Foundations of Modelling and Simulation

Hans Vangheluwe

Modelling, Simulation and Design Lab (MSDL)

Department of Mathematics and Computer Science,
University of Antwerp, Belgium

School of Computer Science, McGill University,
Montréal, Canada
Hierarchy of System Specification of Structure and Behaviour

- Basis of System Specification:
  sets theory, time base, segments and trajectories

- Hierarchy of System Specification (causal, deterministic)
  1. I/O Observation Frame
  2. I/O Observation Relation
  3. I/O Function Observation
  4. I/O System

- Multicomponent Specifications

- Non-causal models

ref: Wayne Wymore, Bernard Zeigler, George Klar, . . .
Set Theory

Properties:

\{1, 2, \ldots, 9\}

\{a, b, \ldots, z\}

\mathbb{N}, \mathbb{N}^+, \mathbb{N}^+_\infty

\mathbb{R}, \mathbb{R}^+, \mathbb{R}^+_{\infty}

\text{EV} = \{\text{ARRIVAL, DEPARTURE}\}

\text{EV}^\phi = \text{EV} \cup \{\phi\}

Structuring:

A \times B = \{(a, b)|a \in A, b \in B\}

G = (E, V), V \subseteq E \times E
Comparing things
Nominal Scale: e.g., gender

A scale that assigns a *category label* to an individual. Establishes no explicit ordering on the category labels.

Only a notion of *equivalence* “=” is defined with properties:

1. Reflexivity: \( x = x \lor x \neq x \).
2. Symmetry of equivalence: \( x = y \iff y = x \).
3. Transitivity: \( x = y \land y = z \rightarrow x = z \).
Ordinal Scale: e.g., degree of happiness

A scale in which data can be *ranked*, but in which no arithmetic transformations are meaningful. It is meaningless to talk about difference (distance).

In addition to equivalence, a notion of order $< \triangleq$ is defined with properties:

1. Symmetry of equivalence: $x = y \iff y = x$.
2. Asymmetry of order: $x < y \implies y \not< x$.
3. Irreflexivity: $x \not< x$.
4. Transitivity: $x < y \land y < z \implies x < z$. 
Partial ordering

The ordering may be *partial* (some data items cannot be compared).

\[
∀x, y ∈ X : x < y \lor y < x \lor x = y
\]

The ordering may be *total* (all data items can be compared).
Interval Scale: e.g., Shoe Size

A scale where *distances* between data are meaningful. On interval measurement scales, one unit on the scale represents the *same magnitude* of the characteristic being measured across the whole range of the scale. Interval scales do not have a “true” zero point, however, and therefore it is not possible to make statements about how many times higher one value is than another.

In addition to equivalence and order, a notion of *interval* is defined. The choice of a zero point is arbitrary.
Ratio Scale: e.g., age

Both *intervals* between values and *ratios* of values are meaningful. A meaningful *zero* point is known. “A is twice as old as B”.

Time Base

- Simulation of **Dynamic** Systems: irreversible passage of *time*.

- **Time Base** $T$:
  - $\{NOW\}$ (instantaneous)
  - $\mathbb{R}$: *continuous-time*
  - $\mathbb{N}$ or isomorphic: *discrete-time*

- **Ordering**:
  - Ordinal Scale (possibly partial ordering, for concurrency)
  - Interval Scale
  - Ratio Scale
Time Bases for hybrid system models
Time Bases for hybrid system models

“nested time” for nested experiments.
Behaviour \equiv \text{Evolution over Time}

- With time base, describe \textit{evolution over time}

- Time function, \textbf{trajectory}, signal: \( f : T \rightarrow V \)

- Restriction to \( T' \subseteq T \)
  \[
  f|_{T'} : T' \rightarrow V, \ \forall t \in T' : f|_{T'}(t) = f(t)
  \]
  - Past of \( f \): \( f|_{T_t} \)
  - Future of \( f \): \( f|_{T_{\langle t}} \)

- Restriction to an interval: \textbf{segment}
  \[
  \omega : \langle t_1, t_2 \rangle \rightarrow V
  \]
Types of Segments

- Continuous
- Piecewise continuous
- Piecewise constant
- Discrete event
Cashier-Queue System

Physical View

Abstract View

Arrival [IAT distribution]  Queue  Cashier [ST distribution]  Departure

Arrival  Queue  Cashier  Departure
Trajectories

state = queue_length \times cashier_state

Input Events
Arrival

Output Events
Departure

queue_length

cashier_state

state = queue_length \times cashier_state
I/O Observation Frame (causal)

\[ O = \langle T, X, Y \rangle \]

- \( T \) is time-base: \( \mathbb{N} \) (discrete-time), \( \mathbb{R} \) (continuous-time)
- \( X \) input value set: \( \mathbb{R}^n, EV^\phi \)
- \( Y \) output value set: system response
I/O Relation Observation

\[ IORO = \langle T, X, \Omega, Y, R \rangle \]

