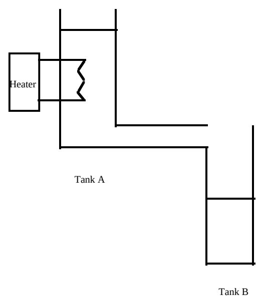
Constraint Based Ontology: QSIM

Heated Tanks



Component Representation

ADD(D1 D2 Q) ← Heater

MULT(Fab T D1)
M+(D2 T_dash)
DERIV(T T_dash)
M+(La Fab)
MINUS(Fba Fab)
DERIV(La Fba) ← Tank A

M+(Fab Lb)
DERIV(Lb Fab) ← Tank B

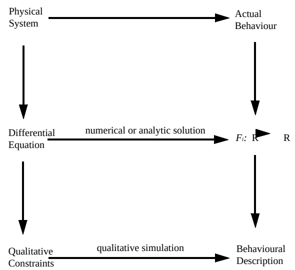
Process Representation

ADD(D1 D2 Q)
MULT(Fab T D1)
M+(D2 T_dash)
DERIV(T T_dash) ← Heating

M+(La Fab)
MINUS(Fba Fab)
DERIV(La Fba) ← Flow

M+(Fab Lb)
DERIV(Lb Fab) ← Containment

Behavioural Abstraction

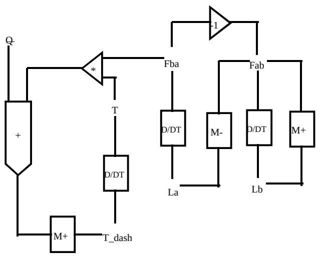


Representation Summary

- Variables represented by two element vectors: <qmag qdir>
- Quantitiy Space
  - Ordered set of points (landmarks) and intervals (always contains *inf*, *0* and *minf*) for magnitude
  - *inc*, *std* and *dec* for derivatives
- Integration phase by means of Transition Rules
  - defined by Mean value and Intermediate value theorems

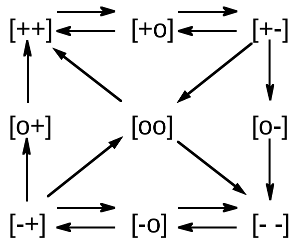
QSIM Model

M+(Fab Lb)
M+(La Fab)
M+(D2 T_dash)
ADD(D1 D2 Q)
MINUS(Fba Fab)
MULT(Fab T D1)
DERIV(La Fba)
DERIV(Lb Fab)
DERIV(T T_dash)



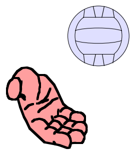
Transition Rules

- Intermediate Value Theorem (IVT)
  - Defines the direction of change of a variable between two points.
- Mean Value Theorem (MVT)
  - States that for a continuous system, a function joining two points of opposite sign must pass through zero.



Example: Thrown Ball

- Presented in the literature
- Simplest meanful example
- Limited operating region
  - bounded by max height and ground (assuming no cliff)
  - transition to spring model when in contact with ground.



DERIV (Y, V)
DERIV(V, A)
A(t) = g < 0

Example Start

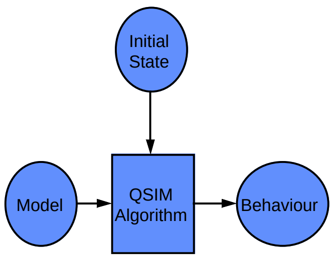
- Initial State
  - An assignment of values <qmag qdir> to all variables of the system at the initial time point, t0: QS(F, t0)

QS(A, t0) = <(g, std>
QS(V, t0) = <(0, ∞), dec>
QS(Y, t0) = <(0, ∞), inc>

- Example Start
  - We join the example after the first transition, where the qualitative state is:

QS(A, t0, t1) = <g, std>
QS(V, t0, t1) = <(0, ∞), dec>
QS(Y, t0, t1) = <(0, ∞), inc>

Qualitative Simulation



QSIM Algorithm

- Select state from ACTIVE
- Apply transition rules to each **Variable**
- Apply **Constraints** to tuples of variables
  - Consistent derivative values
  - Magnitude corresponding values match
- Check variable values in **All** constraints:
  - **Pairwise Consistency** (Waltz)
- Generate all **Global** interpretations
  - Create New States
- Apply **Global Filters** => ACTIVE

Transitions of Variables

QS(A, t_0, t_1) => QS(A, t_1)
I1: <g, std> => <g, std>

QS(Y, t_0, t_1) => QS(Y, t_1)
I4: <(0, ∞), inc> => <(0, ∞), inc>
I8: <(0, ∞), inc> => <L*, std>

QS(V, t_0, t_1) => QS(V, t_1)
I5: <(0, ∞), dec> => <0, std>
I6: <(0, ∞), dec> => <0, dec>
I7: <(0, ∞), dec> => <(0, ∞), dec>
I9: <(0, ∞), dec> => <L*, std>

Global Filters and Heuristics

- Filters
  - No Change
    - Predicted state is the same as the previous state
  - Cycle
    - Predicted state is identical with one of its predecessors
  - Divergence
    - If *inf* or *minf* is predicted - endpoint of simulation reached.
- Heuristics
  - Quiescence
    - All variables have zero derivative - steady state.
  - No Divergence
    - Remove states which diverge to *inf* or *minf*

Filters and Interpretations

- Constraint and Waltz Filtering

DERIV(Y, V)		DERIV(V, A)	
(I4, I5)	c	(I5, I1)	c
(I4, I6)	c	(I6, I1)	
(I4, I7)		(I7, I1)	
(I4, I9)	w	(I9, I1)	c
(I8, I5)	w		
(I8, I6)			
(I8, I7)	c		
(I8, I9)	c		

