

Benchmarks for Hybrid Systems Verification

Ansgar Fehnker¹ and Franjo Ivančić²

¹ Carnegie Mellon University, Pittsburgh

² NEC Laboratories America Inc, Princeton

Contents

- Introduction
- Prior benchmarks in Hybrid Systems Verification
- Example benchmark problems
 - Navigation problem
 - Leak test
 - Heating problem
- Benchmark characteristics
- Summary and future work

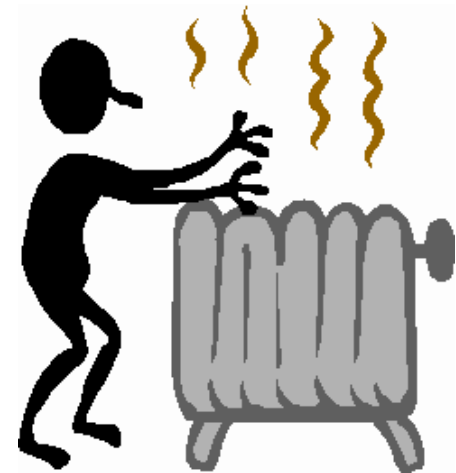
Benchmark Problem

- Evaluate the performance and capacities of different approaches/tools using the **same** problem.
- Many instances, scalable instances
- Examples
 - Traveling salesman problem
 - Job shop problems
 - Dining philosophers problem
 - Fisher mutual exclusion
 - Linpack benchmark
 - USC-SIPI (Image Processing)

Benchmarks in Hybrid Systems Verification

The Thermostat Example

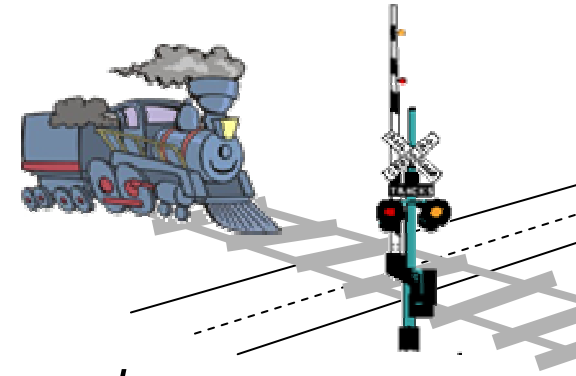
- The prototype hybrid system used to illustrate the concept *Hybrid System*
 - *The Algorithmic Analysis of Hybrid Systems* [ACH⁺95]
 - *HyTech: A Model Checker for Hybrid Systems* [HHW97]
 - *Counterexample Guided Predicate Abstraction of Hybrid Systems* [ADI03]
 - many more



Benchmarks in Hybrid Systems Verification

The Railroad Crossing

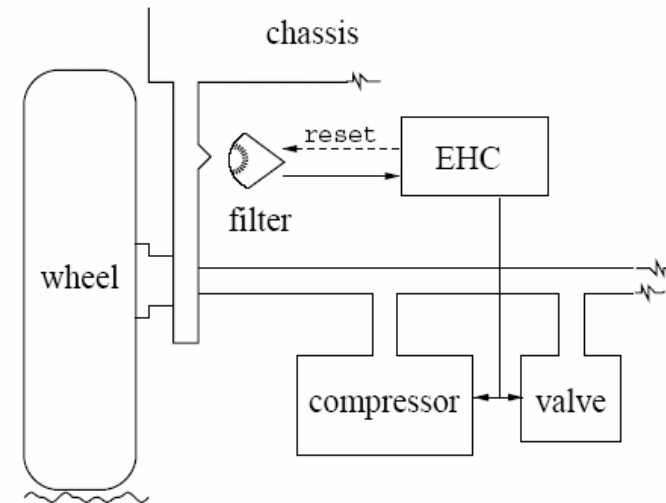
- A benchmark for specification and verification of real-time systems
 - *The Generalized Railroad Crossing* [HL95]
 - *HyTech: The Next Generation* [HHW95]
 - *Deductive Verification of Real-Time Systems using STEP* [BMSU97]
 - *Comparing Verification with Hytech, Kronos, and Uppaal on the Railroad Crossing Example* [BS00]
 - more



Benchmarks in Hybrid Systems Verification

The Electronic Level Controller

- Case study of an automotive level control system
 - *Using HyTech to Verify an Automotive Control System.* [SMF97]
 - *Automotive Control Revisited* [Feh98]
 - *Verification of Hybrid Systems via Mathematical Programming.* [BM99]
 - *Verification of an Automotive Active Leveler* [EB99]
- Incomparable results due to modifications, simplifications, additional assumptions.



Benchmarks in Hybrid Systems Verification

The Adaptive Cruise Controller

- A case study from the PATH project
 - *Counterexample Guided Predicate Abstraction of Hybrid Systems* [ADI03]
 - *Abstraction and Counterexample Guided Refinement of Hybrid Systems* [CFH⁺03]
- Very similar approach, but incomparable. Consider different part of the system.



Benchmarks in Hybrid Systems Verification

Observation

- Many verification examples are constructed to illustrate a concept
 - Fairly small single instances
 - Hand tailored to fit a certain framework
- Others are case studies:
 - A single (possibly big) instance
 - Solved partly by user interaction, smart modeling.
 - Results are often hard to reproduce and compare.

Benchmarks in Hybrid Systems Verification

Many questions remain about various tools

- Does the method scale? (#states, #continuous dim.)
- How much is the method automated?
- Did the tool or experienced user solve the problem?
- Can various methods/tools be “objectively” compared?

Benchmarks in Hybrid Systems Verification

This Paper

- Benchmarks for comparison of approaches to hybrid systems verification

Benchmarks in Hybrid Systems Verification

This Paper

- Benchmarks for comparison of approaches to hybrid systems verification
- Simple and method-independent problems
 - Brief, informal problem description
 - Suitable for comparison of different approaches to verification
 - Little domain knowledge necessary

Benchmarks in Hybrid Systems Verification

This Paper

- Benchmarks for comparison of approaches to hybrid systems verification
- Simple and method-independent problems
- Scalable problems
 - Number of continuous dimensions
 - Number of control locations
 - Switching behavior
 - Dynamic behavior

Benchmarks in Hybrid Systems Verification

This Paper

- Benchmarks for comparison of approaches to hybrid systems verification
- Simple and method-independent problems
- Scalable problems
- Many problems
 - Increases the need for automation
 - Minimizes the influence of user interaction

Three Benchmark Problems

Navigation Problem

Fixed number of continuous variables, increasing discrete size.

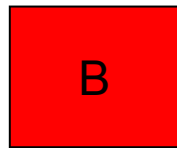
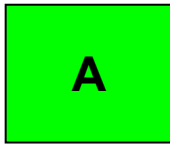
Leak Testing Problem

Simple discrete dynamics, increasing number of continuous variables

Heating Problem

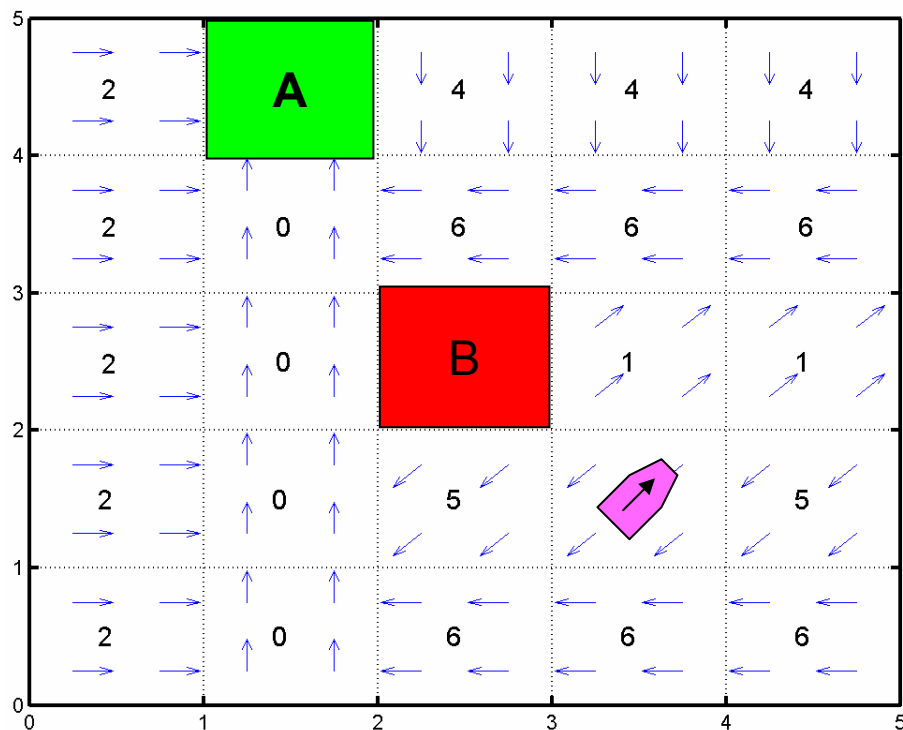
Increasing number of continuous variables and discrete locations.

