From Logical Time Scheduling to Real-Time Scheduling

Frédéric Mallet - Min Zhang

Université Cote d’Azur, France
East China Normal University, China
CCSL and logical clocks

- **The Clock Constraint Specification Language (CCSL)**

- **Logical clocks** (Lamport, synchronous languages, Tag Systems-LSV)
  - Clock = (infinite) sequence of instants \( \leftrightarrow \) events
  - Instant = Event Occurrences

- **Asynchronous** relations (causal dependencies)
  - a \textit{causes} b: \( \forall i \in \mathbb{N}, a[i] \leq b[i] \)
  - Jitter: \( \forall i \in \mathbb{N}, |a[i] - b[i]| < C \)

- **Synchronous** relations (Simultaneity and clock inclusion)
  - a \textit{isSubclockOf} b: \( \forall i \in \mathbb{N}, a[i] \equiv b[f(i)] \)
  - Periodicity: \( f(i) = \text{period} \times i + \text{offset} \) (pure delay: period = 1)
Multiform Logical (discrete) clocks

Clocks are \textit{a priori} independent

\begin{itemize}
  \item [Clock]  
  \item [precedence]  
  \item [exclusion]  
  \item [coincidence]
\end{itemize}
A Simple Task Model

- **Task** $t_i$
  - $A_i =$ arrival time
  - $S_i =$ start time
  - $f_i =$ finish time
  - $D_i =$ deadline

- **Timing / causal constraints**
  - Non-reentrant task: $s_i \sim f_i$
  - Task dependency: $f_i \leq s_j$
  - Periodic arrival: $a_i$ every $P$ unit
  - Duration: $t_i(d) = s_i$ sampled on $d$ unit
  - Interval: $t_i(d_{\text{min}}) \leq f_i \leq t_i(d_{\text{max}})$
  - Deadline: $f_i \leq D_i$

- **Processor** $P_j$
  - $\text{Acq}_j =$ acquisition time
  - $\text{Rel}_j =$ release time

- **Allocation constraints**
  - Mono-processor: $\text{Acq}_j \sim \text{Rel}_j$
  - Allocation: $\text{Acq}_j = s_{i1} + s_{i2}$
    $\text{Rel}_j = f_{i1} + f_{i2}$

- **Preemption**
  - Not preemptive: $\text{unit}_i = _{100\text{ms}}$
  - Preemptive:
    $\text{suspend unit}_i$ in $[s_j, f_j]$
Conclusion

- Use logical clocks to capture causal, timing, resource constraints

**Strength**
- Find a CCSL schedule $<=$ perform real-time schedulability analysis
- Declarative accumulative constraints vs. other *generic* approaches [TIMES]
- TimeSquare computes a partial schedule http://timesquare.inria.fr

**Weaknesses**
- Finding a schedule for CCSL is NP-Hard
- But efficient SMT-based solutions in standard cases