

Higher-Dimensional Timed Automata

Uli Fahrenberg

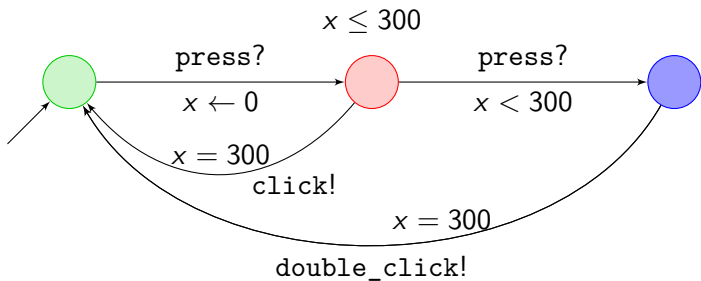
LRE, EPITA, France

FEANICES

December 2022



Recall Timed Automata



Recall Timed Automata

Definition

The set $\Phi(C)$ of **clock constraints** ϕ over a finite set C is defined by the grammar

$$\phi ::= x \bowtie k \mid \phi_1 \wedge \phi_2 \quad (x, y \in C, k \in \mathbb{Z}, \bowtie \in \{\leq, <, \geq, >\}).$$

Definition

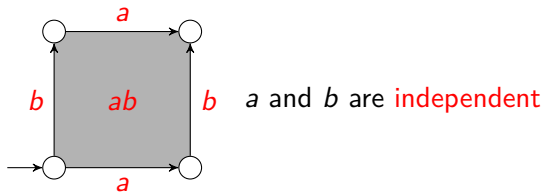
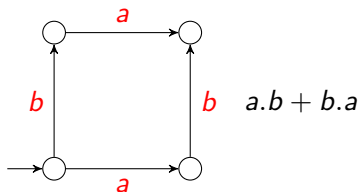
A **timed automaton** is a tuple $(L, \ell_0, C, \Sigma, I, E)$ consisting of a finite set L of locations, an initial location $\ell_0 \in L$, a finite set C of clocks, a finite set Σ of actions, a location invariants mapping $I : L \rightarrow \Phi(C)$, and a set $E \subseteq L \times \Phi(C) \times \Sigma \times 2^C \times L$ of edges.

Recall Timed Automata

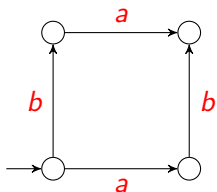
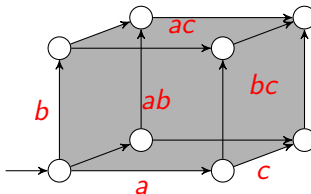
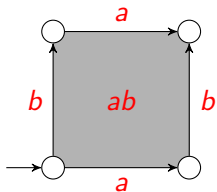
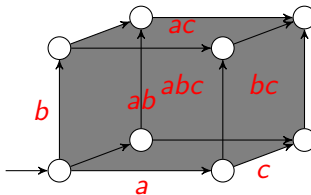
- Useful for modeling **synchronous** real-time systems
- Reachability, emptiness, LTL model checking PSPACE-complete
- Universality undecidable
- Fast on-the-fly algorithms, using zones, for reachability, liveness, and Timed CTL model checking
- **UppAal**
- Extensions to **weighted** timed automata, real-time **games**, etc.

Recall Higher-Dimensional Automata

$a|b$



Recall Higher-Dimensional Automata

 $a|b$

 $a|b|c$

 $a|b + a|c + b|c$
 a

 a

 $\{a, b, c\}$ independent

Recall Higher-Dimensional Automata

An **loset** is a finite, ordered and Σ -labelled set. (a list of events)

A **precubical set** X consists of:

