An Optimal Algorithm for Scheduling Soft-Aperiodic Tasks in Fixed-Priority Preemptive Systems

Junwon Kang

Real-Time Computing and Communications Lab.
Hanyang University
Contents

- Introduction
- Framework and Assumptions
- Slack Stealing Algorithm
- Conclusion
Main idea

- Seek to schedule a mixture of periodic and aperiodic tasks
- All periodic task deadlines are met
- The response times for the aperiodic tasks are as small as possible
- No server for aperiodic tasks
- Use slack (= deadline – current time – execution time) for executing aperiodic tasks
Slack stealing

- Periodic server for aperiodic tasks is not needed
- Steals all the processing time from the periodic tasks without causing their deadlines to be missed
- Periodic tasks are scheduled using a fixed priority algorithm.

Slack stealer

- Relies on the exact schedulability conditions for RM and DM
- The slack stealer provides the maximum possible capacity for aperiodic service.
- Is an optimal algorithm.
  - The shortest aperiodic response time for aperiodic tasks.
Assumptions

- **A1:** All overhead for context swapping is assumed to be zero.
- **A2:** Tasks are already at their periodic and do not suspend themselves or synchronize with any other task.
- **A3:** Any task can be instantly preempted.
- **A4:** There is unlimited buffer space for the aperiodic tasks.
# Slack Stealing Algorithm

## DM Scheduling for periodic tasks

<table>
<thead>
<tr>
<th>C</th>
<th>T</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Task 2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Aperiodic Task</td>
<td>2</td>
<td>-</td>
</tr>
</tbody>
</table>

![Scheduling diagram](image)

- **Task 1**
- **Task 2**
- Slack
Slack Stealing Algorithm

DM Scheduling for periodic tasks

<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>T</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Task 2</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Aperiodic Task</td>
<td>2</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Slack

Task 1

Task 2

Slack
Slack Stealing Algorithm

- **Hyperperiod**

<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>T</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Task 2</td>
<td>3</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Aperiodic Task</td>
<td>2</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- We just focus on the interval of time \([0,H]\).
- \(H\) is the hyperperiod of the task set.
- It is the least common multiple of the task periods.

\[
\begin{align*}
2 & \quad 4 \quad 6 \\
2 & \quad 3
\end{align*}
\]

Hyperperiod = \(2 \times 2 \times 3 = 12\)
Slack Stealing Algorithm

Available Slack (absence of aperiodic tasks)

- Available Slack (absence of aperiodic tasks)

* Jump points: completion times (deadlines) of the jobs

\[ A_1(t) \]

- \( A_1(t) \): The largest amount of aperiodic processing possible at level 1 or higher.

\[ \text{Task1 (C:1, D:1, T:4)} \]

\[ \text{Task2 (C:3, D:6, T:6)} \]

\[ \min_{1 \leq i < V_s, 0 < t < D_{si}} \{ (A_{ij} + P_i(t))/t \} = 1. \]

\( P_i(t) \): the periodic ready work in \([0, t]\) (\(1 \leq p \leq i\))
Slack Stealing Algorithm

Available Slack (absence of aperiodic tasks)

*A slack Stealing Algorithm*

\[ A_2(t) \]

\[ \text{min}_{10 \leq t < D_i} \{ (A_i + P_i(t))/t \} = 1. \]

\[ P_i(t) : \text{the periodic ready work in } [O, t) (1 \leq p \leq i) \]

\[ A_2(t) : \text{The largest amount of aperiodic processing possible at level 2 or higher.} \]
Slack Stealing Algorithm

- Available Slack (absence of aperiodic tasks)

\[ A^*(0,t) = \min \{1 \leq i \leq n \} \ A_i(0,t) \]
Slack Stealing Algorithm

Available Slack (aperiodic task exist)

A1(t)

A2(t)

A*(0,t)

Aperiodic task arrived. Arrival time: 5.5sec
Run time: 2

Task1 (C:1, D:1, T:4)

Task2 (C:3, D:6, T:6)

Aperiodic Task Priority = Task1

Aperiodic Task Priority = Task2

Usable Slack

Real-Time Computing and Communications Lab., Hanyang University
http://rtcc.hanyang.ac.kr
Slack Stealing Algorithm

Aperiodic Task

- Aperiodic Task arrival time = 5.5 sec
- Aperiodic Task run-time = [5.5 , 7.5] sec
- Aperiodic Task response time = 2 sec
Conclusion

- Slack stealing
  - No periodic server
  - Steals time from periodic tasks without causing any deadline
  - It is optimal in the sense that it minimizes the response time of soft aperiodic tasks
Q & A

- Thank you.