Joint Network and Computing Resource Scheduling for Wireless Networked Control Systems

Peng Wu¹  Chencen Fu²  Minming Li²  Yingchao Zhao³  
Jason Xue²  Song Han¹

¹University of Connecticut

²City University of Hong Kong

³Hong Kong Caritas Institute of Higher Education
Wireless Networked Control systems

WNCSs enable wireless networked control, actuating, and feedback
- great advantages in applications such as industrial monitoring and control [1], building automation [2], manufacturing [3], etc.

Structure of WNCSs from the perspective of Control Community
- Multiple control loops
- Shared CPU
- Shared communication medium
- Designated end-to-end timing requirement
- Strict execution order
  - Sensing
  - Computing
  - Actuating

Joint Resource Scheduling (JRS) Model

We observe one control loop of WNCSs from the perspective of real-time system community

The joint task model is defined as:

$$\tau: \tilde{s}(h) \rightarrow \tilde{c}(m) \rightarrow \tilde{a}(k)$$

- Sensing segment $\tilde{s}$ uses network resource
- Computing segment $\tilde{c}$ uses CPU resource
- Actuating segment $\tilde{a}$ uses network resource

where we use $h$, $m$ and $k$ to represent the execution/transmission time of the three segments, respectively.

**Assumption:** Unit size of network and computing resource slices are the same. The wireless network has only one channel and the CPU is a preemptive uniprocessor.
Joint Scheduling Problem Formulation

Joint Network and Computing Resource Scheduling (JNCRS) Problem:
- Given a set of real-time joint tasks \(\{\tau_1, \tau_2, \ldots, \tau_n\}\). Each task \(\tau_i\) is associated with a release time \(r_i\), a deadline \(d_i\), and consists of three consecutive and dependent segments. The objective of the JNCRS problem is to find a feasible schedule so that the deadlines of all real-time tasks are met.

Divide the **JNCRS problem** into 4 subproblems based on the size of the execution time of each segment.

<table>
<thead>
<tr>
<th>Task Model</th>
<th>Complexity or Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\tau: \tilde{s}(1) \rightarrow \tilde{c}(1) \rightarrow \tilde{a}(1))</td>
<td>Polynomial-time solvable (Alg.1)</td>
</tr>
<tr>
<td>(\tau: \tilde{s}(1) \rightarrow \tilde{c}(m &gt; 1) \rightarrow \tilde{a}(1))</td>
<td>Polynomial-time solvable (Alg.2)</td>
</tr>
<tr>
<td>(\tau: \tilde{s}(1) \rightarrow \tilde{c}(m &gt; 0) \rightarrow \tilde{a}(1))</td>
<td>Unknown</td>
</tr>
<tr>
<td>(\tau: \tilde{s}(h) \rightarrow \tilde{c}(m) \rightarrow \tilde{a}(k))</td>
<td>NP-hard</td>
</tr>
</tbody>
</table>
Joint Scheduling Algorithms (Alg. 1 and Alg. 2)

Alg. 1: Unit-Size Joint Resource Scheduling (JRS-US) algorithm
- The key idea is to identify all the intervals with network resource utilization of 100% and modify the deadline of a sensing/actuating segment if its deadline is within but its release time is outside that interval.
  - After the deadline modifications, Earliest Deadline First (EDF) can be used.

Alg. 2: Joint Resource Scheduling (JRS-US) algorithm under 1-m-1 (m>1)
- Modify deadline as Alg. 1, but also needs to modify release time.
- Design Conditional/definite release time/deadline for the network segment, due to different preemption cases of computing segment.
Future Work

- Continue study the remaining subproblems and design the efficient scheduling algorithms on
  - 1-m-1 model (m>0)
  - general model

- Evaluate effectiveness and practicability of the proposed algorithms in real-life systems by implementing them on our WNCS testbed.

- Relax the assumptions on
  - number of wireless network channels and number of CPU cores
  - network topology (sensors, actuators, controllers)