39th IEEE Real-Time Systems Symposium
December 12th 2018

From Logical Time Scheduling to Real-Time Scheduling

Frédéric Mallet - Min Zhang

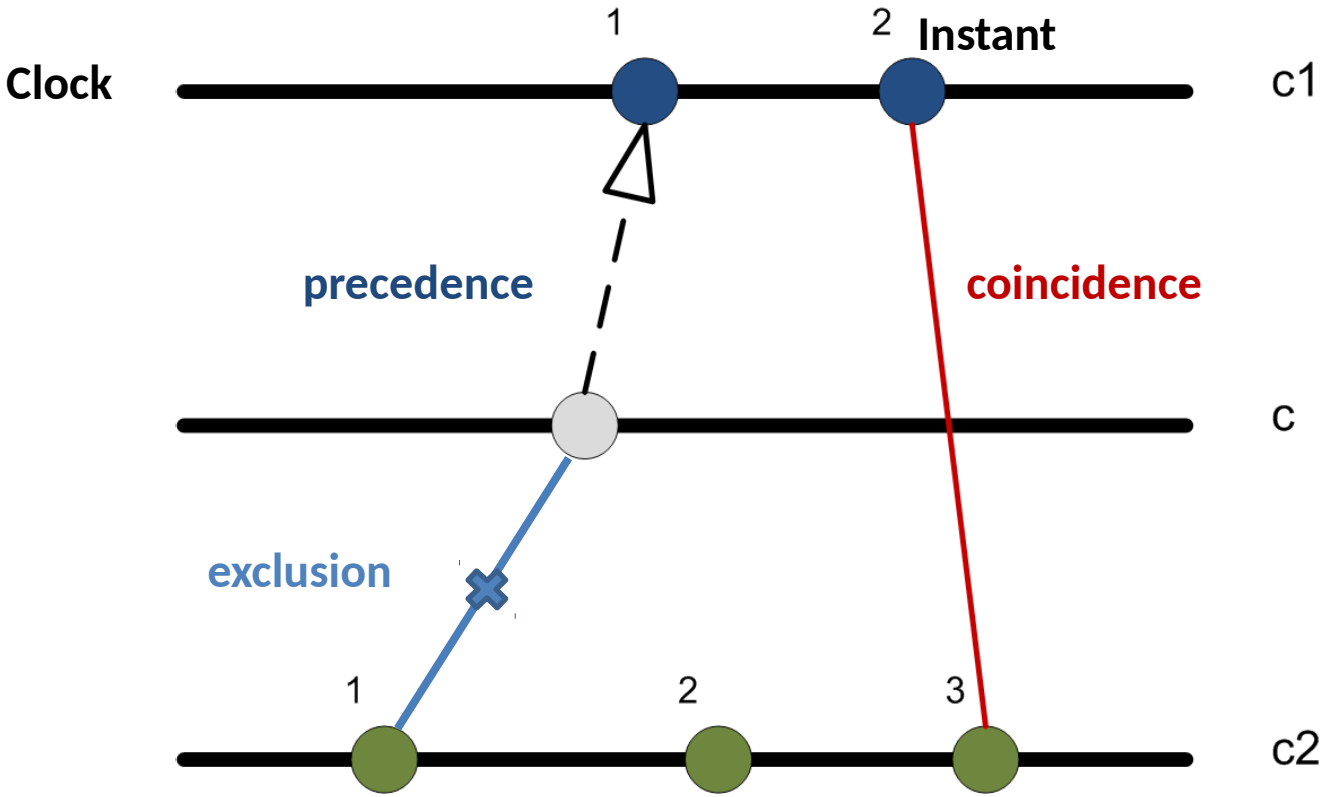
Université Cote d'Azur, France
East China Normal University, China

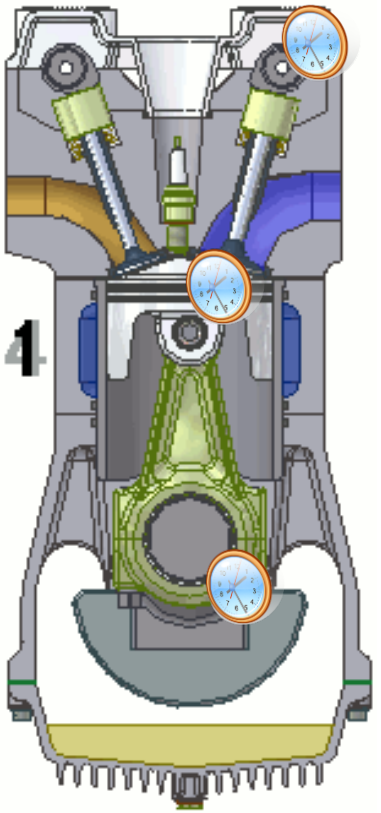
CCSL and logical clocks

- The **C**lock **C**onstraint **S**pecification **L**anguage (CCSL)
- Logical clocks (Lamport, synchronous languages, Tag Systems-LSV)
  - Clock = (infinite) sequence of instants <=> events
  - Instant = Event Occurrences
- **Asynchronous** relations (causal dependencies)
  - a 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 isSubclockOf b: $\forall i \in \mathbb{N}, a[i] \equiv b[\mathbf{f(}i\mathbf{)}]$
  - Periodicity: f(i) = period x i + offset (pure delay: period = 1)

Multiform Logical (discrete) clocks

- Clocks are *a priori* independent





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: $t_i(d_{min}) \leq f_i \leq t_i(d_{max})$
  - Deadline: $f_i \leq D_i$
- Processor P_j
  - Acq_j = acquisition time
  - Rel_j = release time
- Allocation constraints
  - Mono-processor: $Acq_j \sim Rel_j$
  - Allocation: $Acq_j = s_{i1} + s_{i2}$, $Rel_j = f_{i1} + f_{i2}$
- Preemption
  - Not preemptive: $unit_i$ = _100ms
  - Preemptive: 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

[Min Zhang, Feng Dai and F. Mallet. Periodic scheduling for MARTE/CCSL: Theory and practice. Science of Computer Programming 154:42-60, Mar. 2018.]