A Hierarchy of System Specification

- Basis of System Specification
  1. Set Theory
  2. Time Base
  3. 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

5. Multicomponent Specifications
   - Modular
   - Non-modular

- Non-causal models

System Specification

- Start from *observations* of structure and behaviour
- Build progressively more complex/detailed models
- Use models to *answer questions* about structure and behaviour
- OO terminology: *model* composed of
  - objects, with attributes
    - indicative or relational
    - have type (set of possible values)
  - relationships between objects

Set Theory for Abstraction

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

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

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

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

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

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

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

Relationships over Sets

1. Nominal Scale
2. Ordinal Scale
3. Interval Scale
4. Ratio Scale

Nominal Scale
Symbols are used to label or classify data

A scale that assigns a *category label* to an individual. For example, eye color is a categorical scale. Establishes no explicit ordering on the category labels. Categorical scales are also called discrete or symbolic scales, or *nominal* scales when the label (e.g., “green”) is a name.

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

1. Reflexivity: $x = x \lor x \neq x$.
2. Symmetry of equivalence: $x = y \Leftrightarrow y = x$.
3. Transitivity: $x = y \land y = z \rightarrow x = z$.

Ordinal Scale

A scale in which data can be *ranked*, but in which no arithmetic transformations are meaningful. For example, wind speed $\in \{$ high, medium, low$\}$. We would not say that the difference between high and medium wind speed is equal to (or any arithmetic transformation of) the difference between a medium and low wind speed. The distances between points on an ordinal scale are not meaningful.

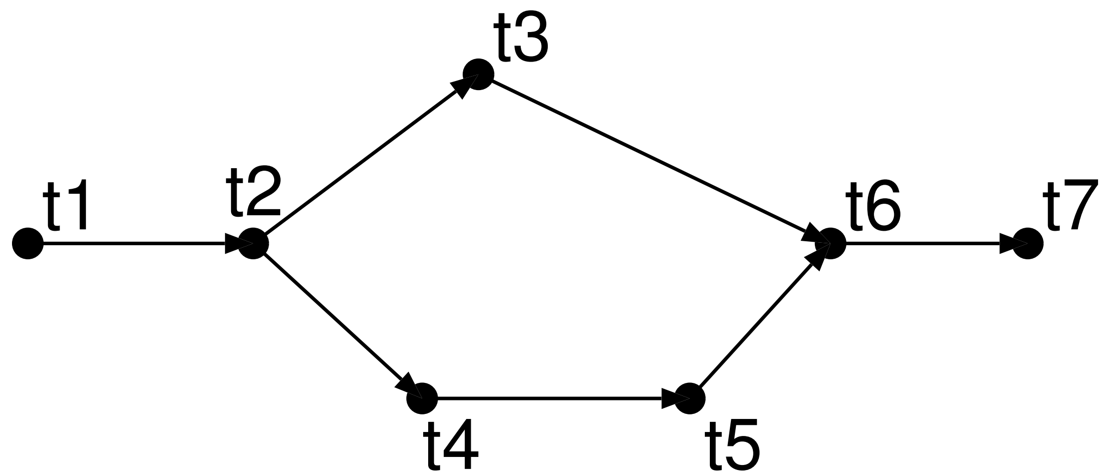
In addition to a notion of equivalence, a notion of *order* $<$ is defined with properties:

1. Symmetry of equivalence: $x = y \Leftrightarrow y = x$.
2. Asymmetry of order: $x < y \rightarrow y \not< x$.
3. Irreflexivity: $x \not< x$.
4. Transitivity: $x < y \wedge y < z \rightarrow x < z$.

Partial ordering

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

Used to model uncertainty, multiplicity, concurrency, ...



The ordering may be *total* (all data items can be compared).

$$\forall x, y \in X : x < y \vee y < x \vee x = y$$

Interval Scale

A scale where *distances* between data are meaningful. On interval measurement scales, one unit on the scale represents the *same magnitude* on 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.

An example is the Celcius scale for temperature. Equal differences on this scale represent equal differences in temperature, but a temperature of 30 degrees is not twice as warm as one of 15 degrees.

In addition to equivalence and order, a notion of *interval* is defined. The choice of a zero point is arbitrary.

Ratio Scale

A scale in which both *intervals* between values and *ratios* of values are meaningful. For example, temperature measured in degrees Kelvin is a ratio scale because we know a meaningful *zero* point (absolute zero). A temperature of 300K is twice as warm as 150K.

Compare this to interval scales in which ratios are not meaningful and ordinal scales in which intervals are not meaningful.

Time Base

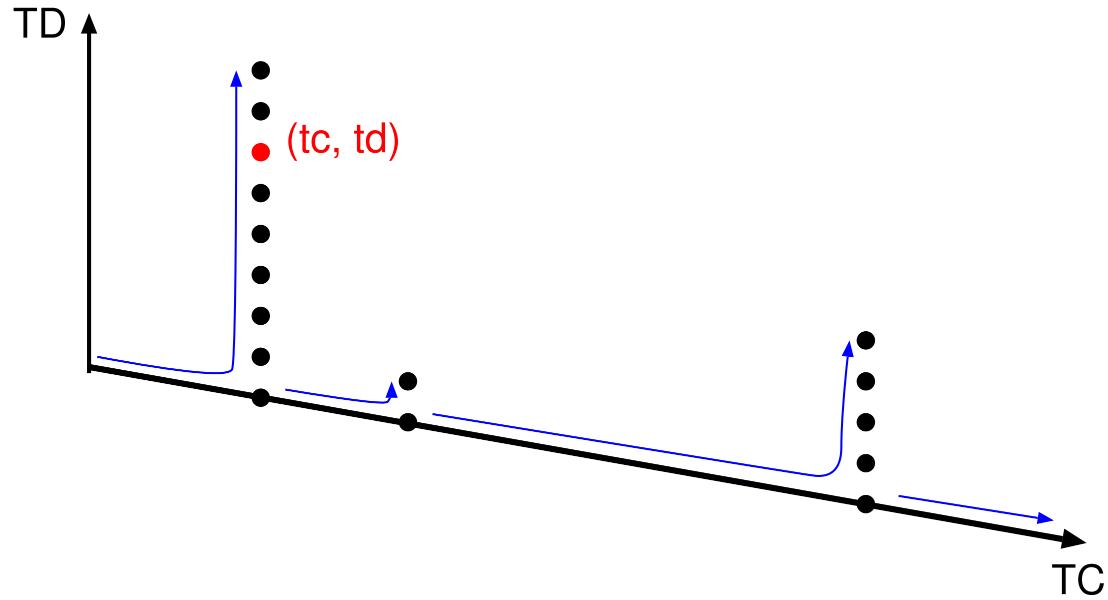
$$time = \langle T, < \rangle$$

- Dynamic system: irreversible passage of *time*.
- Set T, *ordering* relation $<$ on elements of T.
  - transitive: $A < B \wedge B < C \Rightarrow A < C$
  - irreflexive: $A \not< A$
  - antisymmetric: $A < B \Rightarrow B \not< A$
- Ordering:
  - *Total* (linear) ordering
    $\forall t, t' \in T : t < t' \vee t' < t \vee t = t'$
  - *Partial* ordering: uncertainty, multiplicity, concurrency, ...

Past, Future, Intervals

- Past: $T_{t[} = \{\tau | \tau \in T, \tau < t\}$
- Future: $T_{]t} = \{\tau | \tau \in T, t < \tau\}$
- $\langle t$ means $]t$ or $[t$
- Interval $T_{\langle t_b, t_e \rangle}$
- Abelian group $(T, +)$ with zero 0 and inverse $-t$
- Order preserving $+$: $t_1 < t_2 \Rightarrow t_1 + t < t_2 + t$
- Lower bound, upper bound
- Time bases: $\{NOW\}$, $\mathbb{R}$: *continuous*, $\mathbb{N}$ or isomorphic: *discrete*, partial ordering.

