Invariant-Based Verification and Synthesis for Hybrid Systems

Naijun Zhan

State Key Lab. of Comp. Sci., Institute of Software, Chinese Academy of Sciences

Summer School on Symbolic Computation

Nanning, Jul.16-22, 2017

Outline

1. Background

2. Talk1: Preliminaries
   - Polynomials and Polynomial Ideals
   - First-order Theory of Reals
   - Continuous Dynamical Systems
   - Hybrid Automata

3. Talk2: Computing Invariants for Hybrid Systems
   - Generating Continuous Invariants in Simple Case
   - Generating Continuous Invariants in General Case
   - Generating Semi-algebraic Global Invariants
   - Abstraction of Elementary Hybrid Systems by Variable Transformation
   - An Industrial Case Study: Soft Landing

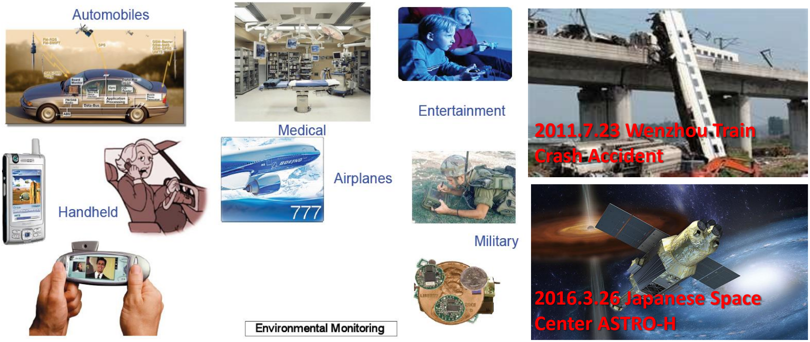
4. Talk3: Controller Synthesis
   - Controller Synthesis with Safety
   - Controller Synthesis with Safety and Optimality
   - An Industrial Case Study: The Oil Pump Control Problem

5. Conclusions

Hybrid systems

Hybrid systems exhibit combinations of discrete jumps and continuous evolution.

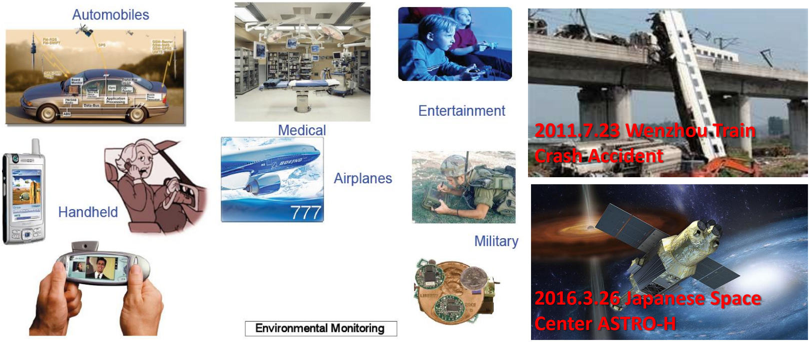
Examples



Hybrid systems

Hybrid systems exhibit combinations of discrete jumps and continuous evolution.

Examples



Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, ...
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, ...
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, . . .
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, ...
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, . . .
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, ...
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, ...
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, . . .
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems

- **Modelling:**
  - To establish a model for the system to be developed with precise mathematical semantics
  - Have to consider: **concurrency, deterministic vs nondeterministic, continuous vs discrete, communication, static vs dynamic (mobility, adaptability), qualitative vs quantitative (predicability), real-time**, ...
- **Simulation:**
  - To obtain a possible execution of the model upto a finite time horizon using numerical methods
  - Well accepted in industrial practice
- **Verification:**
  - Using mathematical approach to prove if a model satisfies the desired properties (specification)
  - Main methods include: **model-checking, theorem proving, abstract interpretation**

Issues of hybrid systems (Cont'd)

- **Synthesis:** The process of computing an implementation (the "how") from a specification of the desired behavior and performance (the "what") and the assumptions on the environment (the "where")
- **Qualitative issues:**
  - Total absence of undesirable behavior is an overly ambitious goal, being economically unattainable or even technically impossible due to
    - uncontrollable environment influences;
    - unavoidable manufacturing tolerance;
    - component breakdown, etc.
  - The existing qualitative safety analysis methods for hybrid systems have to be complemented quantitative methods, quantifying the likelihood of residual errors or the related performance figures in systems subject to uncertain, stochastic behavior as well as noise.
- **Other issues:** stability, controllability, observability, . . .

Issues of hybrid systems (Cont'd)

- **Synthesis:** The process of computing an implementation (the "how") from a specification of the desired behavior and performance (the "what") and the assumptions on the environment (the "where")
- **Qualitative issues:**
  - Total absence of undesirable behavior is an overly ambitious goal, being economically unattainable or even technically impossible due to
    - uncontrollable environment influences;
    - unavoidable manufacturing tolerance;
    - component breakdown, etc.
  - The existing qualitative safety analysis methods for hybrid systems have to be complemented quantitative methods, quantifying the likelihood of residual errors or the related performance figures in systems subject to uncertain, stochastic behavior as well as noise.
- **Other issues:** stability, controllability, observability, ...

Issues of hybrid systems (Cont'd)

- **Synthesis:** The process of computing an implementation (the "how") from a specification of the desired behavior and performance (the "what") and the assumptions on the environment (the "where")
- **Qualitative issues:**
  - Total absence of undesirable behavior is an overly ambitious goal, being economically unattainable or even technically impossible due to
    - uncontrollable environment influences;
    - unavoidable manufacturing tolerance;
    - component breakdown, etc.
  - The existing qualitative safety analysis methods for hybrid systems have to be complemented quantitative methods, quantifying the likelihood of residual errors or the related performance figures in systems subject to uncertain, stochastic behavior as well as noise.
- **Other issues:** stability, controllability, observability, ...

Issues of hybrid systems (Cont'd)

- **Synthesis:** The process of computing an implementation (the "how") from a specification of the desired behavior and performance (the "what") and the assumptions on the environment (the "where")
- **Qualitative issues:**
  - Total absence of undesirable behavior is an overly ambitious goal, being economically unattainable or even technically impossible due to
    - uncontrollable environment influences;
    - unavoidable manufacturing tolerance;
    - component breakdown, etc.
  - The existing qualitative safety analysis methods for hybrid systems have to be complemented quantitative methods, quantifying the likelihood of residual errors or the related performance figures in systems subject to uncertain, stochastic behavior as well as noise.
- **Other issues:** stability, controllability, observability, ...

Issues of hybrid systems (Cont'd)

- **Synthesis:** The process of computing an implementation (the "how") from a specification of the desired behavior and performance (the "what") and the assumptions on the environment (the "where")
- **Qualitative issues:**
  - Total absence of undesirable behavior is an overly ambitious goal, being economically unattainable or even technically impossible due to
    - uncontrollable environment influences;
    - unavoidable manufacturing tolerance;
    - component breakdown, etc.
  - The existing qualitative safety analysis methods for hybrid systems have to be complemented quantitative methods, quantifying the likelihood of residual errors or the related performance figures in systems subject to uncertain, stochastic behavior as well as noise.
- **Other issues:** stability, controllability, observability, ...

Issues of hybrid systems (Cont'd)

- **Synthesis:** The process of computing an implementation (the "how") from a specification of the desired behavior and performance (the "what") and the assumptions on the environment (the "where")
- **Qualitative issues:**
  - Total absence of undesirable behavior is an overly ambitious goal, being economically unattainable or even technically impossible due to
    - uncontrollable environment influences;
    - unavoidable manufacturing tolerance;
    - component breakdown, etc.
  - The existing qualitative safety analysis methods for hybrid systems have to be complemented quantitative methods, quantifying the likelihood of residual errors or the related performance figures in systems subject to uncertain, stochastic behavior as well as noise.
- **Other issues:** stability, controllability, observability, ...

Issues of hybrid systems (Cont'd)

- **Synthesis:** The process of computing an implementation (the "how") from a specification of the desired behavior and performance (the "what") and the assumptions on the environment (the "where")
- **Qualitative issues:**
  - Total absence of undesirable behavior is an overly ambitious goal, being economically unattainable or even technically impossible due to
    - uncontrollable environment influences;
    - unavoidable manufacturing tolerance;
    - component breakdown, etc.
  - The existing qualitative safety analysis methods for hybrid systems have to be complemented quantitative methods, quantifying the likelihood of residual errors or the related performance figures in systems subject to uncertain, stochastic behavior as well as noise.
- **Other issues:** stability, controllability, observability, ...

Issues of hybrid systems (Cont'd)

- **Synthesis:** The process of computing an implementation (the "how") from a specification of the desired behavior and performance (the "what") and the assumptions on the environment (the "where")
- **Qualitative issues:**
  - Total absence of undesirable behavior is an overly ambitious goal, being economically unattainable or even technically impossible due to
    - uncontrollable environment influences;
    - unavoidable manufacturing tolerance;
    - component breakdown, etc.
  - The existing qualitative safety analysis methods for hybrid systems have to be complemented quantitative methods, quantifying the likelihood of residual errors or the related performance figures in systems subject to uncertain, stochastic behavior as well as noise.
- **Other issues:** **stability, controllability, observability,** ...

Related Work

Automata-based techniques

- **Modeling:** **Phase transition systems** [Manna&Pnueli,1993] **Hybrid automata** [Alur et al, 1995]
  - **Advantages:** intuitive, easy to model the behavior of systems, the basis for model-checking.
  - **Disadvantages:** lacks of structured information, not easy to model complex system.
- **Verification by computing reachable set:** **model-checking** [Alur et al, 1995], **decision procedure** [LPY, 2001],
  - **Basic idea:** partitioning infinite state space into finite many equivalent classes according to the solution of ODEs, or representing by O-minimal structures
  - **Advantages:** automatic
  - **Disadvantages:** cannot scale up
  - **Focuses:** **symbolic computation, abstraction, approximation**

Related Work

Automata-based techniques

- **Modeling:** **Phase transition systems** [Manna&Pnueli,1993] **Hybrid automata** [Alur et al, 1995]
  - **Advantages:** intuitive, easy to model the behavior of systems, the basis for model-checking.
  - **Disadvantages:** lacks of structured information, not easy to model complex system.
- **Verification by computing reachable set:** **model-checking** [Alur et al, 1995], **decision procedure** [LPY, 2001],
  - **Basic idea:** partitioning infinite state space into finite many equivalent classes according to the solution of ODEs, or representing by O-minimal structures
  - **Advantages:** automatic
  - **Disadvantages:** cannot scale up
  - **Focuses:** **symbolic computation, abstraction, approximation**

Related Work

Automata-based techniques

- **Modeling: Phase transition systems** [Manna&Pnueli,1993] **Hybrid automata** [Alur et al, 1995]
  - **Advantages:** intuitive, easy to model the behavior of systems, the basis for model-checking.
  - **Disadvantages:** lacks of structured information, not easy to model complex system.
- **Verification by computing reachable set: model-checking** [Alur et al, 1995], **decision procedure** [LPY, 2001],
  - **Basic idea:** partitioning infinite state space into finite many equivalent classes according to the solution of ODEs, or representing by O-minimal structures
  - **Advantages:** automatic
  - **Disadvantages:** cannot scale up
  - **Focuses: symbolic computation, abstraction, approximation**

Related Work

Automata-based techniques

- **Modeling:** **Phase transition systems** [Manna&Pnueli,1993] **Hybrid automata** [Alur et al, 1995]
  - **Advantages:** intuitive, easy to model the behavior of systems, the basis for model-checking.
  - **Disadvantages:** lacks of structured information, not easy to model complex system.
- **Verification by computing reachable set:** **model-checking** [Alur et al, 1995], **decision procedure** [LPY, 2001],
  - **Basic idea:** partitioning infinite state space into finite many equivalent classes according to the solution of ODEs, or representing by O-minimal structures
  - **Advantages:** automatic
  - **Disadvantages:** cannot scale up
  - **Focuses:** **symbolic computation, abstraction, approximation**

Related Work

Automata-based techniques

- **Modeling: Phase transition systems** [Manna&Pnueli,1993] **Hybrid automata** [Alur et al, 1995]
  - **Advantages:** intuitive, easy to model the behavior of systems, the basis for model-checking.
  - **Disadvantages:** lacks of structured information, not easy to model complex system.
- **Verification by computing reachable set: model-checking** [Alur et al, 1995], **decision procedure** [LPY, 2001],
  - **Basic idea:** partitioning infinite state space into finite many equivalent classes according to the solution of ODEs, or representing by O-minimal structures
  - **Advantages:** automatic
  - **Disadvantages:** cannot scale up
  - **Focuses:** **symbolic computation, abstraction, approximation**

Related Work

Automata-based techniques

- **Modeling: Phase transition systems** [Manna&Pnueli,1993] **Hybrid automata** [Alur et al, 1995]
  - **Advantages:** intuitive, easy to model the behavior of systems, the basis for model-checking.
  - **Disadvantages:** lacks of structured information, not easy to model complex system.
- **Verification by computing reachable set: model-checking** [Alur et al, 1995], **decision procedure** [LPY, 2001],
  - **Basic idea:** partitioning infinite state space into finite many equivalent classes according to the solution of ODEs, or representing by O-minimal structures
  - **Advantages:** automatic
  - **Disadvantages:** cannot scale up
  - **Focuses:** symbolic computation, abstraction, approximation

Related Work

Automata-based techniques

- **Modeling: Phase transition systems** [Manna&Pnueli,1993] **Hybrid automata** [Alur et al, 1995]
  - **Advantages:** intuitive, easy to model the behavior of systems, the basis for model-checking.
  - **Disadvantages:** lacks of structured information, not easy to model complex system.
- **Verification by computing reachable set: model-checking** [Alur et al, 1995], **decision procedure** [LPY, 2001],
  - **Basic idea:** partitioning infinite state space into finite many equivalent classes according to the solution of ODEs, or representing by O-minimal structures
  - **Advantages:** automatic
  - **Disadvantages:** cannot scale up
  - **Focuses:** symbolic computation, abstraction, approximation

Related Work

Automata-based techniques

- **Modeling: Phase transition systems** [Manna&Pnueli,1993] **Hybrid automata** [Alur et al, 1995]
  - **Advantages:** intuitive, easy to model the behavior of systems, the basis for model-checking.
  - **Disadvantages:** lacks of structured information, not easy to model complex system.
- **Verification by computing reachable set: model-checking** [Alur et al, 1995], **decision procedure** [LPY, 2001],
  - **Basic idea:** partitioning infinite state space into finite many equivalent classes according to the solution of ODEs, or representing by O-minimal structures
  - **Advantages:** automatic
  - **Disadvantages:** cannot scale up
  - **Focuses: symbolic computation, abstraction, approximation**

Related Work (Cont'd)

Compositional modeling approaches

- Modeling environment: SHIFT [DGV 1996]
- Hierarchical modeling: PTOLEMY [Lee et al 2003]
- Modular modeling: I/O hybrid automata [Lynch et al 1996], hybrid modules [Alur et al 2003], CHARON [Alur&Henzinger 1997]
- Algebraic approach: Hybrid CSP [He 1994, Zhou et al 1995]

Problem

It lacks of verification techniques for these compositional modelling techniques

Related Work (Cont'd)

Compositional modeling approaches

- **Modeling environment: SHIFT** [DGV 1996]
- **Hierarchical modeling: PTOLEMY** [Lee et al 2003]
- **Modular modeling: I/O hybrid automata** [Lynch et al 1996], **hybrid modules** [Alur et al 2003], **CHARON** [Alur&Henzinger 1997]
- **Algebraic approach: Hybrid CSP** [He 1994, Zhou et al 1995]

Problem

It lacks of verification techniques for these compositional modelling techniques

Related Work (Cont'd)

Compositional modeling approaches

- **Modeling environment:** **SHIFT** [DGV 1996]
- **Hierarchical modeling:** **PTOLEMY** [Lee et al 2003]
- **Modular modeling:** **I/O hybrid automata** [Lynch et al 1996], **hybrid modules** [Alur et al 2003], **CHARON** [Alur&Henzinger 1997]
- **Algebraic approach:** **Hybrid CSP** [He 1994, Zhou et al 1995]

Problem

It lacks of verification techniques for these compositional modelling techniques

Related Work (Cont'd)

Compositional modeling approaches

- **Modeling environment: SHIFT** [DGV 1996]
- **Hierarchical modeling: PTOLEMY** [Lee et al 2003]
- **Modular modeling: I/O hybrid automata** [Lynch et al 1996], **hybrid modules** [Alur et al 2003], **CHARON** [Alur&Henzinger 1997]
- **Algebraic approach: Hybrid CSP** [He 1994, Zhou et al 1995]

Problem

It lacks of verification techniques for these compositional modelling techniques

Related Work (Cont'd)

Compositional modeling approaches

- **Modeling environment:** **SHIFT** [DGV 1996]
- **Hierarchical modeling:** **PTOLEMY** [Lee et al 2003]
- **Modular modeling:** **I/O hybrid automata** [Lynch et al 1996], **hybrid modules** [Alur et al 2003], **CHARON** [Alur&Henzinger 1997]
- **Algebraic approach:** **Hybrid CSP** [He 1994, Zhou et al 1995]

Problem

It lacks of verification techniques for these compositional modelling techniques

Related Work (Cont'd)

Compositional modeling approaches

- **Modeling environment: SHIFT** [DGV 1996]
- **Hierarchical modeling: PTOLEMY** [Lee et al 2003]
- **Modular modeling: I/O hybrid automata** [Lynch et al 1996], **hybrid modules** [Alur et al 2003], **CHARON** [Alur&Henzinger 1997]
- **Algebraic approach: Hybrid CSP** [He 1994, Zhou et al 1995]

Problem

It lacks of verification techniques for these compositional modelling techniques

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - A compositional modelling laguage
  - A Hoare logic-like specification logic
  - Invariant generation
- **Well-known compositional modelling languages:** hybrid programs [Platzer&Clarke 2008], HCSP [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics:** DDL [Platzer2008], DADL [Platzer2010], EDC [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - A compositional modelling laguage
  - A Hoare logic-like specification logic
  - Invariant generation
- **Well-known compositional modelling languages:** **hybrid programs** [Platzer&Clarke 2008], **HCSP** [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics:** **DDL** [Platzer2008], **DADL** [Platzer2010], **EDC** [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - **A compositional modelling laguage**
  - **A Hoare logic-like specification logic**
  - **Invariant generation**
- **Well-known compositional modelling languages: hybrid programs** [Platzer&Clarke 2008], **HCSP** [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics: DDL** [Platzer2008], **DADL** [Platzer2010], **EDC** [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - **A compositional modelling laguage**
  - **A Hoare logic-like specification logic**
  - **Invariant generation**
- **Well-known compositional modelling languages:** **hybrid programs** [Platzer&Clarke 2008], **HCSP** [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics:** **DDL** [Platzer2008], **DADL** [Platzer2010], **EDC** [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - **A compositional modelling laguage**
  - **A Hoare logic-like specification logic**
  - **Invariant generation**
- **Well-known compositional modelling languages:** **hybrid programs** [Platzer&Clarke 2008], **HCSP** [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics:** **DDL** [Platzer2008], **DADL** [Platzer2010], **EDC** [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - **A compositional modelling laguage**
  - **A Hoare logic-like specification logic**
  - **Invariant generation**
- **Well-known compositional modelling languages: hybrid programs** [Platzer&Clarke 2008], **HCSP** [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics: DDL** [Platzer2008], **DADL** [Platzer2010], **EDC** [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - **A compositional modelling laguage**
  - **A Hoare logic-like specification logic**
  - **Invariant generation**
- **Well-known compositional modelling languages: hybrid programs** [Platzer&Clarke 2008], **HCSP** [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics: DDL** [Platzer2008], **DADL** [Platzer2010], **EDC** [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - **A compositional modelling laguage**
  - **A Hoare logic-like specification logic**
  - **Invariant generation**
- **Well-known compositional modelling languages: hybrid programs** [Platzer&Clarke 2008], **HCSP** [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics: DDL** [Platzer2008], **DADL** [Platzer2010], **EDC** [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

Related work (Cont'd)

Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
    - **A compositional modelling laguage**
    - **A Hoare logic-like specification logic**
    - **Invariant generation**
- **Well-known compositional modelling languages: hybrid programs** [Platzer&Clarke 2008], **HCSP** [He 1994, Zhou et al 1995], ...
- **Hybrid specification logics: DDL** [Platzer2008], **DADL** [Platzer2010], **EDC** [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation

A grand challenge

How to design correct safety-critical hybrid-systems is a grand challenge in computer science and control theory

Our goal

to establish a systematic approach to formal design, analysis and verification of hybrid systems

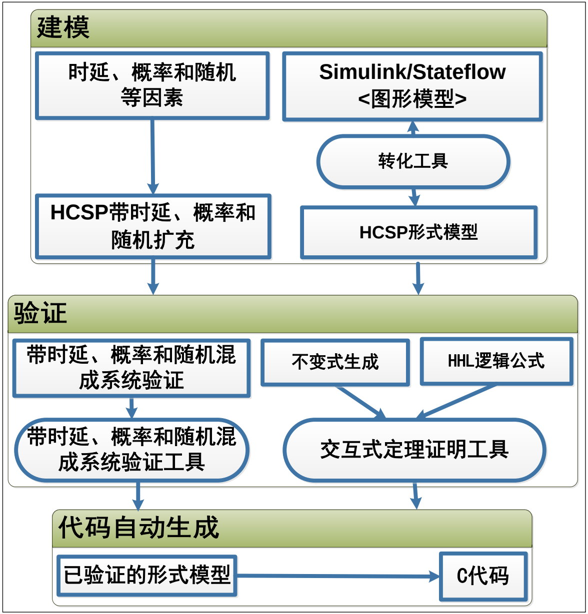
A grand challenge

How to design correct safety-critical hybrid-systems is a grand challenge in computer science and control theory

Our goal

to establish a systematic approach to formal design, analysis and verification of hybrid systems

Overview of Our Approach



Schedule and References

Schedule

- Talk 1: Preliminaries
- Talk 2: Differential invariant generation
- Talk 3: Controller synthesis

References

- N. Zhan, S. Wang and H. Zhao (2013): Formal Verification of Simulink/Stateflow Diagrams: A Deductive Way. Springer, 2016.
- N. Zhan, S. Wang and H. Zhao (2013): Formal Modelling, Analysis and Verification of Hybrid Systems. In the Theories of Programming, LNCS 8050.
- J. Liu, J. Lv, Z. Quan, N. Zhan, H. Zhao, C. Zhou and L. Zou (2010): A calculus for HCSP. Proc. of APLAS 2010, LNCS 6461.
- J. Liu, N. Zhan and H. Zhao (2011): Computing semi-algebraic invariants for polynomial dynamical systems. Proc. of EMSOFT'11.
- H. Zhao, N. Zhan, D. Kapur, and K.G. Larsen (2012): A "hybrid" approach for synthesizing optimal controllers of hybrid systems: A case study of the oil pump industrial example. Proc. of FM 2012, LNCS 7436.
- H. Zhao, M. Yang, N. Zhan, B. Gu, L. Zou and Y. Chen (2014): Formal verification of a descent guidance control program of a lunar lander. Proc. of FM 2014, LNCS 8442.
- L. Zou, M. Fraenzle, N. Zhan and P. Mosaad (2015): Automatic stability and safety verification for delay differential equations. Proc. of CAV 2015, LNCS.

Outline

Outline

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^{n} \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
  - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
  - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
  - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $u_{\boldsymbol{\alpha}} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
  - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
  - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
  - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
  - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
  - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $u_{\boldsymbol{\alpha}} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
  - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
  - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^{n} \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
  - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
  - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
  - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $u_{\boldsymbol{\alpha}} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
  - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
  - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
    - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
    - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
    - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $u_{\boldsymbol{\alpha}} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
    - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
    - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^{n} \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
  - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
  - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
  - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $u_{\boldsymbol{\alpha}} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
  - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
  - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^{n} \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
  - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
  - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
  - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\alpha} u_{\alpha} \mathbf{x}^{\alpha}$, where $u_{\alpha} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
  - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
  - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
    - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
    - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
    - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $u_{\boldsymbol{\alpha}} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
    - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
    - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^{n} \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
  - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
  - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
  - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $u_{\boldsymbol{\alpha}} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
  - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
  - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

- Let $\mathbb{K}$ be a number field, which can be either $\mathbb{Q}$ or $\mathbb{R}$.
- A ***monomial*** in n variables $x_1, x_2, \ldots, x_n$ (or briefly $\mathbf{x}$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $\mathbf{x}^{\boldsymbol{\alpha}}$, where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^{n} \alpha_i$ is called the ***degree*** of $\mathbf{x}^{\boldsymbol{\alpha}}$.
- A *polynomial* $p(\mathbf{x})$ in $\mathbf{x}$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where all $c_{\boldsymbol{\alpha}} \in \mathbb{K}$.
  - The ***degree*** $\deg(p)$ of p is the maximal degree of its component monomials.
  - A polynomial in $x_1, x_2, \ldots, x_n$ with degree d has at most $\binom{n+d}{d}$ many monomials.
  - The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $\mathbb{K}$ form a *polynomial ring* $\mathbb{K}[\mathbf{x}]$.
- A ***parametric polynomial*** is of the form $\sum_{\boldsymbol{\alpha}} u_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $u_{\boldsymbol{\alpha}} \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(\mathbf{u}, \mathbf{x})$ in $\mathbb{R}[\mathbf{u}, \mathbf{x}]$.
  - A parametric polynomial with degree d (in $\mathbf{x}$) has at most $\binom{n+d}{d}$ many indeterminates.
  - For any $\mathbf{u}_0 \in \mathbb{R}^w$, $p_{\mathbf{u}_0}(\mathbf{x}) \in \mathbb{R}[\mathbf{x}]$ obtained by substituting $\mathbf{u}_0$ for $\mathbf{u}$ in $p(\mathbf{u}, \mathbf{x})$ is an *instantiation* of $p(\mathbf{u}, \mathbf{x})$.

Polynomial ideal

Polynomial ideal

- A subset $I \subseteq \mathbb{K}[\mathbf{x}]$ is called an **ideal** if the following conditions are satisfied:
  1. $0 \in I$;
  2. If p, $g \in I$, then $p + g \in I$;
  3. If $p \in I$ and $h \in \mathbb{K}[\mathbf{x}]$, then $hp \in I$.
- Let $g_1, g_2, \ldots, g_s \in \mathbb{K}[\mathbf{x}]$, then $\langle g_1, g_2, \ldots, g_s \rangle \hat{=} \{\sum_{i=1}^{s} h_i g_i : h_1, h_2, \ldots, h_s \in \mathbb{K}[\mathbf{x}]\}$ is an ideal *generated* by $g_1, g_2, \ldots, g_s$.
- If $I = \langle g_1, g_2, \ldots, g_s \rangle$, then $\{g_1, g_2, \ldots, g_s\}$ is called a ***basis*** of I.