- A set of cells X (cubes)
- Every cell $x \in X$ has an loset $\text{ev}(x)$ (list of events active in x)
- We write $X[U] = \{x \in X \mid \text{ev}(x) = U\}$ for an loset U (cells of type U)
- For every loset U and $A \subseteq U$ there are:
 - upper face map** $\delta_A^1 : X[U] \rightarrow X[U - A]$ (terminating events A)
 - lower face map** $\delta_A^0 : X[U] \rightarrow X[U - A]$ (unstarting events A)
- **Precube identities:** $\delta_A^\mu \delta_B^\nu = \delta_B^\nu \delta_A^\mu$ for $A \cap B = \emptyset$ and $\mu, \nu \in \{0, 1\}$

A **higher dimensional automaton (HDA)** is a precubical set X with **start cells** $X_\perp \subseteq X$ and **accept cells** $X^\top \subseteq X$ (not necessarily vertices)

Recall Higher-Dimensional Automata

HDA as a model for concurrency:

- points $x \in X_0$: **states**
- edges $a \in X_1$: **transitions** (labeled with **events**)
- n -squares $\alpha \in X_n$ ($n \geq 2$): **independency** relations (concurrently executing events)

van Glabbeek (TCS 2006): Up to history-preserving bisimilarity, HDA generalize “the main models of concurrency proposed in the literature”

- (for example **Petri nets**)

The Marriage between Real Time and Concurrency

- In real-time formalisms, everything is **synchronous**
 - timed automata, timed Petri nets, hybrid automata, etc.
- and concurrency is **interleaving**
- In formalisms for (non-interleaving) concurrency, **no real time**
 - same for distributed computing theory
 - (Petri nets have a concurrent semantics; timed Petri nets don't)
- Our goal: formalisms for **real-time concurrent** systems
- Application: for example **distributed cyber-physical systems**
- Here: the marriage between **timed** and **higher-dimensional** automata

Actions Take Time?

- **Cardelli 1982 (ICALP)**: Actions **take time**.
 - ‘We read $p \xrightarrow[t]{a} q$ as “ p moves to q performing a for an interval t ”’
- since **Alur-Dill 1990** (even before?): Actions are **immediate**.
 - $(l, v) \xrightarrow{d} (l, v + d) \xrightarrow{s} (l', v + d)$
- **Kim G. Larsen** (personal discussions): Actions are immediate mostly because of **technical** reasons. (“We know how to do; and it’s sufficient”)
- **Henzinger-Manna-Pnueli 1990**: same
- **Chatain-Jard 2013**: In the concurrent semantics for time Petri nets, time has to (locally) be allowed to **run backwards??**
- **U.F. 2018**: In real-time concurrency, actions **cannot** be immediate.
 - and it appears that the “technical reasons” argument is quite weak!

- 1 Motivation
- 2 Higher-Dimensional Timed Automata
- 3 Languages of HDTA
- 4 Higher-Dimensional Hybrid Automata
- 5 Conclusion