- Global Interpretations

Y	V	A
I4	I7	I1
I8	I6	I1

State 1:
QS(Y, t1) = <(0, ∞), dec>
QS(V, t1) = <(0, ∞), dec>
QS(A, t1) = <g, std>

State 2:
QS(Y, t1) = <Y_{new}, std>
QS(V, t1) = <0, dec>
QS(A, t1) = <g, std>

The Next State

QS(Y, t_1) = <Y_{new}, std>
QS(V, t_1) = <0, dec>
QS(A, t_1) = <g, std>

New landmark discovered:
$0 < Y_{new} < \infty$

Corresponding Values

Variables: X, Y, Z

Landmarks: X = {0 X1 X2 X3 inf}
Y = {0 Y1 inf}
Z = {0 Z1 Z2 inf}

Constraint:
ADD(X, Y, Z) (0, 0, 0) (X3, Y1, Z2)

The Problem of Spurious Behaviour Generation

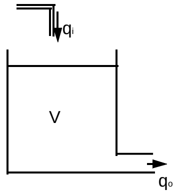
Contributions of Ontologies

- Device (Component) Centred
  - Modularization: Component library
  - Envisionment
  - Qualitative operators
- Process Centred
  - Compositional Modelling (complex ontology)
  - Explicit reasoning about state change
- Constraint Centred
  - Simple abstraction of ODE's
  - Focus on simulation
- Domains of Application
  - Diagnosis
  - Training
  - Control
  - Explanation

Soundness and Completeness

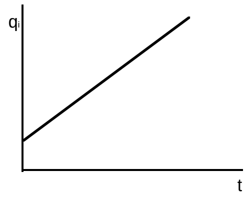
- Sound
  - Guarantees to find all possible behaviours of system
- Incomplete
  - Unfortunately also finds non-existent (spurious) behaviours
- Still useful for ascertaining that a dangerous state cannot be reached.
- Large research effort to remove spurious behaviours
  - we will skim the surfarce of the surface!

Single Tank System



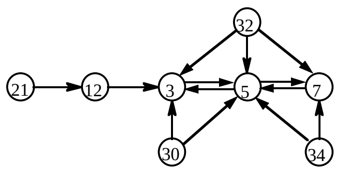
M+(V,q_o)
ADD(V_dash, q_o, q_i)
DERIV(V, V_dash)

•Input: Stepped Ramp



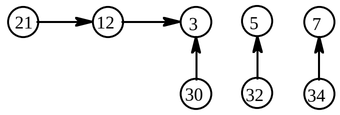
2 Element Vector Envisionment

State	Vector
21	+ - +
12	+ 0 +
3	+ + +
5	+ + 0
7	+ + -
30	0 + +
32	0 + 0
34	0 + -



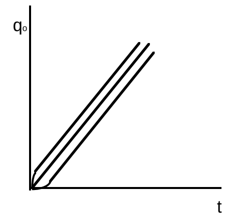
3 Element Vector Envisionment

State	Vector
21	+ - + -
12	+ 0 + -
3	+ + + -
5	+ + 0 0
7	+ + - +
30	0 + + -
32	0 + 0 0
34	0 + - +

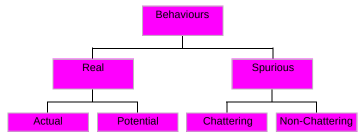


Behaviour Categorisation

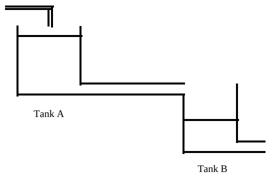
- Distinct Behaviours



- Categorisation of Behaviours

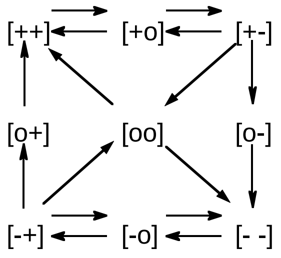


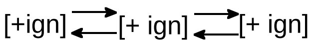
Ignore *qdirs:* Cascaded Tanks



inflowA = IF*
pressureA = M+(amountA)
pressureB = M+(amountB)
outflowA = M+(pressureA)
outflowB = M+(pressureB)
netflowA = inflowA - outflowA
netflowB = outflowA - outflowB
d/dt(amountA) = netflowA
d/dt(amountB) = netflowB

Ignore *qdirs*

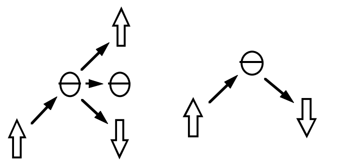




Ignore *qdirs*: Satifiability Filter

- Quiescence when magnitude reaches zero.
- Make sure there are no discontinuities:
  - Check that a consistent state exists by replacing *ign* with each of {inc, std, dec}
  - Check for valid transition

Higher-Order Derivatives (HOD's)



netflowB(t) at a critical pointnetflowB''(t) < 0
netflowB'(t) = 0

Use knowledge of system to automatically generate the second derivative of the HODs

HOD Algorithm

- Identify HODs in the system:
  - necessary and applicable where there is intractible branching from points where HOD’ = 0
- For each HOD derive and expression, valid where HOD’ = 0, for HOD’’ in terms of the other system variables
  - determine sign of HOD’’ and derive valid transition
- Smoothness Assumption
  - M+ relations linear in region of a critical point

And the rest . . .

- QSIM Development
  - Time scale abstraction
  - Semi- quanitative information
  - Energy constraints
  - Phase plane analysis
- Other Systems
  - Order of Magnitude Reasoning
  - Hyper-real simulation
  - Causal ordering
  - Fuzzy simulation
  - Constructive and Non-constructive algorithms
  - Synchronous and asynchronous simulation
- See CS46310
  - . . . .