Polynomial ideal

Polynomial ideal

- A subset $I \subseteq \mathbb{K}[\mathbf{x}]$ is called an **ideal** if the following conditions are satisfied:
  1. $0 \in I$;
  2. If p, $g \in I$, then $p + g \in I$;
  3. If $p \in I$ and $h \in \mathbb{K}[\mathbf{x}]$, then $hp \in I$.
- Let $g_1, g_2, \ldots, g_s \in \mathbb{K}[\mathbf{x}]$, then $\langle g_1, g_2, \ldots, g_s \rangle \widehat{=} \{\sum_{i=1}^{s} h_i g_i : h_1, h_2, \ldots, h_s \in \mathbb{K}[\mathbf{x}]\}$ is an ideal *generated* by $g_1, g_2, \ldots, g_s$.
- If $I = \langle g_1, g_2, \ldots, g_s \rangle$, then $\{g_1, g_2, \ldots, g_s\}$ is called a ***basis*** of I.

Polynomial ideal

Polynomial ideal

- A subset $I \subseteq \mathbb{K}[\mathbf{x}]$ is called an **ideal** if the following conditions are satisfied:
  1. $0 \in I$;
  2. If p, $g \in I$, then $p + g \in I$;
  3. If $p \in I$ and $h \in \mathbb{K}[\mathbf{x}]$, then $hp \in I$.
- Let $g_1, g_2, \ldots, g_s \in \mathbb{K}[\mathbf{x}]$, then $\langle g_1, g_2, \ldots, g_s \rangle \widehat{=} \{\sum_{i=1}^{s} h_i g_i : h_1, h_2, \ldots, h_s \in \mathbb{K}[\mathbf{x}]\}$ is an ideal *generated* by $g_1, g_2, \ldots, g_s$.
- If $I = \langle g_1, g_2, \ldots, g_s \rangle$, then $\{g_1, g_2, \ldots, g_s\}$ is called a ***basis*** of I.

Hilbert Basis Theorem

Every ideal $I \subseteq \mathbb{K}[\mathbf{x}]$ has a finite basis, that is, $I = \langle g_1, g_2, \ldots, g_s \rangle$ for some $g_1, g_2, \ldots, g_s \in \mathbb{K}[\mathbf{x}]$.

Ascending Chain Theorem

For any ascending chain of ideals $I_1 \subseteq I_2 \subseteq \cdots \subseteq I_k \subseteq \cdots$ in $\mathbb{K}[\mathbf{x}]$, there exists an $N \in \mathbb{N}$ such that $I_k = I_N$ for any $k \geq N$.

Outline

First-order Theory $T(\mathbb{R})$ of Reals

Syntax

- The language of $T(\mathbb{R})$ consists of
  - variables: $x, y, z, \ldots, x_1, x_2, \ldots$, which are interpreted over $\mathbb{R}$;
  - relational symbols: $>, <, \geq, \leq, =, \neq$;
  - Boolean connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow, \ldots$; and
  - quantifiers: $\forall, \exists$.
- A *term* of $T(\mathbb{R})$ over a finite set of variables $\{x_1, x_2, \ldots, x_n\}$ is a polynomial $p \in \mathbb{R}[x_1, x_2, \ldots, x_n]$.
- An *atomic formula* of $T(\mathbb{R})$ is of the form $p \rhd 0$, where $\rhd$ is any relational symbol.
- A *quantifier-free formula* (QFF) of $T(\mathbb{R})$ is a Boolean combination of atomic formulas.
- A generic formula of $T(\mathbb{R})$ is built up from atomic formulas using Boolean connectives as well as quantifiers.

First-order Theory $T(\mathbb{R})$ of Reals

Syntax

- The language of $T(\mathbb{R})$ consists of
  - variables: $x, y, z, \ldots, x_1, x_2, \ldots$, which are interpreted over $\mathbb{R}$;
  - relational symbols: $>, <, \geq, \leq, =, \neq$;
  - Boolean connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow, \ldots$; and
  - quantifiers: $\forall, \exists$.
- A *term* of $T(\mathbb{R})$ over a finite set of variables $\{x_1, x_2, \ldots, x_n\}$ is a polynomial $p \in \mathbb{R}[x_1, x_2, \ldots, x_n]$.
- An *atomic formula* of $T(\mathbb{R})$ is of the form $p \rhd 0$, where $\rhd$ is any relational symbol.
- A *quantifier-free formula* (QFF) of $T(\mathbb{R})$ is a Boolean combination of atomic formulas.
- A generic formula of $T(\mathbb{R})$ is built up from atomic formulas using Boolean connectives as well as quantifiers.

First-order Theory $T(\mathbb{R})$ of Reals

Syntax

- The language of $T(\mathbb{R})$ consists of
  - variables: $x, y, z, \ldots, x_1, x_2, \ldots$, which are interpreted over $\mathbb{R}$;
  - relational symbols: $>, <, \geq, \leq, =, \neq$;
  - Boolean connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow, \ldots$; and
  - quantifiers: $\forall, \exists$.
- A *term* of $T(\mathbb{R})$ over a finite set of variables $\{x_1, x_2, \ldots, x_n\}$ is a polynomial $p \in \mathbb{R}[x_1, x_2, \ldots, x_n]$.
- An *atomic formula* of $T(\mathbb{R})$ is of the form $p \triangleright 0$, where $\triangleright$ is any relational symbol.
- A *quantifier-free formula* (QFF) of $T(\mathbb{R})$ is a Boolean combination of atomic formulas.
- A generic formula of $T(\mathbb{R})$ is built up from atomic formulas using Boolean connectives as well as quantifiers.

First-order Theory $T(\mathbb{R})$ of Reals

Syntax

- The language of $T(\mathbb{R})$ consists of
  - variables: $x, y, z, \ldots, x_1, x_2, \ldots$, which are interpreted over $\mathbb{R}$;
  - relational symbols: $>, <, \geq, \leq, =, \neq$;
  - Boolean connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow, \ldots$; and
  - quantifiers: $\forall, \exists$.
- A *term* of $T(\mathbb{R})$ over a finite set of variables $\{x_1, x_2, \ldots, x_n\}$ is a polynomial $p \in \mathbb{R}[x_1, x_2, \ldots, x_n]$.
- An *atomic formula* of $T(\mathbb{R})$ is of the form $p \rhd 0$, where $\rhd$ is any relational symbol.
- A *quantifier-free formula* (QFF) of $T(\mathbb{R})$ is a Boolean combination of atomic formulas.
- A generic formula of $T(\mathbb{R})$ is built up from atomic formulas using Boolean connectives as well as quantifiers.

First-order Theory $T(\mathbb{R})$ of Reals

Syntax

- The language of $T(\mathbb{R})$ consists of
  - variables: $x, y, z, \ldots, x_1, x_2, \ldots$, which are interpreted over $\mathbb{R}$;
  - relational symbols: $>, <, \geq, \leq, =, \neq$;
  - Boolean connectives: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow, \ldots$; and
  - quantifiers: $\forall, \exists$.
- A *term* of $T(\mathbb{R})$ over a finite set of variables $\{x_1, x_2, \ldots, x_n\}$ is a polynomial $p \in \mathbb{R}[x_1, x_2, \ldots, x_n]$.
- An *atomic formula* of $T(\mathbb{R})$ is of the form $p \triangleright 0$, where $\triangleright$ is any relational symbol.
- A *quantifier-free formula* (QFF) of $T(\mathbb{R})$ is a Boolean combination of atomic formulas.
- A generic formula of $T(\mathbb{R})$ is built up from atomic formulas using Boolean connectives as well as quantifiers.

Quantifier Elimination

Quantifier Elimination Property

- A theory $\mathcal{T}$ is said to have **quantifier elimination property**, if for any formula φ in $\mathcal{T}$, there exists a quantifier-free formula φ_{QF} which only contains *free* variables of φ such that $\varphi \Leftrightarrow \varphi_{QF}$.
- $T(\mathbb{R})$ admits **quantifier elimination**.
- The ***decidability*** of $T(\mathbb{R})$

Example

$$\begin{aligned} \exists x. ax^2 + bx + c = 0 \quad \Leftrightarrow \quad & a = b = c = 0 \vee \\ & (a = 0 \wedge b \neq 0) \vee \\ & (a \neq 0 \wedge b^2 - 4ac \geq 0) \end{aligned}$$

Quantifier Elimination

Quantifier Elimination Property

- A theory $\mathcal{T}$ is said to have **quantifier elimination property**, if for any formula φ in $\mathcal{T}$, there exists a quantifier-free formula φ_{QF} which only contains *free* variables of φ such that $\varphi \Leftrightarrow \varphi_{QF}$.
- $T(\mathbb{R})$ admits **quantifier elimination**.
- The ***decidability*** of $T(\mathbb{R})$

Example

$$\begin{aligned} \exists x. ax^2 + bx + c = 0 \quad \Leftrightarrow \quad & a = b = c = 0 \vee \\ & (a = 0 \wedge b \neq 0) \vee \\ & (a \neq 0 \wedge b^2 - 4ac \geq 0) \end{aligned}$$

Quantifier Elimination

Quantifier Elimination Property

- A theory $\mathcal{T}$ is said to have **quantifier elimination property**, if for any formula φ in $\mathcal{T}$, there exists a quantifier-free formula φ_{QF} which only contains *free* variables of φ such that $\varphi \Leftrightarrow \varphi_{QF}$.
- $T(\mathbb{R})$ admits **quantifier elimination**.
- The ***decidability*** of $T(\mathbb{R})$

Example

$$\begin{aligned} \exists x. ax^2 + bx + c = 0 \quad \Leftrightarrow \quad & a = b = c = 0 \vee \\ & (a = 0 \wedge b \neq 0) \vee \\ & (a \neq 0 \wedge b^2 - 4ac \geq 0) \end{aligned}$$

Quantifier Elimination (Cont'd)

Semi-algrbraic Set

- A subset $A \subseteq \mathbb{R}^n$ is called a ***semi-algebraic set*** (SAS), if there exists a QFF $\phi \in T(\mathbb{R})$, such that $A = \{\mathbf{x} \in \mathbb{R}^n \mid \phi(\mathbf{x}) \text{ is true}\}$.
- SASs are closed under common set operations:
  - $\mathcal{A}(\phi_1) \cap \mathcal{A}(\phi_2) = \mathcal{A}(\varphi_1 \wedge \varphi_2)$;
  - $\mathcal{A}(\phi_1) \cup \mathcal{A}(\phi_2) = \mathcal{A}(\varphi_1 \vee \phi_2)$;
  - $\mathcal{A}(\phi_1)^c = \mathcal{A}(\neg\phi_1)$;
  - $\mathcal{A}(\phi_1) \setminus \mathcal{A}(\phi_2) = \mathcal{A}(\phi_1) \cap \mathcal{A}(\phi_2)^c = \mathcal{A}(\phi_1 \wedge \neg\phi_2)$.
- Any SAS can be represented by a QFF in the form of $\phi(\mathbf{x}) \widehat{=} \bigvee_{k=1}^{K} \bigwedge_{j=1}^{J_k} p_{kj}(\mathbf{x}) \triangleright 0$, where $p_{kj}(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ and $\triangleright \in \{\geq, >\}$.

Semi-algebraic Template

A ***semi-algebraic template*** with degree d is of the form

$$\phi(\mathbf{u}, \mathbf{x}) \widehat{=} \vee_{k=1}^{K} \wedge_{j=1}^{J_k} p_{kj}(\mathbf{u}_{kj}, \mathbf{x}) \triangleright 0.$$

Quantifier Elimination (Cont'd)

Semi-algrbraic Set

- A subset $A \subseteq \mathbb{R}^n$ is called a ***semi-algebraic set*** (SAS), if there exists a QFF $\phi \in T(\mathbb{R})$, such that $A = \{\mathbf{x} \in \mathbb{R}^n \mid \phi(\mathbf{x}) \text{ is true}\}$.
- SASs are closed under common set operations:
  - $\mathcal{A}(\phi_1) \cap \mathcal{A}(\phi_2) = \mathcal{A}(\varphi_1 \wedge \varphi_2)$;
  - $\mathcal{A}(\phi_1) \cup \mathcal{A}(\phi_2) = \mathcal{A}(\varphi_1 \vee \phi_2)$;
  - $\mathcal{A}(\phi_1)^c = \mathcal{A}(\neg\phi_1)$;
  - $\mathcal{A}(\phi_1) \setminus \mathcal{A}(\phi_2) = \mathcal{A}(\phi_1) \cap \mathcal{A}(\phi_2)^c = \mathcal{A}(\phi_1 \wedge \neg\phi_2)$.
- Any SAS can be represented by a QFF in the form of $\phi(\mathbf{x}) \widehat{=} \bigvee_{k=1}^{K} \bigwedge_{j=1}^{J_k} p_{kj}(\mathbf{x}) \rhd 0$, where $p_{kj}(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ and $\rhd \in \{\geq, >\}$.

Semi-algebraic Template

A ***semi-algebraic template*** with degree d is of the form

$$\phi(\mathbf{u}, \mathbf{x}) \widehat{=} \vee_{k=1}^{K} \wedge_{j=1}^{J_k} p_{kj}(\mathbf{u}_{kj}, \mathbf{x}) \rhd 0.$$

Quantifier Elimination (Cont'd)

Semi-algrbraic Set

- A subset $A \subseteq \mathbb{R}^n$ is called a ***semi-algebraic set*** (SAS), if there exists a QFF $\phi \in T(\mathbb{R})$, such that $A = \{\mathbf{x} \in \mathbb{R}^n \mid \phi(\mathbf{x}) \text{ is true}\}$.
- SASs are closed under common set operations:
  - $\mathcal{A}(\phi_1) \cap \mathcal{A}(\phi_2) = \mathcal{A}(\varphi_1 \wedge \varphi_2)$;
  - $\mathcal{A}(\phi_1) \cup \mathcal{A}(\phi_2) = \mathcal{A}(\varphi_1 \vee \phi_2)$;
  - $\mathcal{A}(\phi_1)^c = \mathcal{A}(\neg\phi_1)$;
  - $\mathcal{A}(\phi_1) \setminus \mathcal{A}(\phi_2) = \mathcal{A}(\phi_1) \cap \mathcal{A}(\phi_2)^c = \mathcal{A}(\phi_1 \wedge \neg\phi_2)$.
- Any SAS can be represented by a QFF in the form of $\phi(\mathbf{x}) \widehat{=} \bigvee_{k=1}^{K} \bigwedge_{j=1}^{J_k} p_{kj}(\mathbf{x}) \triangleright 0$, where $p_{kj}(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ and $\triangleright \in \{\geq, >\}$.

Semi-algebraic Template

A ***semi-algebraic template*** with degree d is of the form

$$\phi(\mathbf{u}, \mathbf{x}) \widehat{=} \vee_{k=1}^{K} \wedge_{j=1}^{J_k} p_{kj}(\mathbf{u}_{kj}, \mathbf{x}) \triangleright 0.$$

Quantifier Elimination (Cont'd)

Semi-algrbraic Set

- A subset $A \subseteq \mathbb{R}^n$ is called a ***semi-algebraic set*** (SAS), if there exists a QFF $\phi \in T(\mathbb{R})$, such that $A = \{\mathbf{x} \in \mathbb{R}^n \mid \phi(\mathbf{x}) \text{ is true}\}$.
- SASs are closed under common set operations:
  - $\mathcal{A}(\phi_1) \cap \mathcal{A}(\phi_2) = \mathcal{A}(\varphi_1 \wedge \varphi_2)$;
  - $\mathcal{A}(\phi_1) \cup \mathcal{A}(\phi_2) = \mathcal{A}(\varphi_1 \vee \phi_2)$;
  - $\mathcal{A}(\phi_1)^c = \mathcal{A}(\neg\phi_1)$;
  - $\mathcal{A}(\phi_1) \setminus \mathcal{A}(\phi_2) = \mathcal{A}(\phi_1) \cap \mathcal{A}(\phi_2)^c = \mathcal{A}(\phi_1 \wedge \neg\phi_2)$.
- Any SAS can be represented by a QFF in the form of $\phi(\mathbf{x}) \mathrel{\widehat{=}} \bigvee_{k=1}^{K} \bigwedge_{j=1}^{J_k} p_{kj}(\mathbf{x}) \rhd 0$, where $p_{kj}(\mathbf{x}) \in \mathbb{Q}[\mathbf{x}]$ and $\rhd \in \{\geq, >\}$.

Semi-algebraic Template

A ***semi-algebraic template*** with degree d is of the form

$$\phi(\mathbf{u}, \mathbf{x}) \mathrel{\widehat{=}} \vee_{k=1}^{K} \wedge_{j=1}^{J_k} p_{kj}(\mathbf{u}_{kj}, \mathbf{x}) \rhd 0.$$

Quantifier Elimination (Cont'd)

Survey of QE Algorithms

- **Tarski's algorithm** [Tarski 51]: the first one, but its complexity is nonelementary, impratical, simplified by Seidenberg [Seidenberg 54].
- **Collins' algorithm** [Collins 76]: based on **cylindrical algebraic decomposition (CAD)**, **double exponential** in the number of variables, improved by Hoon Hong [Hoon Hong 92] by combining with SAT engine **partial cylindrical algebraic decomposition (PCAD)**, implemented in many computer algebra tools, e.g., **QEBCAD, REDLOG, ...**
- **Ben-Or, Kozen and Reif's algorithm** [Ben-Or, Kozen&Reif 1986]: double exponential in the number of variables using sequential computation, single exponential using parallel computation, based on **Sturm sequence** and **Sturm Theorem**, some mistake.
- More efficient algorithms [Grigor'ev & Vorobjov 1988, Grigor'ev 1988], [Renegar 1989], [Heintz, Roy&Solerno, 1989], [Basu,Pollack&Roy, 1996]: mainly based on **Ben-Or, Kozen and Reif's work**, double exponential in the number of quantifier alternation, no implementation yet.

Quantifier Elimination (Cont'd)

Survey of QE Algorithms

- **Tarski's algorithm** [Tarski 51]: the first one, but its complexity is nonelementary, impratical, simplified by Seidenberg [Seidenberg 54].
- **Collins' algorithm** [Collins 76]: based on **cylindrical algebraic decomposition (CAD)**, **double exponential** in the number of variables, improved by Hoon Hong [Hoon Hong 92] by combining with SAT engine **partial cylindrical algebraic decomposition (PCAD)**, implemented in many computer algebra tools, e.g., **QEBCAD, REDLOG, ...**
- **Ben-Or, Kozen and Reif's algorithm** [Ben-Or, Kozen&Reif 1986]: double exponential in the number of variables using sequential computation, single exponential using parallel computation, based on **Sturm sequence** and **Sturm Theorem**, some mistake.
- More efficient algorithms [Grigor'ev & Vorobjov 1988, Grigor'ev 1988], [Renegar 1989], [Heintz, Roy&Solerno, 1989], [Basu,Pollack&Roy, 1996]: mainly based on **Ben-Or, Kozen and Reif's work**, double exponential in the number of quantifier alternation, no implementation yet.

Quantifier Elimination (Cont'd)

Survey of QE Algorithms

- **Tarski's algorithm** [Tarski 51]: the first one, but its complexity is nonelementary, impratical, simplified by Seidenberg [Seidenberg 54].
- **Collins' algorithm** [Collins 76]: based on **cylindrical algebraic decomposition (CAD)**, **double exponential** in the number of variables, improved by Hoon Hong [Hoon Hong 92] by combining with SAT engine **partial cylindrical algebraic decomposition (PCAD)**, implemented in many computer algebra tools, e.g., **QEBCAD, REDLOG, ...**
- **Ben-Or, Kozen and Reif's algorithm** [Ben-Or, Kozen&Reif 1986]: double exponential in the number of variables using sequential computation, single exponential using parallel computation, based on **Sturm sequence** and **Sturm Theorem**, some mistake.
- More efficient algorithms [Grigor'ev & Vorobjov 1988, Grigor'ev 1988], [Renegar 1989], [Heintz, Roy&Solerno, 1989], [Basu,Pollack&Roy, 1996]: mainly based on **Ben-Or, Kozen and Reif's work**, double exponential in the number of quantifier alternation, no implementation yet.

Quantifier Elimination (Cont'd)

Survey of QE Algorithms

- **Tarski's algorithm** [Tarski 51]: the first one, but its complexity is nonelementary, impratical, simplified by Seidenberg [Seidenberg 54].
- **Collins' algorithm** [Collins 76]: based on **cylindrical algebraic decomposition (CAD)**, **double exponential** in the number of variables, improved by Hoon Hong [Hoon Hong 92] by combining with SAT engine **partial cylindrical algebraic decomposition (PCAD)**, implemented in many computer algebra tools, e.g., **QEBCAD, REDLOG, ...**
- **Ben-Or, Kozen and Reif's algorithm** [Ben-Or, Kozen&Reif 1986]: double exponential in the number of variables using sequential computation, single exponential using parallel computation, based on **Sturm sequence** and **Sturm Theorem**, some mistake.
- More efficient algorithms [Grigor'ev & Vorobjov 1988, Grigor'ev 1988], [Renegar 1989], [Heintz, Roy&Solerno, 1989], [Basu,Pollack&Roy, 1996]: mainly based on **Ben-Or, Kozen and Reif's work**, double exponential in the number of quantifier alternation, no implementation yet.

Outline

Continuous Dynamical Systems

- A **continuous dynamical systems (CDS)** is of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a ***vector field***.

- If $\mathbf{f}$ in (1) satisfies **local Lipschitz condition**, then given $\mathbf{x}_0 \in \mathbb{R}^n$, there exists a unique solution $\mathbf{x}(\mathbf{x}_0; t) : (a, b) \to \mathbb{R}^n$ such that $\mathbf{x}(\mathbf{x}_0; 0) = \mathbf{x}_0$ and $\forall t \in (a, b). \frac{\mathrm{d}\mathbf{x}(\mathbf{x}_0;t)}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}(\mathbf{x}_0; t))$.
- If $\mathbf{f}$ in (1) satisfies **global Lipschitz condition**, then the **existence, uniqueness** and **completeness** of solutions to (1) can be guaranteed.
- The k-th **Lie derivatives** $L^k_{\mathbf{f}}\sigma : \mathbb{R}^n \to \mathbb{R}$ of σ along $\mathbf{f}$ is defined by:
  - $L^0_{\mathbf{f}}\sigma(\mathbf{x}) = \sigma(\mathbf{x})$,
  - $L^k_{\mathbf{f}}\sigma(\mathbf{x}) = \left(\nabla L^{k-1}_{\mathbf{f}}\sigma(\mathbf{x}), \mathbf{f}(\mathbf{x})\right)$, for $k > 0$,

  where $\nabla\varrho(\mathbf{x}) \widehat{=} \left(\frac{\partial\varrho(\mathbf{x})}{\partial x_1}, \frac{\partial\varrho(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial\varrho(\mathbf{x})}{\partial x_n}\right)$ and $(\cdot, \cdot)$ is the *inner product* of two vectors.

Continuous Dynamical Systems

- A **continuous dynamical systems (CDS)** is of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \qquad (1)$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a ***vector field***.

- If $\mathbf{f}$ in (1) satisfies **local Lipschitz condition**, then given $\mathbf{x}_0 \in \mathbb{R}^n$, there exists a unique solution $\mathbf{x}(\mathbf{x}_0; t) : (a, b) \to \mathbb{R}^n$ such that $\mathbf{x}(\mathbf{x}_0; 0) = \mathbf{x}_0$ and $\forall t \in (a, b). \frac{\mathrm{d}\mathbf{x}(\mathbf{x}_0;t)}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}(\mathbf{x}_0; t))$.
- If $\mathbf{f}$ in (1) satisfies **global Lipschitz condition**, then the **existence, uniqueness** and **completeness** of solutions to (1) can be guaranteed.
- The k-th **Lie derivatives** $L_{\mathbf{f}}^k \sigma : \mathbb{R}^n \to \mathbb{R}$ of σ along $\mathbf{f}$ is defined by:
  - $L_{\mathbf{f}}^0 \sigma(\mathbf{x}) = \sigma(\mathbf{x})$,
  - $L_{\mathbf{f}}^k \sigma(\mathbf{x}) = \left(\nabla L_{\mathbf{f}}^{k-1} \sigma(\mathbf{x}), \mathbf{f}(\mathbf{x})\right)$, for $k > 0$,

  where $\nabla \varrho(\mathbf{x}) \widehat{=} \left(\frac{\partial \varrho(\mathbf{x})}{\partial x_1}, \frac{\partial \varrho(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial \varrho(\mathbf{x})}{\partial x_n}\right)$ and $(\cdot, \cdot)$ is the *inner product* of two vectors.

Continuous Dynamical Systems

- A **continuous dynamical systems (CDS)** is of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a ***vector field***.

- If $\mathbf{f}$ in (1) satisfies **local Lipschitz condition**, then given $\mathbf{x}_0 \in \mathbb{R}^n$, there exists a unique solution $\mathbf{x}(\mathbf{x}_0; t) : (a, b) \to \mathbb{R}^n$ such that $\mathbf{x}(\mathbf{x}_0; 0) = \mathbf{x}_0$ and $\forall t \in (a, b).\ \frac{\mathrm{d}\mathbf{x}(\mathbf{x}_0;t)}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}(\mathbf{x}_0; t))$.
- If $\mathbf{f}$ in (1) satisfies **global Lipschitz condition**, then the **existence, uniqueness** and **completeness** of solutions to (1) can be guaranteed.
- The k-th **Lie derivatives** $L_{\mathbf{f}}^k \sigma : \mathbb{R}^n \to \mathbb{R}$ of σ along $\mathbf{f}$ is defined by:
  - $L_{\mathbf{f}}^0 \sigma(\mathbf{x}) = \sigma(\mathbf{x})$,
  - $L_{\mathbf{f}}^k \sigma(\mathbf{x}) = (\nabla L_{\mathbf{f}}^{k-1} \sigma(\mathbf{x}), \mathbf{f}(\mathbf{x}))$, for $k > 0$,

  where $\nabla \varrho(\mathbf{x}) \mathrel{\widehat{=}} \left(\frac{\partial \varrho(\mathbf{x})}{\partial x_1}, \frac{\partial \varrho(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial \varrho(\mathbf{x})}{\partial x_n} \right)$ and $(\cdot, \cdot)$ is the *inner product* of two vectors.