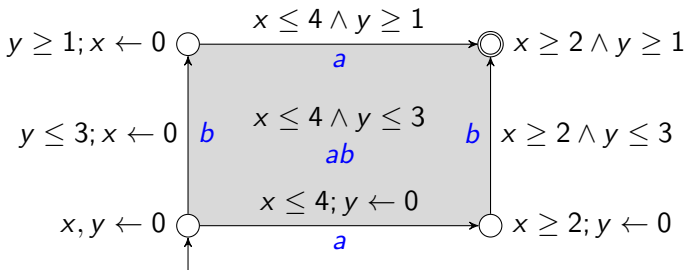
Higher-Dimensional Timed Automata

Definition

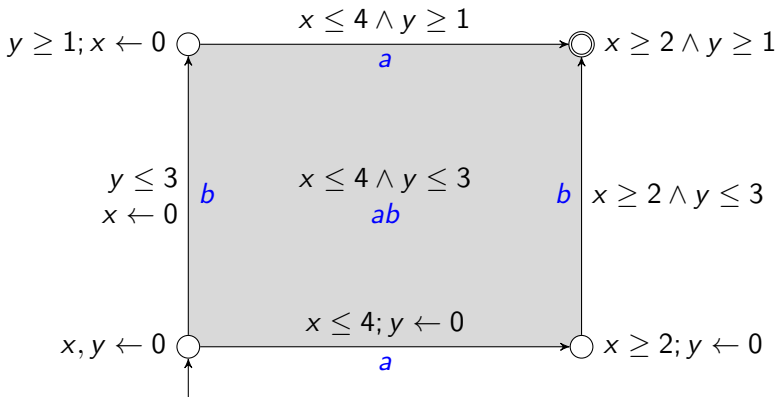
A **HDTA** is a structure $(L, l^0, L^f, \lambda, C, \text{inv}, \text{exit})$, where (L, l^0, L^f, λ) is a finite HDA, C is a finite set of clocks, and $\text{inv} : L \rightarrow \Phi(C)$, $\text{exit} : L \rightarrow 2^C$ give **invariant** and **exit** conditions for each n -cube.

Intuition:

- $\text{inv}(l)$: conditions on the clock values while **delaying** in l
- $\text{exit}(l)$: clocks to be **reset** to 0 when leaving l .

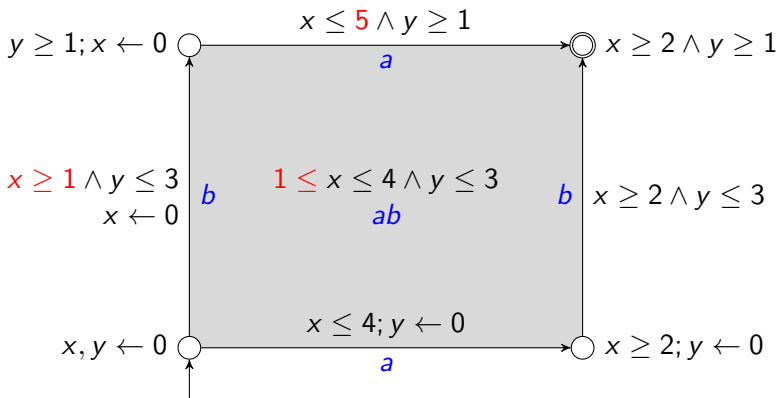


Examples



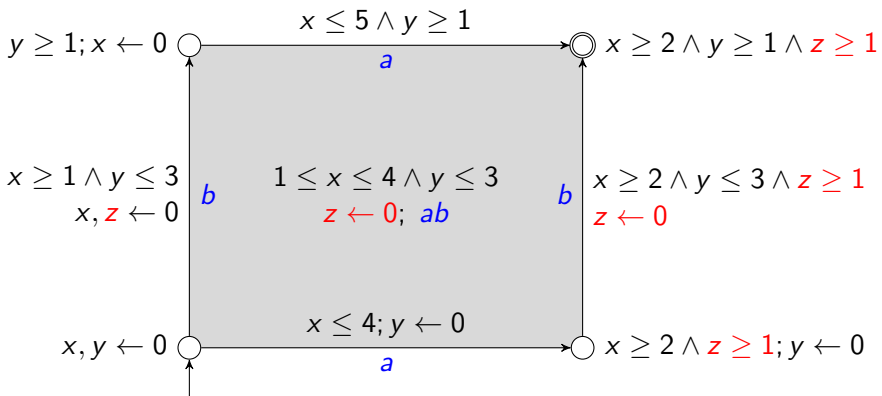
- *a* takes $[2, 4]$ time units, *b* takes $[1, 3]$ time units

Examples



- a takes $[2, 4]$ time units, b takes $[1, 3]$ time units
- unless b is done before a
- b can only start 1 time unit after a