Time Bases for hybrid system models



Behaviour over Time: Segments and Trajectories

- With time base, describe *behaviour over time*
- Time function, *trajectory*, signal: $f : T \rightarrow A$
- Restriction to $T' \subseteq T$
  $f|T' : T' \rightarrow A, \forall t \in T' : f|T'(t) = f(t)$
  - Past of f: $f|T_{t\rangle}$
  - Future of f: $f|T_{\langle t}$
- Restriction to an interval: *segment* $\equiv$ **behaviour**
  $\omega : \langle t_1, t_2 \rangle \rightarrow A$
- $\Omega = (A, T)$ set of all segments

Segments

- Length $l : \Omega \rightarrow T_0^+$
- Contiguous segments if domains are contiguous $\langle t_1, t_2 \rangle, \langle t_3, t_4 \rangle, t_2 = t_3$
- Concatenation of contiguous segments: $\omega_1 \bullet \omega_2$

  $\omega_1 \bullet \omega_2(t) = \omega_1(t), \forall t \in dom(\omega_1)$

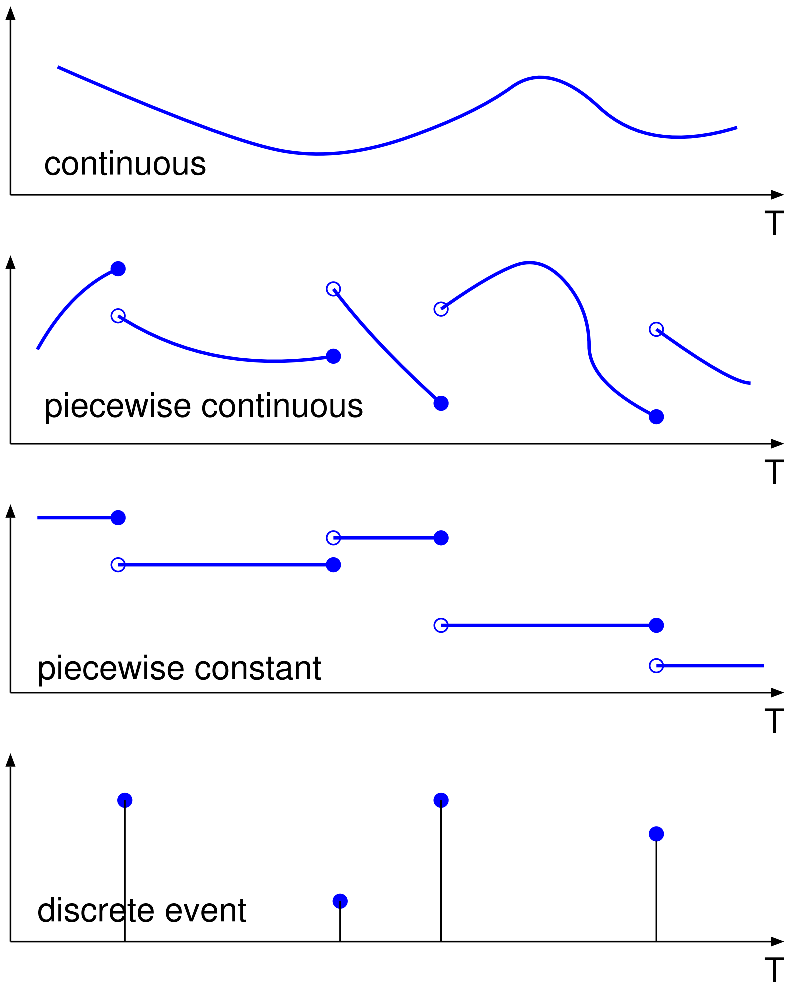
  $\omega_1 \bullet \omega_2(t) = \omega_2(t), \forall t \in dom(\omega_2)$
- Must remain *function*: unique values !
- Ω closed under concatenation
- Left and right segments:

  $\omega_{t\rangle} \bullet \omega_{\langle t} = \omega$

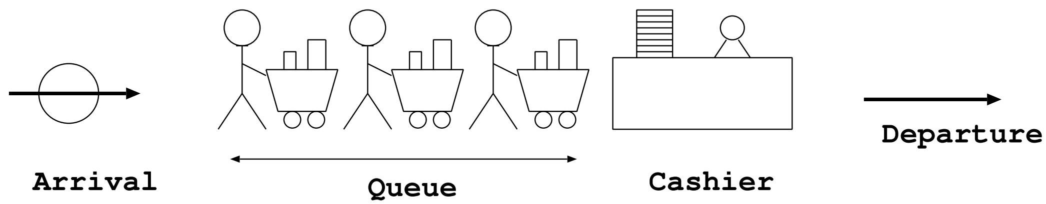
Types of Segments

- Continuous: $\omega : \langle t_1, t_2 \rangle \to \mathbb{R}^n$
- Piecewise continuous
- Piecewise constant
- Event segments: $\omega : \langle t_1, t_2 \rangle \to A \cup \{\phi\}$
- Correspondence between
  piecewise constant and event segments (later, state trajectory)