- \( \langle T, X, Y \rangle \) is Observation Frame
- \( \Omega \) is the set of all possible input segments
- \( R \) is the I/O relation
  \( \Omega \subseteq (X, T), \ R \subseteq \Omega \times (Y, T) \)
  \( (\omega, \rho) \in R \Rightarrow \text{dom}(\omega) = \text{dom}(\rho) \)
- \( \omega : \langle t_i, t_f \rangle \to X \): input segment
- \( \rho : \langle t_i, t_f \rangle \to Y \): output segment
- note: not really necessary to observe over same time domain
I/O Function Observation

\[ IOFO = \langle T, X, \Omega, Y, F \rangle \]

- \( \langle T, X, \Omega, Y, R \rangle \) is a Relation Observation
- \( \Omega \) is the set of all possible input segments
- \( F \) is the set of I/O functions
  \[ f \in F \Rightarrow f \subset \Omega \times (Y, T) \]
  where
  \( f \) is a function such that \( \text{dom}(f(\omega)) = \text{dom}(\omega) \)
- \( f = \text{initial state: unique} \) response to \( \omega \)
- \( R = \bigcup_{f \in F} f \)
I/O System

- From **Descriptive Variables** (properties) to **State**.
- **State** summarizes the past behaviour of the system.
- Future is uniquely determined by
  - current state
  - future input
\[ \text{SYS} = \langle T, X, \Omega, Q, \delta, Y, \lambda \rangle \]

\begin{itemize}
\item \( T \) \quad \text{time base}
\item \( X \) \quad \text{input set}
\item \( \omega : T \to X \) \quad \text{input segment}
\item \( Q \) \quad \text{state set}
\item \( \delta : \Omega \times Q \to Q \) \quad \text{transition function}
\item \( Y \) \quad \text{output set}
\item \( \lambda : Q \to Y \) \quad \text{(or} \quad Q \times X \to Y \text{)} \quad \text{output function}
\end{itemize}

\[ \forall t_x \in [t_i, t_f]: \delta(\omega_{[t_i, t_f]}, q_i) = \delta(\omega_{[t_x, t_f]}, \delta(\omega_{[t_i, t_x]}, q_i)) \]
For a given initial condition $q$ and a given input segment $\omega$, we can define a state trajectory $STRAJ_{q,\omega}$ from $SYS$

$$STRAJ_{q,\omega} : \text{dom}(\omega) \rightarrow Q,$$

with

$$STRAJ_{q,\omega}(t) = \delta(\omega_t), \forall t \in \text{dom}(\omega).$$

From this state trajectory, an output trajectory $OTRAJ_{q,\omega}$ may be constructed

$$OTRAJ_{q,\omega} : \text{dom}(\omega) \rightarrow Y,$$

with

$$OTRAJ_{q,\omega}(t) = \lambda(STRAJ_{q,\omega}(t), \omega(t)), \forall t \in \text{dom}(\omega).$$
Thus, for every $q$ (initial state), it is possible to construct

$$T_q : \Omega \rightarrow (Y, T),$$

where

$$T_q(\omega) = OTRA_{J_q,\omega}, \forall \omega \in \Omega.$$  

The I/O Function Observation associated with $SYS$ is then

$$IOFO = \langle T, X, \Omega, Y, \{T_q(\omega)\mid q \in Q\} \rangle.$$  

Subsequently, we may derive the I/O Relation Observation by constructing the relation $R$ as the union of all I/O functions:

$$R = \{(\omega, \rho) \mid \omega \in \Omega, \rho = OTRA_{J_q,\omega}, q \in Q\}.$$
Composition Property

\[
\begin{align*}
\delta_{(t_i \rightarrow t_x)} & \quad \delta_{(t_x \rightarrow t_f)} \\
\omega_{[t_i, t_x]} & \quad \omega_{[t_x, t_f]} \\
\end{align*}
\]
Simulator: step through time
Formalism classification
based on general system model

<table>
<thead>
<tr>
<th>T: Continuous</th>
<th>T: Discrete</th>
<th>T: {NOW}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q: Continuous</td>
<td>ODE, DEVS</td>
<td>Difference Eqns. (DTSS)</td>
</tr>
<tr>
<td>Q: Discrete</td>
<td>Discrete-event</td>
<td>Finite State Automata</td>
</tr>
</tbody>
</table>

Basis for **general, standard software architecture of simulators**

Further classifications based on **structure of formalisms**
(in particular of $\delta$)
Rule-based specification of \( \delta \)

**Rule 1 (priority 3)**

Locate Initial Current State

Rule 2 (priority 1)

State Transition

Rule 3 (priority 2)