Continuous Dynamical Systems

- A **continuous dynamical systems (CDS)** is of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a ***vector field***.

- If $\mathbf{f}$ in (1) satisfies **local Lipschitz condition**, then given $\mathbf{x}_0 \in \mathbb{R}^n$, there exists a unique solution $\mathbf{x}(\mathbf{x}_0; t) : (a, b) \to \mathbb{R}^n$ such that $\mathbf{x}(\mathbf{x}_0; 0) = \mathbf{x}_0$ and $\forall t \in (a, b). \frac{\mathrm{d}\mathbf{x}(\mathbf{x}_0;t)}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}(\mathbf{x}_0; t))$.
- If $\mathbf{f}$ in (1) satisfies **global Lipschitz condition**, then the **existence, uniqueness** and **completeness** of solutions to (1) can be guaranteed.
- The k-th **Lie derivatives** $L_{\mathbf{f}}^k\sigma : \mathbb{R}^n \to \mathbb{R}$ of σ along $\mathbf{f}$ is defined by:
    - $L_{\mathbf{f}}^0\sigma(\mathbf{x}) = \sigma(\mathbf{x})$,
    - $L_{\mathbf{f}}^k\sigma(\mathbf{x}) = \left(\nabla L_{\mathbf{f}}^{k-1}\sigma(\mathbf{x}), \mathbf{f}(\mathbf{x})\right)$, for $k > 0$,

  where $\nabla\varrho(\mathbf{x}) \,\widehat{=}\, \left(\frac{\partial\varrho(\mathbf{x})}{\partial x_1}, \frac{\partial\varrho(\mathbf{x})}{\partial x_2}, \ldots, \frac{\partial\varrho(\mathbf{x})}{\partial x_n}\right)$ and $(\cdot, \cdot)$ is the *inner product* of two vectors.

Outline

Hybrid Automaton

A **hybrid automaton** (HA) is a system $\mathcal{H} \mathrel{\widehat{=}} (Q, X, f, D, E, G, R, \Xi)$, where

- $Q = \{q_1, \ldots, q_m\}$ is a finite set of modes;
- $X = \{x_1, \ldots, x_n\}$ is a finite set of continuous state variables, with $\mathbf{x} = (x_1, \ldots, x_n)$ ranging over $\mathbb{R}^n$; $Q \times \mathbb{R}^n$ is the state space of $\mathcal{H}$;
- $f : Q \to (\mathbb{R}^n \to \mathbb{R}^n)$ assigns to each mode $q \in Q$ a vector field $\mathbf{f}_q$;
- $D : Q \to 2^{\mathbb{R}^n}$ assigns to each mode $q \in Q$ a domain $D_q \subseteq \mathbb{R}^n$;
- $E \subseteq Q \times Q$ is a set of discrete transitions;
- $G : E \to 2^{\mathbb{R}^n}$ assigns to each transition $e \in E$ a switching guard $G_e \subseteq \mathbb{R}^n$.
- R assigns to each transition $e \in E$ a reset function R_e: $\mathbb{R}^n \to \mathbb{R}^n$;
- Ξ assigns to each $q \in Q$ a set of initial states $\Xi_q \subseteq \mathbb{R}^n$.

Hybrid Trajectories Accepted by HA [Tomlin et al. 00]

Definition (Hybrid Time Set)

A hybrid time set is a sequence of time intervals $\tau = \{I_i\}_{i=0}^{N}$ (N can be ∞) s.t.

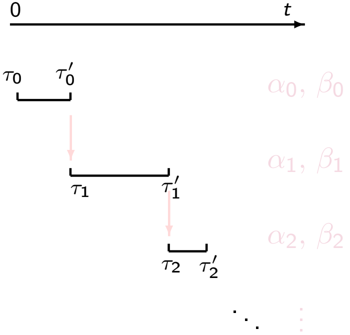
- $I_i = [\tau_i, \tau_i']$ with $\tau_i \leq \tau_i' = \tau_{i+1}$ for all $i < N$;
- if $N < \infty$, then $I_N = [\tau_N, \tau_N'\rangle$ is a right-closed or right-open nonempty interval (τ_N' may be ∞);
- $\tau_0 = 0$.

Hybrid Trajectories Accepted by HA [Tomlin et al. 00]

Definition (Hybrid Time Set)

A hybrid time set is a sequence of time intervals $\tau = \{I_i\}_{i=0}^{N}$ (N can be ∞) s.t.

- $I_i = [\tau_i, \tau_i']$ with $\tau_i \leq \tau_i' = \tau_{i+1}$ for all $i < N$;
- if $N < \infty$, then $I_N = [\tau_N, \tau_N'\rangle$ is a right-closed or right-open nonempty interval (τ_N' may be ∞);
- $\tau_0 = 0$.

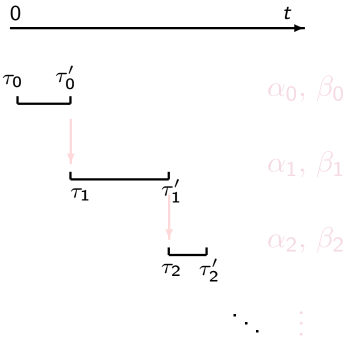


Hybrid Trajectories Accepted by HA [Tomlin et al. 00]

Definition (Hybrid Trajectory)

A hybrid trajectory is a triple $\omega = (\tau, \alpha, \beta)$, where $\tau = \{I_i\}_{i=0}^{N}$ is a hybrid time set and $\alpha = \{\alpha_i : I_i \to Q\}$ and $\beta = \{\beta_i : I_i \to \mathbb{R}^n\}$ are two sequences of functions satisfying

1. Initial condition: $\alpha_0[0] = q_0$ and $\beta_0[0] = \mathbf{x}_0$;
2. Discrete transition: for all $i < \langle\tau\rangle$, $e = (\alpha_i(\tau_i'), \alpha_{i+1}(\tau_{i+1})) \in E$, $\beta_i(\tau_i') \in G_e$ and $\beta_{i+1}(\tau_{i+1}) = R_e(\beta_i(\tau_i'))$;
3. Continuous evolution: for all $i \leq \langle\tau\rangle$ with $\tau_i < \tau_i'$, if $q = \alpha_i(\tau_i)$, then
   (1) for all $t \in I_i$, $\alpha_i(t) = q$,
   (2) $\beta_i(t)$ is the solution to the differential equation $\dot{\mathbf{x}} = \mathbf{f}_q(\mathbf{x})$ over I_i with initial value $\beta_i(\tau_i)$, and
   (3) for all $t \in [\tau_i, \tau_i')$, $\beta_i(t) \in D_q$.

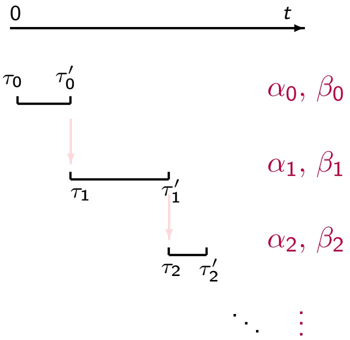


Hybrid Trajectories Accepted by HA [Tomlin et al. 00]

Definition (Hybrid Trajectory)

A hybrid trajectory is a triple $\omega = (\tau, \alpha, \beta)$, where $\tau = \{I_i\}_{i=0}^{N}$ is a hybrid time set and $\alpha = \{\alpha_i : I_i \to Q\}$ and $\beta = \{\beta_i : I_i \to \mathbb{R}^n\}$ are two sequences of functions satisfying

1. Initial condition: $\alpha_0[0] = q_0$ and $\beta_0[0] = \mathbf{x}_0$;
2. Discrete transition: for all $i < \langle\tau\rangle$, $e = \big(\alpha_i(\tau_i'), \alpha_{i+1}(\tau_{i+1})\big) \in E$, $\beta_i(\tau_i') \in G_e$ and $\beta_{i+1}(\tau_{i+1}) = R_e(\beta_i(\tau_i'))$;
3. Continuous evolution: for all $i \leq \langle\tau\rangle$ with $\tau_i < \tau_i'$, if $q = \alpha_i(\tau_i)$, then
   (1) for all $t \in I_i$, $\alpha_i(t) = q$,
   (2) $\beta_i(t)$ is the solution to the differential equation $\dot{\mathbf{x}} = \mathbf{f}_q(\mathbf{x})$ over I_i with initial value $\beta_i(\tau_i)$, and
   (3) for all $t \in [\tau_i, \tau_i')$, $\beta_i(t) \in D_q$.

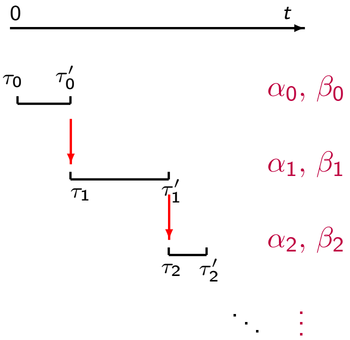


Hybrid Trajectories Accepted by HA [Tomlin et al. 00]

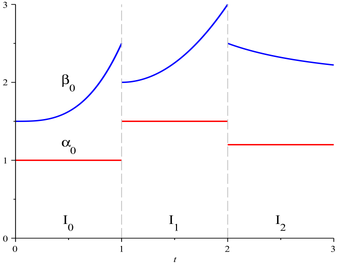
Definition (Hybrid Trajectory)

A hybrid trajectory is a triple $\omega = (\tau, \alpha, \beta)$, where $\tau = \{I_i\}_{i=0}^{N}$ is a hybrid time set and $\alpha = \{\alpha_i : I_i \to Q\}$ and $\beta = \{\beta_i : I_i \to \mathbb{R}^n\}$ are two sequences of functions satisfying

1. Initial condition: $\alpha_0[0] = q_0$ and $\beta_0[0] = \mathbf{x}_0$;
2. Discrete transition: for all $i < \langle\tau\rangle$, $e = \big(\alpha_i(\tau_i'), \alpha_{i+1}(\tau_{i+1})\big) \in E$, $\beta_i(\tau_i') \in G_e$ and $\beta_{i+1}(\tau_{i+1}) = R_e(\beta_i(\tau_i'))$;
3. Continuous evolution: for all $i \leq \langle\tau\rangle$ with $\tau_i < \tau_i'$, if $q = \alpha_i(\tau_i)$, then
   (1) for all $t \in I_i$, $\alpha_i(t) = q$,
   (2) $\beta_i(t)$ is the solution to the differential equation $\dot{\mathbf{x}} = \mathbf{f}_q(\mathbf{x})$ over I_i with initial value $\beta_i(\tau_i)$, and
   (3) for all $t \in [\tau_i, \tau_i')$, $\beta_i(t) \in D_q$.



Hybrid Trajectories Accepted by HA (Cont'd) [Tomlin et al. 00]

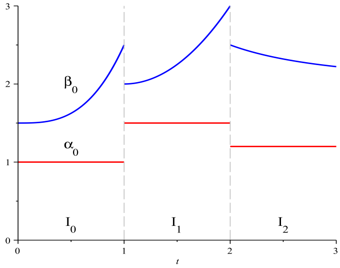


A hybrid trajectory (τ, α, β) is called *infinite* if

- $\langle \tau \rangle = N$ is ∞, or
- $\|\tau\| = \sum_{i=0}^{N} (\tau_i' - \tau_i)$ is ∞.

A hybrid automaton is called *non-blocking* if there is an infinite trajectory starting from any initial state (q_0, x_0), and *blocking* otherwise.

Hybrid Trajectories Accepted by HA (Cont'd) [Tomlin et al. 00]

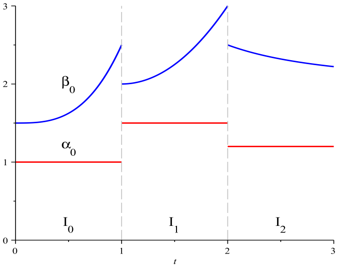


A hybrid trajectory (τ, α, β) is called *infinite* if

- $\langle \tau \rangle = N$ is ∞, or
- $\|\tau\| = \sum_{i=0}^{N}(\tau_i' - \tau_i)$ is ∞.

A hybrid automaton is called *non-blocking* if there is an infinite trajectory starting from any initial state (q_0, x_0), and *blocking* otherwise.

Hybrid Trajectories Accepted by HA (Cont'd) [Tomlin et al. 00]



A hybrid trajectory (τ, α, β) is called *infinite* if

- $\langle\tau\rangle = N$ is ∞, or
- $\|\tau\| = \sum_{i=0}^{N}(\tau_i' - \tau_i)$ is ∞.

A hybrid automaton is called *non-blocking* if there is an infinite trajectory starting from any initial state (q_0, x_0), and *blocking* otherwise.

Reachable Set of HA

Definition (Reachable Set)

Given an HA $\mathcal{H}$, the **reachable set** $\mathcal{R}_{\mathcal{H}}$ of $\mathcal{H}$ consists of those $(q, \mathbf{x})$ for which there exists a finite sequence

$$(q_0, \mathbf{x}_0), (q_1, \mathbf{x}_1), \ldots, (q_l, \mathbf{x}_l)$$

such that $(q_0, \mathbf{x}_0) \in \Xi_{\mathcal{H}}$, $(q_l, \mathbf{x}_l) = (q, \mathbf{x})$, and for any $0 \leq i \leq l - 1$, one of the following two conditions holds:

- **(Discrete Jump):** $e = (q_i, q_{i+1}) \in E$, $\mathbf{x}_i \in G_e$ and $\mathbf{x}_{i+1} = R_e(\mathbf{x}_i)$; or
- **(Continuous Evolution):** $q_i = q_{i+1}$, and there exists a $\delta \geq 0$ s.t. the solution $\mathbf{x}(\mathbf{x}_i; t)$ to $\dot{\mathbf{x}} = \mathbf{f}_{q_i}$ satisfies
  - $\mathbf{x}(\mathbf{x}_i; t) \in D_{q_i}$ for all $t \in [0, \delta]$; and
  - $\mathbf{x}(\mathbf{x}_i; \delta) = \mathbf{x}_{i+1}$.

Outline

Continuous vs Global Invariants

Note that

- **Hybrid systems** consists of a set of CDSs, a set of transitions between these CDSs, and a transition may be equipped with a guard and reset
- **Invariant** plays a key role in analysis, verification, synthesis of hybrid systems
- **Global invariant** keeps invariant during continuous and discrete evolutions
- **Continuous invariant** keeps invariant in a mode
- Interplay between global and continuous invariant
- Both can be reduced to constraint solving
- Continuous invariant (differential invariant) generation is more complicated

Global Invariant

Definition (Global Invariant)

An invariant of an HA $\mathcal{H}$ maps to each $q \in Q$ a subset $I_q \subseteq \mathbb{R}^n$, such that for all $(q, \mathbf{x}) \in \mathcal{R}_{\mathcal{H}}$ (the reachable set), we have $\mathbf{x} \in I_q$.

Definition (Inductive Invariant)

Given an HA $\mathcal{H}$, an **inductive invariant** maps to each $q \in Q$ a subset $I_q \subseteq \mathbb{R}^n$, such that the following conditions are satisfied:

1. $\Xi_q \subseteq I_q$ for all $q \in Q$;
2. for any $e = (q, q') \in E$, if $\mathbf{x} \in I_q \cap G_e$, then $\mathbf{x}' = R_e(\mathbf{x}) \in I_{q'}$;
3. for any $q \in Q$ and any $\mathbf{x}_0 \in I_q$, if there exists a $\delta \geq 0$ s.t. the solution $\mathbf{x}(\mathbf{x}_0; t)$ to $\dot{\mathbf{x}} = \mathbf{f}_q$ satisfies: (i) $\mathbf{x}(\mathbf{x}_0; \delta) = \mathbf{x}'$; and (ii) $\mathbf{x}(\mathbf{x}_0; t) \in D_q$ for all $t \in [0, \delta]$, then $\mathbf{x}' \in I_q$.

Global Invariant

Definition (Global Invariant)

An invariant of an HA $\mathcal{H}$ maps to each $q \in Q$ a subset $I_q \subseteq \mathbb{R}^n$, such that for all $(q, \mathbf{x}) \in \mathcal{R}_{\mathcal{H}}$ (the reachable set), we have $\mathbf{x} \in I_q$.

Definition (Inductive Invariant)

Given an HA $\mathcal{H}$, an **inductive invariant** maps to each $q \in Q$ a subset $I_q \subseteq \mathbb{R}^n$, such that the following conditions are satisfied:

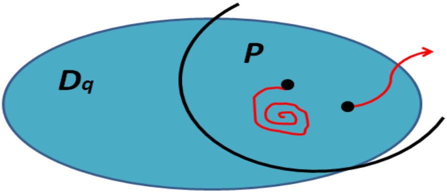
1. $\Xi_q \subseteq I_q$ for all $q \in Q$;
2. for any $e = (q, q') \in E$, if $\mathbf{x} \in I_q \cap G_e$, then $\mathbf{x}' = R_e(\mathbf{x}) \in I_{q'}$;
3. for any $q \in Q$ and any $\mathbf{x}_0 \in I_q$, if there exists a $\delta \geq 0$ s.t. the solution $\mathbf{x}(\mathbf{x}_0; t)$ to $\dot{\mathbf{x}} = \mathbf{f}_q$ satisfies: (i) $\mathbf{x}(\mathbf{x}_0; \delta) = \mathbf{x}'$; and (ii) $\mathbf{x}(\mathbf{x}_0; t) \in D_q$ for all $t \in [0, \delta]$, then $\mathbf{x}' \in I_q$.

Continuous Invariant

Definition (Continuous Invariant see also [Platzer & Clarke 08] **)**

Given $(D_q, \mathbf{f}_q)$, we call $P \subseteq \mathbb{R}^n$ a **continuous invariant** of $(D_q, \mathbf{f}_q)$ if for all $\mathbf{x}_0 \in P$ and all $T \geq 0$,

$$(\forall t \in [0, T].\, \mathbf{x}(t) \in D_q) \implies (\forall t \in [0, T].\, \mathbf{x}(t) \in P) \ .$$



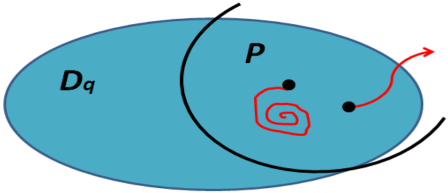
- A continuous invariant of a PDS is called a *semi-algebraic invariant* (SAI) if it is a semi-algebraic set.

Continuous Invariant

Definition (Continuous Invariant see also [Platzer & Clarke 08] **)**

Given $(D_q, \mathbf{f}_q)$, we call $P \subseteq \mathbb{R}^n$ a **continuous invariant** of $(D_q, \mathbf{f}_q)$ if for all $\mathbf{x}_0 \in P$ and all $T \geq 0$,

$$(\forall t \in [0, T].\, \mathbf{x}(t) \in D_q) \implies (\forall t \in [0, T].\, \mathbf{x}(t) \in P) \ .$$



- A continuous invariant of a PDS is called a ***semi-algebraic invariant*** (SAI) if it is a semi-algebraic set.

Related Work

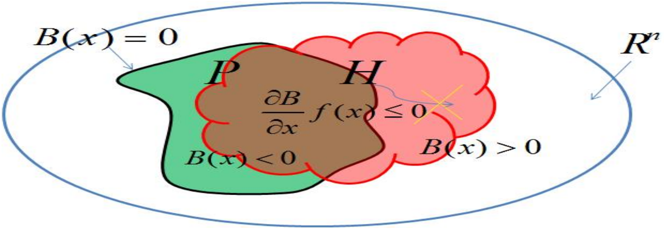
- **Barrier-certificate** [Prajna&Jadbadbaie 2004, Plazer&Clarke 2008]
  - Basic idea: Let $\mathcal{D} = \{\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})\}$ and $H = \{h(\mathbf{x}) \geq 0\}$. A function $B : \mathbb{R}^n \to \mathbb{R}$ is a barrier certificate if it is differentiable and satisfying

$$\forall \mathbf{x} \in H. \frac{\partial B}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}) \leq 0.$$

or

$$\forall \mathbf{x} \in H(B(x) = 0 \Rightarrow \frac{\partial B}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}) < 0).$$

Let $P := \{\mathbf{x} \mid B(\mathbf{x}) \leq 0\}$. Then P is an invariant of $(\mathcal{D}, H)$.



Related Work (Cont'd)

- **Boundary method** [Taly, Gulwani&Tiwari, VMCAI 2009]
  Let $\mathcal{D} = \{\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})\}$ and $H = \{h(\mathbf{x}) \geq 0\}$. If $P := \{\mathbf{x} \mid p(\mathbf{x}) \geq 0\}$ has the following property: For each $\mathbf{x}$ s.t. $p(\mathbf{x}) = 0$, there is a $\delta > 0$ s.t.

  $$\forall \mathbf{y} : (p(\mathbf{y}) = 0 \wedge \|\mathbf{y} - \mathbf{x}\| < \delta \Rightarrow L_{\mathbf{f}} p(\mathbf{y}) \geq 0 \wedge \frac{\partial p}{\partial \mathbf{y}} \neq \mathbf{0}),$$

  then P is an invariant of $(\mathcal{D}, H)$.
  - It imposes a strong assumption on the boundary.
- **Ideal fixed point method** [Sankaranarayanan, HSCC 2010]
  - Basic idea: If an ideal $\mathcal{I} \subseteq \mathcal{R}[\mathbf{x}]$ has the property:
    1. $(\forall p \in \mathcal{I}, \mathbf{x} \in H) p(\mathbf{x}) = 0$,
    2. $(\forall p \in \mathcal{I}), L_{\mathbf{f}} p \in \mathcal{I}$;

    then $P := \{\mathbf{x} \mid p(\mathbf{x}) = 0, \forall p \in \mathcal{I}\}$ is an invariant of $(\mathcal{D}, H)$.
  - It cannot cope with invariants as general semi-algebraic sets.

Related Work (Cont'd)

- **Boundary method** [Taly, Gulwani&Tiwari, VMCAI 2009]

  Let $\mathcal{D} = \{\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})\}$ and $H = \{h(\mathbf{x}) \geq 0\}$. If $P := \{\mathbf{x} \mid p(\mathbf{x}) \geq 0\}$ has the following property: For each $\mathbf{x}$ s.t. $p(\mathbf{x}) = 0$, there is a $\delta > 0$ s.t.

  $$\forall \mathbf{y} : (p(\mathbf{y}) = 0 \wedge \|\mathbf{y} - \mathbf{x}\| < \delta \Rightarrow L_{\mathbf{f}} p(\mathbf{y}) \geq 0 \wedge \frac{\partial p}{\partial \mathbf{y}} \neq \mathbf{0}),$$

  then P is an invariant of $(\mathcal{D}, H)$.

  - It imposes a strong assumption on the boundary.

- **Ideal fixed point method** [Sankaranarayanan, HSCC 2010]
  - Basic idea: If an ideal $\mathcal{I} \subseteq \mathcal{R}[\mathbf{x}]$ has the property:
    1. $(\forall p \in \mathcal{I}, \mathbf{x} \in H) p(\mathbf{x}) = 0$,
    2. $(\forall p \in \mathcal{I}), L_{\mathbf{f}} p \in \mathcal{I}$;

    then $P := \{\mathbf{x} \mid p(\mathbf{x}) = 0, \forall p \in \mathcal{I}\}$ is an invariant of $(\mathcal{D}, H)$.
  - It cannot cope with invariants as general semi-algebraic sets.

Related Work (Cont'd)

Open Problem

- **Open problem** [Sankaranarayanan, HSCC 2010, Taly&Tiwari, FSTTCS 2009]: **Can we find a complete method to generate all semi-algebraic invariants of a polynomial dynamical system?**
- We addressed this problem and gave an affirmative answer in [Liu, Zhan&Zhao 2011].

Outline

Basic Idea

- Let $(D, \mathbf{f})$ be a PDS, $\mathbf{x}(t)$ is a trajectory of $(D, \mathbf{f})$ from $\mathbf{x}_0$, and $P \widehat{=} p(\mathbf{x}) \geq 0$. Then P be a differential invariant of $(D, \mathbf{f})$ iff

$$\forall \mathbf{x}_0 \in \partial P \cap D, \exists \epsilon > 0, \forall t \in [0, \epsilon]. p(\mathbf{x}(t)) \geq 0 \tag{2}$$

- $p(\mathbf{x}(t))$'s Taylor's expansion at $t = 0$

$$p(\mathbf{x}(t)) = L_{\mathbf{f}}^1 p(\mathbf{x}_0).t + L_{\mathbf{f}}^2 p(\mathbf{x}_0).\frac{t^2}{2!} + \cdots L_{\mathbf{f}}^i p(\mathbf{x}_0).\frac{t^i}{i!} + \cdots$$

- (2) holds iff
  1. either for all $i \geq 0, L_{\mathbf{f}}^i p(\mathbf{x}_0) = 0$
  2. or there is some $k > i \geq 0$, such that $L_{\mathbf{f}}^i p(\mathbf{x}_0) = 0$ and $L_{\mathbf{f}}^k p(\mathbf{x}_0) > 0$.
- The *pointwise rank* of p with respect to $\mathbf{f}$ as the function $\gamma_{p,\mathbf{f}} : \mathbb{R}^n \to \mathbb{N} \cup \{\infty\}$ defined by

$$\gamma_{p,\mathbf{f}}(\mathbf{x}) = \min\{k \in \mathbb{N} \mid L_{\mathbf{f}}^k p(\mathbf{x}) \neq 0\}$$

if such k exists, and $\gamma_{p,\mathbf{f}}(\mathbf{x}) = \infty$ otherwise.