Navigation Benchmark



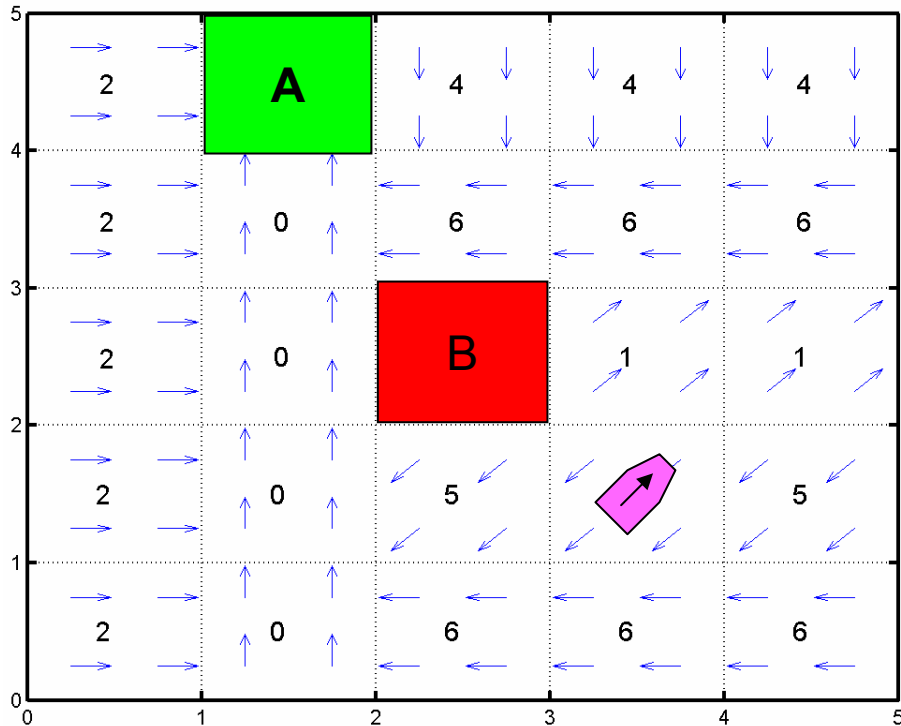
- The object should reach **A**, while it avoids **B**

Navigation Benchmark



- The object should reach **A**, while it avoids **B**
- The desired direction v_d depends on the position in a grid

Navigation Benchmark

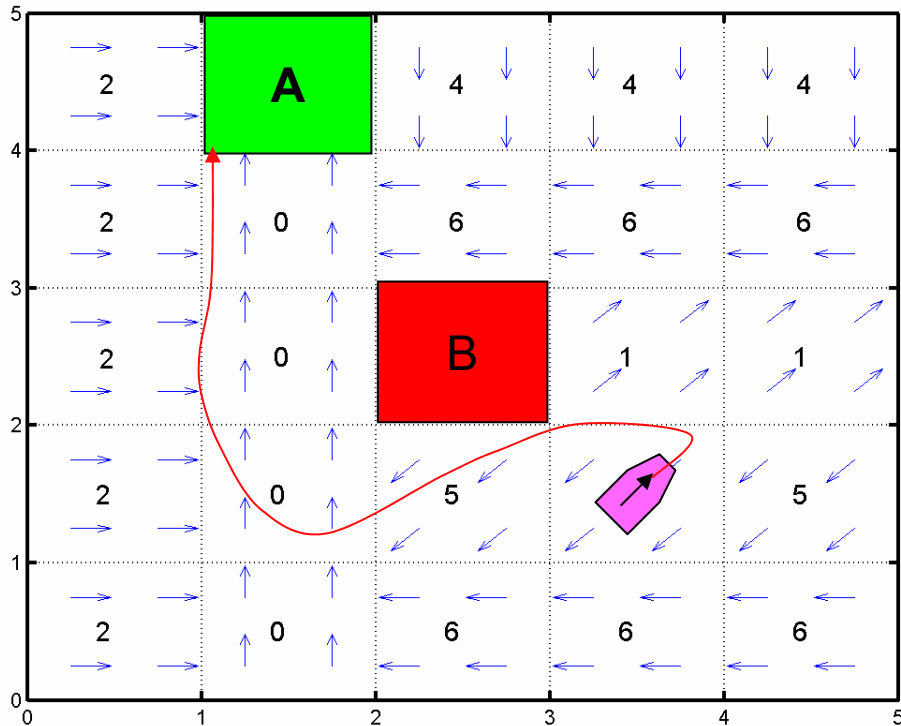


- The object should reach **A**, while it avoids **B**
- The desired direction v_d depends on the position in a grid
- The actual velocity is governed by a linear differential equation

$$\dot{x} = v$$

$$\dot{v} = A (v - v_d)$$

Navigation Benchmark

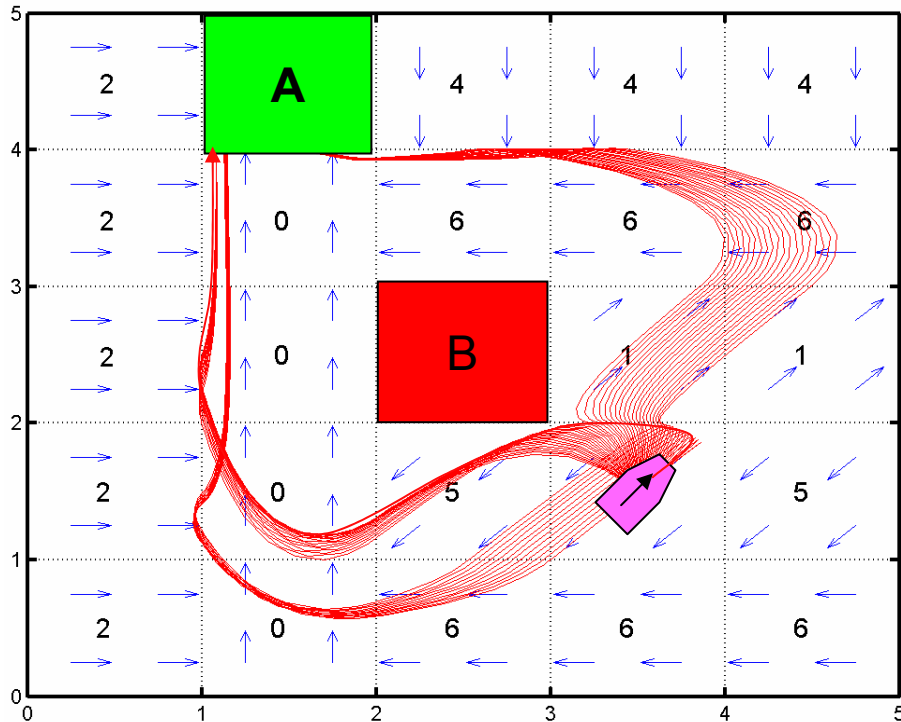


- The object should reach **A**, while it avoids **B**
- The desired direction v_d depends on the position in a grid
- The actual velocity is governed by a linear differential equation

