Density of Multi-Task Real-Time Applications

Prof. Victor Nikiforov, SPIIRAS
Prof. Sergey Baranov, SPIIRAS, ITMO, member of IEEE

nik@iias.spb.su; snbaranov@gmail.com
Outline

- Model parameters of a real-time application
- Scheduling modes
- Protocols to access shared resources
- Necessary and sufficient conditions for mutual blocking of tasks in terms of graphs
- New protocol of inter-partite contours
- Compound and chained blocking of tasks
- Simulation technique for estimating real-time application properties
- Conclusions
Hardness and Density of RTA – 1

Hardness

\[ H_i = \frac{T_i}{D_i} \]

RT system

Processor

Performance \( P \) (op/second)

Input messages

\( T_1 \)

\( T_2 \)

\( T_n \)

\( T_i \) – period (seconds)

Output messages

\( f_1 \)

\( f_2 \)

\( f_n \)

\( D_1 \)

\( D_2 \)

\( D_n \)

\( D_i \) – deadline (seconds)

RTA – Real-Time Application

Association FRUCT, Yaroslavl, Russia, 20-24 April 2015
Hardness and Density of RTA – 2

Hardness

\[ H_i = \frac{T_i}{D_i} \]

RT system

Processor

Performance \( P \) (op/second)

Application:

\( T_i \) – period (seconds)

\( W_i, C_i, u_i \)

Absolute weight \( W_i \) (op)

Utility

\[ u_i = \frac{C_i}{T_i} \]

Relative weight \( C_i = \frac{W_i}{P} \) (seconds)

\( D_i \) – deadline (seconds)

Density

\[ Dens = U_{\text{max}} \]

Association FRUCT, Yaroslavl, Russia, 20-24 April 2015
Utility Depends on CPU Performance

Application

\[ T = 15\text{ms}, \ W = 4 \times 10^6\text{op} \]

\[ T = 19\text{ms}, \ W = 5 \times 10^6\text{op} \]

\[ T = 25\text{ms}, \ W = 7 \times 10^6\text{op} \]

Rate-monotonic scheduling

\[ H_i = 1 : \ Dens = 0.81 \]

Early deadline first scheduling

\[ H_i = 1 : \ Dens = 1 \]

Association FRUCT, Yaroslavl, Russia, 20-24 April 2015
Hierarchy of Scheduling Modes

- **RM** (Rate Monotonic)
- **EDF** (Early Deadline First)
- **RR** (Round Robin)
- **LLF** (Least Laxity First)
- **LWR** (Least Work Remaining)
- **FIFO** (First In First Out)
- **Pfair** (Proportional-fair)
- **DM** (Deadline Monotonic)

**$S_0$** – scheduling with static task priorities

**$S_1$** – scheduling with constant job priorities

**$S_2$** – no constraints for changing job priorities

Association FRUCT, Yaroslavl, Russia, 20-24 April 2015
## Efficiency of Scheduling Modes

<table>
<thead>
<tr>
<th>Scheduling mode</th>
<th>Task priorities, hardness</th>
<th>Number of processor cores</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RM</strong> rate-monotonic</td>
<td>Static</td>
<td>1</td>
<td>$Dens &gt; \ln 2$ (~ 69%)</td>
</tr>
<tr>
<td><strong>RM-US</strong> modification of RM</td>
<td>$H_i = 1$</td>
<td>$m &gt; 1$</td>
<td>$Dens &gt; m/(3m-2)$ (from 33% to 50%)</td>
</tr>
<tr>
<td><strong>EDF</strong> early deadline first</td>
<td>Dynamic</td>
<td>1</td>
<td>$Dens = 1$ (100%)</td>
</tr>
<tr>
<td><strong>Pfair</strong> proportional fair</td>
<td>$H_i = 1$</td>
<td>$m &gt; 1$</td>
<td>$Dens = 1$ (100%)</td>
</tr>
</tbody>
</table>

Scheduling with dynamic task priorities is more efficient, especially for multi-core processors.
Protocols for Access to Resources

The simplest protocol (ensures only integrity protection)

Condition to entry a critical interval: the resource is unlocked
Action at entering a critical interval: lock the resource
Action at exiting a critical interval: unlock the resource

Other protocols

When entering a critical interval:
check additional conditions, perform additional actions

When exiting a critical interval:
perform additional actions
## Properties of Access Protocols

<table>
<thead>
<tr>
<th>Protocol name</th>
<th>Protocol characteristics</th>
<th>Use with dynamic task priorities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Priority inheritance</td>
<td>Protection against mutual task blocking</td>
</tr>
<tr>
<td>Simplest</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Priority inheritance</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Preventive priority inheritance</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Priority ceiling</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
The necessary condition of mutual blocking — existence of bundles (intersections) of critical intervals of the tasks
Bundles Dependencies