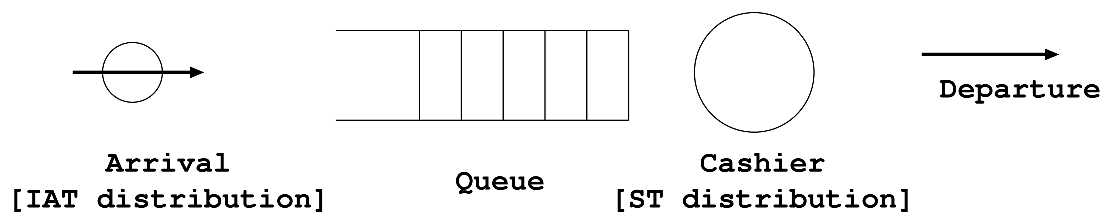
Types of Segments



Cashier-Queue System

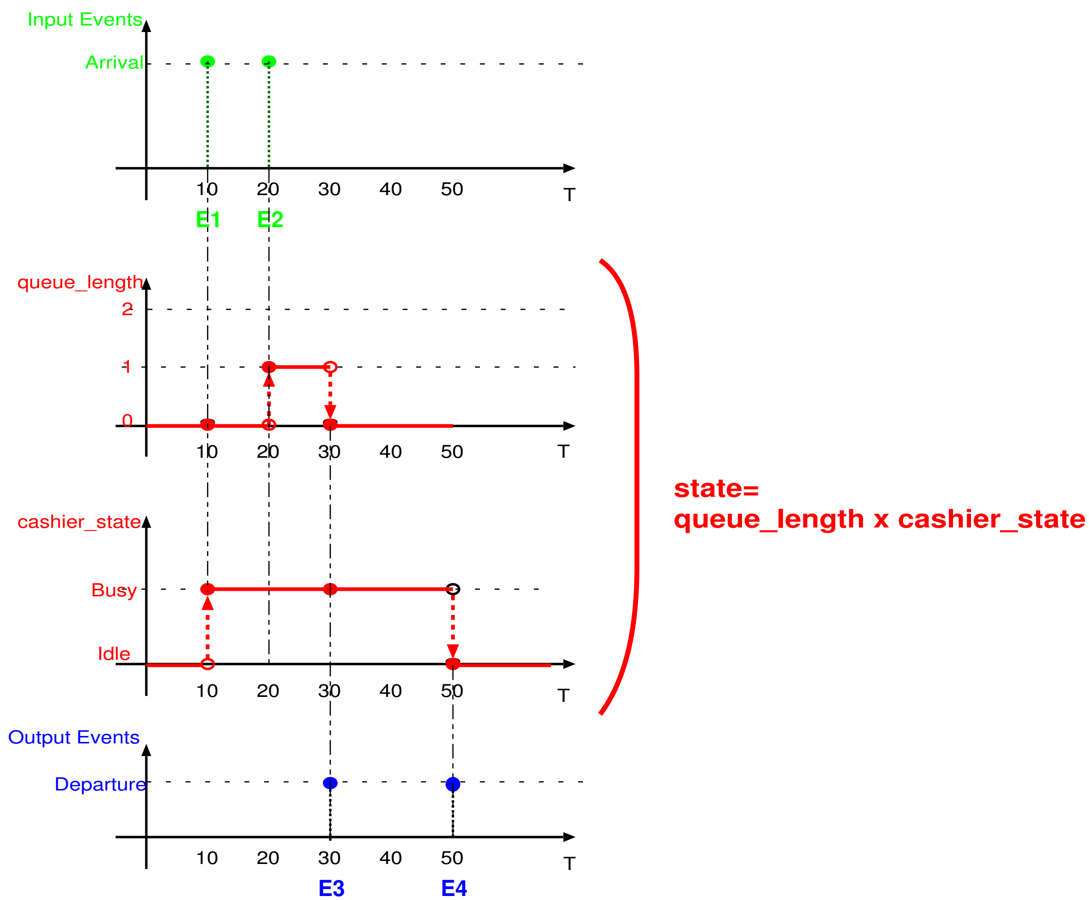


Physical View



Abstract View

Trajectories



I/O Observation Frame

$$O = \langle T, X, Y \rangle$$

- 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
- Ω is the set of all possible input segments
- R is the *I/O relation*
  $\Omega \subseteq (X, T)$, $R \subseteq \Omega \times (Y, T)$
  $(\omega, \rho) \in R \Rightarrow dom(\omega) = dom(\rho)$
- $\omega : \langle t_i, t_f \rangle \to X$: 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
- Ω 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 $dom(f(\omega)) = dom(\omega)$
- *f = initial state*: **unique** response to ω
- $R = \bigcup_{f \in F} f$

I/O System

- From *Descriptive Variables* to *State*.
- *State* summarizes the past of the system.
- Future is uniquely determined by
  - current state
  - future input

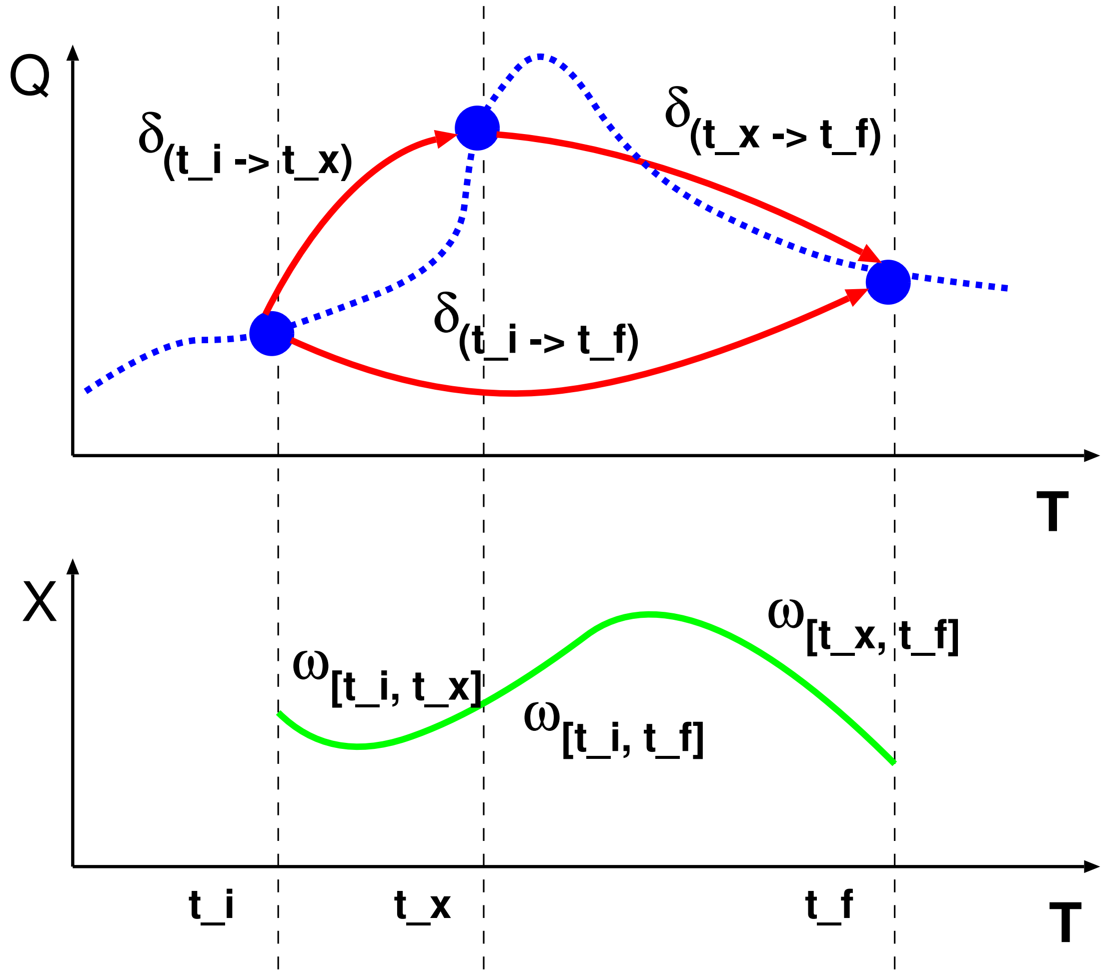
$$SYS = \langle T, X, \Omega, Q, \delta, Y, \lambda \rangle$$

T	time base
X	input set
$\omega : T \rightarrow X$	input segment
Q	state set
$\delta : \Omega \times Q \rightarrow Q$	transition function
Y	output set
$\lambda : Q \rightarrow Y$ (or $Q \times X \rightarrow Y$)	output function

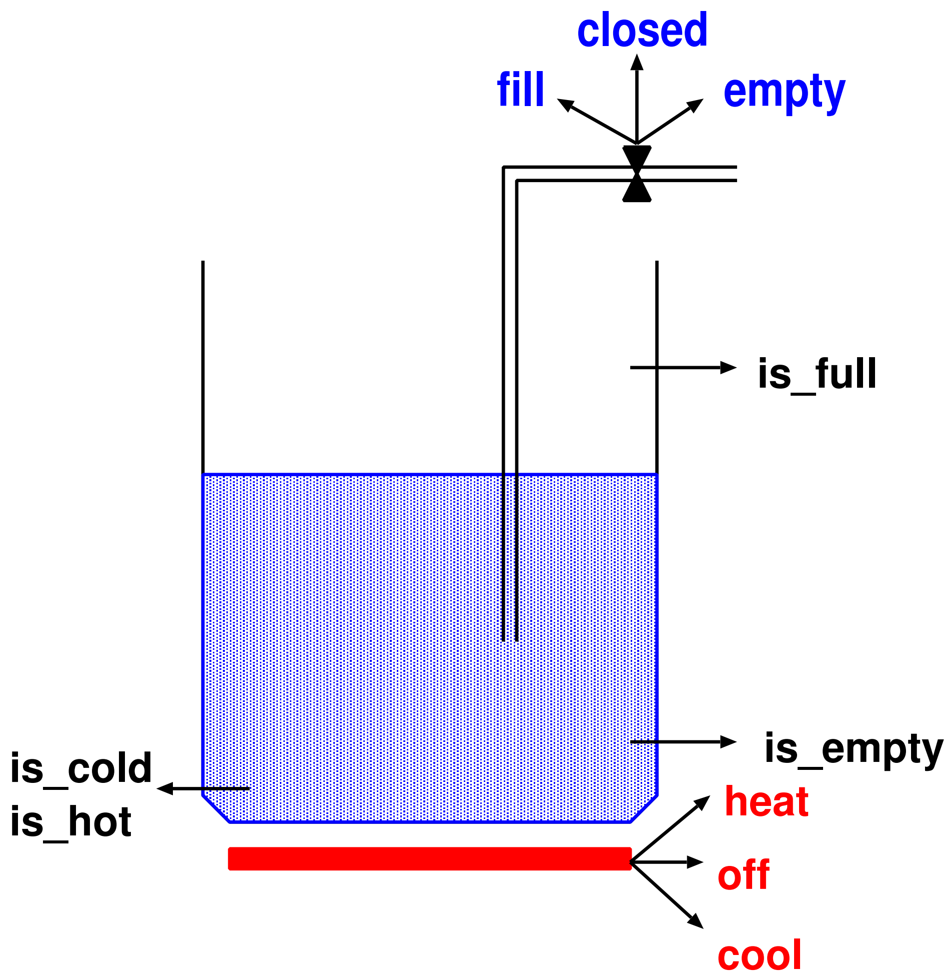
$$\forall t_x \in [t_i, t_f] : \delta(\omega_{[t_i,t_f]}, q_i) = \delta(\omega_{[t_x,t_f]}, \delta(\omega_{[t_i,t_x]}, q_i))$$