Basic Idea

- Let $(D, \mathbf{f})$ be a PDS, $\mathbf{x}(t)$ is a trajectory of $(D, \mathbf{f})$ from $\mathbf{x}_0$, and $P \widehat{=} p(\mathbf{x}) \geq 0$. Then P be a differential invariant of $(D, \mathbf{f})$ iff

$$\forall \mathbf{x}_0 \in \partial P \cap D, \exists \epsilon > 0, \forall t \in [0, \epsilon]. p(\mathbf{x}(t)) \geq 0 \tag{2}$$

- $p(\mathbf{x}(t))$'s **Taylor's expansion** at $t = 0$

$$p(\mathbf{x}(t)) = L_{\mathbf{f}}^1 p(\mathbf{x}_0).t + L_{\mathbf{f}}^2 p(\mathbf{x}_0).\frac{t^2}{2!} + \cdots L_{\mathbf{f}}^i p(\mathbf{x}_0).\frac{t^i}{i!} + \cdots$$

- (2) holds iff
  1. either for all $i \geq 0, L_{\mathbf{f}}^i p(\mathbf{x}_0) = 0$
  2. or there is some $k > i \geq 0$, such that $L_{\mathbf{f}}^i p(\mathbf{x}_0) = 0$ and $L_{\mathbf{f}}^k p(\mathbf{x}_0) > 0$.
- The *pointwise rank* of p with respect to $\mathbf{f}$ as the function $\gamma_{p,\mathbf{f}} : \mathbb{R}^n \to \mathbb{N} \cup \{\infty\}$ defined by

$$\gamma_{p,\mathbf{f}}(\mathbf{x}) = \min\{k \in \mathbb{N} \mid L_{\mathbf{f}}^k p(\mathbf{x}) \neq 0\}$$

if such k exists, and $\gamma_{p,\mathbf{f}}(\mathbf{x}) = \infty$ otherwise.

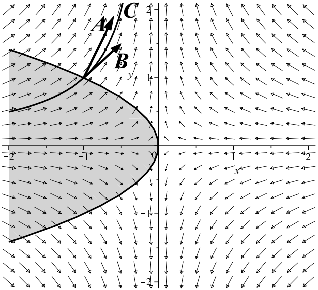
Basic Idea

- Let $(D, \mathbf{f})$ be a PDS, $\mathbf{x}(t)$ is a trajectory of $(D, \mathbf{f})$ from $\mathbf{x}_0$, and $P \widehat{=} p(\mathbf{x}) \geq 0$. Then P be a differential invariant of $(D, \mathbf{f})$ iff
$$\forall \mathbf{x}_0 \in \partial P \cap D, \exists \epsilon > 0, \forall t \in [0, \epsilon]. p(\mathbf{x}(t)) \geq 0 \qquad (2)$$
- $p(\mathbf{x}(t))$'s **Taylor's expansion** at $t = 0$
$$p(\mathbf{x}(t)) = L^1_{\mathbf{f}} p(\mathbf{x}_0).t + L^2_{\mathbf{f}} p(\mathbf{x}_0).\frac{t^2}{2!} + \cdots L^i_{\mathbf{f}} p(\mathbf{x}_0).\frac{t^i}{i!} + \cdots$$
- (2) holds iff
  1. either for all $i \geq 0, L^i_{\mathbf{f}} p(\mathbf{x}_0) = 0$
  2. or there is some $k > i \geq 0$, such that $L^i_{\mathbf{f}} p(\mathbf{x}_0) = 0$ and $L^k_{\mathbf{f}} p(\mathbf{x}_0) > 0$.
- The *pointwise rank* of p with respect to $\mathbf{f}$ as the function $\gamma_{p,\mathbf{f}} : \mathbb{R}^n \to \mathbb{N} \cup \{\infty\}$ defined by
$$\gamma_{p,\mathbf{f}}(\mathbf{x}) = \min\{k \in \mathbb{N} \mid L^k_{\mathbf{f}} p(\mathbf{x}) \neq 0\}$$
  if such k exists, and $\gamma_{p,\mathbf{f}}(\mathbf{x}) = \infty$ otherwise.

Example

Let $\mathbf{f} = (-x, y)$ and
$p(x, y) = x + y^2$. Then

$$
\begin{aligned}
L^0_{\mathbf{f}} p(x,y) &= x + y^2 \\
L^1_{\mathbf{f}} p(x,y) &= -x + 2y^2 \\
L^2_{\mathbf{f}} p(x,y) &= x + 4y^2 \\
&\vdots
\end{aligned}
$$



I: 1-order Lie Derivative and Gradient

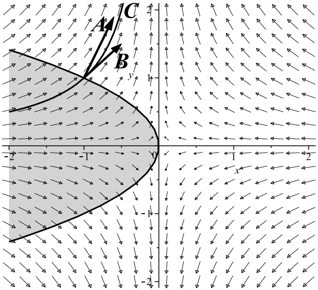
Consider point $(-1, 1)$ (see the picture),

- The points on the parabola $p(x, y) = 0$ with zero energy, and the points in the white area have positive energy, i.e. $p(x, y) > 0$.
- B denotes the evolution direction of $\mathbf{f}$ at the point.
- A is the gradient $\nabla p|_{(-1,1)}$ of $p(x, y)$.
- $L^1_{\mathbf{f}} p|_{(-1,1)} = 3$ predicts that the trajectory starting at $(-1, 1)$ will enter the white area.

Example

Let $\mathbf{f} = (-x, y)$ and $p(x,y) = x + y^2$. Then

$$\begin{aligned} L^0_{\mathbf{f}} p(x,y) &= x + y^2 \\ L^1_{\mathbf{f}} p(x,y) &= -x + 2y^2 \\ L^2_{\mathbf{f}} p(x,y) &= x + 4y^2 \\ &\vdots \end{aligned}$$



I: 1-order Lie Derivative and Gradient

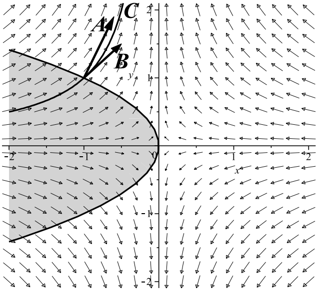
Consider point $(-1,1)$ (see the picture),

- The points on the parabola $p(x,y) = 0$ with zero energy, and the points in the white area have positive energy, i.e. $p(x,y) > 0$.
- B denotes the evolution direction of $\mathbf{f}$ at the point.
- A is the gradient $\nabla p|_{(-1,1)}$ of $p(x,y)$.
- $L^1_{\mathbf{f}} p|_{(-1,1)} = 3$ predicts that the trajectory starting at $(-1,1)$ will enter the white area.

Example

Let $\mathbf{f} = (-x, y)$ and $p(x,y) = x + y^2$. Then

$$\begin{aligned} L^0_{\mathbf{f}} p(x,y) &= x + y^2 \\ L^1_{\mathbf{f}} p(x,y) &= -x + 2y^2 \\ L^2_{\mathbf{f}} p(x,y) &= x + 4y^2 \\ &\vdots \end{aligned}$$



I: 1-order Lie Derivative and Gradient

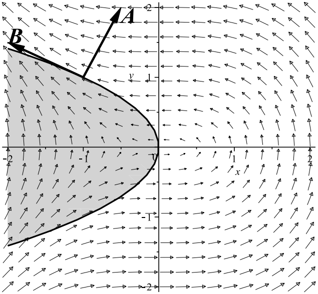
Consider point $(-1, 1)$ (see the picture),

- The points on the parabola $p(x,y) = 0$ with zero energy, and the points in the white area have positive energy, i.e. $p(x,y) > 0$.
- B denotes the evolution direction of $\mathbf{f}$ at the point.
- A is the gradient $\nabla p|_{(-1,1)}$ of $p(x,y)$.
- $L^1_{\mathbf{f}} p|_{(-1,1)} = 3$ predicts that the trajectory starting at $(-1, 1)$ will enter the white area.

Example

Let $\mathbf{f}(x, y) = (-2y, x^2)$ and $h(x, y) = x + y^2$. Then

$$
\begin{aligned}
L_{\mathbf{f}}^0 h(x, y) &= x + y^2 \\
L_{\mathbf{f}}^1 h(x, y) &= -2y + 2x^2 y \\
L_{\mathbf{f}}^2 h(x, y) &= -8y^2 x - (2 - 2x^2)x^2 \\
&\vdots
\end{aligned}
$$



II: Demand for Higher Order Lie Derivative

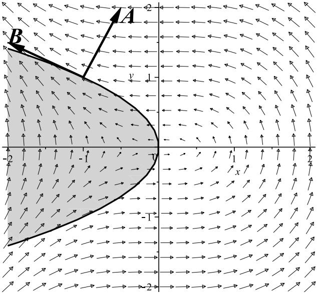
Also consider point $(-1, 1)$ on $h(x, y) = 0$ (see the picture),

- the gradient of h is $(1, 2)$ (vector A);
- the evolution direction is $(-2, 1)$ (vector B);
- their inner product is zero, i.e., $L_{\mathbf{f}}^1 h(-1, 1) = 0$, thus it is impossible to predict the tendency of the trajectory starting from $(-1, 1)$ via the 1-order Lie derivative;
- By a simple computation, $L_{\mathbf{f}}^2 h(-1, 1) = 8$. Hence $\gamma_{h,\mathbf{f}}(-1, 1) = 2$.

Example

Let $\mathbf{f}(x, y) = (-2y, x^2)$ and $h(x, y) = x + y^2$. Then

$$\begin{aligned} L^0_{\mathbf{f}} h(x, y) &= x + y^2 \\ L^1_{\mathbf{f}} h(x, y) &= -2y + 2x^2 y \\ L^2_{\mathbf{f}} h(x, y) &= -8y^2 x - (2 - 2x^2)x^2 \\ &\vdots \end{aligned}$$



II: Demand for Higher Order Lie Derivative

Also consider point $(-1, 1)$ on $h(x, y) = 0$ (see the picture),

- the gradient of h is $(1, 2)$ (vector A);
- the evolution direction is $(-2, 1)$ (vector B);
- their inner product is zero, i.e., $L^1_{\mathbf{f}} h(-1, 1) = 0$, thus it is impossible to predict the tendency of the trajectory starting from $(-1, 1)$ via the 1-order Lie derivative;
- By a simple computation, $L^2_{\mathbf{f}} h(-1, 1) = 8$. Hence $\gamma_{h,\mathbf{f}}(-1, 1) = 2$.

Theoretical Results

Theorem (Rank Theorem)

Given a polynomial p and a PVF $\mathbf{f}$, there is a natural number $N_{p,\mathbf{f}}$ such that for any $\mathbf{x} \in \mathbb{R}^n$, if $\gamma_{p,\mathbf{f}}(\mathbf{x}) < \infty$, then $\gamma_{p,\mathbf{f}}(\mathbf{x}) \leq N_{p,\mathbf{f}}$.

Theorem (Parametric Rank Theorem)

Given a parametric polynomial $p(\mathbf{u}, \mathbf{x})$ and a PVF $\mathbf{f}$, there is an integer $N_{p,\mathbf{f}} \in \mathbb{N}$ such that $\gamma_{p_{\mathbf{u}_0},\mathbf{f}}(\mathbf{x}) < \infty$ implies $\gamma_{p_{\mathbf{u}_0},\mathbf{f}}(\mathbf{x}) \leq N_{p,\mathbf{f}}$ for all $\mathbf{x} \in \mathbb{R}^n$ and all $\mathbf{u}_0 \in \mathbb{R}^w$.

Theorem (Criterion Theorem)

Given a polynomial p, $p(\mathbf{x}) \geq 0$ is an SCI of the PCCDS $(h(\mathbf{x}) \geq 0, \mathbf{f})$ iff

$$\theta(h, p, \mathbf{f}, \mathbf{x}) \widehat{=} (p(\mathbf{x}) = 0 \wedge \pi(p, \mathbf{f}, \mathbf{x})) \rightarrow \pi(h, \mathbf{f}, \mathbf{x}), \tag{3}$$

holds for all $\mathbf{x} \in \mathbb{R}^n$, where

$$\pi^{(i)}(p, \mathbf{f}, \mathbf{x}) \;\widehat{=}\; \left(\bigwedge_{0 \leq j < i} L^j_{\mathbf{f}} p(\mathbf{x}) = 0 \right) \wedge L^i_{\mathbf{f}} p(\mathbf{x}) < 0,$$

$$\pi(p, \mathbf{f}, \mathbf{x}) \;\widehat{=}\; \bigvee_{0 \leq i \leq N_{p,\mathbf{f}}} \pi^{(i)}(p, \mathbf{f}, \mathbf{x}).$$

Theoretical Results

Theorem (Rank Theorem)

Given a polynomial p and a PVF $\mathbf{f}$, there is a natural number $N_{p,\mathbf{f}}$ such that for any $\mathbf{x} \in \mathbb{R}^n$, if $\gamma_{p,\mathbf{f}}(\mathbf{x}) < \infty$, then $\gamma_{p,\mathbf{f}}(\mathbf{x}) \leq N_{p,\mathbf{f}}$.

Theorem (Parametric Rank Theorem)

Given a parametric polynomial $p(\mathbf{u}, \mathbf{x})$ and a PVF $\mathbf{f}$, there is an integer $N_{p,\mathbf{f}} \in \mathbb{N}$ such that $\gamma_{p_{\mathbf{u_0}},\mathbf{f}}(\mathbf{x}) < \infty$ implies $\gamma_{p_{\mathbf{u_0}},\mathbf{f}}(\mathbf{x}) \leq N_{p,\mathbf{f}}$ for all $\mathbf{x} \in \mathbb{R}^n$ and all $\mathbf{u_0} \in \mathbb{R}^w$.

Theorem (Criterion Theorem)

Given a polynomial p, $p(\mathbf{x}) \geq 0$ is an SCI of the PCCDS $(h(\mathbf{x}) \geq 0, \mathbf{f})$ iff

$$\theta(h, p, \mathbf{f}, \mathbf{x}) \widehat{=} (p(\mathbf{x}) = 0 \wedge \pi(p, \mathbf{f}, \mathbf{x})) \rightarrow \pi(h, \mathbf{f}, \mathbf{x}), \tag{3}$$

holds for all $\mathbf{x} \in \mathbb{R}^n$, where

$$\pi^{(i)}(p, \mathbf{f}, \mathbf{x}) \quad \widehat{=} \quad \left(\bigwedge_{0 \leq j < i} L_{\mathbf{f}}^j p(\mathbf{x}) = 0 \right) \wedge L_{\mathbf{f}}^i p(\mathbf{x}) < 0 \,,$$

$$\pi(p, \mathbf{f}, \mathbf{x}) \quad \widehat{=} \quad \bigvee_{0 \leq i \leq N_{p,\mathbf{f}}} \pi^{(i)}(p, \mathbf{f}, \mathbf{x}) \,.$$

Theoretical Results

Theorem (Rank Theorem)

Given a polynomial p and a PVF $\mathbf{f}$, there is a natural number $N_{p,\mathbf{f}}$ such that for any $\mathbf{x} \in \mathbb{R}^n$, if $\gamma_{p,\mathbf{f}}(\mathbf{x}) < \infty$, then $\gamma_{p,\mathbf{f}}(\mathbf{x}) \leq N_{p,\mathbf{f}}$.

Theorem (Parametric Rank Theorem)

Given a parametric polynomial $p(\mathbf{u}, \mathbf{x})$ and a PVF $\mathbf{f}$, there is an integer $N_{p,\mathbf{f}} \in \mathbb{N}$ such that $\gamma_{p_{\mathbf{u_0}},\mathbf{f}}(\mathbf{x}) < \infty$ implies $\gamma_{p_{\mathbf{u_0}},\mathbf{f}}(\mathbf{x}) \leq N_{p,\mathbf{f}}$ for all $\mathbf{x} \in \mathbb{R}^n$ and all $\mathbf{u_0} \in \mathbb{R}^w$.

Theorem (Criterion Theorem)

Given a polynomial p, $p(\mathbf{x}) \geq 0$ is an SCI of the PCCDS $(h(\mathbf{x}) \geq 0, \mathbf{f})$ iff

$$\theta(h, p, \mathbf{f}, \mathbf{x}) \,\widehat{=}\, \big(p(\mathbf{x}) = 0 \wedge \pi(p, \mathbf{f}, \mathbf{x})\big) \rightarrow \pi(h, \mathbf{f}, \mathbf{x}), \tag{3}$$

holds for all $\mathbf{x} \in \mathbb{R}^n$, where

$$\begin{aligned} \pi^{(i)}(p, \mathbf{f}, \mathbf{x}) \quad &\widehat{=} \quad \left(\bigwedge_{0 \leq j < i} L^j_{\mathbf{f}} p(\mathbf{x}) = 0 \right) \wedge L^i_{\mathbf{f}} p(\mathbf{x}) < 0\,, \\ \pi(p, \mathbf{f}, \mathbf{x}) \quad &\widehat{=} \quad \bigvee_{0 \leq i \leq N_{p,\mathbf{f}}} \pi^{(i)}(p, \mathbf{f}, \mathbf{x})\,. \end{aligned}$$

Algorithm

I. **First, set a simple semi-algebraic template $P \,\widehat{=}\, p(\mathbf{u}, \mathbf{x}) \geq 0$ using a parametric polynomial $p(\mathbf{u}, \mathbf{x})$.**

II. Then apply QE to the formula $\forall \mathbf{x}.\theta(h, p, \mathbf{f}, \mathbf{x})$. In practice, QE may be applied to a formula $\forall \mathbf{x}.(\theta \wedge \phi)$, where ϕ is a formula imposing some additional constraint on the SCI P. If the output of QE is *false*, then there is no SCI in the form of the predefined P; otherwise, a constraint on $\mathbf{u}$, denoted by $R(\mathbf{u})$, will be returned.

III. Now, use an SMT solver like Z3 to pick a $\mathbf{u}_0 \in R(\mathbf{u})$ and then $p_{\mathbf{u}_0}(\mathbf{x}) \geq 0$ is an SCI of $(h(\mathbf{x}) \geq 0, \mathbf{f})$.

Algorithm

I. **First, set a simple semi-algebraic template $P \widehat{=} p(\mathbf{u}, \mathbf{x}) \geq 0$ using a parametric polynomial $p(\mathbf{u}, \mathbf{x})$.**

II. **Then apply QE to the formula $\forall \mathbf{x}.\theta(h, p, \mathbf{f}, \mathbf{x})$. In practice, QE may be applied to a formula $\forall \mathbf{x}.(\theta \wedge \phi)$, where ϕ is a formula imposing some additional constraint on the SCI P. If the output of QE is *false*, then there is no SCI in the form of the predefined P; otherwise, a constraint on $\mathbf{u}$, denoted by $R(\mathbf{u})$, will be returned.**

III. Now, use an SMT solver like **Z3** to pick a $\mathbf{u}_0 \in R(\mathbf{u})$ and then $p_{\mathbf{u}_0}(\mathbf{x}) \geq 0$ is an SCI of $(h(\mathbf{x}) \geq 0, \mathbf{f})$.

Algorithm

I. First, set a simple semi-algebraic template $P \,\widehat{=}\, p(\mathbf{u}, \mathbf{x}) \geq 0$ using a parametric polynomial $p(\mathbf{u}, \mathbf{x})$.

II. Then apply **QE** to the formula $\forall \mathbf{x}.\theta(h, p, \mathbf{f}, \mathbf{x})$. In practice, **QE** may be applied to a formula $\forall \mathbf{x}.(\theta \wedge \phi)$, where ϕ is a formula imposing some additional constraint on the SCI P. If the output of **QE** is *false*, then there is no SCI in the form of the predefined P; otherwise, a constraint on $\mathbf{u}$, denoted by $R(\mathbf{u})$, will be returned.

III. Now, use an SMT solver like **Z3** to pick a $\mathbf{u}_0 \in R(\mathbf{u})$ and then $p_{\mathbf{u}_0}(\mathbf{x}) \geq 0$ is an SCI of $(h(\mathbf{x}) \geq 0, \mathbf{f})$.

Running Example

Consider a PDS $(D = -x - y^2 \geq 0, \mathbf{f}(x, y) = (-2y, x^2))$.
Apply procedure (I-III), we have:

I Set a template $P \mathrel{\widehat{=}} p(\mathbf{u}, \mathbf{x}) \geq 0$ with $p(\mathbf{u}, \mathbf{x}) \mathrel{\widehat{=}} ay(x - y)$, where $\mathbf{u} \mathrel{\widehat{=}} (a)$. By a simple computation we get $N_{p,\mathbf{f}} = 2$.

II Compute the corresponding formula

$$\begin{aligned}\theta(h, p, \mathbf{f}, \mathbf{x}) \quad &\widehat{=} \quad p = 0 \wedge (\pi^{(0)}_{p,\mathbf{f},\mathbf{x}} \vee \pi^{(1)}_{p,\mathbf{f},\mathbf{x}} \vee \pi^{(2)}_{p,\mathbf{f},\mathbf{x}}) \longrightarrow \\ &\quad (\pi^{(0)}_{h,\mathbf{f},\mathbf{x}} \vee \pi^{(1)}_{h,\mathbf{f},\mathbf{x}} \vee \pi^{(2)}_{h,\mathbf{f},\mathbf{x}})\end{aligned}$$

where

$$\begin{aligned}
\pi^{(0)}_{h,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad -x - y^2 < 0, \\
\pi^{(1)}_{h,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad -x - y^2 = 0 \wedge 2y - 2x^2y < 0, \\
\pi^{(2)}_{h,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad -x - y^2 = 0 \wedge 2y - 2x^2y = 0 \ \wedge 8xy^2 + 2x^2 - 2x^4 < 0, \\
\pi^{(0)}_{p,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad ay(x - y) < 0, \\
\pi^{(1)}_{p,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad ay(x - y) = 0 \wedge -2ay^2 + ax^3 - 2yax^2 < 0, \\
\pi^{(2)}_{p,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad ay(x - y) = 0 \wedge -2ay^2 + ax^3 - 2yax^2 = 0 \\
&\qquad \wedge 40axy^2 - 16ay^3 + 32ax^3y - 10ax^4 < 0.
\end{aligned}$$

Running Example

Consider a PDS $(D = -x - y^2 \geq 0, \mathbf{f}(x, y) = (-2y, x^2))$.
Apply procedure (I-III), we have:

I Set a template $P \,\widehat{=}\, p(\mathbf{u}, \mathbf{x}) \geq 0$ with $p(\mathbf{u}, \mathbf{x}) \,\widehat{=}\, ay(x - y)$, where $\mathbf{u} \,\widehat{=}\, (a)$. By a simple computation we get $N_{p,\mathbf{f}} = 2$.

II Compute the corresponding formula

$$\begin{aligned}\theta(h, p, \mathbf{f}, \mathbf{x}) \quad &\widehat{=} \quad p = 0 \wedge (\pi^{(0)}_{p,\mathbf{f},\mathbf{x}} \vee \pi^{(1)}_{p,\mathbf{f},\mathbf{x}} \vee \pi^{(2)}_{p,\mathbf{f},\mathbf{x}}) \longrightarrow \\ &\quad\quad (\pi^{(0)}_{h,\mathbf{f},\mathbf{x}} \vee \pi^{(1)}_{h,\mathbf{f},\mathbf{x}} \vee \pi^{(2)}_{h,\mathbf{f},\mathbf{x}})\end{aligned}$$

where

$$\begin{aligned}\pi^{(0)}_{h,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad -x - y^2 < 0, \\ \pi^{(1)}_{h,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad -x - y^2 = 0 \wedge 2y - 2x^2y < 0, \\ \pi^{(2)}_{h,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad -x - y^2 = 0 \wedge 2y - 2x^2y = 0 \wedge 8xy^2 + 2x^2 - 2x^4 < 0, \\ \pi^{(0)}_{p,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad ay(x - y) < 0, \\ \pi^{(1)}_{p,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad ay(x - y) = 0 \wedge -2ay^2 + ax^3 - 2yax^2 < 0, \\ \pi^{(2)}_{p,\mathbf{f},\mathbf{x}} \quad &\widehat{=} \quad ay(x - y) = 0 \wedge -2ay^2 + ax^3 - 2yax^2 = 0 \\ &\quad\quad \wedge 40axy^2 - 16ay^3 + 32ax^3y - 10ax^4 < 0.\end{aligned}$$

Running Example

Consider a PDS $(D = -x - y^2 \geq 0, \mathbf{f}(x, y) = (-2y, x^2))$.
Apply procedure (I-III), we have:

I Set a template $P \widehat{=} p(\mathbf{u}, \mathbf{x}) \geq 0$ with $p(\mathbf{u}, \mathbf{x}) \widehat{=} ay(x - y)$, where $\mathbf{u} \widehat{=} (a)$. By a simple computation we get $N_{p,\mathbf{f}} = 2$.

II Compute the corresponding formula

$$\begin{aligned} \theta(h, p, \mathbf{f}, \mathbf{x}) \quad \widehat{=} \quad & p = 0 \wedge (\pi^{(0)}_{p,\mathbf{f},\mathbf{x}} \vee \pi^{(1)}_{p,\mathbf{f},\mathbf{x}} \vee \pi^{(2)}_{p,\mathbf{f},\mathbf{x}}) \longrightarrow \\ & (\pi^{(0)}_{h,\mathbf{f},\mathbf{x}} \vee \pi^{(1)}_{h,\mathbf{f},\mathbf{x}} \vee \pi^{(2)}_{h,\mathbf{f},\mathbf{x}}) \end{aligned}$$

where

$$\begin{aligned} \pi^{(0)}_{h,\mathbf{f},\mathbf{x}} \quad & \widehat{=} \quad -x - y^2 < 0, \\ \pi^{(1)}_{h,\mathbf{f},\mathbf{x}} \quad & \widehat{=} \quad -x - y^2 = 0 \wedge 2y - 2x^2 y < 0, \\ \pi^{(2)}_{h,\mathbf{f},\mathbf{x}} \quad & \widehat{=} \quad -x - y^2 = 0 \wedge 2y - 2x^2 y = 0 \ \wedge 8xy^2 + 2x^2 - 2x^4 < 0, \\ \pi^{(0)}_{p,\mathbf{f},\mathbf{x}} \quad & \widehat{=} \quad ay(x - y) < 0, \\ \pi^{(1)}_{p,\mathbf{f},\mathbf{x}} \quad & \widehat{=} \quad ay(x - y) = 0 \wedge -2ay^2 + ax^3 - 2yax^2 < 0, \\ \pi^{(2)}_{p,\mathbf{f},\mathbf{x}} \quad & \widehat{=} \quad ay(x - y) = 0 \wedge -2ay^2 + ax^3 - 2yax^2 = 0 \\ & \qquad \wedge 40axy^2 - 16ay^3 + 32ax^3 y - 10ax^4 < 0. \end{aligned}$$

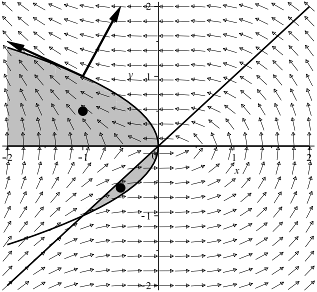
Running Example (Cont'd)

III In addition, we require the two points $\{(-1, 0.5), (-0.5, -0.6)\}$ to be contained in P. Then apply **QE** to the formula

$$\forall x \forall y. (\theta(h, p, \mathbf{f}, \mathbf{x}) \wedge 0.5a(-1 - 0.5) \geq 0 \wedge -0.6a(-0.5 + 0.6) \geq 0).$$

The result is $a \leq 0$.