Local State Transition

---

Hans Vangheluwe

Hans.Vangheluwe@uantwerpen.be

Modelling and Simulation Foundations

27
System under study: $T, h$ controlled liquid
Detailed (continuous) view, ALG + ODE

Inputs (discontinuous $\rightarrow$ hybrid model):
- Emptying, filling flow rate $\phi$
- Rate of adding/removing heat $W$

Parameters:
- Temperature of influent $T_{in}$
- Cross-section surface of vessel $A$
- Specific heat of liquid $c$
- Density of liquid $\rho$

State variables:
- Temperature $T$
- Level of liquid $l$

Outputs (sensors):
- $is_{low}$, $is_{high}$, $is_{cold}$, $is_{hot}$

\[
\begin{align*}
\frac{dT}{dt} &= \frac{1}{l} \left[ \frac{W}{c\rho A} - \phi(T - T_{in}) \right] \\
\frac{dl}{dt} &= \phi \\
is_{low} &= (l < l_{low}) \\
is_{high} &= (l > l_{high}) \\
is_{cold} &= (T < T_{cold}) \\
is_{hot} &= (T > T_{hot})
\end{align*}
\]
\[ SYS^{ODE}_{VESSEL} = \langle \mathcal{T}, X, \Omega, Q, \delta, Y, \lambda \rangle \]

\[ \mathcal{T} = \mathbb{R} \]
\[ X = \mathbb{R} \times \mathbb{R} = \{ (W, \phi) \} \]
\[ \omega : \mathcal{T} \rightarrow X \]
\[ Q = \mathbb{R}^+ \times \mathbb{R}^+ = \{ (T, l) \} \]
\[ \delta : \Omega \times Q \rightarrow Q \]
\[ \delta(\omega_{[t_i,t_f]}, (T(t_i), l(t_i))) = \]
\[ \left( T(t_i) + \int_{t_i}^{t_f} \frac{1}{l(\alpha)} \left[ \frac{W(\alpha)}{c \rho A} - \phi(\alpha)T(\alpha) \right] d\alpha, \right. \]
\[ \left. l(t_i) + \int_{t_i}^{t_f} \phi(\alpha) d\alpha \right) \]

\[ Y = \mathbb{B} \times \mathbb{B} \times \mathbb{B} \times \mathbb{B} = \{ (\text{is low}, \text{is high}, \text{is cold}, \text{is hot}) \} \]
\[ \lambda : Q \rightarrow Y \]
\[ \lambda(T, l) = ((l < l_{low}), (l > l_{high}), (T < T_{cold}), (T > T_{hot})) \]
High-abstraction-level (discrete) view: FSA

at this level: verification of properties possible
don’t build simulator (Operational Semantics) but Transform (Transformational Semantics)
Non-determinism: Traffic network Petri Net
All traces → Reachability Graph
Probabilistic → Monte-Carlo Simulation

www.engr.utexas.edu/trafficSims/
Causality: Modelica vs. Matlab/Simulink
Multicomponent Specification

- Collections of *interacting* components
- *Compositional* modelling
  - *Modular* (interaction through ports only).
    Encapsulated. Allows for *hierarchical* (de-)composition.
  - *non-modular* (direct interaction between components).
    Not encapsulated. “global” variable access. Direct interaction through transition function
Causal Block Diagram
solution:

- co-simulation
- formalism transformation (using graph transformation)
Transform to common Formalism

- DEVS
- Process Interaction
- Discrete Event
- State trajectory data (observation frame)
- Petri Nets
- Statecharts
- scheduling-hybrid-DAE
- DEVS&DESS
- Activity Scanning
- Bond Graph a-causal
- Bond Graph causal
- DAE non-causal set
- DAE causal set
- DAE causal sequence (sorted)
- Transfer Function
- System Dynamics
- Causal Block Diagram
- KTG
- Bond Graph causal
- Bond Graph a-causal
- Cellular Automata
- Petri Nets
- Timed Automata
- Event Scheduling
- 3 Phase Approach
- Discrete Event
- Process Interaction
- Discrete Event
- Difference Equations
- state trajectory data (observation frame)
Hybrid Simulation
Simulation Trace
A Zoo of Formalisms

Hierarchy of System Specifications

Modelling Language Engineering

is theory behind

used to implement

Formalisms

Causal Block Diagrams

- time-less CBDs
- discrete-time CBDs
- continuous-time CBDs

Continuous-Time

- CSSLs
- Population Dynamics
- Forrester System Dynamics
- Modelica (multi-physics)

Discrete Event

- Finite State Automata (FSA)
- Event Scheduling
- Petri Nets
- Activity Scanning
- Statecharts
- Process Interaction
- GPSS
- DEVS

Hybrid

Animation

Tackling Complexity: challenges

visualized using

Hans Vangheluwe
Hans.Vangheluwe@uantwerpen.be
Modelling and Simulation Foundations 43