SEN9110 Simulation Masterclass Lecture 7: Distributed Simulation

Prof.dr.ir Alexander Verbraeck
a.verbraeck@tudelft.nl

Brightspace: SEN9110

Why distribute simulation models?

- for combining several independent (simulation) models
- for combining different background disciplines
- for reusing already existing models
- for collaborative work between different organizations
- for parallel design and development
- for hiding internal information
- for correspondence to reality: several distinguishable objects in reality
- for speed of the model: spreading the load over more processors
- for enabling to spread the model geographically
- for maintenance of several smaller models versus one large model
- for real-time interaction with the model

Distributed versus parallel simulation

Parallel

- Using multiple processors to execute a simulation model
  - speeding up a single model run by parallel execution
  - GPU / CPU possible
- Using multiple processors to execute a simulation experiment
  - multiple runs / replications in parallel (they are independent)

Distributed

- Using multiple computers to execute a simulation model
  - distributing a single model over multiple computers
- Using multiple computers to execute a simulation experiment
  - multiple runs / replications on different computers (they are independent)

There can be overlap: distributed execution *can implicate* parallel execution

Distributed versus parallel simulation

Different **focus:**

- Parallel discrete-event simulation (PDES) focuses on accelerating the speed of simulation model execution
- Distributed simulation focuses on interconnecting separately developed models running on different computers

Requirements

Strict

Simulation model execution shall be reproducible

- independent of the distribution over computers
- independent of the types of computers
- and provide the same results when distributed in any way
- and provide the same results as when executed on a single processor

Loose

Simulation model execution shall be reproducible

- independent of the types of computers
- and provide the same results when distributed in the same way

(so-called 'race conditions' in software should be avoided)

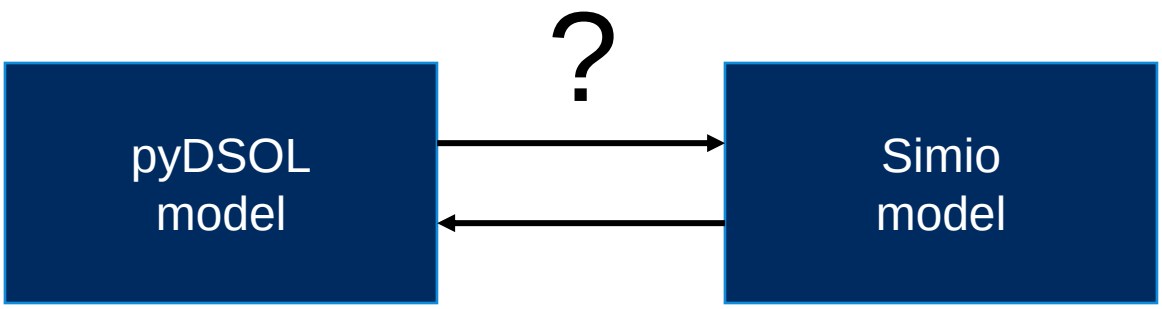
How to synchronize clocks

- Wall clock versus simulation clock
  - advantages?
  - disadvantages?
- How do you solve model parts that are (sometimes) slower than reality?

Definitions of time

- **Physical time**
  - time in the physical system that is modeled
  - can be in the past or future
  - example: supermarket between 7 and 8 on August 17, 2016
- **Simulation time** (logical time)
  - representation of physical time in simulation
  - example: 312212.342
- **Wall clock time**
  - time during execution of a simulation model, e.g. read from the hardware clock
  - example: 09:02:11 on Sep 20, 2017

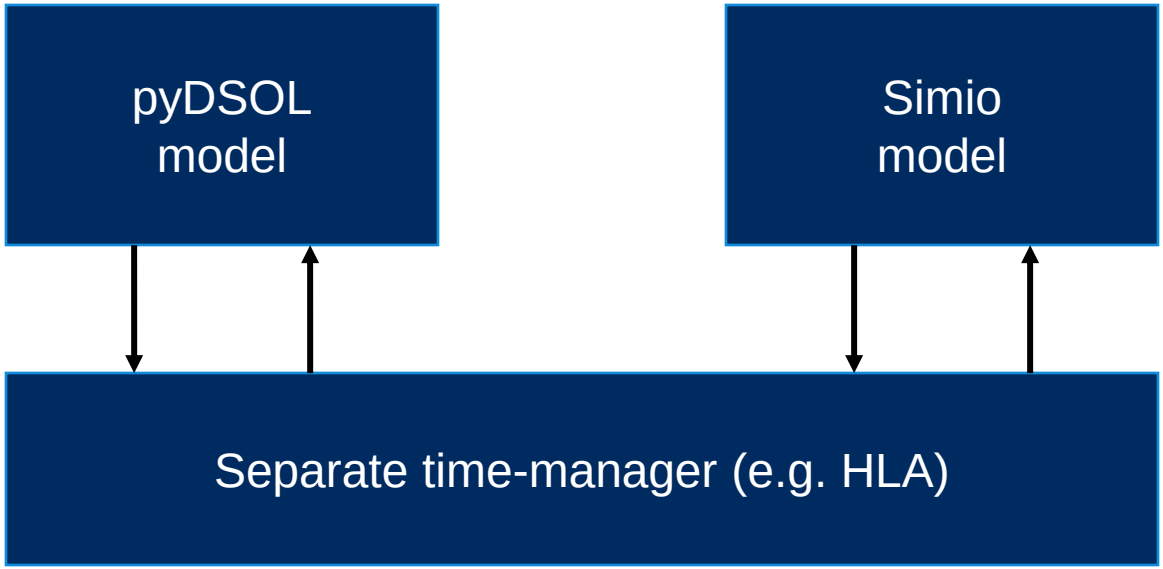
Example 1:



Own event calendar

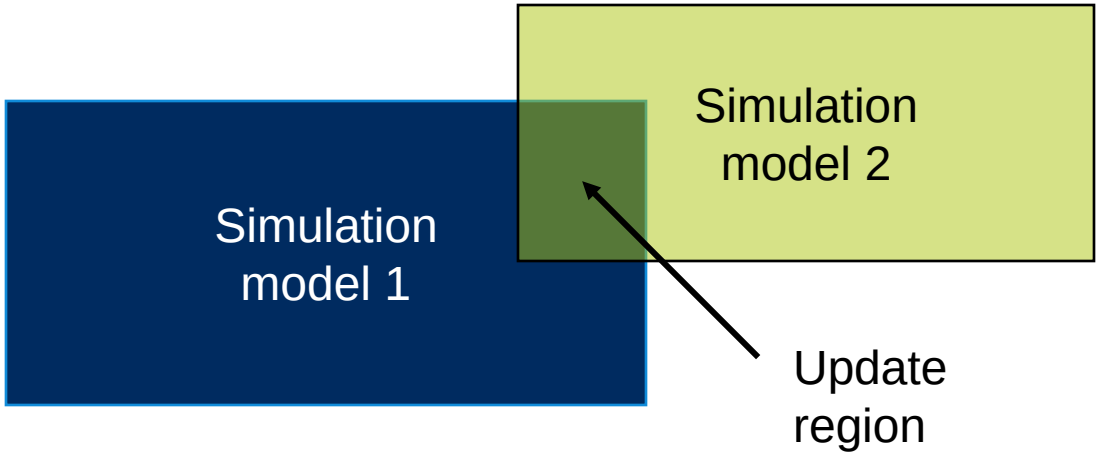
Own event heap

Possible solution



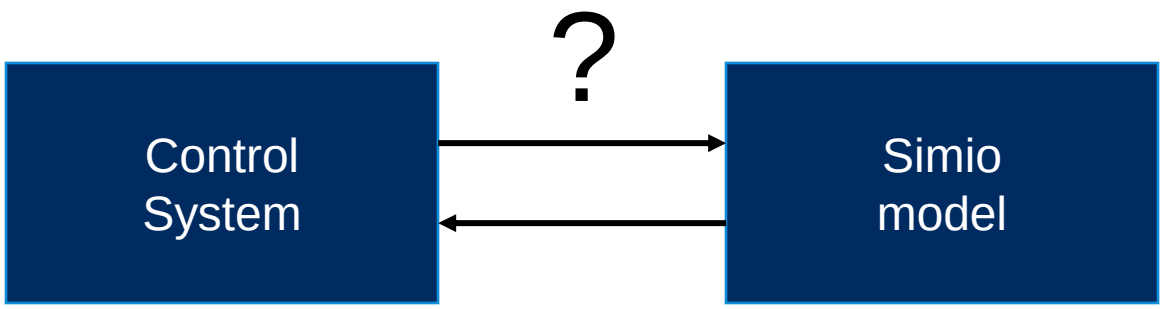
Important choices

- ALL events synchronized
- Only events that matter for another model
- Only events that can be influenced by another model



- Expected problems?
- How to determine?

Example 2:



Takes wall time

Own event heap

Synchronization: conservative techniques

Conservative:

- fully reproducible
- execution using TSO (Time Stamped Order)
- sometimes RO (Receive Order) is used; TSO is considered better (why?), but RO is easier (why?)
- a model does not execute an event until it can guarantee that no event with a smaller timestamp will later be received by that model

- Lock-step time-driven
  - fixed steps
- Fixed time-bucket
  - global lookahead value L
- Chandy Misra
  - message queues for dependent objects

Synchronization: optimistic techniques

Optimistic:

- models do nut run synchronously
- they can "repair" if events from other models come in-between

- Time-warp
  - lookahead per model
  - rollback (lazy cancellation)
  - copy state saving
  - incremental state saving
  - anti-messages (reverse computation)
- Adaptive time-warp
  - lookahead per model re-calculated based on rollback

Synchronization: relaxed techniques

Relaxed

- relaxes the constraint that events be strictly processed in TSO
- models are allowed to progress a small simulation duration in parallel

How to guarantee reproducibility?

- only process inter-model communication at the end of the small duration
  - delay communication
  - store communication in a queue
- process all inter-model events at that time in a given order
- change the state of each model as a result
- start a new cycle

Reproducible, but not equal to single-computer/processor model (why?)

Encountered problems with parallel and distributed simulation

- Reliability
- Loss of messages
- Sequencing of messages
- Safety overhead
- Waste of bandwidth by external processes

Can you think of solutions to each of these?

General question

- What are the possible advantages of distributed simulations?
- Are distributed simulations faster in general?
- If no in general, under which conditions are they?
- When to use conservative/optimistic/relaxed scheduling?
- What about standard simulation packages and distribution?
- How to synchronize two simulations with regards to:
  - syntax
  - semantics
  - pragmatics

Homework

- Simulation assignment 2
- Paper meeting next week: send me the structure of the paper (and the first content), as well as a list of literature that you have considered until now the afternoon/evening before the meeting
- Read about distributed simulation in the Reader and try to understand the difficulties
- If you are interested in this topic, also browse the HLA materials (background reading) on the SEN9110 site