Closure requirement: Ω closed under concatenation and left segmentation.

Composition Property



System under study: T, h controlled liquid



Detailed (continuous) view, ALG + ODE formalism

Inputs (discontinuous $\rightarrow$ hybrid model):

- Emptying, filling flow rate ϕ
- 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 ρ

State variables:

- Temperature T
- Level of liquid l

Outputs (sensors):

- $is\_low, is\_high, is\_cold, is\_hot$

$$\begin{cases} \frac{dT}{dt} = \frac{1}{l}\left[\frac{W}{c\rho A} - \phi(T - T_{in})\right] \\ \frac{dl}{dt} = \phi \\ is\_low = (l < l_{low}) \\ is\_high = (l > l_{high}) \\ is\_cold = (T < T_{cold}) \\ is\_hot = (T > T_{hot}) \end{cases}$$

$$SYS_{VESSEL}^{ODE} = \langle \mathcal{T}, X, \Omega, Q, \delta, Y, \lambda \rangle$$

$$\mathcal{T} = \mathbb{R}$$

$$X = \mathbb{R} \times \mathbb{R} = \{(W, \phi)\}$$

$$\omega : \mathcal{T} \rightarrow X$$

$$Q = \mathbb{R}^+ \times \mathbb{R}^+ = \{(T, l)\}$$

$$\delta : \Omega \times Q \rightarrow Q$$

$$\delta(\omega_{[t_i,t_f]}, (T(t_i), l(t_i))) =$$

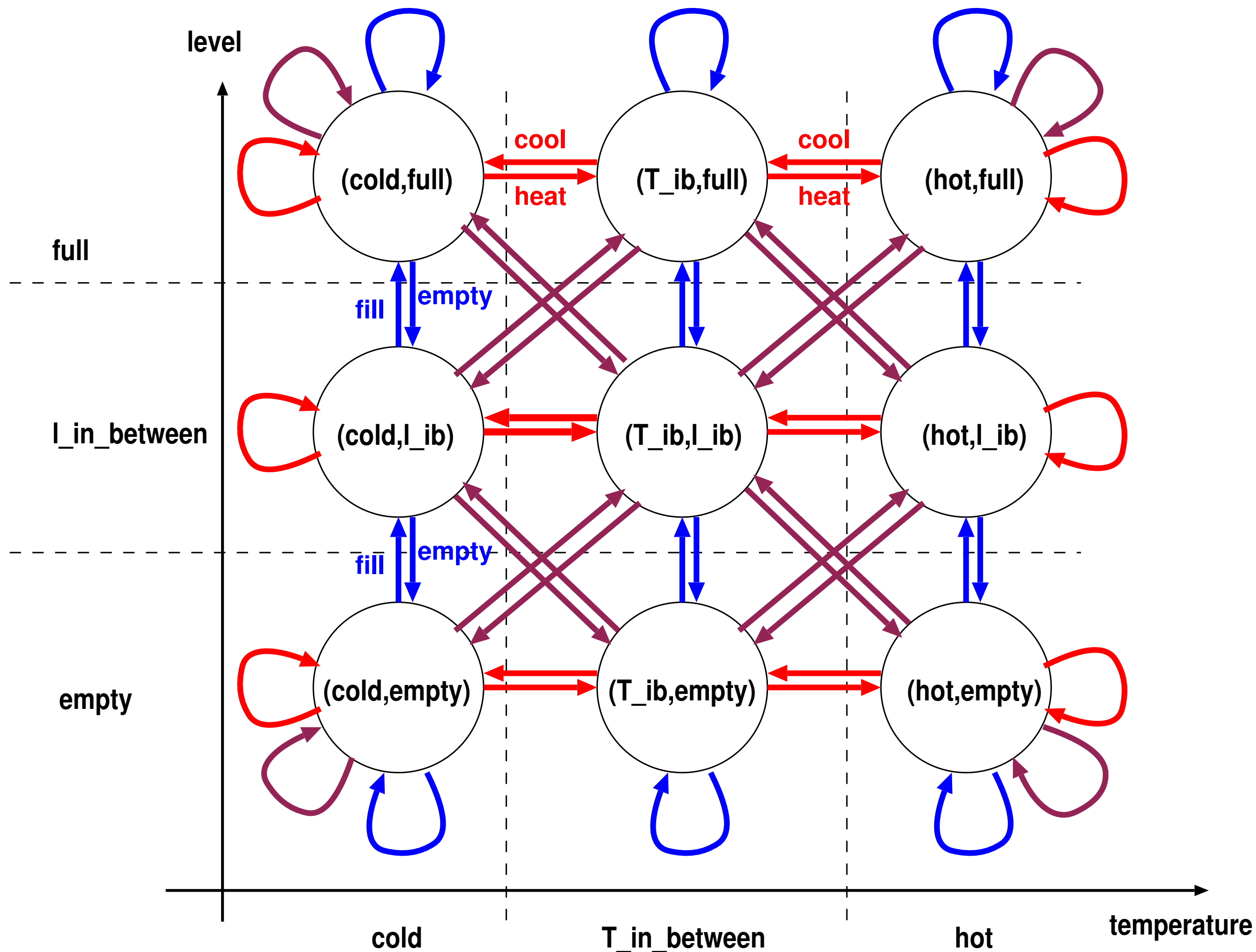
$$(T(t_i) + \int_{t_i}^{t_f} \frac{1}{l(\alpha)} [\frac{W(\alpha)}{c\rho A} - \phi(\alpha) T(\alpha)] d\alpha,\ l(t_i) + \int_{t_i}^{t_f} \phi(\alpha) d\alpha)$$

$$Y = \mathbb{B} \times \mathbb{B} \times \mathbb{B} \times \mathbb{B} = \{(is\_low, is\_high, is\_cold, is\_hot)\}$$

$$\lambda : Q \rightarrow Y$$

$$\lambda(T, l) = ((l < l_{low}), (l > l_{high}), (T < T_{cold}), (T > T_{hot}))$$

High-level (discrete) view, FSA formalism



$$SYS^{FSA}_{VESSEL} = \langle \mathcal{T}, X, \Omega, Q, \delta, Y, \lambda \rangle$$

$$\mathcal{T} = \mathbb{N}$$

$$X = \{heat, cool, off\} \times \{fill, empty, closed\}$$

$$\omega : \mathcal{T} \rightarrow X$$

$$Q = \{cold, T_{between}, hot\} \times \{empty, l_{between}, full\}$$

$$\delta : \Omega \times Q \rightarrow Q$$

$$\delta((off, fill)_{[n,n+1[}, (cold, empty)) = (cold, l_{between})$$

$$\delta((off, fill)_{[n,n+1[}, (cold, l_{between})) = (cold, full))$$

$$\delta((off, fill)_{[n,n+1[}, (cold, full)) = (cold, full))$$

$$\vdots$$

$$\delta((heat, fill)_{[n,n+1[}, (hot, full)) = (hot, full))$$

$$Y = \mathbb{B} \times \mathbb{B} \times \mathbb{B} \times \mathbb{B}$$

$$\lambda : Q \rightarrow Y$$

$$\lambda(T, l) = ((l == low), (l == high), (T == cold), (T == hot))$$

From I/O System specification to I/O Function observation

Given: initial state q and a given input segment ω.