$$\dot{x} = v$$

$$\dot{v} = A (v - v_d)$$

Navigation Benchmark

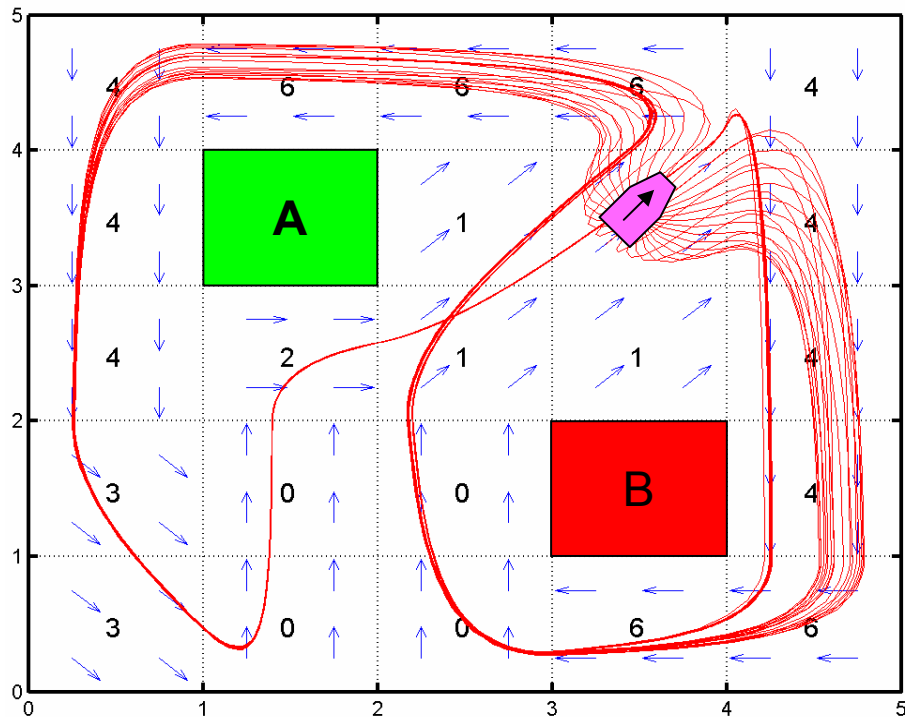


- The object should reach **A**, while it avoids **B**
- The desired direction v_d depends on the position in a grid
- The actual velocity is governed by a linear differential equation

$$\dot{x} = v$$

$$\dot{v} = A (v - v_d)$$

Navigation Benchmark



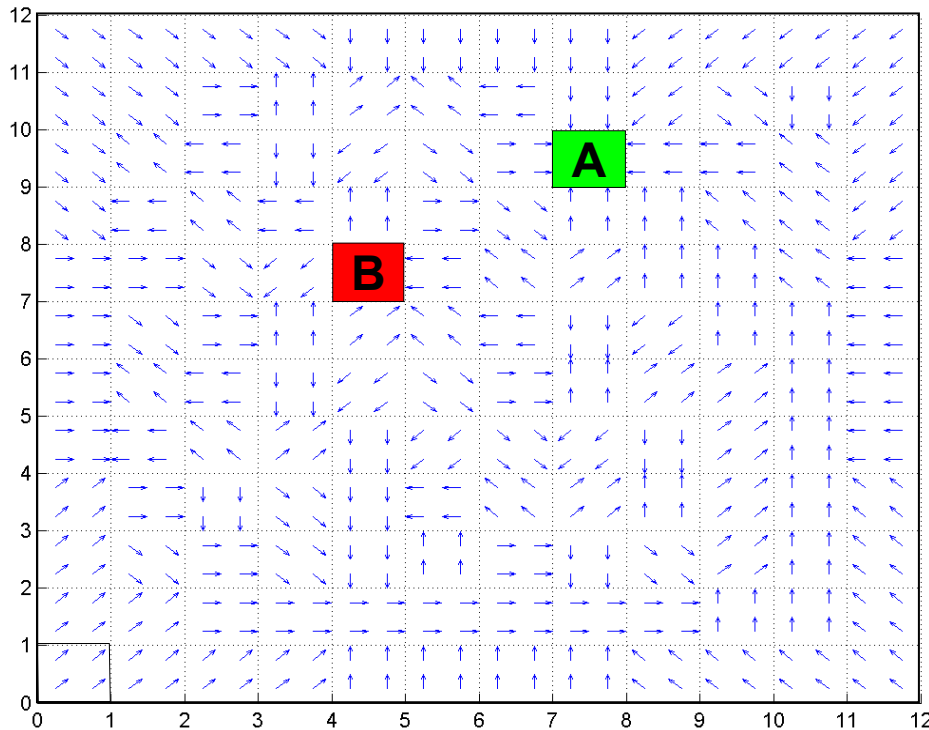
- The object should reach **A**, while it avoids **B**
- The desired direction v_d depends on the position in a grid
- The actual velocity is governed by a linear differential equation

$$\dot{x} = v$$

$$\dot{v} = A(v - v_d)$$

- Many instances

Navigation Benchmark



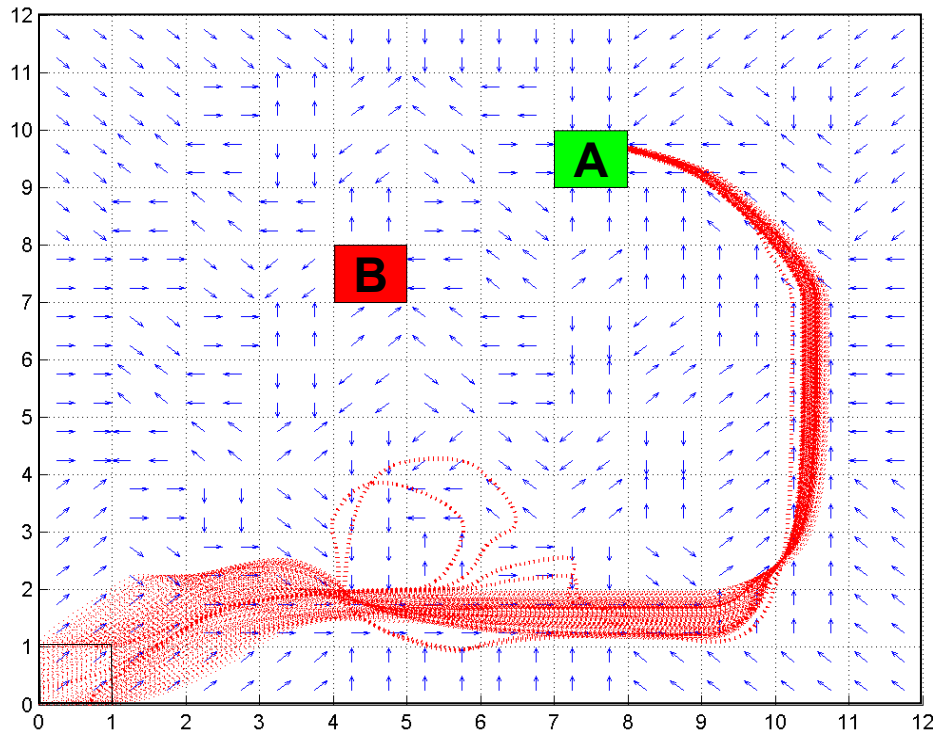
- The object should reach **A**, while it avoids **B**
- The desired direction v_d depends on the position in a grid
- The actual velocity is governed by a linear differential equation

$$\dot{x} = v$$

$$\dot{v} = A (v - v_d)$$

- Many instances
- Scalable

Navigation Benchmark



- The object should reach **A**, while it avoids **B**
- The desired direction v_d depends on the position in a grid
- The actual velocity is governed by a linear differential equation

