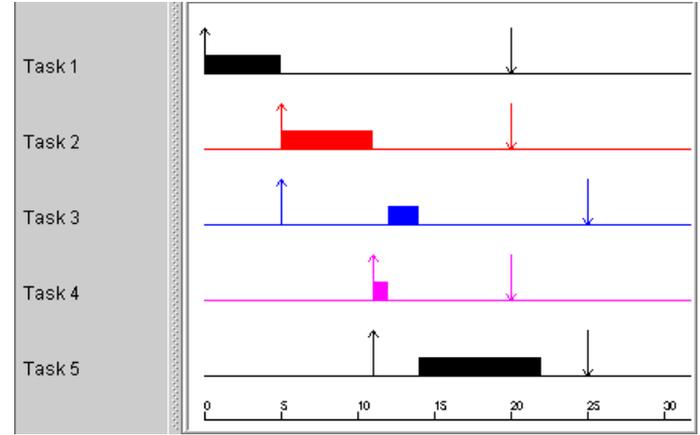
- High overhead
- Some path deadlines might become shorter than is required
- More likely that tasks miss their deadlines

Comparison

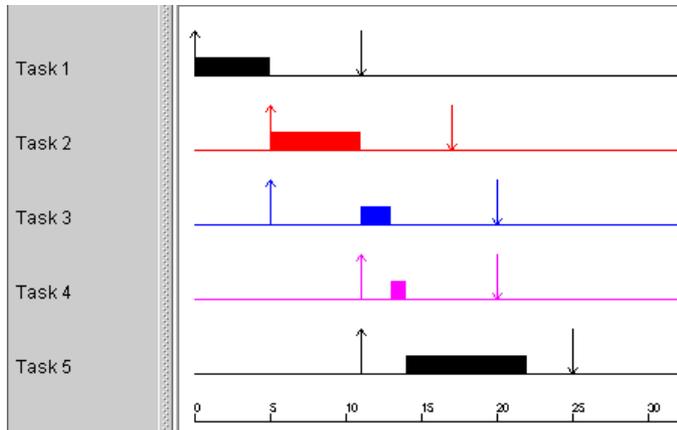
EA



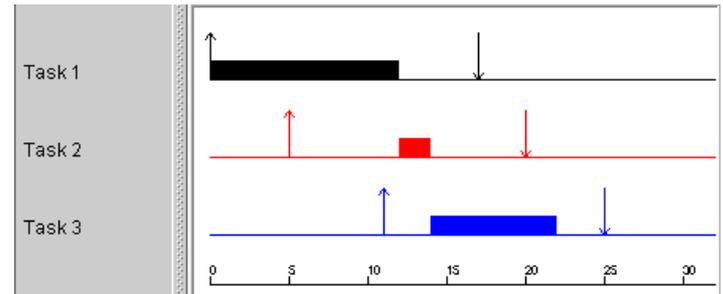
LA



RD



JRD



Conclusions and Future Work

- Algorithm comparison
 - Metrics
 - Number of tasks
 - Schedulability
 - Overhead on the scheduler
 - Probably, there is no best choice
 - Different applications would require different algorithms
- Extensions
 - Fixed priority schedulers and non-preemptive schedulers
 - HW/SW Co-design
 - Multi-processor scheduling
 - Distributed systems