Examples

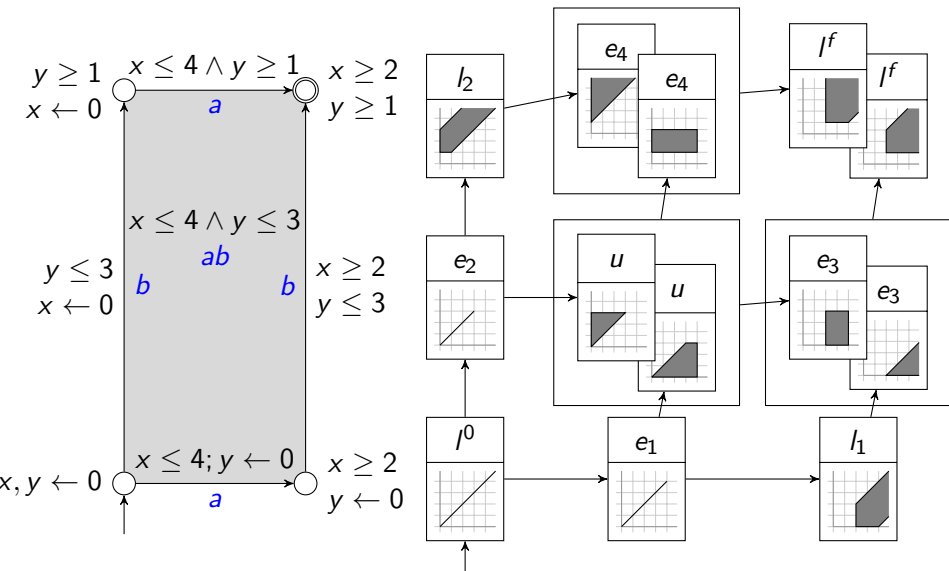


- a takes $[2, 4]$ time units, b takes $[1, 3]$ time units
- b can only start 1 time unit after a
- b has to finish 1 time unit before a

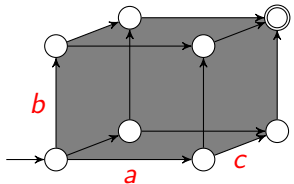
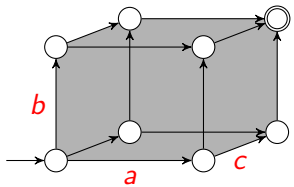
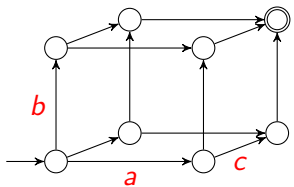
Good News

- **Reachability** for HDTA is PSPACE-complete
- and can be checked using **zone**-based algorithms
- (Everything works like for timed automata)
- Universality probably still undecidable

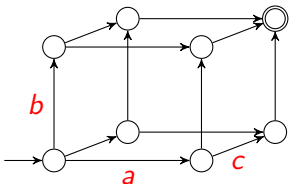
Zone-Based Reachability



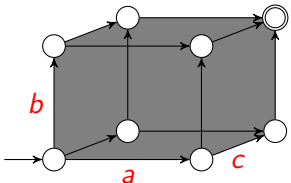
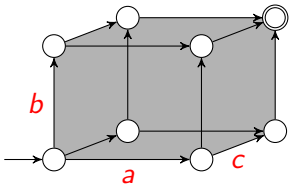
Languages of HDA



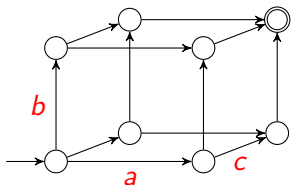
Languages of HDA



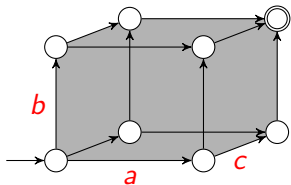
$$L_1 = \{abc, acb, bac, bca, cab, cba\}$$



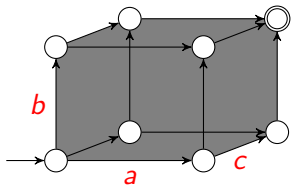
Languages of HDA



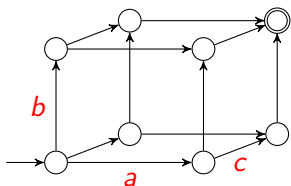
$$L_1 = \{abc, acb, bac, bca, cab, cba\}$$



