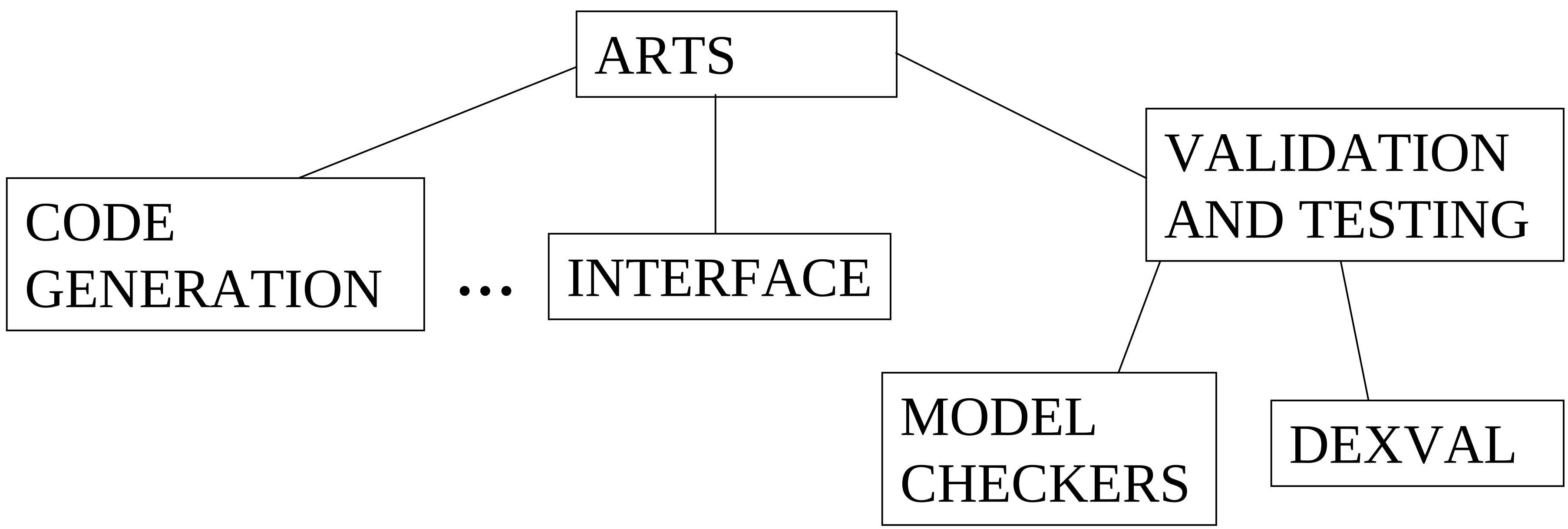
ARTS/DExVal
Derivation of Meaningful Experiments for Validation

- Prof. A. Haeberer, PUC-Rio
- Prof. M. Wirsing, LMU Munich
- Dr. A. Ciarlini, PUC-Rio
- Dr. T. Fruehwirth, LMU Munich

ARTS

- Formal basis for software development, funded partially by Siemens, Brazil



Validation and Testing

- Critical
- Expensive
- Revealing maximum number of bugs
- Meaningful experiments

Model Checking

- Verification of properties
- Modal temporal logic
- Prop. holds or there is a counterexample
- Approximation
  - Infinite state machines $\rightarrow$ Finite state machines
  - Continuous variables $\rightarrow$ Discrete variables
  - State explosion

The Goal

- Verification and derivation of properties of concurrent transition systems
- Continuous variables and non-linear expressions
- Expressiveness: variables at different times

The Approach

- Symbolical execution
- Constraint Logic Programming
- User descriptions $\rightarrow$ all paths and corresponding derived properties
- E.g. Constraints on output $\rightarrow$ constraints on input

Hybrid Automata

- Continuous activities
- Discrete transitions
- Components
  - Variables
  - States: name, invariant and iteration
  - Transitions: source and target states, guarded actions, events

Hybrid System

- Timed hybrid automata
  - Synchronization: machine clock
  - Modifications according to last state
- Coordination: sharing of variables and events
- Simultaneous modifications
- Variable modified by only one automaton

Constraint Logic Programming

- Logic programming
  - Declarative rules defining relations
  - Search for all solutions using backtracking
  - Non-deterministic
- Constraint solving
  - Efficient algorithms
  - Solving sets of distinguished relations
  - Deterministic

Constraint Logic Programming

- LP + CS:
  - Expressiveness and efficiency
  - LP sends constraints to CS
  - Constraints solved in parallel
  - Inconsistency $\rightarrow$ cut branch
  - Ex:
    - $X+Y<5$ and $Y>0$
    - $X=6 \rightarrow$ fail