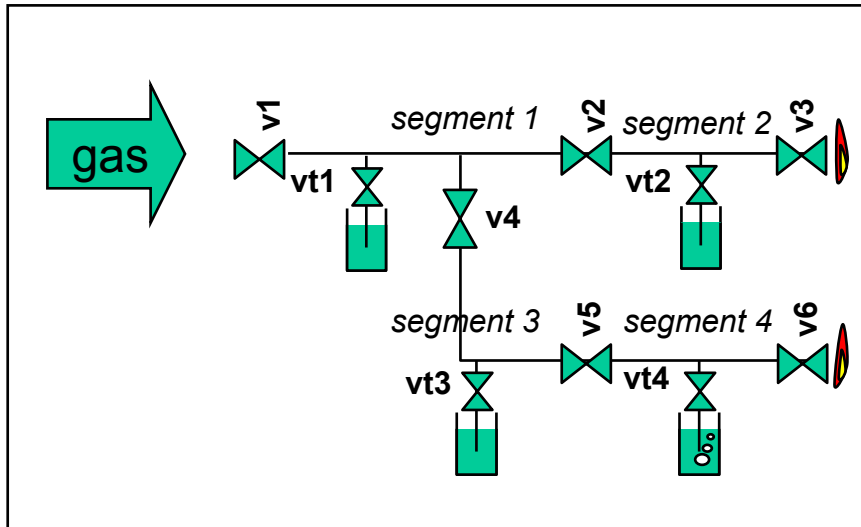
$$\dot{x} = v$$

$$\dot{v} = A (v - v_d)$$

- Many instances
- Scalable
- Interesting features can be included

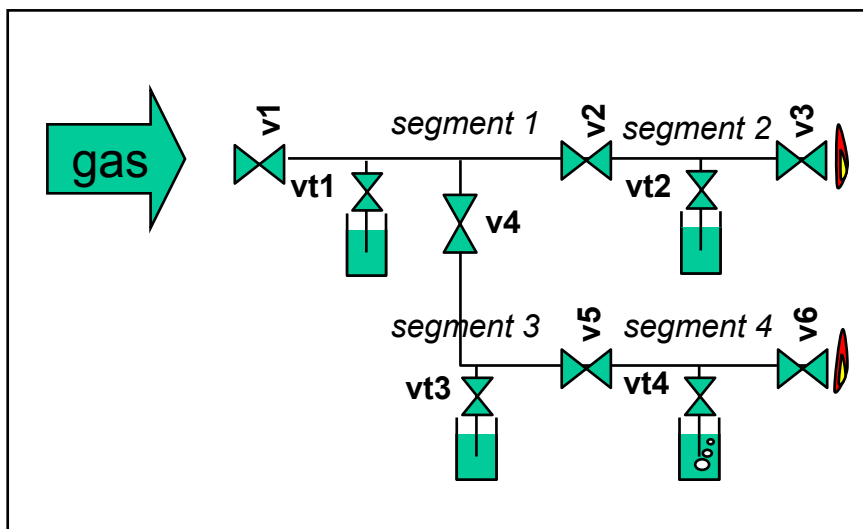
Leak Test Benchmark

Detect leaking valves in a pressurized network



Leak Test Benchmark

Detect leaking valves in a pressurized network



Non-linear dynamics:

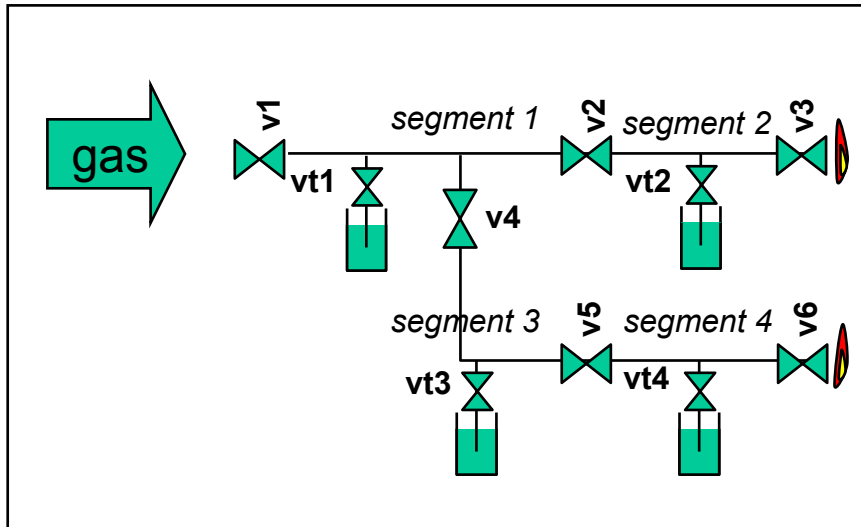
$$\dot{x}_{i_0} = \sum_{j=1, \dots, k} c_j f(x_{i_0}, x_{i_k}) + dg(x_{i_0})$$

$$f(x, y) = \begin{cases} -\sqrt{x - y} & \text{if } x \geq y \\ \sqrt{y - x} & \text{otherwise} \end{cases}$$

$$g(x) = \begin{cases} -\sqrt{x - z_{off}} & \text{if } x \geq z_{off} \\ 0 & \text{otherwise} \end{cases}$$

Leak Test Benchmark

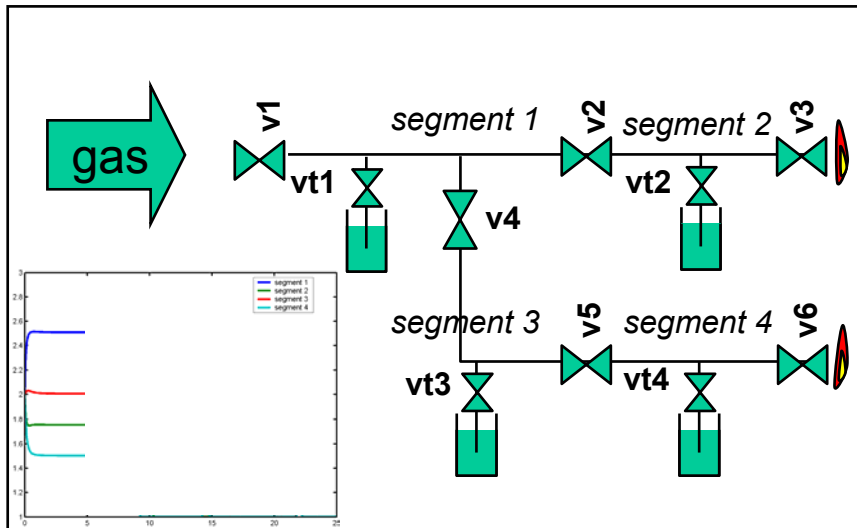
Detect leaking valves in a pressurized network



- **Pressurize** the network
- **Close** all valves
- **Test** segment as soon as all upstream segments have been tested.
 - **Wait** for some time
 - **Open** tap valve
No bubbles => upstream leak
 - **Open** upstream valves
 - **Wait** for some time
Still bubbling => downstream leak

Leak Test Benchmark

Detect leaking valves in a pressurized network



➔ **Pressurize** the network

➔ **Close** all valves

- **Test** segment as soon as all upstream segments have been tested.