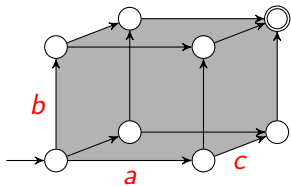
$$L_3 = \left\{ \begin{pmatrix} a \\ b \\ c \end{pmatrix}, \dots \right\}$$



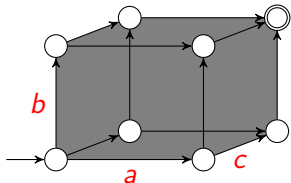
Languages of HDA



$$L_1 = \{abc, acb, bac, bca, cab, cba\}$$

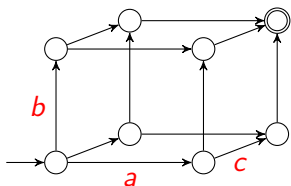


$$L_2 = \left\{ \begin{pmatrix} a \\ b \rightarrow c \end{pmatrix}, \begin{pmatrix} a \\ c \rightarrow b \end{pmatrix}, \begin{pmatrix} b \\ a \rightarrow c \end{pmatrix}, \right. \\ \left. \begin{pmatrix} b \\ c \rightarrow a \end{pmatrix}, \begin{pmatrix} c \\ a \rightarrow b \end{pmatrix}, \begin{pmatrix} c \\ b \rightarrow a \end{pmatrix}, \dots \right\}$$

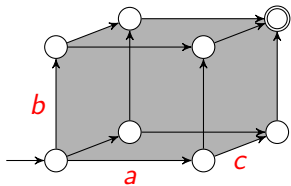


$$L_3 = \left\{ \begin{pmatrix} a \\ b \\ c \end{pmatrix}, \dots \right\}$$

Languages of HDA

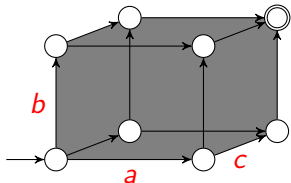


$$L_1 = \{abc, acb, bac, bca, cab, cba\}$$



$$L_2 = \left\{ \begin{pmatrix} a \\ b \rightarrow c \end{pmatrix}, \begin{pmatrix} a \\ c \rightarrow b \end{pmatrix}, \begin{pmatrix} b \\ a \rightarrow c \end{pmatrix}, \right. \\ \left. \begin{pmatrix} b \\ c \rightarrow a \end{pmatrix}, \begin{pmatrix} c \\ a \rightarrow b \end{pmatrix}, \begin{pmatrix} c \\ b \rightarrow a \end{pmatrix} \right\} \cup L_1$$

sets of pomsets



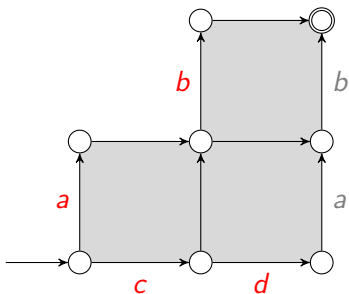
$$L_3 = \left\{ \begin{pmatrix} a \\ b \\ c \end{pmatrix} \right\} \cup L_2$$

Pomsets

A (finite) **pomset** (“partially ordered multiset”) (P, \leq, ℓ) :

- a finite partially ordered set (P, \leq)
- with labeling $\ell : P \rightarrow \Sigma$
- (AKA **labeled partial order**)
- [Lamport 1978]

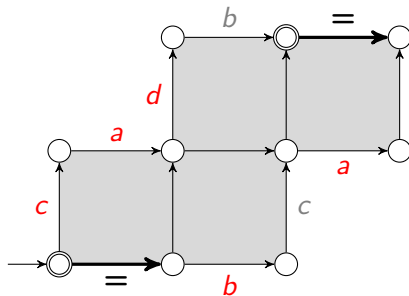
Example



$$\left(\begin{array}{l} a \rightarrow b \\ c \rightarrow d \end{array} \right)$$

- not series-parallel!