DExVal Tool

- Input:
  - Automata
  - Initial and final states (not mandatory)
  - Properties:Values or ranges(input, intermediate and output)
- Output: Paths and corresponding constraints relating selected variables
- Using output for testing

OUT>100 $\rightarrow$ 10<IN<20

OUT$\leq$ 100 $\rightarrow$ (IN$\leq$10) $\vee$ (IN$\geq$20)

Better testing IN=1,10,15,20,30 than IN=12,13,14,15,16

Examples of Properties

- Since X>Y, Z=1
- For all states, X has a higher value than its value in the previous state
- If, at some time, X>Y, then at most 5 clocks later Z=1
- Obs: Existential and universal quantification

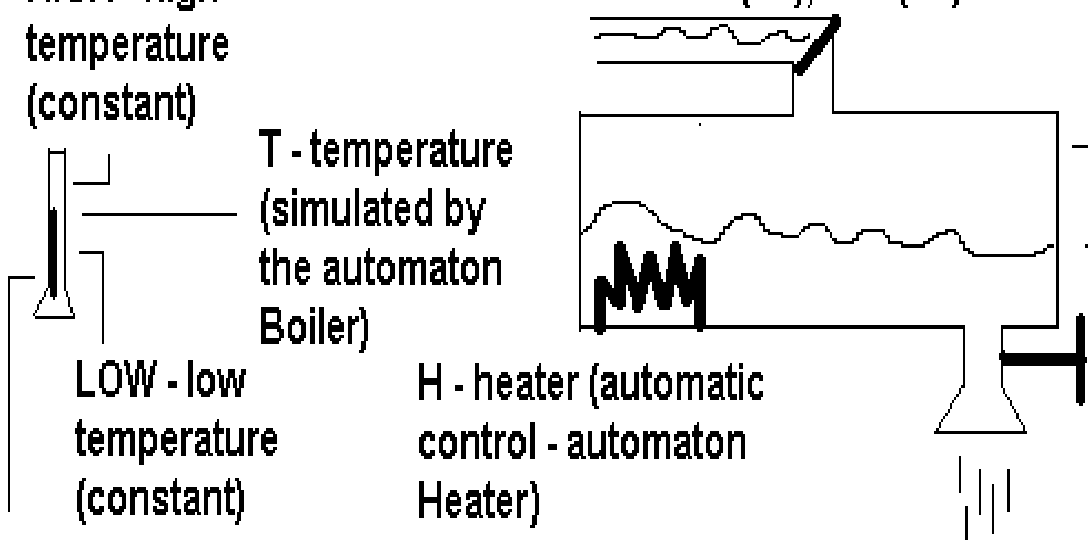
Implementation

- Preparation stage
  - Data structure $\rightarrow$ variables' history
  - Translation of descriptions into constraints
- Symbolic execution
  - search for paths
  - addition of new constraints corresponding to invariants, iterations and transitions