State Trajectory $STRAJ_{q,\omega}$ from SYS

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

with

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

From this state trajectory, construct an *output trajectory* $OTRAJ_{q,\omega}$

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

with

$$OTRAJ_{q,\omega}(t) = \lambda(STRAJ_{q,\omega}(t), \omega(t)), \forall t \in dom(\omega).$$

Thus, for every q (initial state), it is possible to construct

$$\mathcal{T}_q : \Omega \rightarrow (Y, T),$$

where

$$\mathcal{T}_q(\omega) = OTRAJ_{q,\omega}, \forall \omega \in \Omega.$$

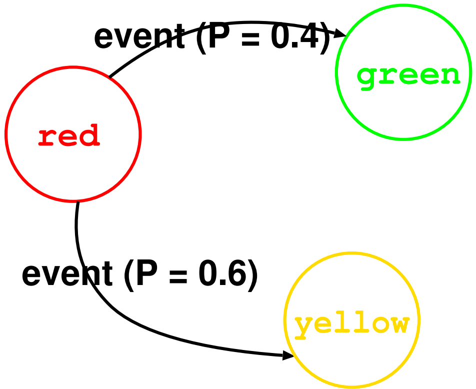
The I/O Function Observation associated with SYS is then

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

I/O Relation Observation relation R constructed as the union of all I/O functions:

$$R = \{(\omega, \rho) | \omega \in \Omega, \rho = OTRAJ_{q,\omega}, q \in Q\}.$$

In SYS: δ is deterministic, but . . .



1. Transform non-deterministic into deterministic model (*e.g.,* NFA to DFA).
2. Monte Carlo simulation: sample from probability distribution; perform multiple deterministic runs and thus obtain an estimate for performance variables.

Discrete-event models ($T = \mathbb{R}$, finite non-ϕ)

- Specification and analysis of behaviour
  - physical systems (time-scale, parameter abstraction)
    queueing systems
  - non-physical systems (software)
- Traditionally: World Views
  1. Event Scheduling
  2. Activity Scanning
  3. Three Phase Approach
  4. Process Interaction
- Emulate non-determinism by deterministic + pseudo RNG

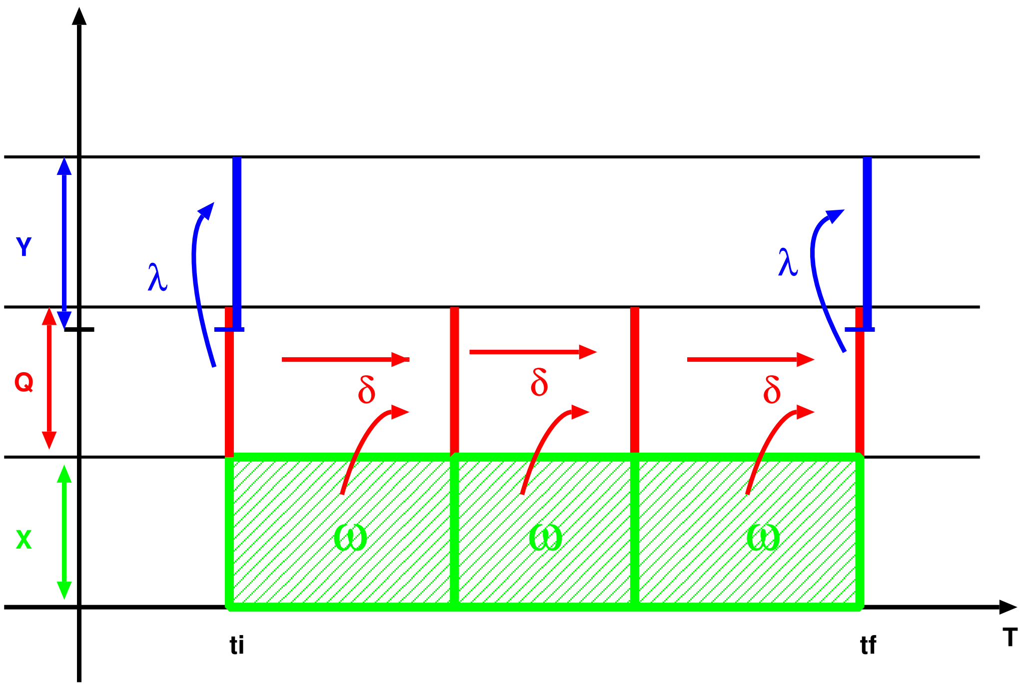
Formalism classification based on general system model

	T: **Continuous**	T: **Discrete**	T: $\{$**NOW**$\}$
Q: **Continuous**	ODE	Difference Eqns.	Algebraic Eqns.
Q: **Discrete**	Discrete-event	Finite State Automata	Integer Eqns.
	Naive Physics	Petri Nets	

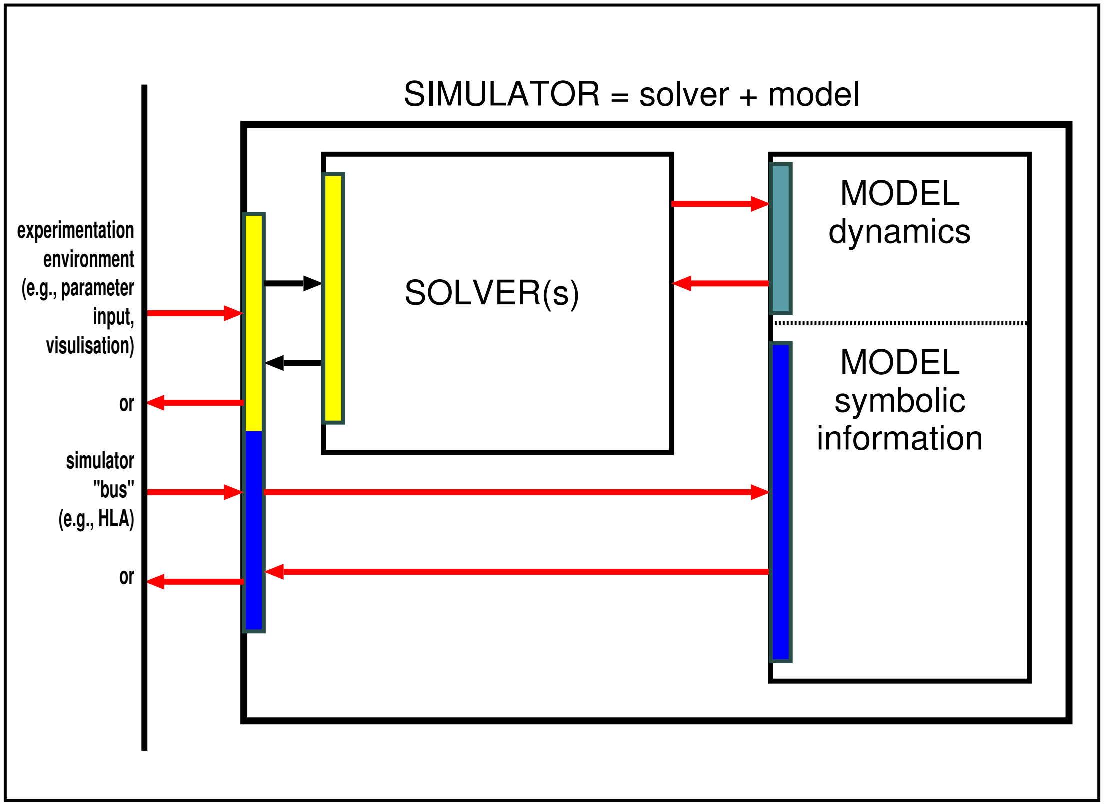
Basis for **general, standard software architecture of simulators**