A loop



$$\left(\begin{array}{l} a \rightarrow b \\ c \rightarrow d \end{array} \right)$$

$$\left(\begin{array}{l} a \rightarrow b \rightarrow a \rightarrow b \\ c \rightarrow d \rightarrow c \rightarrow d \end{array} \right)$$

$$\left(\begin{array}{l} a \rightarrow b \rightarrow a \rightarrow b \rightarrow a \rightarrow b \\ c \rightarrow d \rightarrow c \rightarrow d \rightarrow c \rightarrow d \end{array} \right)$$

...

Are all pomsets generated by HDA?

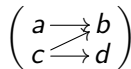
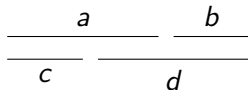
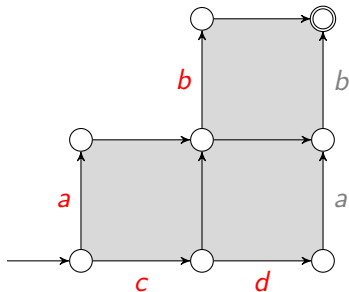
No, only (labeled) **interval orders**

- Poset (P, \leq) is an interval order iff it does not contain (\implies)
 - (iff it is “2+2-free”)
- iff it has an **interval representation**:
 - a set $I = \{[l_i, r_i]\}$ of real intervals
 - with order $[l_i, r_i] \preceq [l_j, r_j]$ iff $r_i \leq l_j$
 - and an order isomorphism $(P, \leq) \leftrightarrow (I, \preceq)$
- [Fishburn 1970]

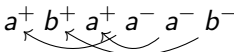
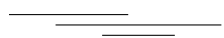
Are all pomsets generated by HDA?

No, only (labeled) **interval orders**

- Poset (P, \leq) is an interval order iff it does not contain (\implies)
 - (iff it is “2+2-free”)
- iff it has an **interval representation**:
 - a set $I = \{[l_i, r_i]\}$ of real intervals
 - with order $[l_i, r_i] \preceq [l_j, r_j]$ iff $r_i \leq l_j$
 - and an order isomorphism $(P, \leq) \leftrightarrow (I, \preceq)$



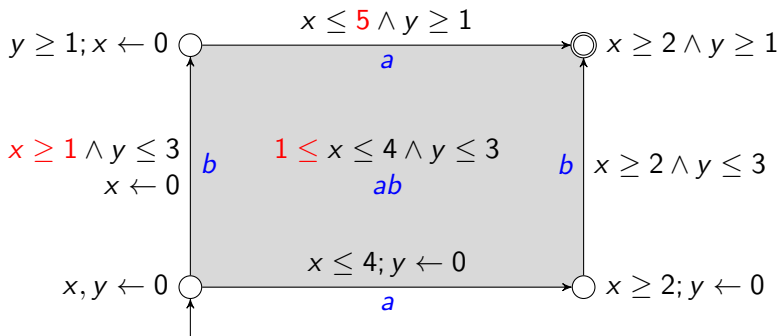
Interval orders vs ST-traces

- An **ST-trace**: $a^+ b^+ a^+ a^- a^- b^-$ [van Glabbeek 1990]
 
- as intervals: 
- **Lemma**: ST-traces up to the equivalence generated by $a^+ b^+ \sim b^+ a^+$ and $a^- b^- \sim b^- a^-$ are in bijection with interval orders.

Languages of HDTA

- Timed automata generate **timed words** (w, t) :
 - $w = w_1 \dots w_n \in \Sigma^*$
 - $t = (t_1, \dots, t_n) \in \mathbb{R}_{\geq 0}^n$ increasing sequence of **time stamps**
 - example: $\left(\begin{array}{cccc} a & c & a & a \\ .7 & 1.1 & 1.1 & 1.7 \end{array} \right)$
- Higher-dimensional automata generate **labeled interval orders** (I, ℓ) :
 - $I = \{[l_i, r_i]\} \subseteq \mathbb{N} \times \mathbb{N}$ finite set of intervals ($l_i \leq r_i$)
 - $\ell : I \rightarrow \Sigma$
 - example: $\left(\begin{array}{cc} a & b \\ \hline c & a \end{array} \right)$
- Proposal: HDTA generate **timed interval orders** (I, ℓ) :
 - $I = \{[l_i, r_i]\} \subseteq \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}$ finite **multiset** of **real** intervals ($l_i \leq r_i$)
 - $\ell : I \rightarrow \Sigma$