IV Just pick $a = -1$, and then $-xy + y^2 \geq 0$ is an SCI of $(D, \mathbf{f})$. The grey part of Picture III is the intersection of the invariant P and domain D.



III: SCI in Simple Case

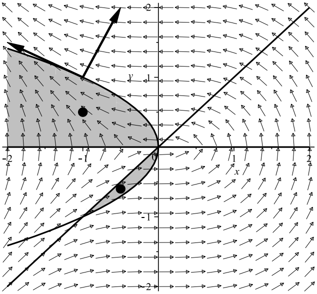
Running Example (Cont'd)

III In addition, we require the two points $\{(-1, 0.5), (-0.5, -0.6)\}$ to be contained in P. Then apply **QE** to the formula

$$\forall x \forall y. (\theta(h, p, \mathbf{f}, \mathbf{x}) \wedge 0.5a(-1-0.5) \geq 0 \wedge -0.6a(-0.5+0.6) \geq 0).$$

The result is $a \leq 0$.

IV Just pick $a = -1$, and then $-xy + y^2 \geq 0$ is an SCI of $(D, \mathbf{f})$. The grey part of Picture III is the intersection of the invariant P and domain D.



III: SCI in Simple Case

Outline

General Case

- **Problem:** Consider a PDS $(D, \mathbf{f})$ with

$$D = \bigvee_{i=1}^{I} \bigwedge_{j=1}^{J_i} p_{ij}(\mathbf{x}) \triangleright 0,$$

and $\mathbf{f} \in \mathbb{Q}^n[\mathbf{x}]$, where $\triangleright \in \{\geq, >\}$, to generate SAIs automatically with a general template

$$P = \bigvee_{k=1}^{K} \bigwedge_{l=1}^{L_k} p_{kl}(\mathbf{u}_{kl}, \mathbf{x}) \triangleright 0\,, \ \triangleright \in \{\geq, >\}$$

- **Basic idea** The procedure is essentially same as in the simple case, but have to sophisticatedly handle the complex combinations due to the complicated boundaries.

General Case

- **Problem:** Consider a PDS $(D, \mathbf{f})$ with

$$D = \bigvee_{i=1}^{I} \bigwedge_{j=1}^{J_i} p_{ij}(\mathbf{x}) \triangleright 0,$$

 and $\mathbf{f} \in \mathbb{Q}^n[\mathbf{x}]$, where $\triangleright \in \{\geq, >\}$, to generate SAIs automatically with a general template

$$P = \bigvee_{k=1}^{K} \bigwedge_{l=1}^{L_k} p_{kl}(\mathbf{u}_{kl}, \mathbf{x}) \triangleright 0\,,\ \triangleright \in \{\geq, >\}$$

- **Basic idea** The procedure is essentially same as in the simple case, but have to sophisticatedly handle the complex combinations due to the complicated boundaries.

Theorem (Main Result)

A semi-algebraic template $P(\mathbf{u}, \mathbf{x})$ *defined by*

$$\bigvee_{k=1}^{K} \left(\bigwedge_{j=1}^{j_k} p_{kj}(\mathbf{u}_{kj}, \mathbf{x}) \geq 0 \quad \wedge \bigwedge_{j=j_k+1}^{J_k} p_{kj}(\mathbf{u}_{kj}, \mathbf{x}) > 0 \right)$$

is a CI of the PCCDS $(D, \mathbf{f})$ *with*

$$D \mathrel{\widehat{=}} \bigvee_{m=1}^{M} \left(\bigwedge_{l=1}^{l_m} p_{ml}(\mathbf{x}) \geq 0 \quad \wedge \bigwedge_{l=l_m+1}^{L_m} p_{ml}(\mathbf{x}) > 0 \right),$$

iff $\mathbf{u}$ *satisfies*

$$\forall \mathbf{x}. \Big(\big(P \wedge D \wedge \Phi_D \rightarrow \Phi_P\big) \wedge \big(\neg P \wedge D \wedge \Phi_D^{Iv} \rightarrow \neg \Phi_P^{Iv}\big) \Big),$$

where

Theorem (Main Result (Cont'd))

$$\Phi_D \widehat{=} \bigvee_{m=1}^{M} \left(\bigwedge_{l=1}^{l_m} \psi_0^+(p_{ml}, \mathbf{f}) \wedge \bigwedge_{l=l_m+1}^{L_m} \psi^+(p_{ml}, \mathbf{f}) \right),$$

$$\Phi_P \widehat{=} \bigvee_{k=1}^{K} \left(\bigwedge_{j=1}^{j_k} \psi_0^+(p_{kj}, \mathbf{f}) \wedge \bigwedge_{j=j_k+1}^{J_k} \psi^+(p_{kj}, \mathbf{f}) \right),$$

$$\Phi_D^{Iv} \widehat{=} \bigvee_{m=1}^{M} \left(\bigwedge_{l=1}^{l_m} \varphi_0^+(p_{ml}, \mathbf{f}) \wedge \bigwedge_{l=l_m+1}^{L_m} \varphi^+(p_{ml}, \mathbf{f}) \right),$$

$$\Phi_P^{Iv} \widehat{=} \bigvee_{k=1}^{K} \left(\bigwedge_{j=1}^{j_k} \varphi_0^+(p_{kj}, \mathbf{f}) \wedge \bigwedge_{j=j_k+1}^{J_k} \varphi^+(p_{kj}, \mathbf{f}) \right),$$

$$\psi^+(p, \mathbf{f}) \widehat{=} \bigvee_{0 \le i \le N_{p,\mathbf{f}}} \psi^{(i)}(p, \mathbf{f}) \textit{ with } \psi^{(i)}(p, \mathbf{f}) \widehat{=} \Big(\bigwedge_{0 \le j < i} L_{\mathbf{f}}^j p = 0 \Big) \wedge L_{\mathbf{f}}^i p > 0, \textit{ and}$$

$$\psi_0^+(p, \mathbf{f}) \widehat{=} \psi^+(p, \mathbf{f}) \vee \Big(\bigwedge_{0 \le j \le N_{p,\mathbf{f}}} L_{\mathbf{f}}^j p = 0 \Big)$$

$$\varphi^+(p, \mathbf{f}) \widehat{=} \bigvee_{0 \le i \le N_{p,\mathbf{f}}} \varphi^{(i)}(p, \mathbf{f}) \textit{ with } \varphi^{(i)}(p, \mathbf{f}) \widehat{=} \Big(\bigwedge_{0 \le j < i} L_{\mathbf{f}}^j p = 0 \Big) \wedge (-1)^i \cdot L_{\mathbf{f}}^i p > 0, \textit{ and}$$

$$\varphi_0^+(p, \mathbf{f}) \widehat{=} \varphi^+(p, \mathbf{f}) \vee \Big(\bigwedge_{0 \le j \le N_{p,\mathbf{f}}} L_{\mathbf{f}}^j p = 0 \Big).$$

Running Example

- Let $\mathbf{f}(x, y) = (-2y, x^2)$ and $D \mathrel{\widehat{=}} \mathbb{R}^2$.
- Take a template: $P(\mathbf{u}, \mathbf{x}) \mathrel{\widehat{=}} x - a \geq 0 \vee y - b > 0$ with $\mathbf{u} = (a, b)$.
- So, P is an SCI of $(D, \mathbf{f})$ iff a, b satisfy

$$\forall x \forall y. (P \to \zeta) \wedge (\neg P \to \neg \xi),$$

where

$$\begin{aligned}
\zeta \mathrel{\widehat{=}} & (x - a > 0) \vee (x - a = 0 \wedge -2y > 0) \\
& \vee (x - a = 0 \wedge -2y = 0 \wedge -2x^2 \geq 0) \\
& \vee (y - b > 0) \vee (y - b = 0 \wedge x^2 > 0) \\
& \vee (y - b = 0 \wedge x^2 = 0 \wedge -4yx > 0) \\
& \vee (y - b = 0 \wedge x^2 = 0 \wedge -4yx = 0 \wedge 8y^2 - 4x^3 > 0) \\
\xi \mathrel{\widehat{=}} & (x - a > 0) \vee (x - a = 0 \wedge -2y < 0) \\
& \vee (x - a = 0 \wedge -2y = 0 \wedge -2x^2 \geq 0) \\
& \vee (y - b > 0) \vee (y - b = 0 \wedge x^2 < 0) \\
& \vee (y - b = 0 \wedge x^2 = 0 \wedge -4yx > 0) \\
& \vee (y - b = 0 \wedge x^2 = 0 \wedge -4yx = 0 \wedge 8y^2 - 4x^3 < 0)
\end{aligned}$$

Running Example

- Let $\mathbf{f}(x,y) = (-2y, x^2)$ and $D \mathrel{\widehat{=}} \mathbb{R}^2$.
- Take a template: $P(\mathbf{u},\mathbf{x}) \mathrel{\widehat{=}} x - a \geq 0 \vee y - b > 0$ with $\mathbf{u} = (a,b)$.
- So, P is an SCI of $(D, \mathbf{f})$ iff a, b satisfy
$$\forall x \forall y. (P \to \zeta) \wedge (\neg P \to \neg \xi),$$
where
$$\begin{aligned}
\zeta \mathrel{\widehat{=}} & (x-a>0) \vee (x-a=0 \wedge -2y>0) \\
& \vee (x-a=0 \wedge -2y=0 \wedge -2x^2 \geq 0) \\
& \vee (y-b>0) \vee (y-b=0 \wedge x^2>0) \\
& \vee (y-b=0 \wedge x^2=0 \wedge -4yx>0) \\
& \vee (y-b=0 \wedge x^2=0 \wedge -4yx=0 \wedge 8y^2-4x^3>0) \\
\xi \mathrel{\widehat{=}} & (x-a>0) \vee (x-a=0 \wedge -2y<0) \\
& \vee (x-a=0 \wedge -2y=0 \wedge -2x^2 \geq 0) \\
& \vee (y-b>0) \vee (y-b=0 \wedge x^2<0) \\
& \vee (y-b=0 \wedge x^2=0 \wedge -4yx>0) \\
& \vee (y-b=0 \wedge x^2=0 \wedge -4yx=0 \wedge 8y^2-4x^3<0)
\end{aligned}$$

Running Example

- Let $\mathbf{f}(x, y) = (-2y, x^2)$ and $D \mathrel{\widehat{=}} \mathbb{R}^2$.
- Take a template: $P(\mathbf{u}, \mathbf{x}) \mathrel{\widehat{=}} x - a \geq 0 \vee y - b > 0$ with $\mathbf{u} = (a, b)$.
- So, P is an SCI of $(D, \mathbf{f})$ iff a, b satisfy

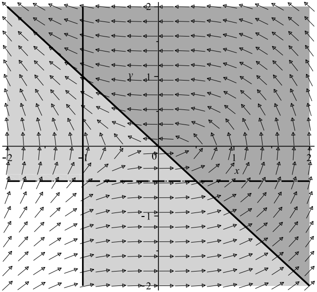
$$\forall x \forall y.(P \rightarrow \zeta) \wedge (\neg P \rightarrow \neg \xi),$$

where

$$\begin{aligned}
\zeta \mathrel{\widehat{=}} & (x - a > 0) \vee (x - a = 0 \wedge -2y > 0) \\
& \vee (x - a = 0 \wedge -2y = 0 \wedge -2x^2 \geq 0) \\
& \vee (y - b > 0) \vee (y - b = 0 \wedge x^2 > 0) \\
& \vee (y - b = 0 \wedge x^2 = 0 \wedge -4yx > 0) \\
& \vee (y - b = 0 \wedge x^2 = 0 \wedge -4yx = 0 \wedge 8y^2 - 4x^3 > 0) \\
\xi \mathrel{\widehat{=}} & (x - a > 0) \vee (x - a = 0 \wedge -2y < 0) \\
& \vee (x - a = 0 \wedge -2y = 0 \wedge -2x^2 \geq 0) \\
& \vee (y - b > 0) \vee (y - b = 0 \wedge x^2 < 0) \\
& \vee (y - b = 0 \wedge x^2 = 0 \wedge -4yx > 0) \\
& \vee (y - b = 0 \wedge x^2 = 0 \wedge -4yx = 0 \wedge 8y^2 - 4x^3 < 0)
\end{aligned}$$

Running Example (Cont'd)

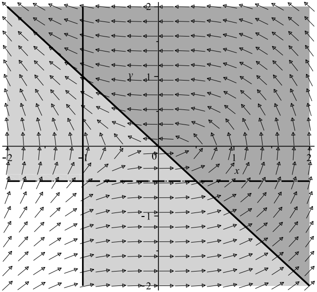
- In addition, we require the set $x + y \geq 0$ to be contained in P.
- By applying **QE**, we get $a + b \leq 0 \wedge b \leq 0$.
- Let $a = -1$ and $b = -0.5$, and we obtain an SCI $P \hat{=} x + 1 \geq 0 \vee y + 0.5 > 0$.



IV: SCI in General Case

Running Example (Cont'd)

- In addition, we require the set $x + y \geq 0$ to be contained in P.
- By applying **QE**, we get $a + b \leq 0 \wedge b \leq 0$.
- Let $a = -1$ and $b = -0.5$, and we obtain an SCI $P \widehat{=} x + 1 \geq 0 \vee y + 0.5 > 0$.



IV: SCI in General Case

Outline

Algorithm

I. Predefine a familiy of semi-algebraic templates $I_q(\mathbf{u}, \mathbf{x})$ with degree bound d for each $q \in Q$, as the SCI to be generated at mode q.

II. Translate conditions for the family of $I_q(\mathbf{u}, \mathbf{x})$ to be a GI of $\mathcal{H}$, i.e.
- $\Xi_q \subseteq I_q$ for all $q \in Q$;
- for any $e = (q, q') \in E$, if $\mathbf{x} \in I_q \cap G_e$, then $\mathbf{x}' = R_e(\mathbf{x}) \in I_{q'}$;
- for any $q \in Q$, I_q is a CI of $(D_q, \mathbf{f}_q)$

into a set of first-order real arithmetic formulas, i.e.

(1) $\forall \mathbf{x}. (\Xi_q \rightarrow I_q(\mathbf{u}, \mathbf{x}))$ for all $q \in Q$;

(2) $\forall \mathbf{x}, \mathbf{x}'. (I_q(\mathbf{u}, \mathbf{x}) \wedge G_e \wedge \mathbf{x}' = R_e(\mathbf{x}) \rightarrow I_{q'}(\mathbf{u}, \mathbf{x}'))$ for all $q \in Q$ and all $e = (q, q') \in E$;

(3) $\forall \mathbf{x}. ((I_q(\mathbf{u}, \mathbf{x}) \wedge D_q \wedge \Phi_{D_q} \rightarrow \Phi_{I_q}) \wedge (\neg I_q(\mathbf{u}, \mathbf{x}) \wedge D_q \wedge \Phi^{\mathrm{Iv}}_{D_q} \rightarrow \neg \Phi^{\mathrm{Iv}}_{I_q}))$, for each $q \in Q$.

For safety property $\mathcal{S}$, there may be a fourth set of formulas:

(4) $\forall \mathbf{x}. (I_q(\mathbf{u}, \mathbf{x}) \longrightarrow S_q)$ for all $q \in Q$.

III. Take the conjunction of all the formulas in Step 2 and apply QE to get a QFF $\phi(\mathbf{u})$. Then choose a specific $\mathbf{u}_0$ from $\phi(\mathbf{u})$ with a tool like **Z3**, and the set of instantiations $I_{q,\mathbf{u}_0}(\mathbf{x})$ form a GI of $\mathcal{H}$.

Algorithm

I. Predefine a familiy of semi-algebraic templates $I_q(\mathbf{u}, \mathbf{x})$ with degree bound d for each $q \in Q$, as the SCI to be generated at mode q.

II. Translate conditions for the family of $I_q(\mathbf{u}, \mathbf{x})$ to be a GI of $\mathcal{H}$, i.e.

- $\Xi_q \subseteq I_q$ for all $q \in Q$;
- for any $e = (q, q') \in E$, if $\mathbf{x} \in I_q \cap G_e$, then $\mathbf{x}' = R_e(\mathbf{x}) \in I_{q'}$;
- for any $q \in Q$, I_q is a CI of $(D_q, \mathbf{f}_q)$

into a set of first-order real arithmetic formulas, i.e.

(1) $\forall \mathbf{x}. \big(\Xi_q \rightarrow I_q(\mathbf{u}, \mathbf{x})\big)$ for all $q \in Q$;

(2) $\forall \mathbf{x}, \mathbf{x}'. \big(I_q(\mathbf{u}, \mathbf{x}) \wedge G_e \wedge \mathbf{x}' = R_e(\mathbf{x}) \rightarrow I_{q'}(\mathbf{u}, \mathbf{x}')\big)$ for all $q \in Q$ and all $e = (q, q') \in E$;

(3) $\forall \mathbf{x}. \big((I_q(\mathbf{u}, \mathbf{x}) \wedge D_q \wedge \Phi_{D_q} \rightarrow \Phi_{I_q}) \wedge (\neg I_q(\mathbf{u}, \mathbf{x}) \wedge D_q \wedge \Phi^{\mathrm{Iv}}_{D_q} \rightarrow \neg \Phi^{\mathrm{Iv}}_{I_q})\big)$, for each $q \in Q$.

For safety property $\mathcal{S}$, there may be a fourth set of formulas:

(4) $\forall \mathbf{x}. (I_q(\mathbf{u}, \mathbf{x}) \longrightarrow S_q)$ for all $q \in Q$.

III. Take the conjunction of all the formulas in Step 2 and apply QE to get a QFF $\phi(\mathbf{u})$. Then choose a specific $\mathbf{u}_0$ from $\phi(\mathbf{u})$ with a tool like **Z3**, and the set of instantiations $I_{q,\mathbf{u}_0}(\mathbf{x})$ form a GI of $\mathcal{H}$.

Algorithm

I. Predefine a familiy of semi-algebraic templates $I_q(\mathbf{u}, \mathbf{x})$ with degree bound d for each $q \in Q$, as the SCI to be generated at mode q.

II. Translate conditions for the family of $I_q(\mathbf{u}, \mathbf{x})$ to be a GI of $\mathcal{H}$, i.e.
   - $\Xi_q \subseteq I_q$ for all $q \in Q$;
   - for any $e = (q, q') \in E$, if $\mathbf{x} \in I_q \cap G_e$, then $\mathbf{x}' = R_e(\mathbf{x}) \in I_{q'}$;
   - for any $q \in Q$, I_q is a CI of $(D_q, \mathbf{f}_q)$

   into a set of first-order real arithmetic formulas, i.e.

   (1) $\forall \mathbf{x}. \big(\Xi_q \rightarrow I_q(\mathbf{u}, \mathbf{x})\big)$ for all $q \in Q$;

   (2) $\forall \mathbf{x}, \mathbf{x}'. \big(I_q(\mathbf{u}, \mathbf{x}) \wedge G_e \wedge \mathbf{x}' = R_e(\mathbf{x}) \rightarrow I_{q'}(\mathbf{u}, \mathbf{x}')\big)$ for all $q \in Q$ and all $e = (q, q') \in E$;

   (3) $\forall \mathbf{x}. \big((I_q(\mathbf{u}, \mathbf{x}) \wedge D_q \wedge \Phi_{D_q} \rightarrow \Phi_{I_q}) \wedge (\neg I_q(\mathbf{u}, \mathbf{x}) \wedge D_q \wedge \Phi^{\mathrm{Iv}}_{D_q} \rightarrow \neg \Phi^{\mathrm{Iv}}_{I_q})\big)$, for each $q \in Q$.

   For safety property $\mathcal{S}$, there may be a fourth set of formulas:

   (4) $\forall \mathbf{x}. (I_q(\mathbf{u}, \mathbf{x}) \longrightarrow S_q)$ for all $q \in Q$.

III. Take the conjunction of all the formulas in Step 2 and apply QE to get a QFF $\phi(\mathbf{u})$. Then choose a specific $\mathbf{u}_0$ from $\phi(\mathbf{u})$ with a tool like **Z3**, and the set of instantiations $I_{q,\mathbf{u}_0}(\mathbf{x})$ form a GI of $\mathcal{H}$.

Algorithm

I. Predefine a familiy of semi-algebraic templates $I_q(\mathbf{u}, \mathbf{x})$ with degree bound d for each $q \in Q$, as the SCI to be generated at mode q.

II. Translate conditions for the family of $I_q(\mathbf{u}, \mathbf{x})$ to be a GI of $\mathcal{H}$, i.e.
   - $\Xi_q \subseteq I_q$ for all $q \in Q$;
   - for any $e = (q, q') \in E$, if $\mathbf{x} \in I_q \cap G_e$, then $\mathbf{x}' = R_e(\mathbf{x}) \in I_{q'}$;
   - for any $q \in Q$, I_q is a CI of $(D_q, \mathbf{f}_q)$

   into a set of first-order real arithmetic formulas, i.e.

   (1) $\forall \mathbf{x}. \big(\Xi_q \to I_q(\mathbf{u}, \mathbf{x})\big)$ for all $q \in Q$;

   (2) $\forall \mathbf{x}, \mathbf{x}'. \big(I_q(\mathbf{u}, \mathbf{x}) \wedge G_e \wedge \mathbf{x}' = R_e(\mathbf{x}) \to I_{q'}(\mathbf{u}, \mathbf{x}')\big)$ for all $q \in Q$ and all $e = (q, q') \in E$;

   (3) $\forall \mathbf{x}. \big((I_q(\mathbf{u}, \mathbf{x}) \wedge D_q \wedge \Phi_{D_q} \to \Phi_{I_q}) \wedge (\neg I_q(\mathbf{u}, \mathbf{x}) \wedge D_q \wedge \Phi^{\mathrm{Iv}}_{D_q} \to \neg \Phi^{\mathrm{Iv}}_{I_q})\big)$, for each $q \in Q$.

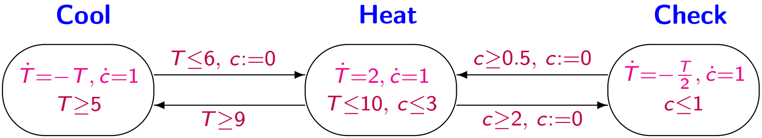
   For safety property $\mathcal{S}$, there may be a fourth set of formulas:

   (4) $\forall \mathbf{x}. (I_q(\mathbf{u}, \mathbf{x}) \longrightarrow S_q)$ for all $q \in Q$.

III. Take the conjunction of all the formulas in Step 2 and apply QE to get a QFF $\phi(\mathbf{u})$. Then choose a specific $\mathbf{u}_0$ from $\phi(\mathbf{u})$ with a tool like **Z3**, and the set of instantiations $I_{q,\mathbf{u_0}}(\mathbf{x})$ form a GI of $\mathcal{H}$.

Running Example

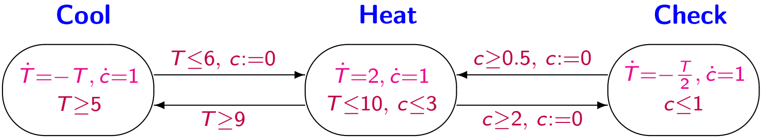
- The Thermostat can be described by the HA in following figure.



- To verify that under the initial condition $\Xi_{\mathcal{H}} \mathrel{\widehat{=}} \{q_{\rm ht}\} \times X_0$ with $X_0 \mathrel{\widehat{=}} c = 0 \wedge 5 \leq T \leq 10$, $S \mathrel{\widehat{=}} T \geq 4.5$ is satisfied at all modes.

Running Example

- The Thermostat can be described by the HA in following figure.



- To verify that under the initial condition $\Xi_{\mathcal{H}} \widehat{=} \{q_{\rm ht}\} \times X_0$ with $X_0 \widehat{=} c = 0 \wedge 5 \leq T \leq 10$, $S \widehat{=} T \geq 4.5$ is satisfied at all modes.

Running Example (Cont'd)

- Firstly, predefine the following set of templates:
  - $I_{q_{\rm ht}} \widehat{=} T + a_1 c + a_0 \geq 0 \wedge c \geq 0$;
  - $I_{q_{\rm cl}} \widehat{=} T + a_2 \geq 0$;
  - $I_{q_{\rm ck}} \widehat{=} T \geq a_3 c^2 - 4.5c + 9 \wedge c \geq 0 \wedge c \leq 1$
- By the second step, we get

$$10a_3 - 9 \leq 0 \wedge 2a_3 - 1 \geq 0 \wedge a_1 + 2 = 0 \wedge a_0 + 2a_1 + 9 = 0 \wedge a_2 - a_0 = 0\,.$$

- By choosing $a_0 = -5, a_1 = -2, a_2 = -5, a_3 = \frac{1}{2}$, obtain the following SGI
  - $I_{q_{\rm ht}} \widehat{=} T \geq 2c + 5 \wedge c \geq 0$;
  - $I_{q_{\rm cl}} \widehat{=} T \geq 5$;
  - $I_{q_{\rm ck}} \widehat{=} 2T \geq c^2 - 9c + 18 \wedge c \geq 0 \wedge c \leq 1$.
- The safety property is successfully verified by the SGI.