- **Wait** for some time

- **Open** tap valve

*No bubbles => **upstream leak***

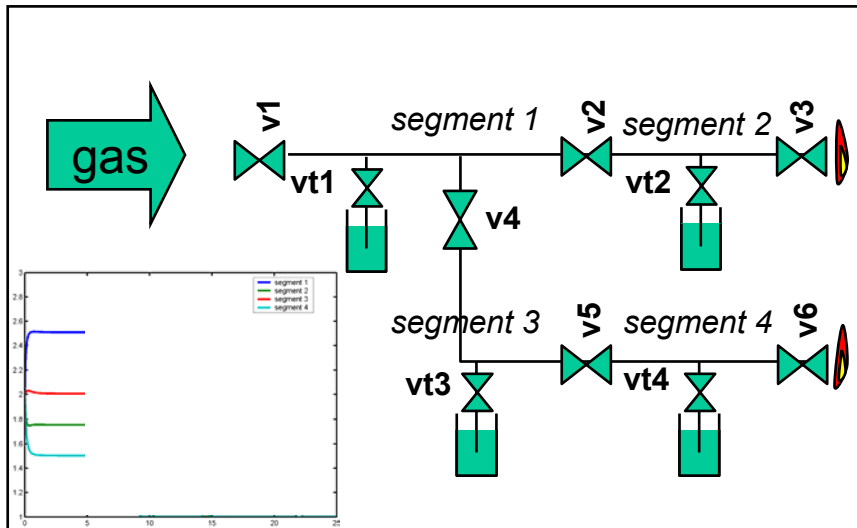
- **Open** upstream valves

- **Wait** for some time

*Still bubbling => **downstream leak***

Leak Test Benchmark

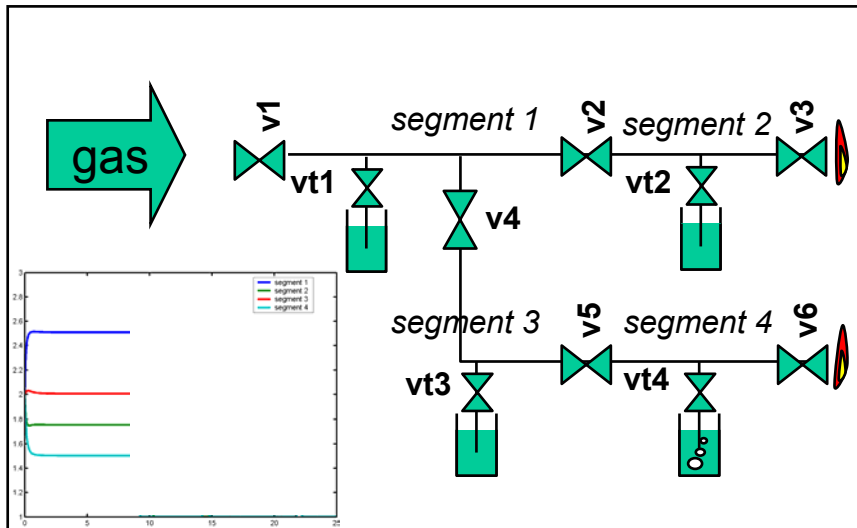
Detect leaking valves in a pressurized network



- Pressurize the network
- Close all valves
- ➔ Test segment as soon as all upstream segments have been tested.
 - Wait for some time
 - Open tap valve
 - No bubbles => upstream leak*
 - Open upstream valves
 - Wait for some time
 - Still bubbling => downstream leak*

Leak Test Benchmark

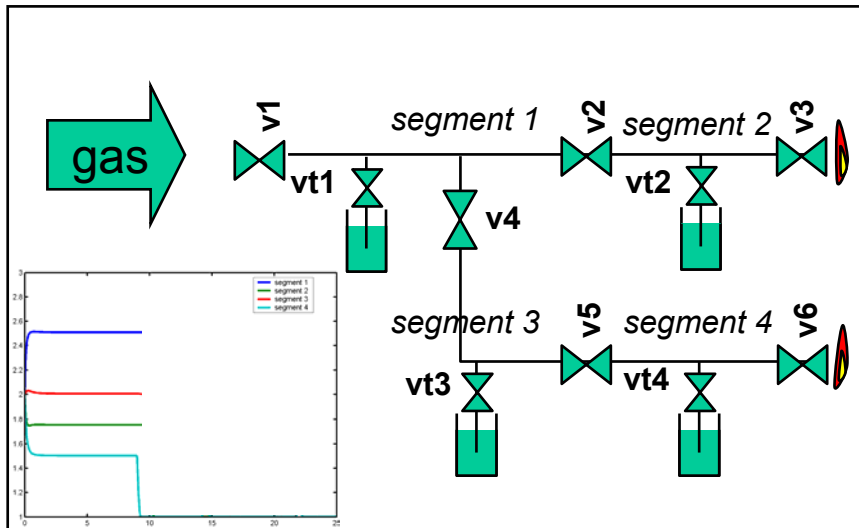
Detect leaking valves in a pressurized network



- **Pressurize** the network
- **Close** all valves
- ➔ **Test** segment as soon as all upstream segments have been tested.
- ➔ **Wait** for some time
- ➔ **Open** tap valve
 - No bubbles => **upstream leak***
- **Open** upstream valves
- **Wait** for some time
 - Still bubbling => **downstream leak***

Leak Test Benchmark

Detect leaking valves in a pressurized network



- **Pressurize** the network
- **Close** all valves
- ➔ **Test** segment as soon as all upstream segments have been tested.

- **Wait** for some time
- **Open** tap valve
- No bubbles => **upstream leak***

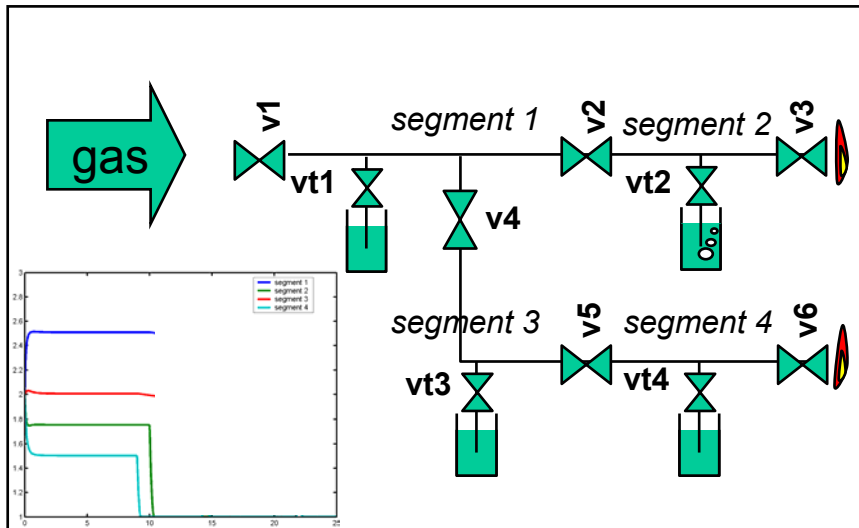
➔ **Open** upstream valves

➔ **Wait** for some time

*Still bubbling => **downstream leak***

Leak Test Benchmark

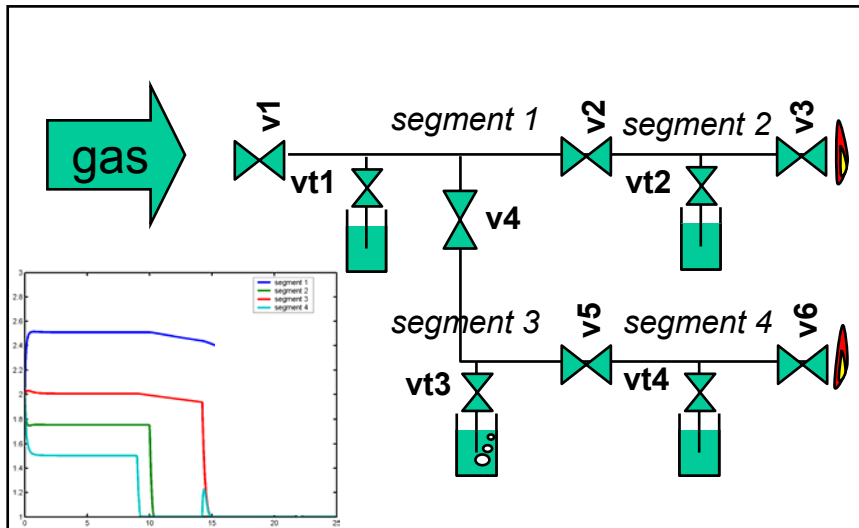
Detect leaking valves in a pressurized network



- **Pressurize** the network
- **Close** all valves
- **Test** segment as soon as all upstream segments have been tested.
 - **Wait** for some time
 - **Open** tap valve
*No bubbles => **upstream leak***
 - **Open** upstream valves
 - **Wait** for some time
*Still bubbling => **downstream leak***