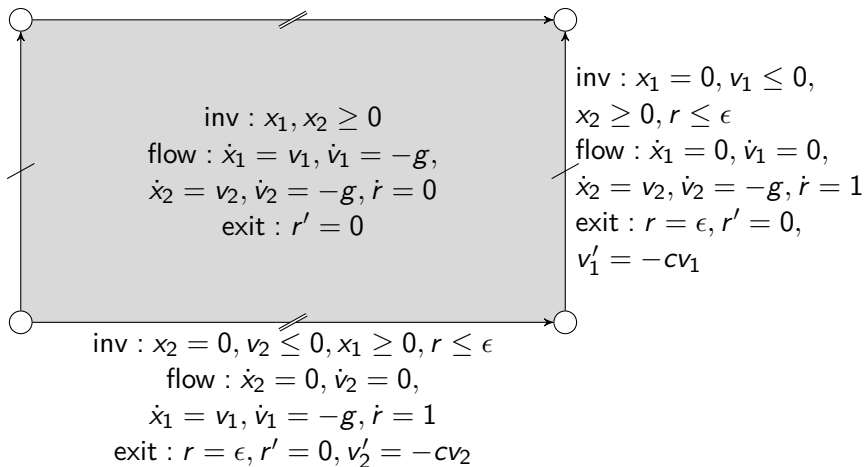
Example



$$L(A) = \left\{ \left\{ [l_1, r_1]^a, [l_2, r_2]^b \right\} \mid \begin{cases} 1 \leq r_2 - l_2 \leq 3 \\ 2 \leq r_1 - l_1 \leq \begin{cases} 4 & \text{if } r_1 < r_2 \\ 5 & \text{if } r_1 \geq r_2 \end{cases} \\ l_2 \geq l_1 + 1 \end{cases} \right. \right\}$$

Higher-Dimensional Hybrid Automata

Two independently bouncing balls (with temporal regularization):



International Workshop on
Methods and Tools for Distributed Hybrid Systems
Aalborg, Denmark, 26 August 2017
Associated with **MFCS 2017**



About

The purpose of DHS is to connect researchers working in *real-time systems*, *hybrid systems*, *control theory*, *distributed computing*, and *concurrency*, in order to advance the subject of **distributed hybrid systems**.

Distributed hybrid systems, or distributed *cyber-physical systems*, are abundant. Many of them are safety-critical, but ensuring their correct functioning is very difficult. We believe that new techniques are needed for the analysis and validation of DHS. More precisely, we believe that convergence and interaction of methods and tools from different areas of computer science, engineering, and mathematics is needed in order to advance the subject.

This first edition of the DHS workshop aims at gathering researchers which work in the above areas in order to facilitate collaboration and discuss how the subject may advance.

Invited Talks

- **Alessandro Abate**, Oxford University, United Kingdom
Hybrid models for heterogeneous populations of photovoltaic panels on the grid
- **Martin Fränzle**, Carl von Ossietzky Universität Oldenburg, Germany
Indecision and delays are the parents of failures
- **Kim G. Larsen**, Aalborg Universitet, Denmark
Synthesis and optimization for cyber-physical systems
- **Martin Raussen**, Aalborg Universitet, Denmark
Topological models for spaces of executions in HDA
- **Rafael Wisniewski**, Aalborg Universitet, Denmark
Safety verification of stochastic hybrid systems

Second International Workshop on Methods and Tools for Distributed Hybrid Systems Palaiseau, France, 4 July 2018



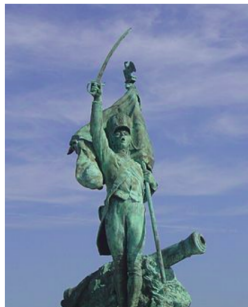
About

The purpose of DHS is to connect researchers working in *real-time systems*, *hybrid systems*, *control theory*, *distributed computing*, and *concurrency theory*, in order to advance the subject of **distributed hybrid systems**.