Other classifications based on **structure of formalisms**

Simulation Kernel Operation: iterative specification



Model-Solver Architecture



Difference Equations (solving may be symbolic)

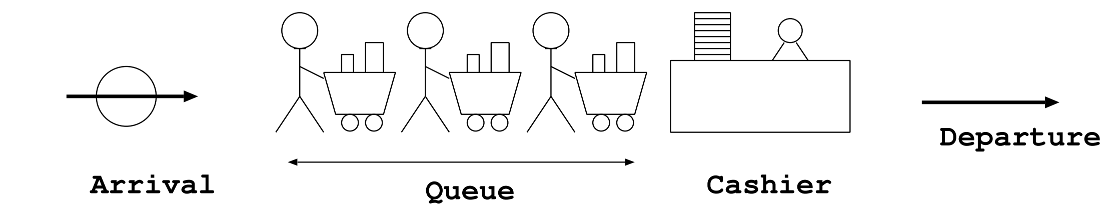
$$\begin{cases} x_1 & = & 1 \\ x_{i+1} & = & ax_i + 1 \end{cases}$$

$$\begin{aligned} x_n &= 1 + a + a^2 + \ldots + a^{n-1} \\ ax_n &= a + a^2 + \ldots + a^{n-1} + a^n \end{aligned}$$

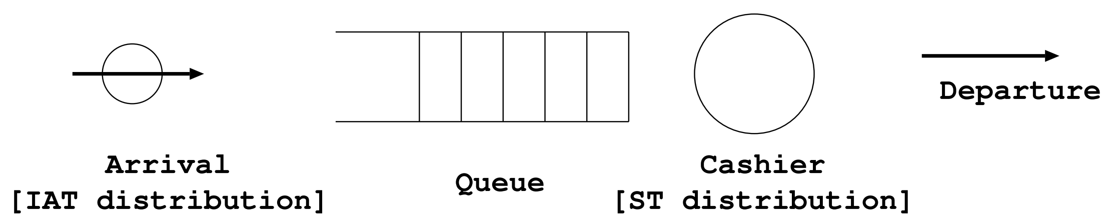
$$\Longrightarrow x_n(1-a) = 1 - a^n$$

$$\Longrightarrow x_n = \frac{1-a^n}{1-a}$$

State set can be product set



Physical View



Abstract View

Adding Structure

- no *structure* is imposed on sets upto now
- additional information: construct sets from primitives
- cross-product $\times$
- building *concrete* systems from building blocks
- system $\rightarrow$ structured system
- structured sets and functions $\sim$ *variables, ports*

Multivariable Sets

Variables, coordinates, ports v_i

$$V = (v_1, v_2, \ldots, v_n)$$

$$S_1, S_2, \ldots S_n$$

$$S = (V, S_1 \times S_2 \times \ldots S_n)$$

Projection operator

$$. : S \times V \rightarrow \bigcup_{j=1}^{n} S_j, S.v_i = s_i$$

$$. : S \times 2^V \rightarrow \bigcup_{v \in 2^V} \times_{j \in V} S_j, S.(v_i, v_j, \ldots) = s.v_i, s.v_j, \ldots$$

Examples

Ports

$$X_1 = ((heatFlow, liquidFlow), \mathbb{R} \times \mathbb{R})$$

$$x \in X_1, x.heatFlow$$

Variables

$$S_1 = ((temperature, level),]0.0, 100.0[\times [0, H])$$

$$S_2 = ((qLength, cashStatus), \mathbb{N} \times \{Idle, Busy\})$$

$$s \in S_2, s.qLength$$

Structured Functions

$$f : A \to B$$

with A and B structured sets

Projection

$$f.b_i : A \to ((b_i), B_i),$$

$$f.b_i(a) = f(a).b_i$$

$$f.(b_i, b_j, \ldots) : A \to ((b_i, b_j, \ldots), B_i \times B_j \times \ldots)$$