Running Example (Cont'd)

- Firstly, predefine the following set of templates:
  - $I_{q_{\rm ht}} \widehat{=} T + a_1 c + a_0 \geq 0 \wedge c \geq 0$;
  - $I_{q_{\rm cl}} \widehat{=} T + a_2 \geq 0$;
  - $I_{q_{\rm ck}} \widehat{=} T \geq a_3 c^2 - 4.5c + 9 \wedge c \geq 0 \wedge c \leq 1$
- By the second step, we get

$$10a_3 - 9 \leq 0 \wedge 2a_3 - 1 \geq 0 \wedge a_1 + 2 = 0 \wedge a_0 + 2a_1 + 9 = 0 \wedge a_2 - a_0 = 0\,.$$

- By choosing $a_0 = -5, a_1 = -2, a_2 = -5, a_3 = \frac{1}{2}$, obtain the following SGI
  - $I_{q_{\rm ht}} \widehat{=} T \geq 2c + 5 \wedge c \geq 0$;
  - $I_{q_{\rm cl}} \widehat{=} T \geq 5$;
  - $I_{q_{\rm ck}} \widehat{=} 2T \geq c^2 - 9c + 18 \wedge c \geq 0 \wedge c \leq 1$.
- **The safety property is successfully verified by the SGI.**

Running Example (Cont'd)

- Firstly, predefine the following set of templates:
  - $I_{q_{\rm ht}} \widehat{=} T + a_1 c + a_0 \geq 0 \wedge c \geq 0$;
  - $I_{q_{\rm cl}} \widehat{=} T + a_2 \geq 0$;
  - $I_{q_{\rm ck}} \widehat{=} T \geq a_3 c^2 - 4.5c + 9 \wedge c \geq 0 \wedge c \leq 1$
- By the second step, we get

  $$10a_3 - 9 \leq 0 \wedge 2a_3 - 1 \geq 0 \wedge a_1 + 2 = 0 \wedge a_0 + 2a_1 + 9 = 0 \wedge a_2 - a_0 = 0\,.$$

- By choosing $a_0 = -5, a_1 = -2, a_2 = -5, a_3 = \frac{1}{2}$, obtain the following SGI
  - $I_{q_{\rm ht}} \widehat{=} T \geq 2c + 5 \wedge c \geq 0$;
  - $I_{q_{\rm cl}} \widehat{=} T \geq 5$;
  - $I_{q_{\rm ck}} \widehat{=} 2T \geq c^2 - 9c + 18 \wedge c \geq 0 \wedge c \leq 1$.
- The safety property is successfully verified by the SGI.

Running Example (Cont'd)

- Firstly, predefine the following set of templates:
  - $I_{q_{\text{ht}}} \,\widehat{=}\, T + a_1 c + a_0 \geq 0 \wedge c \geq 0$;
  - $I_{q_{\text{cl}}} \,\widehat{=}\, T + a_2 \geq 0$;
  - $I_{q_{\text{ck}}} \,\widehat{=}\, T \geq a_3 c^2 - 4.5c + 9 \wedge c \geq 0 \wedge c \leq 1$
- By the second step, we get

  $$10a_3 - 9 \leq 0 \wedge 2a_3 - 1 \geq 0 \wedge a_1 + 2 = 0 \wedge a_0 + 2a_1 + 9 = 0 \wedge a_2 - a_0 = 0\,.$$

- By choosing $a_0 = -5, a_1 = -2, a_2 = -5, a_3 = \frac{1}{2}$, obtain the following SGI
  - $I_{q_{\text{ht}}} \,\widehat{=}\, T \geq 2c + 5 \wedge c \geq 0$;
  - $I_{q_{\text{cl}}} \,\widehat{=}\, T \geq 5$;
  - $I_{q_{\text{ck}}} \,\widehat{=}\, 2T \geq c^2 - 9c + 18 \wedge c \geq 0 \wedge c \leq 1$.
- **The safety property is successfully verified by the SGI.**

Outline

Elementary Functions

$$\begin{aligned} f, g \quad ::= \quad & c \mid x \mid f+g \mid f-g \mid f \times g \mid \\ & \frac{f}{g} \mid f^a \mid e^f \mid \ln(f) \mid \sin(f) \mid \cos(f) \ , \end{aligned}$$

- $c \in \mathbb{R}$, $a \in \mathbb{Q}$, $x \in \{x, y, z, \ldots, x_1, x_2, \ldots, x_n\}$
- elementary (or polynomial) hybrid system (or CDS), EHS/PHS/EDS/PDS:

Elementary Functions

$$\begin{aligned} f, g \quad ::= \quad & c \mid x \mid f+g \mid f-g \mid f \times g \mid \\ & \frac{f}{g} \mid f^{a} \mid e^{f} \mid \ln(f) \mid \sin(f) \mid \cos(f) \quad , \end{aligned}$$

- $c \in \mathbb{R}$, $a \in \mathbb{Q}$, $x \in \{x, y, z, \ldots, x_1, x_2, \ldots, x_n\}$
- elementary (or polynomial) hybrid system (or CDS), EHS/PHS/EDS/PDS:

Univariate Basic Elementary Functions: $\dot{x} = f(x)$

- $f(x) = \frac{1}{x}$: let $v = \frac{1}{x}$, and thus $\dot{v} = -\frac{\dot{x}}{x^2}$, so (1) is transformed to

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= -v^3 \end{cases}$$

- $f(x) = \sqrt{x}$: let $v = \sqrt{x}$, and thus $\dot{v} = \frac{\dot{x}}{2\sqrt{x}}$, so (1) is transformed to

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= \frac{1}{2} \end{cases}$$

- $f(x) = e^x$: let $v = e^x$, and thus $\dot{v} = e^x \cdot \dot{x}$, so (??) is transformed to

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= v^2 \end{cases}$$

Univariate Basic Elementary Functions: $\dot{x} = f(x)$

- $f(x) = \frac{1}{x}$: let $v = \frac{1}{x}$, and thus $\dot{v} = -\frac{\dot{x}}{x^2}$, so (1) is transformed to

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= -v^3 \end{cases}$$

- $f(x) = \sqrt{x}$: let $v = \sqrt{x}$, and thus $\dot{v} = \frac{\dot{x}}{2\sqrt{x}}$, so (1) is transformed to

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= \frac{1}{2} \end{cases}$$

- $f(x) = e^x$: let $v = e^x$, and thus $\dot{v} = e^x \cdot \dot{x}$, so (??) is transformed to

$$\begin{cases} \dot{x} &= v \\ \dot{v} &= v^2 \end{cases}$$

Univariate Basic Elementary Functions: $\dot{x} = f(x)$

- $f(x) = \frac{1}{x}$: let $v = \frac{1}{x}$, and thus $\dot{v} = -\frac{\dot{x}}{x^2}$, so (1) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & -v^3 \end{cases}$$

- $f(x) = \sqrt{x}$: let $v = \sqrt{x}$, and thus $\dot{v} = \frac{\dot{x}}{2\sqrt{x}}$, so (1) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & \frac{1}{2} \end{cases}$$

- $f(x) = e^x$: let $v = e^x$, and thus $\dot{v} = e^x \cdot \dot{x}$, so (??) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & v^2 \end{cases}$$

Univariate Basic Elementary Functions: $\dot{x} = f(x)$

- $f(x) = \ln x$: let $v = \ln x$, and thus $\dot{v} = \frac{\dot{x}}{x}$; further let $u = \frac{1}{x}$, and thus $\dot{u} = -\frac{\dot{x}}{x^2}$. Therefore (1) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & uv \\ \dot{u} & = & -u^2 v \end{cases}$$

- $f(x) = \sin x$: let $v = \sin x$, and thus $\dot{v} = \dot{x} \cdot \cos x$; further let $u = \cos x$, and thus $\dot{u} = -\sin x \cdot \dot{x}$. Therefore (1) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & uv \\ \dot{u} & = & -v^2 \end{cases}$$

- $f(x) = \cos x$: the transformation is analogous to the case of $f(x) = \sin x$.

Univariate Basic Elementary Functions: $\dot{x} = f(x)$

- $f(x) = \ln x$: let $v = \ln x$, and thus $\dot{v} = \frac{\dot{x}}{x}$; further let $u = \frac{1}{x}$, and thus $\dot{u} = -\frac{\dot{x}}{x^2}$. Therefore (1) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & uv \\ \dot{u} & = & -u^2 v \end{cases}$$

- $f(x) = \sin x$: let $v = \sin x$, and thus $\dot{v} = \dot{x} \cdot \cos x$; further let $u = \cos x$, and thus $\dot{u} = -\sin x \cdot \dot{x}$. Therefore (1) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & uv \\ \dot{u} & = & -v^2 \end{cases}$$

- $f(x) = \cos x$: the transformation is analogous to the case of $f(x) = \sin x$.

Univariate Basic Elementary Functions: $\dot{x} = f(x)$

- $f(x) = \ln x$: let $v = \ln x$, and thus $\dot{v} = \frac{\dot{x}}{x}$; further let $u = \frac{1}{x}$, and thus $\dot{u} = -\frac{\dot{x}}{x^2}$. Therefore (1) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & uv \\ \dot{u} & = & -u^2 v \end{cases}$$

- $f(x) = \sin x$: let $v = \sin x$, and thus $\dot{v} = \dot{x} \cdot \cos x$; further let $u = \cos x$, and thus $\dot{u} = -\sin x \cdot \dot{x}$. Therefore (1) is transformed to

$$\begin{cases} \dot{x} & = & v \\ \dot{v} & = & uv \\ \dot{u} & = & -v^2 \end{cases}$$

- $f(x) = \cos x$: the transformation is analogous to the case of $f(x) = \sin x$.

Compositional and Multivariate Functions

- Compositional: if $f(x) = \ln(2+\sin x)$, then let

$$\begin{cases} v & = \sin x \\ u & = \cos x \\ w & = \ln(2+v) = \ln(2+\sin x) \\ z & = \frac{1}{2+v} = \frac{1}{2+\sin x} \end{cases},$$

so (1) is transformed to

$$\begin{cases} \dot{x} & = & w \\ \dot{v} & = & uw \\ \dot{u} & = & -vw \\ \dot{w} & = & zuw \\ \dot{z} & = & -z^2uw \end{cases}.$$

- Multivariate: analogous.

Abstracting EDSs

Abstracting EDS $\mathcal{C}_{\mathbf{x}} \widehat{=} (\Xi_{\mathbf{x}}, \mathbf{f}_{\mathbf{x}}, D_{\mathbf{x}})$ to PDS $C_{\mathbf{y}} \widehat{=} (\Xi_{\mathbf{y}}, \mathbf{f}_{\mathbf{y}}, D_{\mathbf{y}})$

(S1) Introduce new variables to replace all non-polynomial terms in $\mathbf{f}_{\mathbf{x}}$, $\Xi_{\mathbf{x}}$ and $D_{\mathbf{x}}$, and obtain a collection of replacement equations $\mathbf{v} = \Gamma(\mathbf{x})$.

(S2) Differentiate both sides of $\mathbf{v} = \Gamma(\mathbf{x})$ w.r.t. time, and then replace all newly appearing non-polynomial terms by introducing fresh variables.

(S3) Repeat (S2) until no more variables need to be introduced. For simplicity, still denote the final set of replacement equations by $\mathbf{v} = \Gamma(\mathbf{x})$.

(S4) Define the simulation map as $\Theta(\mathbf{x}) = (\mathbf{x}, \Gamma(\mathbf{x}))$. Then use $\mathbf{v} = \Gamma(\mathbf{x})$ to construct $\Xi_{\mathbf{y}}$ and $D_{\mathbf{y}}$ as illustrated by the following example.

Abstracting EDSs

Abstracting EDS $\mathcal{C}_{\mathbf{x}} \widehat{=} (\Xi_{\mathbf{x}}, \mathbf{f}_{\mathbf{x}}, D_{\mathbf{x}})$ to PDS $C_{\mathbf{y}} \widehat{=} (\Xi_{\mathbf{y}}, \mathbf{f}_{\mathbf{y}}, D_{\mathbf{y}})$

(S1) Introduce new variables to replace all non-polynomial terms in $\mathbf{f}_{\mathbf{x}}$, $\Xi_{\mathbf{x}}$ and $D_{\mathbf{x}}$, and obtain a collection of replacement equations $\mathbf{v} = \Gamma(\mathbf{x})$.

(S2) Differentiate both sides of $\mathbf{v} = \Gamma(\mathbf{x})$ w.r.t. time, and then replace all newly appearing non-polynomial terms by introducing fresh variables.

(S3) Repeat (S2) until no more variables need to be introduced. For simplicity, still denote the final set of replacement equations by $\mathbf{v} = \Gamma(\mathbf{x})$.

(S4) Define the simulation map as $\Theta(\mathbf{x}) = (\mathbf{x}, \Gamma(\mathbf{x}))$. Then use $\mathbf{v} = \Gamma(\mathbf{x})$ to construct $\Xi_{\mathbf{y}}$ and $D_{\mathbf{y}}$ as illustrated by the following example.

Abstracting EDSs

Abstracting EDS $\mathcal{C}_{\mathbf{x}} \mathrel{\widehat{=}} (\Xi_{\mathbf{x}}, \mathbf{f}_{\mathbf{x}}, D_{\mathbf{x}})$ to PDS $C_{\mathbf{y}} \mathrel{\widehat{=}} (\Xi_{\mathbf{y}}, \mathbf{f}_{\mathbf{y}}, D_{\mathbf{y}})$

(S1) Introduce new variables to replace all non-polynomial terms in $\mathbf{f}_{\mathbf{x}}$, $\Xi_{\mathbf{x}}$ and $D_{\mathbf{x}}$, and obtain a collection of replacement equations $\mathbf{v} = \Gamma(\mathbf{x})$.

(S2) Differentiate both sides of $\mathbf{v} = \Gamma(\mathbf{x})$ w.r.t. time, and then replace all newly appearing non-polynomial terms by introducing fresh variables.

(S3) Repeat (S2) until no more variables need to be introduced. For simplicity, still denote the final set of replacement equations by $\mathbf{v} = \Gamma(\mathbf{x})$.

(S4) Define the simulation map as $\Theta(\mathbf{x}) = (\mathbf{x}, \Gamma(\mathbf{x}))$. Then use $\mathbf{v} = \Gamma(\mathbf{x})$ to construct $\Xi_{\mathbf{y}}$ and $D_{\mathbf{y}}$ as illustrated by the following example.

Abstracting EDSs

Abstracting EDS $\mathcal{C}_{\mathbf{x}} \widehat{=} (\Xi_{\mathbf{x}}, \mathbf{f}_{\mathbf{x}}, D_{\mathbf{x}})$ to PDS $C_{\mathbf{y}} \widehat{=} (\Xi_{\mathbf{y}}, \mathbf{f}_{\mathbf{y}}, D_{\mathbf{y}})$

(S1) Introduce new variables to replace all non-polynomial terms in $\mathbf{f}_{\mathbf{x}}$, $\Xi_{\mathbf{x}}$ and $D_{\mathbf{x}}$, and obtain a collection of replacement equations $\mathbf{v} = \Gamma(\mathbf{x})$.

(S2) Differentiate both sides of $\mathbf{v} = \Gamma(\mathbf{x})$ w.r.t. time, and then replace all newly appearing non-polynomial terms by introducing fresh variables.

(S3) Repeat (S2) until no more variables need to be introduced. For simplicity, still denote the final set of replacement equations by $\mathbf{v} = \Gamma(\mathbf{x})$.

(S4) Define the simulation map as $\Theta(\mathbf{x}) = (\mathbf{x}, \Gamma(\mathbf{x}))$. Then use $\mathbf{v} = \Gamma(\mathbf{x})$ to construct $\Xi_{\mathbf{y}}$ and $D_{\mathbf{y}}$ as illustrated by the following example.

Abstracting EDSs: An Example

Consider the EDS $\mathcal{C}_{\mathbf{x}} \mathrel{\widehat{=}} (\Xi_{\mathbf{x}}, \mathbf{f}_{\mathbf{x}}, D_{\mathbf{x}})$, where

- $\Xi_{\mathbf{x}} \mathrel{\widehat{=}} (x+0.5)^2 + (y-0.5)^2 - 0.16 \leq 0$;
- $D_{\mathbf{x}} \mathrel{\widehat{=}} -2 \leq x \leq 2 \wedge -2 \leq y \leq 2$; and
- $\mathbf{f}_{\mathbf{x}}$ defines the ODE

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} e^{-x} + y - 1 \\ -\sin^2(x) \end{pmatrix} .$$

Abstracting EDSs: An Example

- (S1-S3): by the replacement relations $\mathbf{v} = \Gamma(\mathbf{x})$

$$(v_1, v_2, v_3) = (\sin x, e^{-x}, \cos x)$$

we get the transformed polynomial ODE (i.e. $\mathbf{f_y}$)

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{v}_1 \\ \dot{v}_2 \\ \dot{v}_3 \end{pmatrix} = \begin{pmatrix} v_2 + y - 1 \\ -v_1^2 \\ v_3(v_2 + y - 1) \\ -v_2(v_2 + y - 1) \\ -v_1(v_2 + y - 1) \end{pmatrix},$$

Abstracting EDSs: An Example

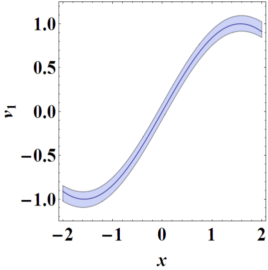
- (S4): the simulation map is $\Theta(x, y) = (x, y, \sin x, e^{-x}, \cos x)$
  - $\Theta(\Xi_{\mathsf{x}}) \mathrel{\widehat{=}} \Xi_{\mathsf{x}} \wedge v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x$
  - $\Theta(D_{\mathsf{x}}) \mathrel{\widehat{=}} D_{\mathsf{x}} \wedge v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x$
  - abstracting $v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x$ by polynomial expressions

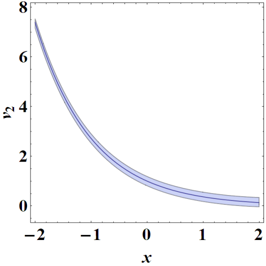
Abstracting EDSs: An Example

- (S4): the simulation map is $\Theta(x, y) = (x, y, \sin x, e^{-x}, \cos x)$
  - $\Theta(\Xi_{\mathbf{x}}) \,\widehat{=}\, \Xi_{\mathbf{x}} \wedge v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x$
  - $\Theta(D_{\mathbf{x}}) \,\widehat{=}\, D_{\mathbf{x}} \wedge v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x$
  - abstracting $v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x$ by polynomial expressions

Polynomial Approximation via Taylor Model

- $D_{\mathbf{x}} \widehat{=} -2 \leq x \leq 2 \wedge -2 \leq y \leq 2$
- $D_{\mathbf{x}} \wedge v_1 = \sin x$, expand up to degree 6
- $D_{\mathbf{x}} \wedge v_2 = e^{-x}$, expand up to degree 6

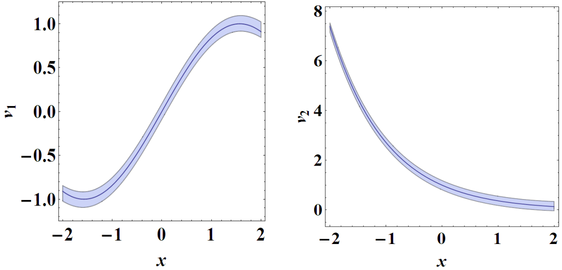




- In this way we can obtain $\Xi_{\mathbf{y}}$, $D_{\mathbf{y}}$

Polynomial Approximation via Taylor Model

- $D_{\mathbf{x}} \widehat{=} -2 \leq x \leq 2 \wedge -2 \leq y \leq 2$
- $D_{\mathbf{x}} \wedge v_1 = \sin x$, expand up to degree 6
- $D_{\mathbf{x}} \wedge v_2 = e^{-x}$, expand up to degree 6



- In this way we can obtain $\Xi_{\mathbf{y}}$, $D_{\mathbf{y}}$

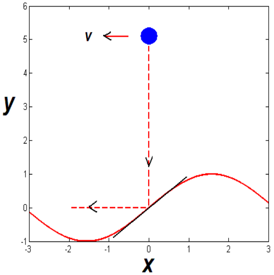
Abstracting EHSs

- abstracting EHS $\mathcal{H}_{\mathbf{x}} \,\widehat{=}\, (Q, X, f_{\mathbf{x}}, D_{\mathbf{x}}, E, G_{\mathbf{x}}, R_{\mathbf{x}}, \Xi_{\mathbf{x}})$ by PHS $\mathcal{H}_{\mathbf{y}} \,\widehat{=}\, (Q, Y, f_{\mathbf{y}}, D_{\mathbf{y}}, E, G_{\mathbf{y}}, R_{\mathbf{y}}, \Xi_{\mathbf{y}})$
- just extend the abstraction approach for EDSs to take into account guard constraints and reset functions
- treat each mode of a HA separately by constructing an individual abstraction map for each of them

Abstracting EHSs: An Example

- Bouncing ball on a sine-waved surface
- $Q = \{q\}$; $X = \{x, y, v_x, v_y\}$;
- $E = \{e\}$ with $e = (q, q)$;
- $D_{\mathbf{x},q} \mathrel{\widehat{=}} y \geq \sin x$; $G_{\mathbf{x},e} \mathrel{\widehat{=}} y = \sin x$;
- $\Xi_{\mathbf{x},q} \mathrel{\widehat{=}} y \geq 4.9 \wedge y \leq 5.1 \wedge x = 0 \wedge v_x = -1 \wedge v_y = 0$;
- $\mathbf{f}_{\mathbf{x},q} = \begin{cases} \dot{x} & = & v_x \\ \dot{y} & = & v_y \\ \dot{v}_x & = & 0 \\ \dot{v}_y & = & -9.8 \end{cases}$
- $R_{\mathbf{x},e}(x, y, v_x, v_y) \mathrel{\widehat{=}} \{(x, y, v_x', v_y')\}$ with

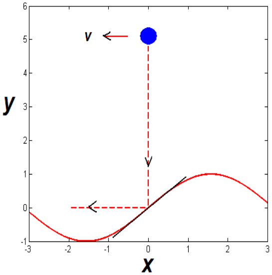
$$\begin{cases} v_x' & = & \frac{(\sin x)^2 \cdot v_x + 2(\cos x)\cdot v_y}{1+(\cos x)^2} \\ v_y' & = & \frac{2(\cos x)\cdot v_x - (\sin x)^2 \cdot v_y}{1+(\cos x)^2} \end{cases}.$$



Abstracting EHSs: An Example

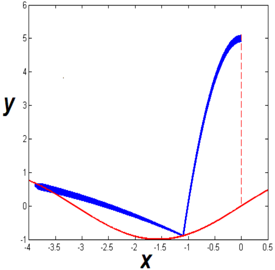
- Bouncing ball on a sine-waved surface
- $Q = \{q\}$; $X = \{x, y, v_x, v_y\}$;
- $E = \{e\}$ with $e = (q, q)$;
- $D_{\mathbf{x},q} \mathrel{\widehat{=}} y \geq \sin x$; $G_{\mathbf{x},e} \mathrel{\widehat{=}} y = \sin x$;
- $\Xi_{\mathbf{x},q} \mathrel{\widehat{=}} y \geq 4.9 \wedge y \leq 5.1 \wedge x = 0 \wedge v_x = -1 \wedge v_y = 0$;
- $\mathbf{f}_{\mathbf{x},q} = \begin{cases} \dot{x} & = & v_x \\ \dot{y} & = & v_y \\ \dot{v}_x & = & 0 \\ \dot{v}_y & = & -9.8 \end{cases}$
- $R_{\mathbf{x},e}(x, y, v_x, v_y) \mathrel{\widehat{=}} \{(x, y, v_x', v_y')\}$ with

$$\begin{cases} v_x' & = & \frac{(\sin x)^2 \cdot v_x + 2(\cos x) \cdot v_y}{1+(\cos x)^2} \\ v_y' & = & \frac{2(\cos x) \cdot v_x - (\sin x)^2 \cdot v_y}{1+(\cos x)^2} \end{cases}.$$



Abstracting EHSs: An Example

- replacement equations: $(u_1, u_2, u_3) = (\sin x, \cos x, \frac{1}{1+(\cos x)^2})$,
- flowpipe computation for the abstract system using Flow* (not applicable on the original system)



The Verification Problem

Consider the EDS $\mathcal{C}_{\mathbf{x}} \mathrel{\widehat{=}} (\Xi_{\mathbf{x}}, \mathbf{f}_{\mathbf{x}}, D_{\mathbf{x}})$, where

- $\Xi_{\mathbf{x}} \mathrel{\widehat{=}} (x+0.5)^2 + (y-0.5)^2 - 0.16 \leq 0$;
- $D_{\mathbf{x}} \mathrel{\widehat{=}} -2 \leq x \leq 2 \wedge -2 \leq y \leq 2$; and
- $\mathbf{f}_{\mathbf{x}}$ defines the ODE

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} e^{-x} + y - 1 \\ -\sin^2(x) \end{pmatrix} .$$

- verify the safety of $\mathcal{C}_{\mathbf{x}}$ w.r.t. an unsafe region $\bar{S}_{\mathbf{x}} \mathrel{\widehat{=}} (x-0.7)^2 + (y+0.7)^2 - 0.09 \leq 0$

The Verification Problem

Consider the EDS $\mathcal{C}_{\mathbf{x}} \widehat{=} (\Xi_{\mathbf{x}}, \mathbf{f}_{\mathbf{x}}, D_{\mathbf{x}})$, where

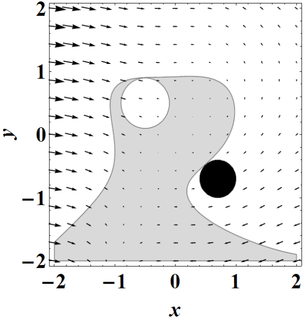
- $\Xi_{\mathbf{x}} \widehat{=} (x+0.5)^2 + (y-0.5)^2 - 0.16 \leq 0$;
- $D_{\mathbf{x}} \widehat{=} -2 \leq x \leq 2 \wedge -2 \leq y \leq 2$; and
- $\mathbf{f}_{\mathbf{x}}$ defines the ODE

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} e^{-x} + y - 1 \\ -\sin^2(x) \end{pmatrix} .$$

- verify the safety of $\mathcal{C}_{\mathbf{x}}$ w.r.t. an unsafe region $\bar{S}_{\mathbf{x}} \widehat{=} (x-0.7)^2 + (y+0.7)^2 - 0.09 \leq 0$

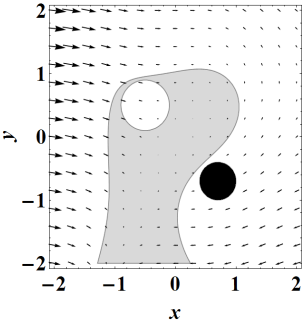
Generating Polynomial Invariants

- $(v_1, v_2, v_3) = (\sin x, e^{-x}, \cos x)$
- Assume a polynomial invariant template of degree 5 without fresh variables

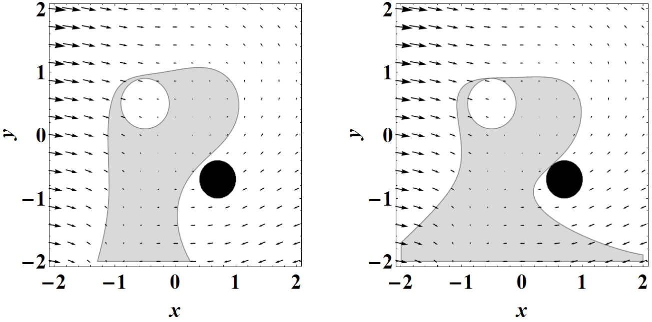


Generating Elementary Invariants

- $(v_1, v_2, v_3) = (\sin x, e^{-x}, \cos x)$
- Assume a polynomial invariant template of degree 4 with fresh variables



Comparison



Outline

Outline

Outline

Problem Decription

- A safety requirement $\mathcal{S}$ assigns to each mode $q \in Q$ a safe region $S_q \subseteq \mathbb{R}^n$, i.e. $\mathcal{S} = \bigcup_{q \in Q}(\{q\} \times S_q)$.