Mutual blocking is **impossible**

Mutual blocking is **possible**

**Bundle**

$L(Task_1, g_1, g_2)$

**Dependent bundles**

$L_x$ depends on $L_y$, if
- they belong to different tasks;
- additional resource $L_x$ coincides with the heading resource $L_y$

$g_2 g_3$ depends on $g_3 g_4$

$g_4 g_5$ depends on $g_5 g_1$
Graph of Critical Interval Bundles – 1

Mutual blocking is possible

Task_1 → g_1 → g_2
Task_2 → g_2 → g_3
Task_3 → g_3 → g_4
Task_4 → g_4 → g_5
Task_5 → g_5 → g_1

Multi-partite graph of bundles

Partite of Task_1
Partite of Task_2
Partite of Task_3
Partite of Task_4
Partite of Task_5

Presence of inter-partite contours — the necessary and sufficient condition for a possibility of mutual task blocking

Association FRUCT, Yaroslavl, Russia, 20-24 April 2015
Graph of Critical Interval Bundles – 2

Presence of inter-partite contours — the necessary and sufficient condition for a possibility mutual task blocking

Association FRUCT, Yaroslavl, Russia, 20-24 April 2015
Protocol of Inter-Partite Contours

Task_1

Task_2

Task_3

Task_4

Active bundle of Task_i

Cursor of task Task_i is in the bundle header

Counter $w_j$ Stores the number of active bundles in the $j^{th}$ inter-partite contour of the length $l_j$

Additional condition to enter the bundle heading interval

The value of the counter $w_j$ of the contour which the bundle heading interval belongs to is not greater than $l_j-2$

Additional actions at entering the bundle heading interval

Increment at 1 the value of the counter $w_j$

Additional actions at entering the bundle additional interval

Decrement at 1 the value of the counter $w_j$
Chained Blocking in a System with Static Task Priorities

Multi-Partite Graph of Bundles and Critical Intervals

Arrows are built:
- from $L_i^{<g_j, g_k>}$ to $L_i^{[*]<g_j^{*}, g_k^{*}>}$ if $i \neq i^{*}$ and $k = j^{*}$
- from $L_i^{<g_j, g_k>}$ to $G_i^{[*]<g_j^{*}>}$ if $i \neq i^{*}$ and $k = j^{*}$
- from $G_i^{<g_j>}$ to $L_i^{[*]<g_j^{*}, g_k^{*}>}$ if $i \neq i^{*}$ and $j = j^{*}$

Chained blocking of task $\tau_i$ is possible if there is an inter-partite route from the partite of $\tau_i$ with two or more arcs which pass through partites of tasks with lower priorities than that of $\tau_i$
Simulation Technique for Execution a Multi-Task Real-Time Application

File with application features ➔ Simulation engine ➔ Sequence of system events ➔ Lists of active jobs ➔ Model time ➔ File with modeling results

Sequence of system events
Task descriptions

Simulation engine

Lists of active jobs
Model time

File with modeling results
Hardness/Density Dependences for EDF and RM Scheduling Modes

<table>
<thead>
<tr>
<th>Task</th>
<th>$\tau_1$</th>
<th>$\tau_2$</th>
<th>$\tau_3$</th>
<th>$\tau_4$</th>
<th>$\tau_5$</th>
<th>$\tau_6$</th>
<th>$\tau_7$</th>
<th>$\tau_8$</th>
<th>$\tau_9$</th>
<th>$\tau_{10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_i$</td>
<td>100</td>
<td>107</td>
<td>114</td>
<td>123</td>
<td>132</td>
<td>141</td>
<td>151</td>
<td>165</td>
<td>174</td>
<td>187</td>
</tr>
<tr>
<td>$C_i$</td>
<td>7,2</td>
<td>7,7</td>
<td>8,3</td>
<td>8,9</td>
<td>9,5</td>
<td>10,2</td>
<td>10,9</td>
<td>11,6</td>
<td>12,4</td>
<td>13,2</td>
</tr>
</tbody>
</table>

![Graph showing EDF and RM scheduling modes](image)

Association FRUCT, Yaroslavl, Russia, 20-24 April 2015
Conclusions

1. The notion of application density permits to compare the efficiency of particular combinations of a scheduling mode and an access protocol for execution of real-time applications.

2. The proposed protocol of inter-partite contours increases application density against other known protocols which prevent mutual task blocking.

3. However, the proposed protocol of inter-partite contours admits compound and chained blocking which decreases the application density.

4. Simulation of application execution allows to decide which protocol to use in order to increase the application density and therefore, to increase the efficiency of the processor usage without the threat of mutual task blocking.

MANY THANKS FOR YOUR ATTENTION!