Implementation

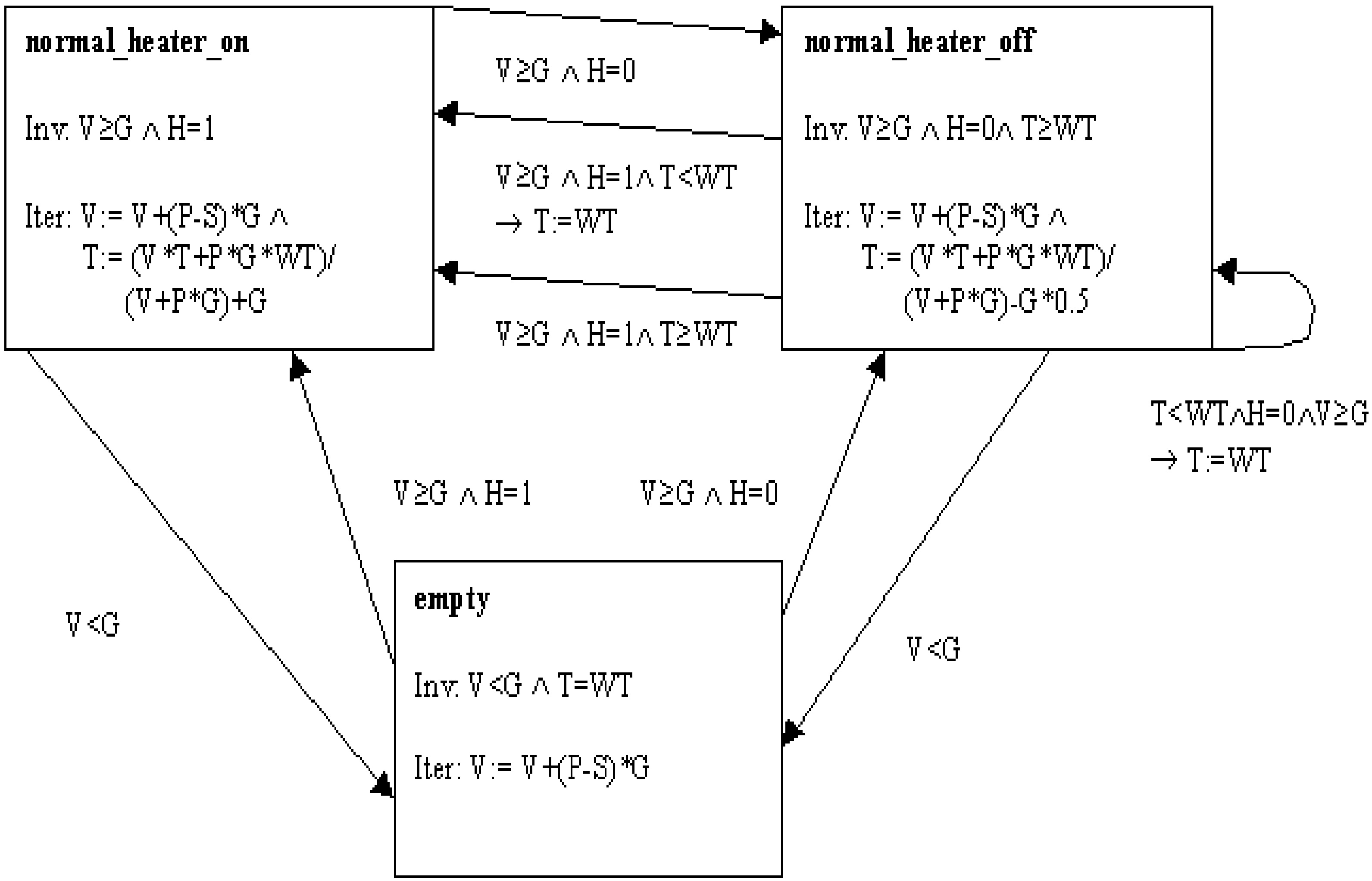
- Production of answers
  - Projection on selected variables
  - Printing
    - States at each clock
    - Remaining constraints resulting from execution and projection

Boiler Example



G - granularity - controls the speed with which the water volume and temperature vary (constant)

Boiler Automaton



Initial temperature for taking a shower without turning on the heater

```
INPUT:
CONSTRAINTS:    heater:1=0, pump:1=1, water_volume:1=10.0,
                shower:1=1,
                all(X,shower:X=1), all(X,heater:X=0)
INITIAL STATES: pump_on, heater_maintain,
                 boiler_normal_heater_off
FINAL STATES:   (not specified)
CLOCKS:         5
PROJECT:        temperature:1 (i.e. initial temperature)

OUTPUT:
Clock   Pump       Heater          Boiler
    1     on       maintain        normal_heater_off
    2     on       maintain        normal_heater_off
    3     on       maintain        normal_heater_off
    4     on       maintain        normal_heater_off
    5     on       maintain        normal_heater_off
temperature:1 > 47.18
```

Behaviour of the shower for the continuous increase of the water level

```
INPUT:
CONSTRAINTS:    heater:1=0, pump:1=1, temperature:1=30.0,
water_volume:1=6.0,
all(X,water_volume:(X+1)>water_volume:X) (increase water)
INITIAL STATES: pump_on, heater_maintain,
boiler_normal_heater_off
FINAL STATES:   (not specified)
CLOCKS:         5
PROJECT:        shower:X, water_volume:X (i.e. at all clocks)

OUTPUT:
Clock   Pump       Heater          Boiler
    1     on       maintain        normal_heater_off
    2     on       turning_on      normal_heater_on
    3     on       maintain        normal_heater_on
    4     on       maintain        normal_heater_on
    5     on       maintain        normal_heater_on
shower:[1..4]=0, shower:5=Var,
water_volume:1=6.0, water_volume:2=8.0, water_volume:3=10.0,
water_volume:4=12.0, water_volume:5=14.0
```

Summary

- We are concerned with validation and testing
- Meaningful experiments
- Derivation of properties
- Symbolic execution
- DExVal tool based on CLP

Future work

- Integration with ARTS’ graphical interface
- Tailoring the behaviour of the constraint solver:
  – Non-linear constraints
  – Non-determinism: disjunction and existential quantification
- Meaningful experiments:
  – Methodology
  – Real applications