Switching controller synthesis for safety [Asarin et al. 00]

Given a hybrid automaton $\mathcal{H}$ and a safety property S, find a hybrid automaton $\mathcal{H}' = (Q, X, f, D', E, G')$ such that

(r1) Refinement: for any $q \in Q$, $D'_q \subseteq D_q$, and for any $e \in E$, $G'_e \subseteq G_e$;

(r2) Safety: for any trajectory ω that $\mathcal{H}'$ accepts, if $(q, \mathbf{x})$ is on ω, then $\mathbf{x} \in S_q$;

(r3) Non-blocking: $\mathcal{H}'$ is non-blocking.

Problem Decription

- A safety requirement $\mathcal{S}$ assigns to each mode $q \in Q$ a safe region $S_q \subseteq \mathbb{R}^n$, i.e. $\mathcal{S} = \bigcup_{q \in Q}(\{q\} \times S_q)$.

Switching controller synthesis for safety [Asarin et al. 00]

Given a hybrid automaton $\mathcal{H}$ and a safety property S, find a hybrid automaton $\mathcal{H}' = (Q, X, f, D', E, G')$ such that

(r1) Refinement: for any $q \in Q$, $D'_q \subseteq D_q$, and for any $e \in E$, $G'_e \subseteq G_e$;

(r2) Safety: for any trajectory ω that $\mathcal{H}'$ accepts, if (q, x) is on ω, then $x \in S_q$;

(r3) Non-blocking: $\mathcal{H}'$ is non-blocking.

Problem Decription

- A safety requirement $\mathcal{S}$ assigns to each mode $q \in Q$ a safe region $S_q \subseteq \mathbb{R}^n$, i.e. $\mathcal{S} = \bigcup_{q \in Q}(\{q\} \times S_q)$.

Switching controller synthesis for safety [Asarin et al. 00]

Given a hybrid automaton $\mathcal{H}$ and a safety property S, find a hybrid automaton $\mathcal{H}' = (Q, X, f, D', E, G')$ such that

(r1) Refinement: for any $q \in Q$, $D'_q \subseteq D_q$, and for any $e \in E$, $G'_e \subseteq G_e$;

(r2) Safety: for any trajectory ω that $\mathcal{H}'$ accepts, if $(q, \mathbf{x})$ is on ω, then $\mathbf{x} \in S_q$;

(r3) Non-blocking: $\mathcal{H}'$ is non-blocking.

Problem Decription

- A safety requirement $\mathcal{S}$ assigns to each mode $q \in Q$ a safe region $S_q \subseteq \mathbb{R}^n$, i.e. $\mathcal{S} = \bigcup_{q \in Q}(\{q\} \times S_q)$.

Switching controller synthesis for safety [Asarin et al. 00]

Given a hybrid automaton $\mathcal{H}$ and a safety property S, find a hybrid automaton $\mathcal{H}' = (Q, X, f, D', E, G')$ such that

(r1) Refinement: for any $q \in Q$, $D'_q \subseteq D_q$, and for any $e \in E$, $G'_e \subseteq G_e$;

(r2) Safety: for any trajectory ω that $\mathcal{H}'$ accepts, if $(q, \mathbf{x})$ is on ω, then $\mathbf{x} \in S_q$;

(r3) Non-blocking: $\mathcal{H}'$ is non-blocking.

Problem Decription

- A safety requirement $\mathcal{S}$ assigns to each mode $q \in Q$ a safe region $S_q \subseteq \mathbb{R}^n$, i.e. $\mathcal{S} = \bigcup_{q \in Q}(\{q\} \times S_q)$.

Switching controller synthesis for safety [Asarin et al. 00]

Given a hybrid automaton $\mathcal{H}$ and a safety property S, find a hybrid automaton $\mathcal{H}' = (Q, X, f, D', E, G')$ such that

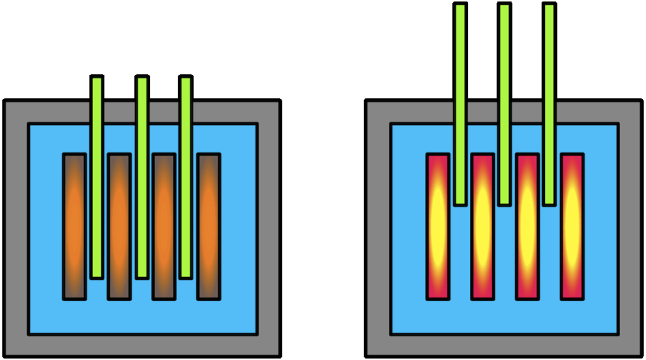
(r1) Refinement: for any $q \in Q$, $D'_q \subseteq D_q$, and for any $e \in E$, $G'_e \subseteq G_e$;

(r2) Safety: for any trajectory ω that $\mathcal{H}'$ accepts, if $(q, \mathbf{x})$ is on ω, then $\mathbf{x} \in S_q$;

(r3) Non-blocking: $\mathcal{H}'$ is non-blocking.

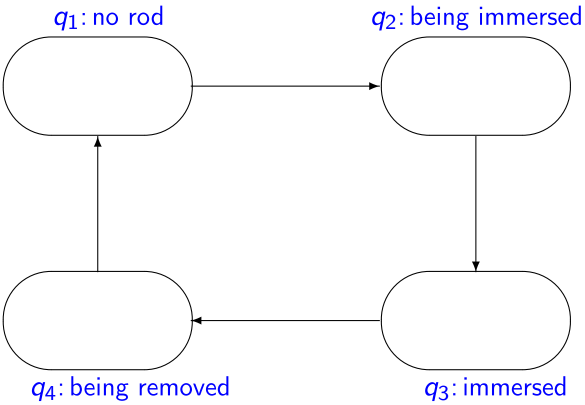
A Nuclear Reactor Example

The nuclear reactor system consists of a reactor core and a cooling rod which is immersed into and removed out of the core periodically to keep the temperature of the core in a certain range.



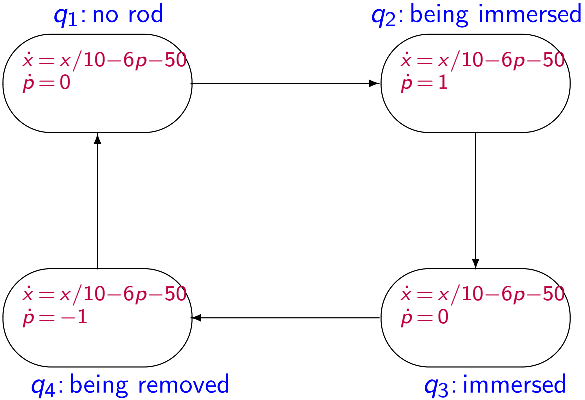
A Nuclear Reactor Example (Cont'd)

- x: temperature;
- p: proportion immersed



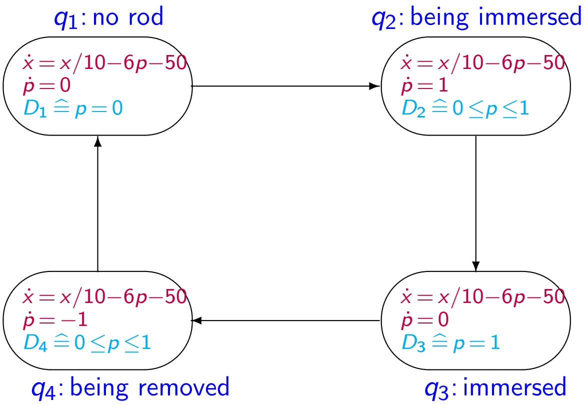
A Nuclear Reactor Example (Cont'd)

- x: temperature;
- p: proportion immersed



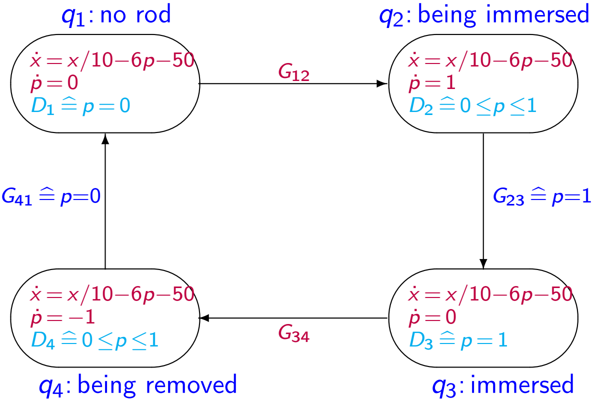
A Nuclear Reactor Example (Cont'd)

- x: temperature;
- p: proportion immersed



A Nuclear Reactor Example (Cont'd)

- x: temperature;
- p: proportion immersed

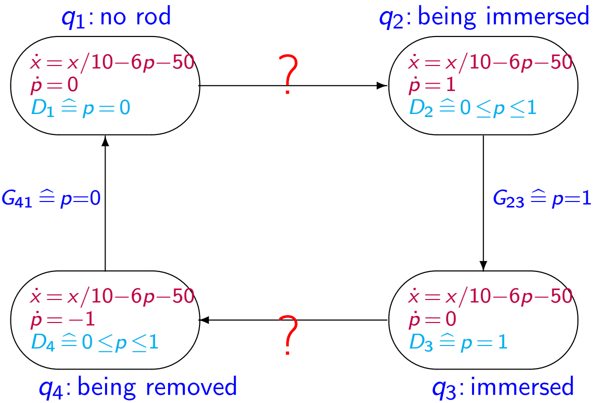


Switching Controller Synthesis for the Reactor

$S \mathrel{\widehat{=}} 510 \leq x \leq 550$ for all modes

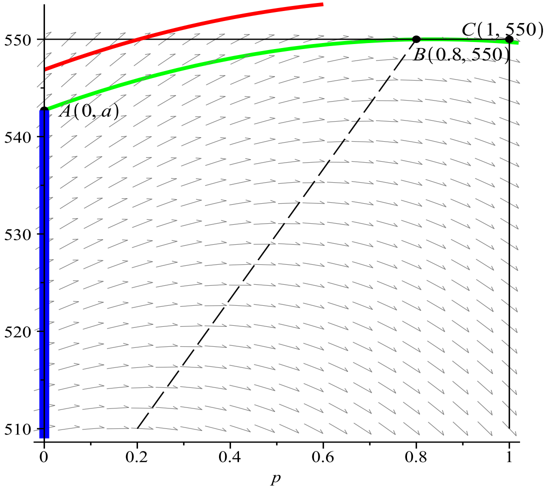
Switching Controller Synthesis for the Reactor

$S \mathrel{\widehat{=}} 510 \leq x \leq 550$ for all modes



Bad Switching Violates Safety Property

Transition from mode q_1 to q_2



Solution to the Controller Synthesis Problem

Abstract Solution

Let $\mathcal{H}$ be a hybrid system and $\mathcal{S}$ be a safety property. If we can find a family of $D'_q \subseteq \mathbb{R}^n$ such that

(c1) for all $q \in Q$, $D'_q \subseteq D_q \cap S_q$;

(c2) for all $q \in Q$, D'_q is a continuous invariant of (H_q, f_q) with

$$H_q \widehat{=} \Big(\bigcup_{e=(q,q')\in E} G'_e \Big)^c ,$$

where $G'_e \widehat{=} G_e \cap D'_{q'}$ for $e = (q, q')$, then the family of G'_e form a safe switching controller.

Solution to the Controller Synthesis Problem

Abstract Solution

Let $\mathcal{H}$ be a hybrid system and $\mathcal{S}$ be a safety property. If we can find a family of $D'_q \subseteq \mathbb{R}^n$ such that

(c1) for all $q \in Q$, $D'_q \subseteq D_q \cap S_q$;

(c2) for all $q \in Q$, D'_q is a continuous invariant of $(H_q, \mathbf{f}_q)$ with

$$H_q \mathrel{\widehat{=}} \Big(\bigcup_{e=(q,q')\in E} G'_e \Big)^c ,$$

where $G'_e \mathrel{\widehat{=}} G_e \cap D'_{q'}$ for $e = (q, q')$, then the family of G'_e form a safe switching controller.

Solution to the Controller Synthesis Problem

Abstract Solution

Let $\mathcal{H}$ be a hybrid system and $\mathcal{S}$ be a safety property. If we can find a family of $D'_q \subseteq \mathbb{R}^n$ such that

(c1) for all $q \in Q$, $D'_q \subseteq D_q \cap S_q$;

(c2) for all $q \in Q$, D'_q is a continuous invariant of $(H_q, \mathbf{f}_q)$ with

$$H_q \mathrel{\widehat{=}} \Big(\bigcup_{e=(q,q')\in E} G'_e \Big)^c ,$$

where $G'_e \mathrel{\widehat{=}} G_e \cap D'_{q'}$ for $e = (q, q')$, then the family of G'_e form a safe switching controller.

Template-Based Synthesis Framework

(s1) Template assignment: assign to each $q \in Q$ a template D'_q as the continuous invariant to be generated at mode q;

(s2) Guard refinement: refine the transition guard G_e for each $e = (q, q') \in E$ by setting $G'_e \,\widehat{=}\, G_e \cap D'_{q'}$;

(s3) Deriving synthesis conditions: encode (c1) and (c2) in the abstract solution into constraints on parameters appearing in the templates;

(s4) Constraint solving: solve the constraints derived from (s3) using quantifier elimination (QE);

(s5) Parameters instantiation: find an appropriate instantiation of D'_q and G'_e from the possible parameter values obtained at (s4)

Template-Based Synthesis Framework

(s1) Template assignment: assign to each $q \in Q$ a template D'_q as the continuous invariant to be generated at mode q ;

(s2) Guard refinement: refine the transition guard G_e for each $e = (q, q') \in E$ by setting $G'_e \,\widehat{=}\, G_e \cap D'_{q'}$;

(s3) Deriving synthesis conditions: encode (c1) and (c2) in the abstract solution into constraints on parameters appearing in the templates;

(s4) Constraint solving: solve the constraints derived from (s3) using quantifier elimination (QE);

(s5) Parameters instantiation: find an appropriate instantiation of D'_q and G'_e from the possible parameter values obtained at (s4)

Template-Based Synthesis Framework

(s1) Template assignment: assign to each $q \in Q$ a template D'_q as the continuous invariant to be generated at mode q;

(s2) Guard refinement: refine the transition guard G_e for each $e = (q, q') \in E$ by setting $G'_e \widehat{=} G_e \cap D'_{q'}$;

(s3) Deriving synthesis conditions: encode (c1) and (c2) in the abstract solution into constraints on parameters appearing in the templates;

(s4) Constraint solving: solve the constraints derived from (s3) using quantifier elimination (QE);

(s5) Parameters instantiation: find an appropriate instantiation of D'_q and G'_e from the possible parameter values obtained at (s4)

Template-Based Synthesis Framework

(s1) Template assignment: assign to each $q \in Q$ a template D'_q as the continuous invariant to be generated at mode q;

(s2) Guard refinement: refine the transition guard G_e for each $e = (q, q') \in E$ by setting $G'_e \mathrel{\widehat{=}} G_e \cap D'_{q'}$;

(s3) Deriving synthesis conditions: encode (c1) and (c2) in the abstract solution into constraints on parameters appearing in the templates;

(s4) Constraint solving: solve the constraints derived from (s3) using quantifier elimination (QE);

(s5) Parameters instantiation: find an appropriate instantiation of D'_q and G'_e from the possible parameter values obtained at (s4)

Template-Based Synthesis Framework

(s1) Template assignment: assign to each $q \in Q$ a template D'_q as the continuous invariant to be generated at mode q;

(s2) Guard refinement: refine the transition guard G_e for each $e = (q, q') \in E$ by setting $G'_e \mathrel{\widehat{=}} G_e \cap D'_{q'}$;

(s3) Deriving synthesis conditions: encode (c1) and (c2) in the abstract solution into constraints on parameters appearing in the templates;

(s4) Constraint solving: solve the constraints derived from (s3) using quantifier elimination (QE);

(s5) Parameters instantiation: find an appropriate instantiation of D'_q and G'_e from the possible parameter values obtained at (s4)

Heuristics for Predefining Templates by Qualitative Analysis

Using qualitative analysis to identify critical points for predefining templates

- Infer the evolution behavior (increasing or decreasing) of continuous variables in each mode from the ODEs
- Identify modes (called critical) at which the evolution behavior of a continuous variable changes, and thus the maximal (or minimal) value of this continuous variable can be achieved
- Equate the maximal (or minimal) value to the corresponding safety upper (or lower) bound to obtain a critical point
- Backward propagate the critical point, by backtracking along the continuous trajectory through the critical point

Heuristics for Predefining Templates by Qualitative Analysis

Using qualitative analysis to identify critical points for predefining templates

- Infer the evolution behavior (increasing or decreasing) of continuous variables in each mode from the ODEs
- Identify modes (called critical) at which the evolution behavior of a continuous variable changes, and thus the maximal (or minimal) value of this continuous variable can be achieved
- Equate the maximal (or minimal) value to the corresponding safety upper (or lower) bound to obtain a critical point
- Backward propagate the critical point, by backtracking along the continuous trajectory through the critical point

Heuristics for Predefining Templates by Qualitative Analysis

Using qualitative analysis to identify critical points for predefining templates

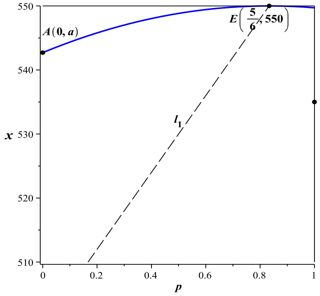
- Infer the evolution behavior (increasing or decreasing) of continuous variables in each mode from the ODEs
- Identify modes (called critical) at which the evolution behavior of a continuous variable changes, and thus the maximal (or minimal) value of this continuous variable can be achieved
- Equate the maximal (or minimal) value to the corresponding safety upper (or lower) bound to obtain a critical point
- Backward propagate the critical point, by backtracking along the continuous trajectory through the critical point

Heuristics for Predefining Templates by Qualitative Analysis

Using qualitative analysis to identify critical points for predefining templates

- Infer the evolution behavior (increasing or decreasing) of continuous variables in each mode from the ODEs
- Identify modes (called critical) at which the evolution behavior of a continuous variable changes, and thus the maximal (or minimal) value of this continuous variable can be achieved
- Equate the maximal (or minimal) value to the corresponding safety upper (or lower) bound to obtain a critical point
- Backward propagate the critical point, by backtracking along the continuous trajectory through the critical point

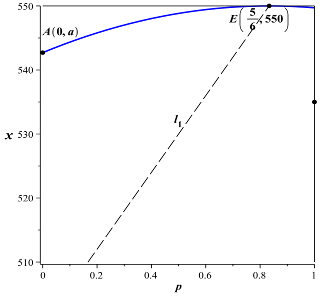
Revisiting the Running Example



For the running example,

- At D_{q_2}, temperature x achieves maximal value when crossing $l_1 \widehat{=} x/10 - 6p - 50 = 0$.
- $E(5/6, 550)$ at q_2 is obtained by taking the intersection of l_1 and safety upper bound $x = 550$
- E is backward propagated to $A(0, a)$, with a a parameter
- Compute a parabola $x - 550 - \frac{36}{25}(a - 550)(p - \frac{5}{6})^2 = 0$ through A and E as part of the template D'_{q_2}

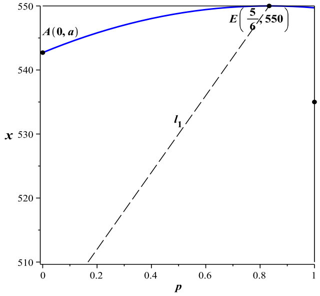
Revisiting the Running Example



For the running example,

- At D_{q_2}, temperature x achieves maximal value when crossing $l_1 \widehat{=} x/10 - 6p - 50 = 0$.
- $E(5/6, 550)$ at q_2 is obtained by taking the intersection of l_1 and safety upper bound $x = 550$
- E is backward propagated to $A(0, a)$, with a a parameter
- Compute a parabola $x - 550 - \frac{36}{25}(a - 550)(p - \frac{5}{6})^2 = 0$ through A and E as part of the template D'_{q_2}

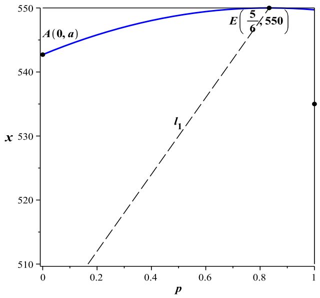
Revisiting the Running Example



For the running example,

- At D_{q_2}, temperature x achieves maximal value when crossing $l_1 \mathrel{\widehat{=}} x/10 - 6p - 50 = 0$.
- $E(5/6, 550)$ at q_2 is obtained by taking the intersection of l_1 and safety upper bound $x = 550$
- E is backward propagated to $A(0, a)$, with a a parameter
- Compute a parabola $x - 550 - \frac{36}{25}(a - 550)(p - \frac{5}{6})^2 = 0$ through A and E as part of the template D'_{q_2}

Revisiting the Running Example



For the running example,

- At D_{q_2}, temperature x achieves maximal value when crossing $l_1 \widehat{=} x/10 - 6p - 50 = 0$.
- $E(5/6, 550)$ at q_2 is obtained by taking the intersection of l_1 and safety upper bound $x = 550$
- E is backward propagated to $A(0, a)$, with a a parameter
- Compute a parabola $x - 550 - \frac{36}{25}(a - 550)(p - \frac{5}{6})^2 = 0$ through A and E as part of the template D'_{q_2}

Revisiting the Running Example (Cont'd)

The set of parameters: a, b, c, d

- $D'_1 \mathrel{\widehat{=}} p = 0 \wedge 510 \leq x \leq a$
- $D'_2 \mathrel{\widehat{=}} 0 \leq p \leq 1 \wedge x - b \geq p(d - b) \wedge$
  $x - 550 - \frac{36}{25}(a - 550)(p - \frac{5}{6})^2 \leq 0$
- $D'_3 \mathrel{\widehat{=}} p = 1 \wedge d \leq x \leq 550$
- $D'_4 \mathrel{\widehat{=}} 0 \leq p \leq 1 \wedge x - a \leq p(c - a) \wedge$
  $x - 510 - \frac{36}{25}(d - 510)(p - \frac{1}{6})^2 \geq 0$

- $G'_{12} \mathrel{\widehat{=}} p = 0 \wedge b \leq x \leq a$
- $G'_{23} \mathrel{\widehat{=}} p = 1 \wedge d \leq x \leq 550$
- $G'_{34} \mathrel{\widehat{=}} p = 1 \wedge d \leq x \leq c$
- $G'_{41} \mathrel{\widehat{=}} p = 0 \wedge 510 \leq x \leq a$

Revisiting the Running Example (Cont'd)

The set of parameters: a, b, c, d

- $D'_1 \mathrel{\widehat{=}} p = 0 \wedge 510 \leq x \leq a$
- $D'_2 \mathrel{\widehat{=}} 0 \leq p \leq 1 \wedge x - b \geq p(d - b) \wedge$
  $x - 550 - \frac{36}{25}(a - 550)(p - \frac{5}{6})^2 \leq 0$
- $D'_3 \mathrel{\widehat{=}} p = 1 \wedge d \leq x \leq 550$
- $D'_4 \mathrel{\widehat{=}} 0 \leq p \leq 1 \wedge x - a \leq p(c - a) \wedge$
  $x - 510 - \frac{36}{25}(d - 510)(p - \frac{1}{6})^2 \geq 0$

- $G'_{12} \mathrel{\widehat{=}} p = 0 \wedge b \leq x \leq a$
- $G'_{23} \mathrel{\widehat{=}} p = 1 \wedge d \leq x \leq 550$
- $G'_{34} \mathrel{\widehat{=}} p = 1 \wedge d \leq x \leq c$
- $G'_{41} \mathrel{\widehat{=}} p = 0 \wedge 510 \leq x \leq a$

Revisiting the Running Example (Cont'd)

- $a = \frac{6575}{12} \wedge b = \frac{4135}{8} \wedge c = \frac{4345}{8} \wedge d = \frac{6145}{12}$.
- From this result we get that the cooling rod should be immersed before temperature rises to $\frac{6575}{12} = 547.92$, and removed before temperature drops to $\frac{6145}{12} = 512.08$.
- By solving differential equations explicitly, the corresponding exact bounds are 547.97 and 512.03

Revisiting the Running Example (Cont'd)

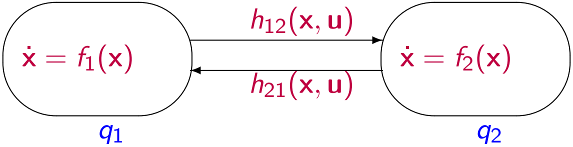
- $a = \frac{6575}{12} \wedge b = \frac{4135}{8} \wedge c = \frac{4345}{8} \wedge d = \frac{6145}{12}$.
- From this result we get that the cooling rod should be immersed before temperature rises to $\frac{6575}{12} = 547.92$, and removed before temperature drops to $\frac{6145}{12} = 512.08$.
- By solving differential equations explicitly, the corresponding exact bounds are 547.97 and 512.03

Revisiting the Running Example (Cont'd)

- $a = \frac{6575}{12} \wedge b = \frac{4135}{8} \wedge c = \frac{4345}{8} \wedge d = \frac{6145}{12}$.
- From this result we get that the cooling rod should be immersed before temperature rises to $\frac{6575}{12} = 547.92$, and removed before temperature drops to $\frac{6145}{12} = 512.08$.
- By solving differential equations explicitly, the corresponding exact bounds are 547.97 and 512.03

Outline

Problem Description



- Given a hybrid system $\mathcal{H}$ in which transition conditions h_{ij} are not determined but parameterized by $\mathbf{u}$, a vector of control parameters
- Our task is to determine $\mathbf{u}$ such that $\mathcal{H}$ can make discrete jumps at desired points, thus guaranteeing that
  - a safety property $\mathcal{S}$ is satisfied, i.e. $\mathbf{x} \in \mathcal{S}$ at any time
  - an optimization goal, e.g. $\min_{\mathbf{u}} g(\mathbf{u})$, is achieved