Leak Test Benchmark

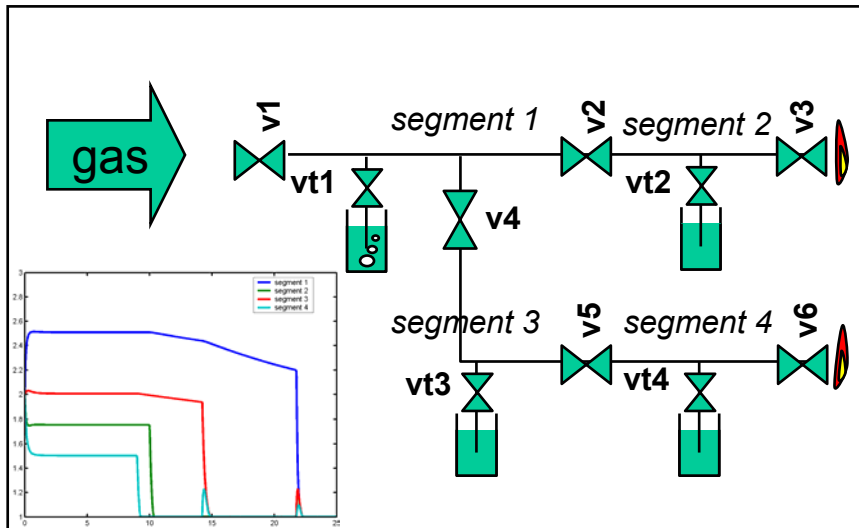
Detect leaking valves in a pressurized network



- **Pressurize** the network
- **Close** all valves
- **Test** segment as soon as all upstream segments have been tested.
 - **Wait** for some time
 - **Open** tap valve
*No bubbles => **upstream leak***
 - **Open** upstream valves
 - **Wait** for some time
*Still bubbling => **downstream leak***

Leak Test Benchmark

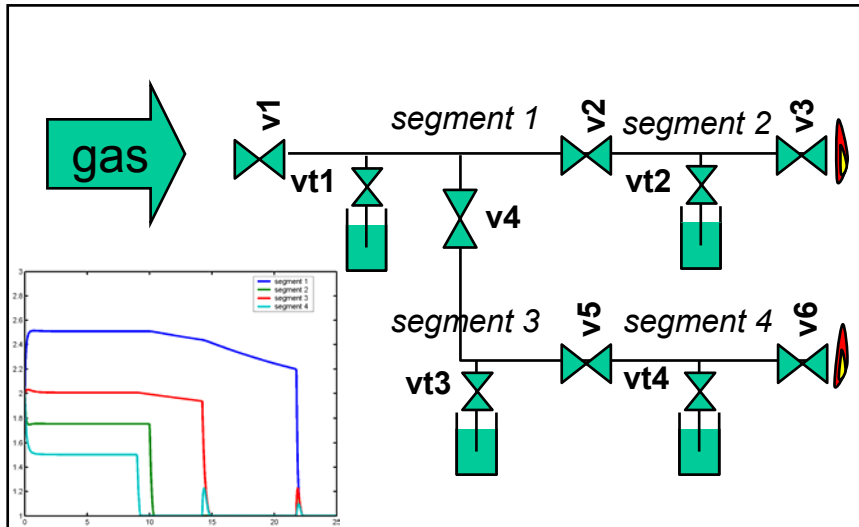
Detect leaking valves in a pressurized network



- **Pressurize** the network
- **Close** all valves
- **Test** segment as soon as all upstream segments have been tested.
 - **Wait** for some time
 - **Open** tap valve
 - No bubbles => **upstream leak***
 - **Open** upstream valves
 - **Wait** for some time
 - Still bubbling => **downstream leak***

Leak Test Benchmark

Detect leaking valves in a pressurized network



- **Pressurize** the network
- **Close** all valves
- **Test** segment as soon as all upstream segments have been tested.

- **Wait** for some time
- **Open** tap valve
No bubbles => upstream leak

- **Open** upstream valves
- **Wait** for some time

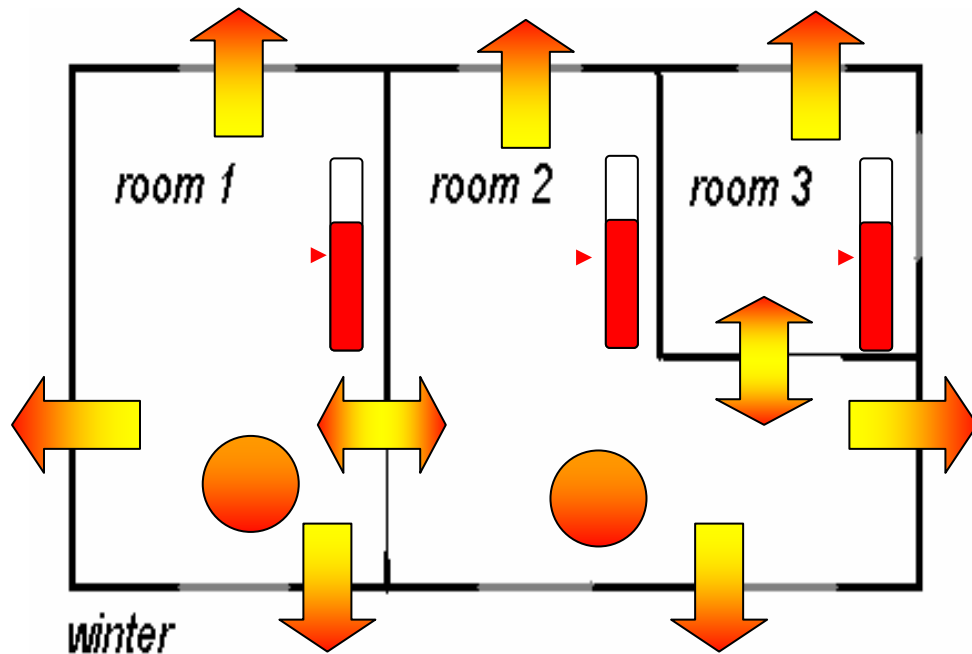
Still bubbling => downstream leak

Desired property

Detect leaks correctly

Heating Benchmark

Heat rooms with a limited number of heaters

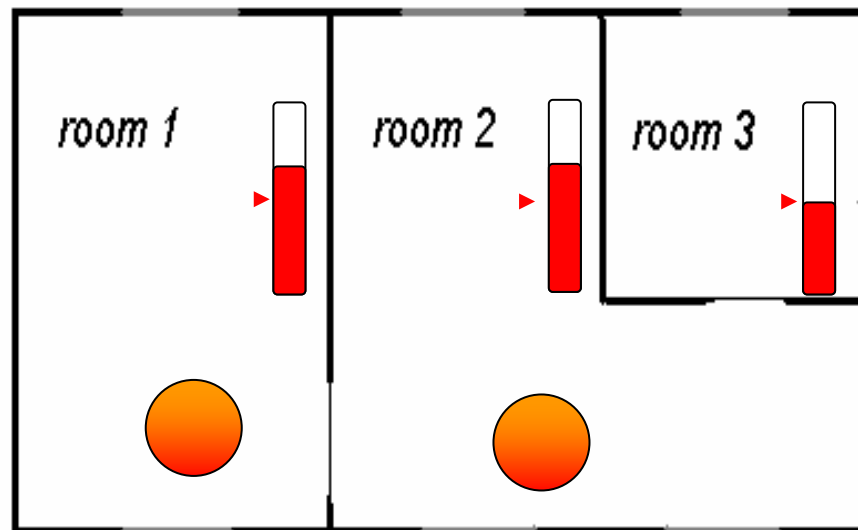


- The temperature in a room depends on
 - Loss to environment
 - Temperature in adjacent rooms
 - Presence of heaters
 - State of heater
- Linear dynamics:

$$\dot{x}_i = c_i h_i + b_i (u - x_i) + \sum_{i \neq j} a_{i,j} (x_j - x_i)$$

Heating Benchmark

Heat rooms with a limited number of heaters

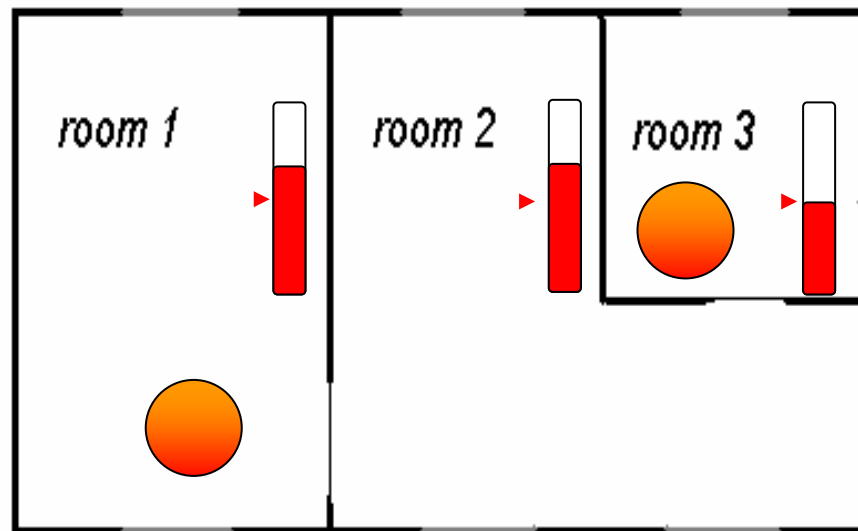


winter

- A room can obtain a heater from another room, if
 - The temperature in the first room drops below a threshold
 - The difference in temperature is above a threshold
 - The first room has no heater
 - The second room has a heater

Heating Benchmark

Heat rooms with a limited number of heaters

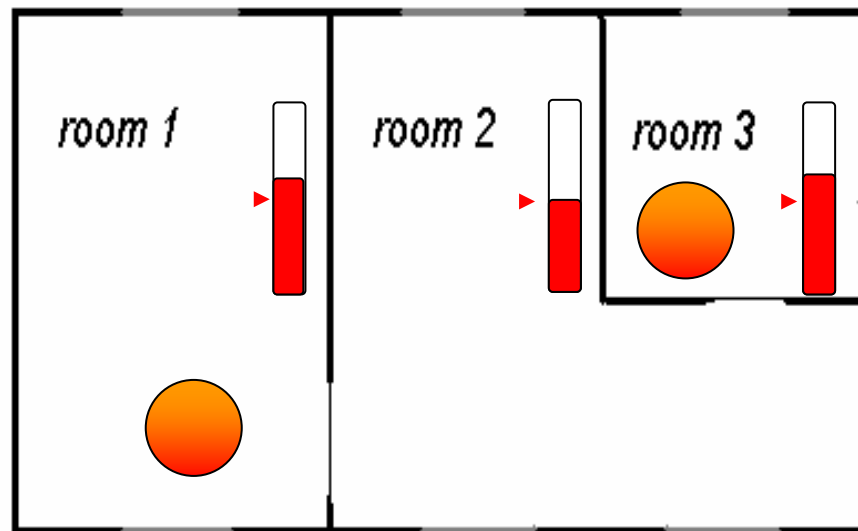


winter

- A room can obtain a heater from another room, if
 - The temperature in the first room drops below a threshold
 - The difference in temperature is above a threshold
 - The first room has no heater
 - The second room has a heater

Heating Benchmark

Heat rooms with a limited number of heaters

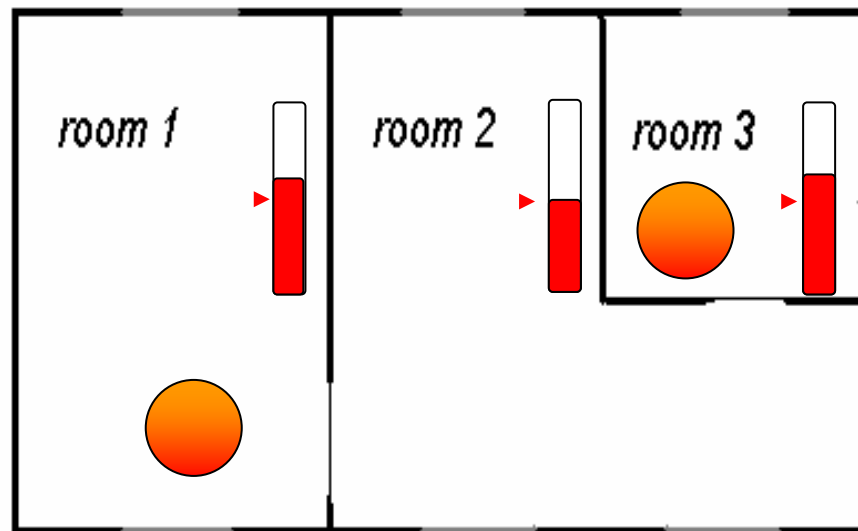


winter

- A room can obtain a heater from another room, if
 - The **temperature** in the first room drops **below a threshold**
 - The **difference** in temperature is **above a threshold**
 - The first room has no heater
 - The second room has a heater

Heating Benchmark

Heat rooms with a limited number of heaters

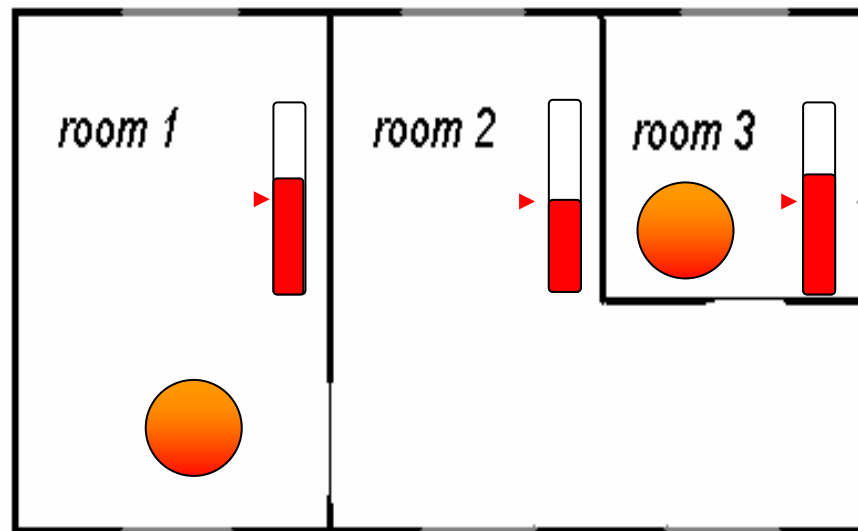


winter

- A room can obtain a heater from another room, if
 - The temperature in the first room drops below a threshold
 - The difference in temperature is above a threshold
 - The first room has no heater
 - The second room has a heater
- Includes non-deterministic choices

Heating Benchmark

Heat rooms with a limited number of heaters



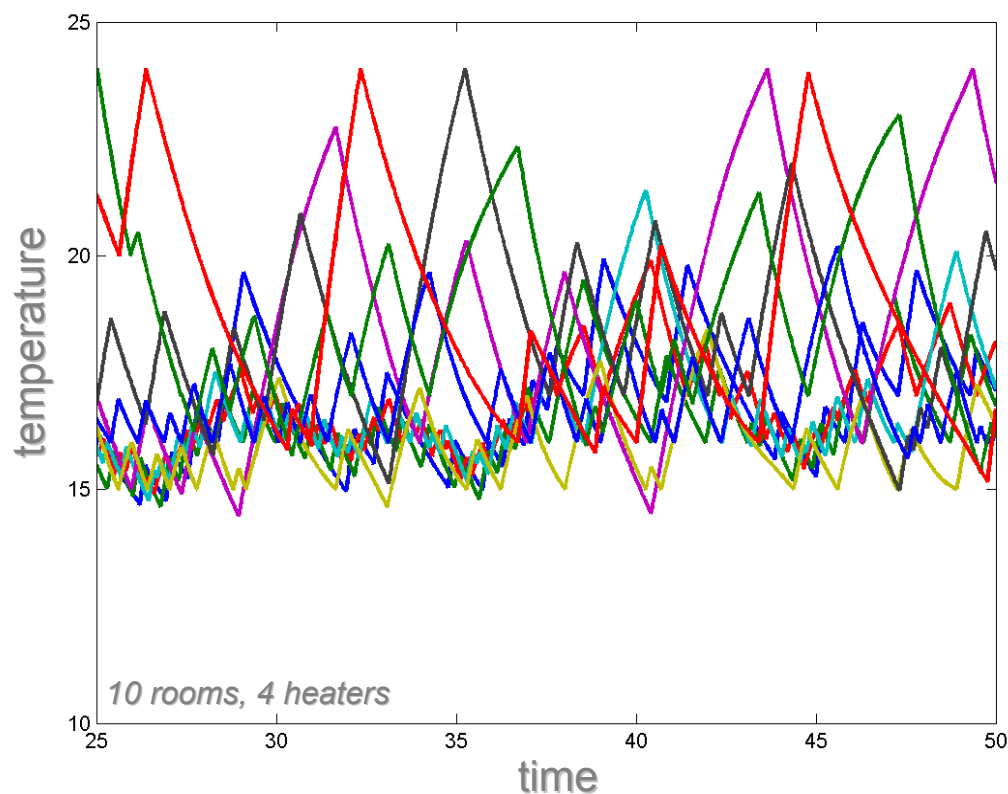
winter

■ Properties

- The **temperature** of all rooms remains **above** a (room specific) **threshold**.
- Every **room eventually obtains** a heater
- Every **room eventually shares** a heater