The **first DHS workshop** was held in Aalborg, Denmark, in August 2017, featuring invited talks by **Alessandro Abate**, **Martin Fränzle**, **Kim G. Larsen**, **Martin Raussen**, and **Rafael Wisniewski**. This second edition aims to continue the conversation.

Invited Talks

- **Luc Jaulin**, ENSTA Bretagne, Brest, France
Distributed localization and control of underwater robots
- **Thao Dang**, Verimag, Grenoble, France
Invariance and stability verification of hybrid systems
- **Lisbeth Fajstrup**, Aalborg University, Denmark
Symmetries in the PV-model and of directed invariants
- **Emmanuel Ledinot**, Dassault Aviation, France
Towards CPS certification reformation: call for effective foundations
- **André Platzer**, Carnegie Mellon University, United States
Logic of distributed hybrid systems



Third International Workshop on
Methods and Tools for Distributed Hybrid Systems
Amsterdam, The Netherlands, 26 August 2019
Associated with CONCUR 2019



DHS 2019 will take place in Amsterdam, the day before CONCUR, 26 August 2019.

[Download workshop poster](#) (or in A4 format)

About

The purpose of DHS is to connect researchers working in *real-time systems*, *hybrid systems*, *control theory*, *distributed computing*, and *concurrency theory*, in order to advance the subject of **distributed hybrid systems**.

Distributed hybrid systems, or distributed *cyber-physical systems*, are abundant. Many of them are safety-critical, but ensuring their correct functioning is very difficult. We believe that new techniques are needed for the analysis and validation of DHS. More precisely, we believe that convergence and interaction of methods and tools from different areas of computer science, engineering, and mathematics is needed in order to advance the subject.

The **first DHS workshop** was held in Aalborg, Denmark, in August 2017 and associated with **MFCS**. It featured invited talks by **Alessandro Abate**, **Martin Fränzle**, **Kim G. Larsen**, **Martin Raussen**, and **Rafael Wisniewski**. The **second edition** was held in Palaiseau, France, in July 2018, with invited talks by **Luc Jaulin**, **Thao Dang**, **Lisbeth Fajstrup**, **Emmanuel Ledinot**, and **André Platzer**.


This third edition of DHS will add **distributed robotics** as a special theme. This emerging field is



My Assigned

Filters

New Submission

- | | | | | | |
|-----|--|--|-------------------------|-------------------|----------------|
| 119 | Tran et al.
Real-Time Verification for Distributed Cyber-Physical Systems |  1 | Production | View | ▼ |
| 121 | Kröger et al.
Bayesian Hybrid Automata: A Formal Model of Justified Belief in Interacting Hybr... | | Production | View | ▼ |
| 118 | Nejati et al.
From Dissipativity Theory to Compositional Construction of Control Barrier Certif... | | Production | View | ▼ |
| 117 | Adimoolam et al.
Safety Verification of Networked Control Systems by Complex Zonotopes | | Production | View | ▼ |
| 114 | Fahrenberg
Higher-Dimensional Timed and Hybrid Automata | | Production | View | ▼ |
| 112 | Courtieu et al.
Swarms of Mobile Robots: towards Versability with Safety | | Production | View | ▼ |
| 110 | Kamburjan et al.
A Hybrid Programming Language for Formal Modeling and Verification of Hybri... | | Production | View | ▼ |

Conclusion

- Higher-dimensional timed automata: a nice formalism for real-time concurrency?
- Also, higher-dimensional hybrid automata
- For HDTA verification, zones
- Tensor product for parallel composition
- “Partial-order reduction built in”
- Actions should take time!?
- Distributed Hybrid Systems?