$$f.(b_i, b_j, \ldots)(a) = f(a).(b_i, b_j, \ldots)$$

Adding Structure to

- IORO
- IOFO
- IOSYS

$$IOSYS = < T, X, \Omega, Q, \delta, Y, \lambda >$$

$$X, Q, \delta, Y, \lambda$$

are structured sets/functions

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

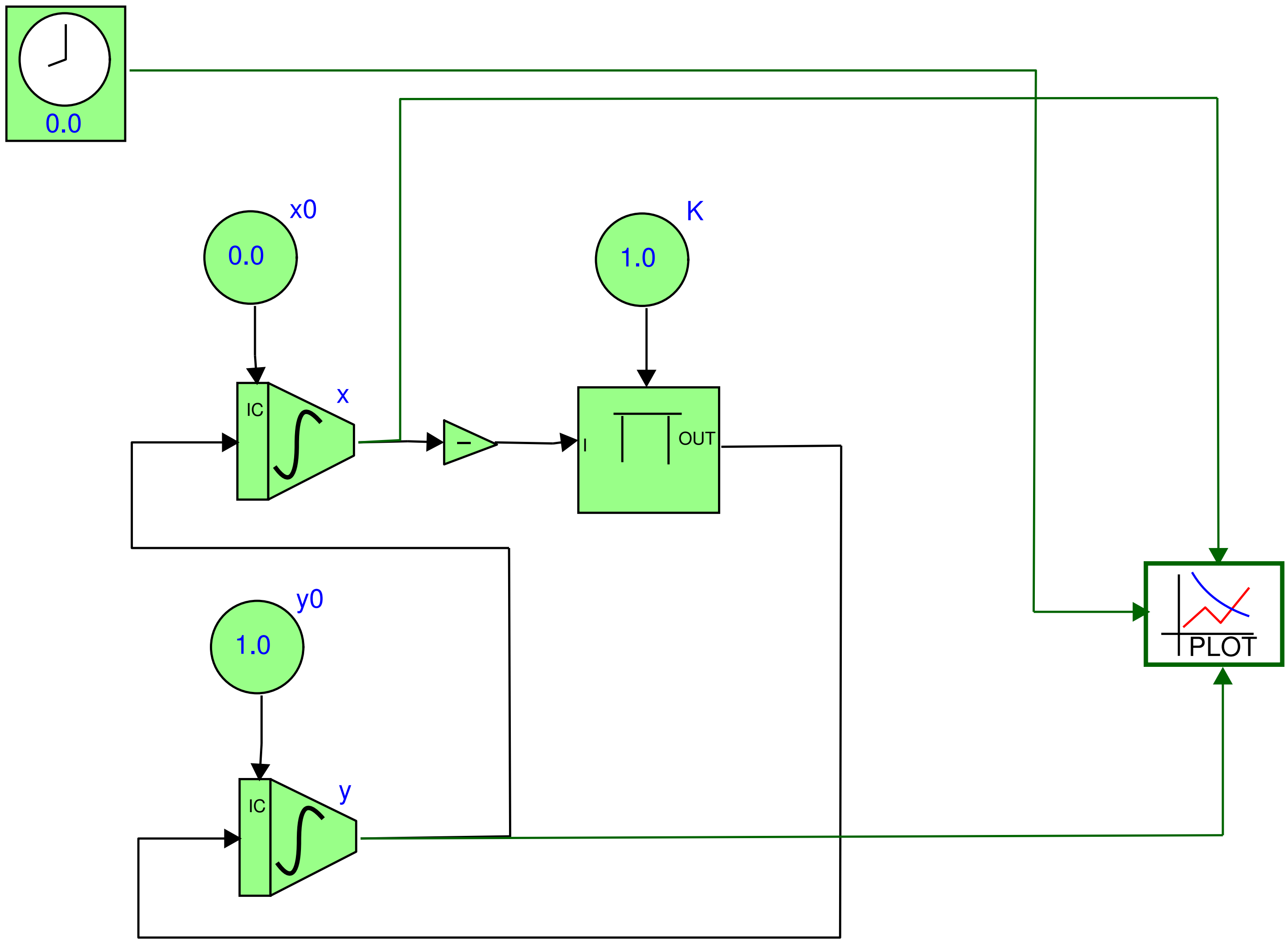
Nonmodular Multicomponent Specification

$$MC = \langle T, X, \Omega, Y, D, \{M_d | d \in D\} \rangle$$

$$M_d = \langle Q_d, E_d, I_d, \delta_d, \lambda_d \rangle, \forall d \in D$$

- D is a set of component *references/names*
- Q_d is the *state set* of component d
- $I_d \subseteq D$ is the set of *influencers* of d
- $E_d \subseteq D$ is the set of *influencees* of d
- δ_d is the *state transition function* of d
  $\delta_d : \times_{i \in I_d} Q_i \times \Omega \rightarrow \times_{j \in E_d} Q_j$
- λ_d is the *output function* of d
  $\lambda_d : \times_{i \in I_d} Q_i \times X \rightarrow Y$

Example: Causal Block Diagram



Time Slicing Causal Block Diagram

- $E_d = \{d\}$
- $Y = \times_{d \in D} Y_d$
- $Q = \times_{d \in D} Q_d$
- $\delta(q, \omega).d = \delta_d(\times_{i \in I_d} q_i, \omega)$
- $\lambda(q, \omega(t)).d = \lambda_d(\times_{i \in I_d} q_i, \omega(t))$
- Less constrained for Discrete Event

Modular Multicomponent (Network) Specification

$$N = \langle T, X_N, Y_N, D, \{M_d | d \in D\}, \{I_d | d \in D \cup \{N\}\}, \{Z_d | d \in D \cup \{N\}\} \rangle$$

- X_N and Y_N are external network inputs and outputs
- D is a set of component *references* or *names*
- $\forall d \in D . M_d$ is an I/O system
- $I_d \subseteq D \cup \{N\}$ is the set of *influencers* of d
- $Z_d : \times_{i \in I_d} YX_i \rightarrow XY_d$ is the *interface map* for d
  $YX_i = X_i$ if $i = N$, $YX_i = Y_i$ if $i \neq N$
  $XY_d = Y_d$ if $d = N$, $YX_d = X_d$ if $d \neq N$

Semantics: Flattening/Closure under coupling

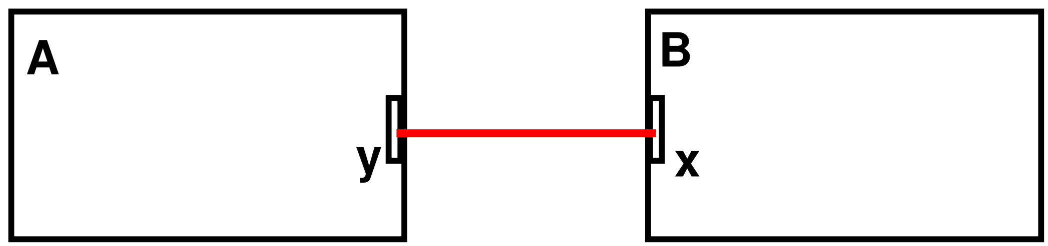
$$\langle T, X_N, Y_N, D, \{M_d | d \in D\}, \{I_d | d \in D \cup \{N\}\}, \{Z_d | d \in D \cup \{N\}\}\rangle$$

$$\rightarrow \langle T, X, \Omega, Q, \delta, Y, \lambda \rangle$$

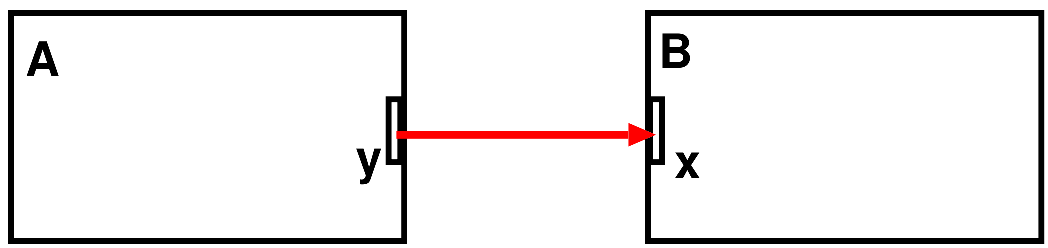
- Continuous
  - unique names (scope resolution)
  - `connect(M1.o,M2.i)` $\equiv M2.i := M1.o \rightarrow \# I_d \leq 1$
  - *closure* of the ALG+ODE formalism
  - Discrete Event (later, DEVS)
- Allows for *hierarchy*