Heating Benchmark

Heat rooms with a limited number of heaters



■ Properties

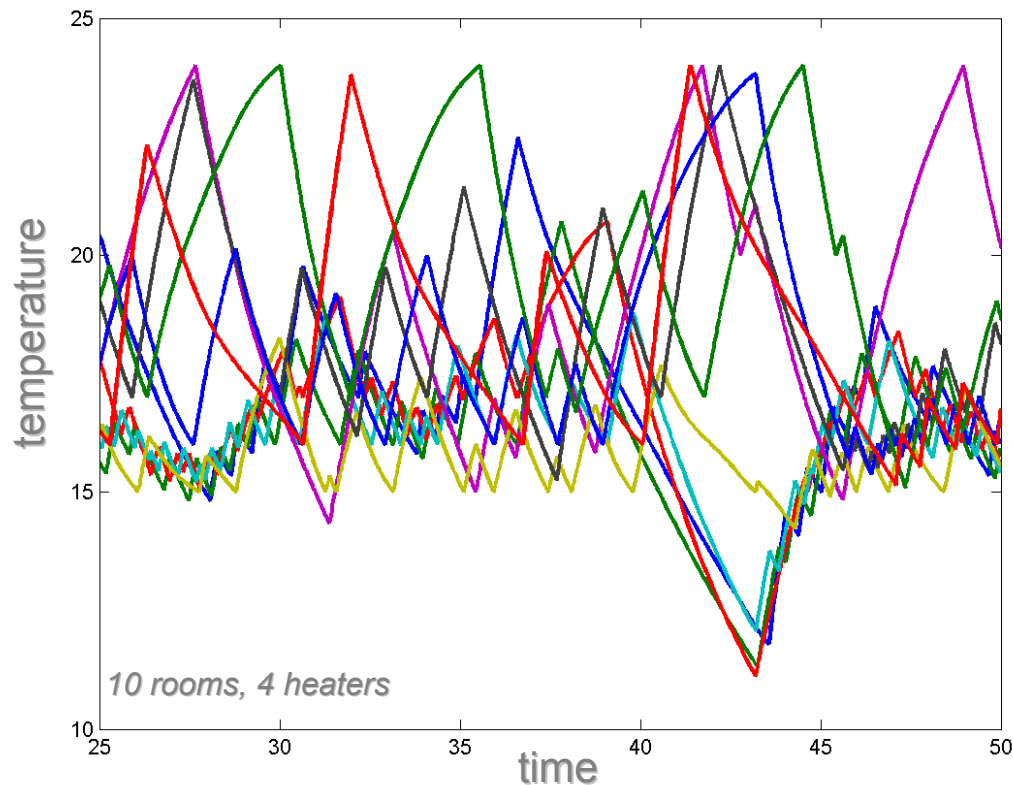
- The **temperature** of all rooms remains **above** a (room specific) **threshold**.
- Every **room eventually obtains** a heater
- Every **room eventually shares** a heater

■ Multiple instances

■ Interesting features

Heating Benchmark

Heat rooms with a limited number of heaters



■ Properties

- The **temperature** of all rooms remains **above** a (room specific) **threshold**.
- Every **room eventually obtains** a heater
- Every **room eventually shares** a heater

■ Multiple instances

■ Interesting features

Benchmark Characteristics

30 instances of each problem

(<http://www.ece.cmu.edu/~ansgar/benchmark>)

```
%% NAV01
MAP= [ B      2      4;
      2      3      4;
      2      2      A]
MatA= [-1.2   0.1;
       0.1  -1.2]
x0 in [2,3]x[1,2]
v0 in [-0.3,0.3]x[-0.3,0]
```

- Concise representation
- MatLab/Simulink model of all instances
- Vary in the number of continuous dimensions and discrete control locations
- May exhibit interesting features
- Defined number of safety/liveness properties for each problem.

Benchmark Characteristics

30 instances of each problem
(<http://www.ece.cmu.edu/~ansgar/benchmark>)

```
%% NAV01
MAP= [ B      2      4;
      2      3      4;
      2      2      A]
MatA= [-1.2   0.1;
       0.1  -1.2]
x0 in [2,3]x[1,2]
v0 in [-0.3,0.3]x[-0.3,0]
```

- First results for instances of the navigation problem
- *Charon* model that over-approximates all behavior
- Compared:
 - Reachability algorithm
 - Counter-example guided predicate abstraction
- More results in [Iva03]

Benchmark Characteristics

Characteristics of the navigation problem

Instance	Grid size
NAV01-06	3 x 3
NAV07-09	4 x 4
NAV10-12	5 x 5
NAV13-15	7 x 7
NAV16-18	9 x 9
NAV19-21	12 x 12
NAV22-24	15 x 15
NAV25-27	20 x 20
NAV28-30	25 x 25

- Linear dynamics
- Four continuous dimensions
- 10 or more discrete locations
- Deterministic (mostly)
- Grid size: see table

Benchmark Characteristics

Characteristics of the leak test problem

instance	segments	valves
LEAK01-03	3	4
LEAK04-06	3	5
LEAK07-09	4	6
LEAK10-12	5	8
LEAK13-15	7	10
LEAK16-18	9	12
LEAK19-21	11	14
LEAK22-24	14	18
LEAK25-27	17	22
LEAK28-30	20	26

- Non-linear dynamics (*sqrt*)
- Continuous dimensions:
 - > 1 per segment
(plus one clock per branch)
- Discrete locations:
 - > 3 per location
- No loops in the control structure
- Probably compositional

Benchmark Characteristics

Characteristics of the heating problem

instance	rooms	heaters	r over h
HEAT01-03	3	1	3
HEAT04-06	3	2	3
HEAT07-09	4	3	4
HEAT10-12	4	2	6
HEAT13-15	5	3	10
HEAT16-18	6	3	20
HEAT19-21	7	4(3)	35
HEAT22-24	8	3	56
HEAT25-27	9	4	126
HEAT28-30	10	4	210

- Linear dynamics
- Symmetric matrices
- Continuous dimensions:
1 per room
- Discrete locations:
 $> (r \text{ over } h) * h^2$
- Non-determinism
- Probably compositional

Summary

- Three method-independent scalable problems
 1. Fixed number of continuous variables, increasing discrete size
 2. Simple discrete dynamics, increasing # of continuous variables
 3. Increasing # of continuous variables and discrete locations
- Different characteristics/features
- Fixed instances with given properties
- First results for the navigation benchmark.
- Provided MatLab/Simulink models.

<http://www.ece.cmu.edu/~ansgar/benchmark>

Future Work

- Solving the proposed problem instances
- Maintain instances
- The proposed problem stress only certain characteristics of HS verification

⇒ Other benchmarks with

- Other non-linear dynamics
- Simpler than linear dynamics
- Different sources of non-determinism
- Data
- Stochastic systems
- ..

<http://www.ece.cmu.edu/~ansgar/benchmark>