Closure in Block Diagrams

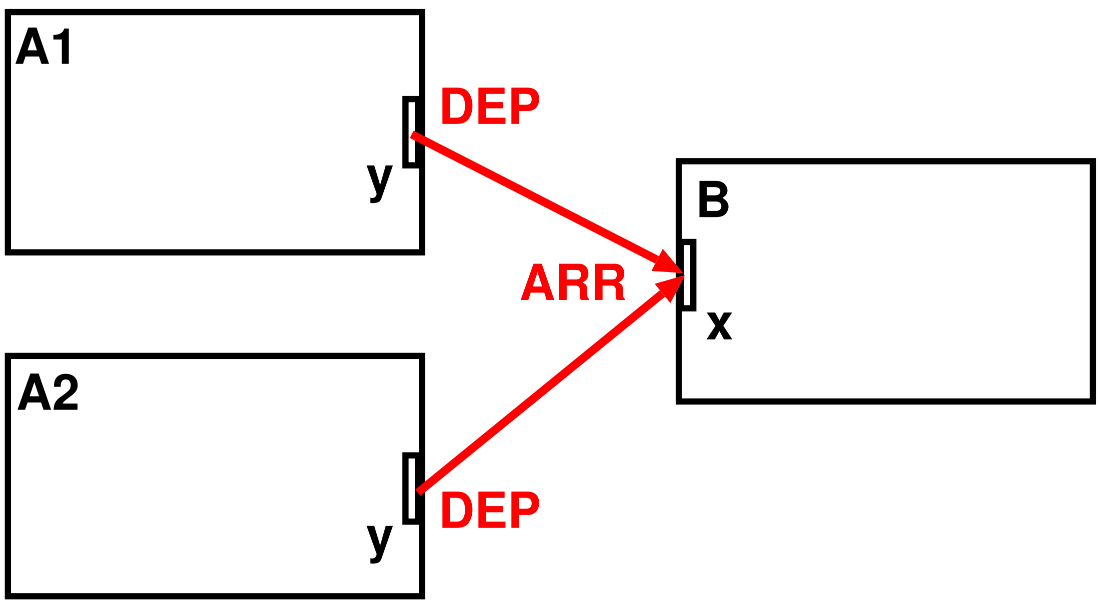
non-causal



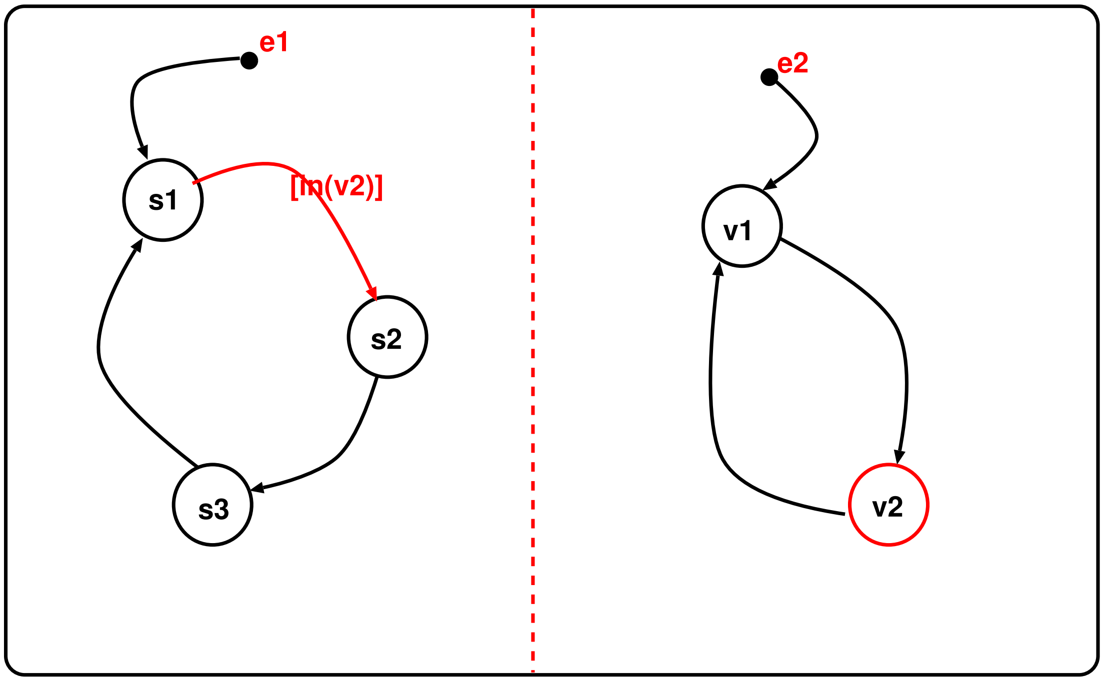
causal



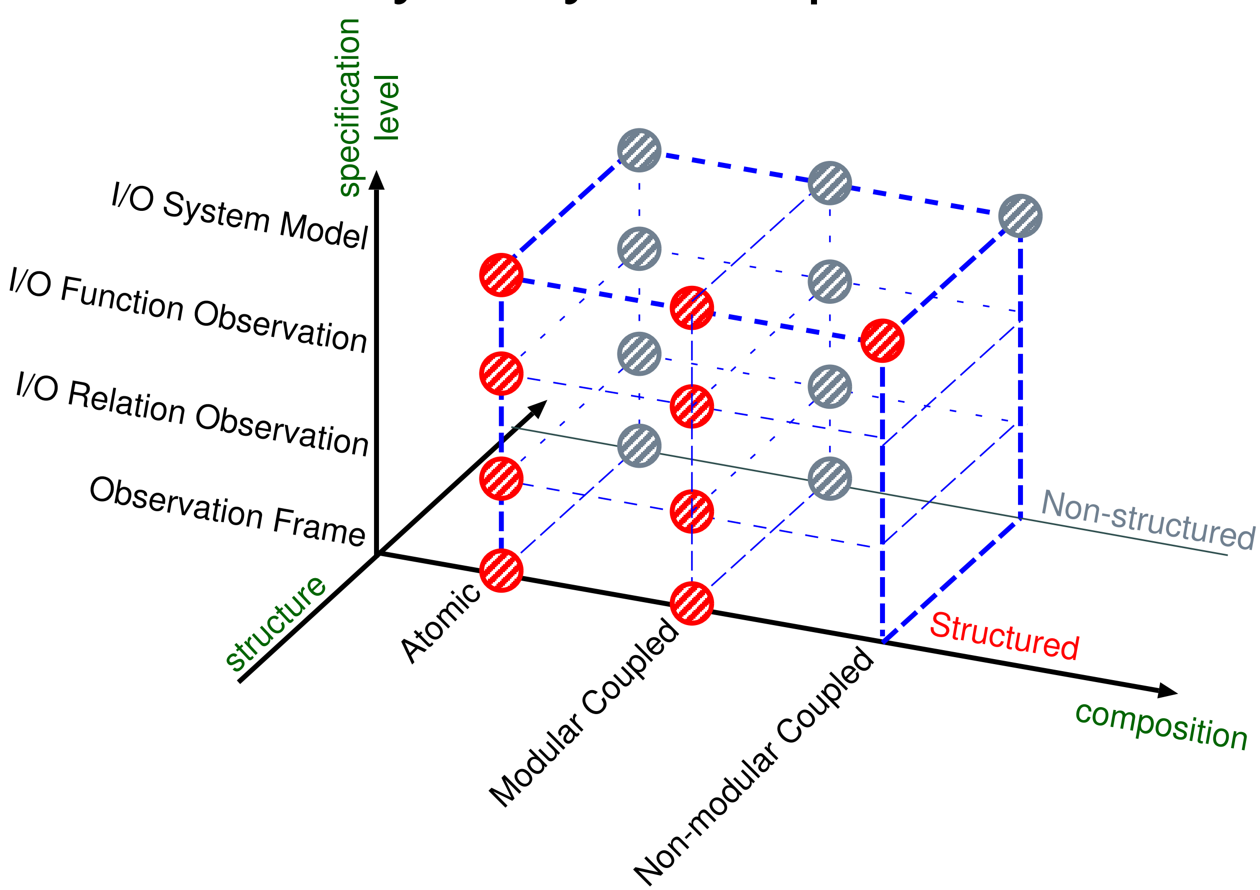
Closure in modular Discrete Event formalisms



Closure in State Charts



Hierarchy of system specification

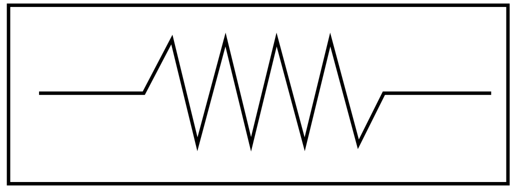


+ relationships (morphisms, transformations)

Transforming Nonmodular into Modular Specifications

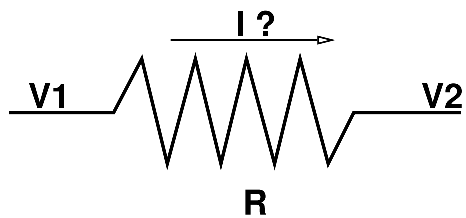
- example: shared memory to distributed memory
- direct access routed through ports
- may use local copy

Noncausal models

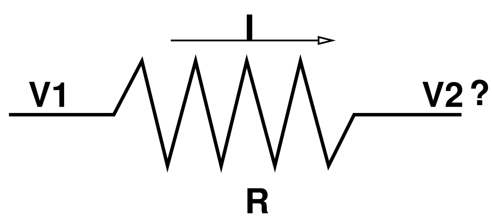


Object "resistor"

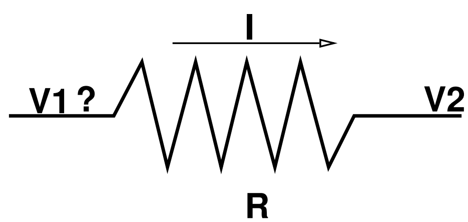
V1 − V2 = R*I



I = (V1−V2)/R



V2 = V1 − R*I



V1 = V2 + R*I

Classification: Different Model Types

- well-defined (white box) vs. ill-defined (black box)
- continuous vs. discrete (time base)
- deterministic vs. stochastic
- graphical vs. textual
- causal vs. noncausal
- ...
- which formalism ?

Arch of Karplus