Our Approach – Step 1

Derive constraint $D(\mathbf{u})$ on $\mathbf{u}$ from the safety requirements $\mathcal{S}$

- Compute
  - the exact reachable set $\mathrm{Reach}_{\mathcal{H}}(\mathbf{x}, \mathbf{u})$ of $\mathcal{H}$, or
  - an inductive invariant $\mathrm{Inv}_{\mathcal{H}}(\mathbf{x}, \mathbf{u})$

  as polynomial formulas
- Suppose $\mathcal{S}$ is also modeled by polynomial formulas, then $D(\mathbf{u})$ can be obtained by applying QE to

$$\forall \mathbf{x}.\ \Big(\mathrm{Reach}_{\mathcal{H}}(\mathbf{x}, \mathbf{u}) \longrightarrow \mathcal{S}\Big)$$

or

$$\forall \mathbf{x}.\ \Big(\mathrm{Inv}_{\mathcal{H}}(\mathbf{x}, \mathbf{u}) \longrightarrow \mathcal{S}\Big)$$

Our Approach – Step 1

Derive constraint $D(\mathbf{u})$ on $\mathbf{u}$ from the safety requirements $\mathcal{S}$

- Compute
  - the exact reachable set $\mathrm{Reach}_{\mathcal{H}}(\mathbf{x}, \mathbf{u})$ of $\mathcal{H}$, or
  - an inductive invariant $\mathrm{Inv}_{\mathcal{H}}(\mathbf{x}, \mathbf{u})$

  as polynomial formulas
- Suppose $\mathcal{S}$ is also modeled by polynomial formulas, then $D(\mathbf{u})$ can be obtained by applying QE to

$$\forall \mathbf{x}.\ \left(\mathrm{Reach}_{\mathcal{H}}(\mathbf{x}, \mathbf{u}) \longrightarrow \mathcal{S}\right)$$

or

$$\forall \mathbf{x}.\ \left(\mathrm{Inv}_{\mathcal{H}}(\mathbf{x}, \mathbf{u}) \longrightarrow \mathcal{S}\right)$$

Our Approach – Step 1

Derive constraint $D(\mathbf{u})$ on $\mathbf{u}$ from the safety requirements $\mathcal{S}$

- Compute
  - the exact reachable set $\mathrm{Reach}_{\mathcal{H}}(\mathbf{x}, \mathbf{u})$ of $\mathcal{H}$, or
  - an inductive invariant $\mathrm{Inv}_{\mathcal{H}}(\mathbf{x}, \mathbf{u})$

  as polynomial formulas
- Suppose $\mathcal{S}$ is also modeled by polynomial formulas, then $D(\mathbf{u})$ can be obtained by applying QE to

$$\forall \mathbf{x}.\ \left(\mathrm{Reach}_{\mathcal{H}}(\mathbf{x}, \mathbf{u}) \longrightarrow \mathcal{S}\right)$$

or

$$\forall \mathbf{x}.\ \left(\mathrm{Inv}_{\mathcal{H}}(\mathbf{x}, \mathbf{u}) \longrightarrow \mathcal{S}\right)$$

Our Approach – Step 1

Derive constraint $D(\mathbf{u})$ on $\mathbf{u}$ from the safety requirements $\mathcal{S}$

- Compute
  - the exact reachable set $\mathrm{Reach}_{\mathcal{H}}(\mathbf{x}, \mathbf{u})$ of $\mathcal{H}$, or
  - an inductive invariant $\mathrm{Inv}_{\mathcal{H}}(\mathbf{x}, \mathbf{u})$

  as polynomial formulas
- Suppose $\mathcal{S}$ is also modeled by polynomial formulas, then $D(\mathbf{u})$ can be obtained by applying QE to

$$\forall \mathbf{x}.\ \Big(\mathrm{Reach}_{\mathcal{H}}(\mathbf{x}, \mathbf{u}) \longrightarrow \mathcal{S}\Big)$$

or

$$\forall \mathbf{x}.\ \Big(\mathrm{Inv}_{\mathcal{H}}(\mathbf{x}, \mathbf{u}) \longrightarrow \mathcal{S}\Big)$$

Our Approach – Step 2

Encode the optimization problem (suppose the objective function g is a polynomial) over constraint $D(\mathbf{u})$ into a quantified first-order polynomial formula $\mathbf{Qu}.\varphi(\mathbf{u}, z)$ by introducing a fresh variable z

- Minimize u^2 on $[-1, 1]$
- Introduce a fresh variable z:
  $u \geq -1 \wedge u \leq 1 \wedge \; u^2 \leq z$
- Projection to the z-axis:
  $\exists u.(u \geq -1 \wedge u \leq 1 \wedge u^2 \leq z)$
- After QE: $z \geq 0$, which means

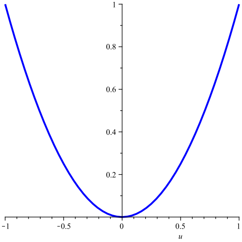
$$\min_{u \in [-1,1]} u^2 = 0$$

Our Approach – Step 2

Encode the optimization problem (suppose the objective function g is a polynomial) over constraint $D(\mathbf{u})$ into a quantified first-order polynomial formula $\mathbf{Qu}.\varphi(\mathbf{u}, z)$ by introducing a fresh variable z

- Minimize u^2 on $[-1, 1]$
- Introduce a fresh variable z:
  $u \geq -1 \wedge u \leq 1 \wedge\ u^2 \leq z$
- Projection to the z-axis:
  $\exists u.(u \geq -1 \wedge u \leq 1 \wedge u^2 \leq z)$
- After QE: $z \geq 0$, which means

$$\min_{u\in[-1,1]} u^2 = 0$$

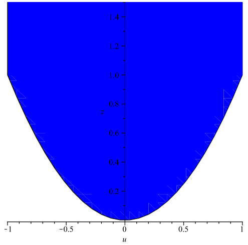


Our Approach – Step 2

Encode the optimization problem (suppose the objective function g is a polynomial) over constraint $D(\mathbf{u})$ into a quantified first-order polynomial formula $\mathbf{Qu}.\varphi(\mathbf{u}, z)$ by introducing a fresh variable z

- Minimize u^2 on $[-1, 1]$
- Introduce a fresh variable z:
  $u \geq -1 \wedge u \leq 1 \wedge u^2 \leq z$
- Projection to the z-axis:
  $\exists u.(u \geq -1 \wedge u \leq 1 \wedge u^2 \leq z)$
- After QE: $z \geq 0$, which means

$$\min_{u \in [-1,1]} u^2 = 0$$

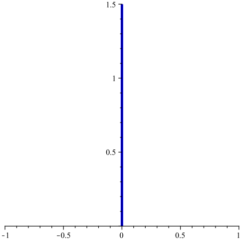


Our Approach – Step 2

Encode the optimization problem (suppose the objective function g is a polynomial) over constraint $D(\mathbf{u})$ into a quantified first-order polynomial formula $\mathbf{Qu}.\varphi(\mathbf{u}, z)$ by introducing a fresh variable z

- Minimize u^2 on $[-1, 1]$
- Introduce a fresh variable z:
  $u \geq -1 \wedge u \leq 1 \wedge\ u^2 \leq z$
- Projection to the z-axis:
  $\exists u.(u \geq -1 \wedge u \leq 1 \wedge u^2 \leq z)$
- After QE: $z \geq 0$, which means

$$\min_{u \in [-1,1]} u^2 = 0$$

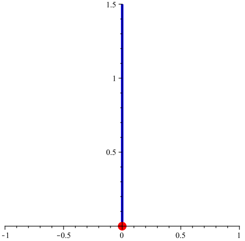


Our Approach – Step 2

Encode the optimization problem (suppose the objective function g is a polynomial) over constraint $D(\mathbf{u})$ into a quantified first-order polynomial formula $\mathbf{Q}\mathbf{u}.\varphi(\mathbf{u}, z)$ by introducing a fresh variable z

- Minimize u^2 on $[-1, 1]$
- Introduce a fresh variable z:
  $u \geq -1 \wedge u \leq 1 \wedge\ u^2 \leq z$
- Projection to the z-axis:
  $\exists u.(u \geq -1 \wedge u \leq 1 \wedge u^2 \leq z)$
- After QE: $z \geq 0$, which means

$$\min_{u \in [-1,1]} u^2 = 0$$



Encoding Optimization Criteria

Lemma

Suppose $g_1(\mathbf{u}_1)$, $g_2(\mathbf{u}_1, \mathbf{u}_2)$, $g_3(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ *are polynomials, and* $D_1(\mathbf{u}_1)$, $D_2(\mathbf{u}_1, \mathbf{u}_2)$, $D_3(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$ *are nonempty compact semi-algebraic sets. Then there exist* c_1, c_2, $c_3 \in \mathbb{R}$ *s.t.*

$$\exists \mathbf{u}_1.(D_1 \wedge g_1 \leq z) \Leftrightarrow z \geq c_1 \tag{4}$$

$$\forall \mathbf{u}_2.\big(\exists \mathbf{u}_1.D_2 \Rightarrow \exists \mathbf{u}_1.(D_2 \wedge g_2 \leq z)\big) \Leftrightarrow z \geq c_2 \tag{5}$$

$$\exists \mathbf{u}_3.\Big((\exists \mathbf{u}_1 \mathbf{u}_2.D_3) \wedge \forall \mathbf{u}_2.\big(\exists \mathbf{u}_1.D_3 \Rightarrow \exists \mathbf{u}_1.(D_3 \wedge g_3 \leq z)\big)\Big) \Leftrightarrow z \rhd c_3 \tag{6}$$

where $\rhd \in \{>, \geq\}$, *and* c_1, c_2, c_3 *satisfy*

$$c_1 = \min_{\mathbf{u}_1} g_1(\mathbf{u}_1) \quad \text{over } D_1(\mathbf{u}_1)\,, \tag{7}$$

$$c_2 = \sup_{\mathbf{u}_2} \min_{\mathbf{u}_1} g_2(\mathbf{u}_1, \mathbf{u}_2) \quad \text{over } D_2(\mathbf{u}_1, \mathbf{u}_2)\,, \tag{8}$$

$$c_3 = \inf_{\mathbf{u}_3} \sup_{\mathbf{u}_2} \min_{\mathbf{u}_1} g_3(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) \quad \text{over } D_3(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)\,. \tag{9}$$

Our Approach – Step 3

Eliminate quantifiers in $\mathbf{Qu}.\varphi(\mathbf{u}, z)$ and from the result we can retrieve the optimal value and the corresponding optimal controller $\mathbf{u}$

- Combine exact QE with numeric computation: (discretization of existentially quantified variables)

$$\exists \mathbf{x} \in A.\, \varphi(\mathbf{x}) \approx \bigvee_{\mathbf{y} \in F_A} \varphi(\mathbf{y}) \quad ,$$

where F_A is a finite subset of A

Our Approach – Step 3

Eliminate quantifiers in $\mathbf{Qu}.\varphi(\mathbf{u}, z)$ and from the result we can retrieve the optimal value and the corresponding optimal controller $\mathbf{u}$

- Combine exact QE with numeric computation: (discretization of existentially quantified variables)

$$\exists \mathbf{x} \in A.\, \varphi(\mathbf{x}) \approx \bigvee_{\mathbf{y} \in F_A} \varphi(\mathbf{y}) \quad ,$$

where F_A is a finite subset of A

Outline

A Reported Case Study

Cassez, F., Jessen, J.J., Larsen, K.G., Raskin, J.F., Reynier, P.A.:
Automatic Synthesis of Robust and Optimal Controllers — An Industrial Case Study. HSCC'09

- Provided by the HYDAC ELECTRONIC GMBH company within the European project *Quasimodo*
- An oil pump control problem
  - safety
  - robustness
  - optimality

A Reported Case Study

Cassez, F., Jessen, J.J., Larsen, K.G., Raskin, J.F., Reynier, P.A.: Automatic Synthesis of Robust and Optimal Controllers — An Industrial Case Study. HSCC'09

- Provided by the HYDAC ELECTRONIC GMBH company within the European project *Quasimodo*
- An oil pump control problem
  - safety
  - robustness
  - optimality

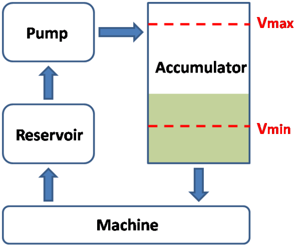
A Reported Case Study

Cassez, F., Jessen, J.J., Larsen, K.G., Raskin, J.F., Reynier, P.A.: Automatic Synthesis of Robust and Optimal Controllers — An Industrial Case Study. HSCC'09

- Provided by the HYDAC ELECTRONIC GMBH company within the European project *Quasimodo*
- An oil pump control problem
  - safety
  - robustness
  - optimality

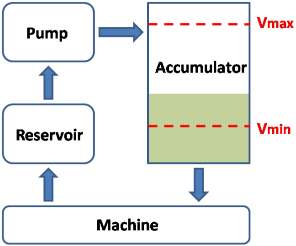
The System

- The system is composed of a machine, an accumulator, a reservoir and a pump
- The machine consumes oil out of the accumulator; the pump adds oil from the reservoir into the accumulator



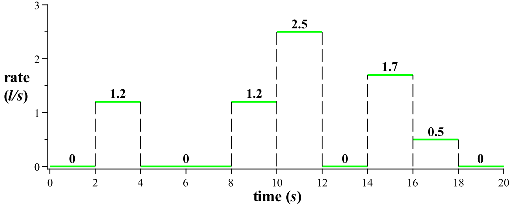
The System

- The system is composed of a machine, an accumulator, a reservoir and a pump
- The machine consumes oil out of the accumulator; the pump adds oil from the reservoir into the accumulator



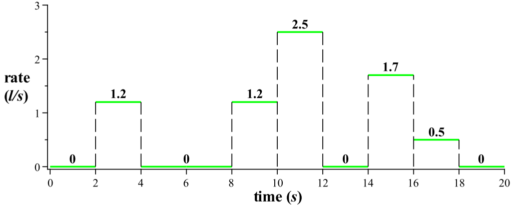
The Consumption Rate

- The oil consumption is periodic. The length of one consumption cycle is 20s (second)
- The profile of consumption rate in one cycle is depicted by



The Consumption Rate

- The oil consumption is periodic. The length of one consumption cycle is 20s (second)
- The profile of consumption rate in one cycle is depicted by



The Pump

- The power of the pump is 2.2 l/s (liter/second)
- 2-second latency: if the pump is switched on (t_{2k+1}) or off (t_{2k+2}) at time points

$$0 \leq t_1 \leq t_2 \leq \cdots \leq t_i \leq t_{i+1} \leq \cdots ,$$

then

$$t_{i+1} - t_i \geq 2$$

for any $i \geq 1$
- It is obvious that the pump can be turned on at most 5 times in one cycle

The Pump

- The power of the pump is 2.2 l/s (liter/second)
- 2-second latency: if the pump is switched on (t_{2k+1}) or off (t_{2k+2}) at time points

$$0 \leq t_1 \leq t_2 \leq \cdots \leq t_i \leq t_{i+1} \leq \cdots ,$$

then

$$t_{i+1} - t_i \geq 2$$

for any $i \geq 1$

- It is obvious that the pump can be turned on at most 5 times in one cycle

The Pump

- The power of the pump is 2.2 l/s (liter/second)
- 2-second latency: if the pump is switched on (t_{2k+1}) or off (t_{2k+2}) at time points

$$0 \leq t_1 \leq t_2 \leq \cdots \leq t_i \leq t_{i+1} \leq \cdots ,$$

then

$$t_{i+1} - t_i \geq 2$$

for any $i \geq 1$

- It is obvious that the pump can be turned on at most 5 times in one cycle

Control Objectives

Determine the t_i's in order to

- $\mathrm{R_s}$ (*safety*): maintain

$$v(t) \in [V_{\min}, V_{\max}], \quad \forall t \in [0, \infty)$$

  - $v(t)$ denotes the oil volume in the accumulator at time t
  - $V_{\min} = 4.9l$ (liter)
  - $V_{\max} = 25.1l$

and considering the energy cost and wear of the system,

- $\mathrm{R_o}$ (*optimality*): minimize the average accumulated oil volume in the limit, i.e. minimize

$$\lim_{T\to\infty} \frac{1}{T} \int_{t=0}^{T} v(t)\mathrm{d}t$$

Control Objectives

Determine the t_i's in order to

- $\mathrm{R_s}$ (*safety*): maintain

$$v(t) \in [V_{\min}, V_{\max}], \quad \forall t \in [0, \infty)$$

  - $v(t)$ denotes the oil volume in the accumulator at time t
  - $V_{\min} = 4.9l$ (liter)
  - $V_{\max} = 25.1l$

and considering the energy cost and wear of the system,

- $\mathrm{R_o}$ (*optimality*): minimize the average accumulated oil volume in the limit, i.e. minimize

$$\lim_{T \to \infty} \frac{1}{T} \int_{t=0}^{T} v(t) \mathrm{d}t$$

Control Objectives (Cont'd)

Both objectives should be achieved under constraints:

- R_{pl} (*pump latency*): $t_{i+1} - t_i \geq 2$
- R_{r} (*robustness*): uncertainties of the system should be taken into account:
  - fluctuation of consumption rate (if it is not 0), up to $f = 0.1 l/s$
  - imprecision in the measurement of oil volume, up to $\epsilon = 0.06 l$
  - imprecision in the measurement of time, up to $\delta = 0.015 s$.

Control Objectives (Cont'd)

Both objectives should be achieved under constraints:

- R_{pl} (*pump latency*): $t_{i+1} - t_i \geq 2$
- R_{r} (*robustness*): uncertainties of the system should be taken into account:
  - fluctuation of consumption rate (if it is not 0), up to $f = 0.1 l/s$
  - imprecision in the measurement of oil volume, up to $\epsilon = 0.06 l$
  - imprecision in the measurement of time, up to $\delta = 0.015 s$.

Control Objectives (Cont'd)

Both objectives should be achieved under constraints:

- R_{pl} (*pump latency*): $t_{i+1} - t_i \geq 2$
- R_{r} (*robustness*): uncertainties of the system should be taken into account:
  - fluctuation of consumption rate (if it is not 0), up to $f = 0.1 l/s$
  - imprecision in the measurement of oil volume, up to $\epsilon = 0.06 l$
  - imprecision in the measurement of time, up to $\delta = 0.015 s$.

Control Objectives (Cont'd)

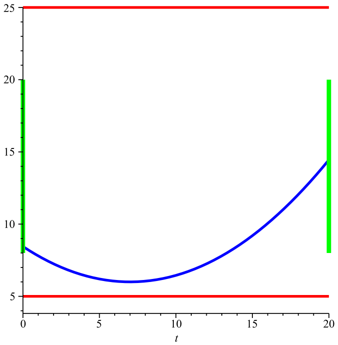
Both objectives should be achieved under constraints:

- R_{pl} (*pump latency*): $t_{i+1} - t_i \geq 2$

- R_{r} (*robustness*): uncertainties of the system should be taken into account:
  - fluctuation of consumption rate (if it is not 0), up to $f = 0.1 l/s$
  - imprecision in the measurement of oil volume, up to $\epsilon = 0.06 l$
  - imprecision in the measurement of time, up to $\delta = 0.015 s$.

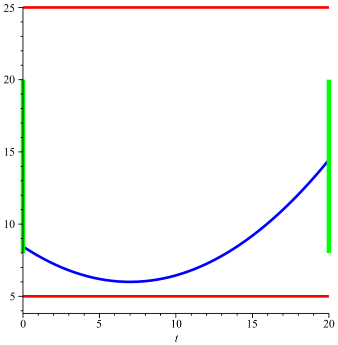
Localize the Controller

- $0 \leq t_1 \leq t_2 \leq \cdots \leq t_i \leq t_{i+1} \leq \cdots$
- Employing the periodicity
- Stable interval $[L, U] \subseteq [V_{\min}, V_{\max}]$



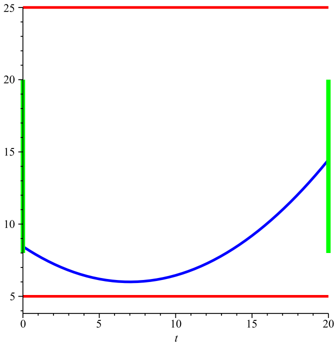
Localize the Controller

- $0 \leq t_1 \leq t_2 \leq \cdots \leq t_i \leq t_{i+1} \leq \cdots$
- Employing the periodicity
- Stable interval $[L, U] \subseteq [V_{\min}, V_{\max}]$

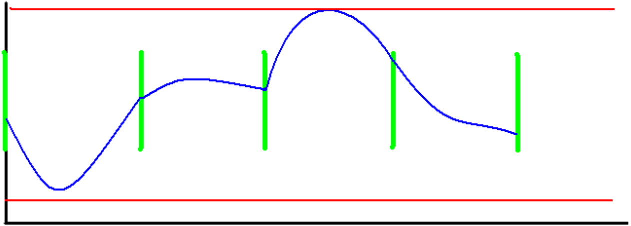


Localize the Controller

- $0 \leq t_1 \leq t_2 \leq \cdots \leq t_i \leq t_{i+1} \leq \cdots$
- Employing the periodicity
- Stable interval $[L, U] \subseteq [V_{\min}, V_{\max}]$



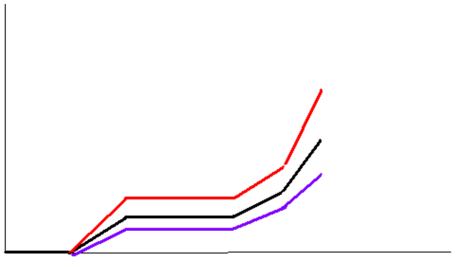
Repeated Cycles



Step 1: Modeling Oil Consumption

time	[2,4]	[8,10]	[10,12]	[14,16]	[16,18]
rate	1.2	1.2	2.5	1.7	0.5

- fluctuation of consumption rate: $f = 0.1$



Step 1: Modeling Oil Consumption

time	[2,4]	[8,10]	[10,12]	[14,16]	[16,18]
rate	1.2	1.2	2.5	1.7	0.5
- fluctuation of consumption rate: $f = 0.1$

$$
C_1 \mathrel{\widehat{=}}
\begin{array}{lcl}
(0 \le t \le 2 & \longrightarrow & V_{out} = 0) \\
\wedge\, (2 \le t \le 4 & \longrightarrow & 1.1(t-2) \le V_{out} \le 1.3(t-2)) \\
\wedge\, (4 \le t \le 8 & \longrightarrow & 2.2 \le V_{out} \le 2.6) \\
\wedge\, (8 \le t \le 10 & \longrightarrow & 2.2 + 1.1(t-8) \le V_{out} \le 2.6 + 1.3(t-8)) \\
\wedge\, (10 \le t \le 12 & \longrightarrow & 4.4 + 2.4(t-10) \le V_{out} \le 5.2 + 2.6(t-10)) \\
\wedge\, (12 \le t \le 14 & \longrightarrow & 9.2 \le V_{out} \le 10.4) \\
\wedge\, (14 \le t \le 16 & \longrightarrow & 9.2 + 1.6(t-14) \le V_{out} \le 10.4 + 1.8(t-14)) \\
\wedge\, (16 \le t \le 18 & \longrightarrow & 12.4 + 0.4(t-16) \le V_{out} \le 14 + 0.6(t-16)) \\
\wedge\, (18 \le t \le 20 & \longrightarrow & 13.2 \le V_{out} \le 15.2)
\end{array}
$$

Step 1: Modeling the Pump

- We will first assume that the pump is activated at most twice in one cycle: t_1, t_2, t_3, t_4
- $t_{i+1} - t_i \geq 2$:

$$C_2 \mathrel{\widehat{=}} \begin{array}{l} \quad (t_1 \geq 2 \wedge t_2 - t_1 \geq 2 \wedge t_3 - t_2 \geq 2 \wedge t_4 - t_3 \geq 2 \wedge t_4 \leq 20) \\ \vee (t_1 \geq 2 \wedge t_2 - t_1 \geq 2 \wedge t_2 \leq 20 \wedge t_3 = 20 \wedge t_4 = 20) \\ \vee (t_1 = 20 \wedge t_2 = 20 \wedge t_3 = 20 \wedge t_4 = 20) \end{array}.$$

- $2.2 l/s$

$$C_3 \mathrel{\widehat{=}} \begin{array}{lcl} \quad (0 \leq t \leq t_1 & \longrightarrow & V_{in} = 0) \\ \wedge (t_1 \leq t \leq t_2 & \longrightarrow & V_{in} = 2.2(t - t_1)) \\ \wedge (t_2 \leq t \leq t_3 & \longrightarrow & V_{in} = 2.2(t_2 - t_1)) \\ \wedge (t_3 \leq t \leq t_4 & \longrightarrow & V_{in} = 2.2(t_2 - t_1) + 2.2(t - t_3)) \\ \wedge (t_4 \leq t \leq 20 & \longrightarrow & V_{in} = 2.2(t_2 + t_4 - t_1 - t_3)) \end{array}.$$

Step 1: Modeling the Pump

- We will first assume that the pump is activated at most twice in one cycle: t_1, t_2, t_3, t_4
- $t_{i+1} - t_i \geq 2$:

$$C_2 \mathrel{\widehat{=}} \begin{array}{l} \quad (t_1 \geq 2 \wedge t_2 - t_1 \geq 2 \wedge t_3 - t_2 \geq 2 \wedge t_4 - t_3 \geq 2 \wedge t_4 \leq 20) \\ \vee (t_1 \geq 2 \wedge t_2 - t_1 \geq 2 \wedge t_2 \leq 20 \wedge t_3 = 20 \wedge t_4 = 20) \\ \vee (t_1 = 20 \wedge t_2 = 20 \wedge t_3 = 20 \wedge t_4 = 20) \end{array} .$$

- $2.2l/s$

$$C_3 \mathrel{\widehat{=}} \begin{array}{lll} \quad (0 \leq t \leq t_1 & \longrightarrow & V_{in} = 0) \\ \wedge (t_1 \leq t \leq t_2 & \longrightarrow & V_{in} = 2.2(t - t_1)) \\ \wedge (t_2 \leq t \leq t_3 & \longrightarrow & V_{in} = 2.2(t_2 - t_1)) \\ \wedge (t_3 \leq t \leq t_4 & \longrightarrow & V_{in} = 2.2(t_2 - t_1) + 2.2(t - t_3)) \\ \wedge (t_4 \leq t \leq 20 & \longrightarrow & V_{in} = 2.2(t_2 + t_4 - t_1 - t_3)) \end{array} .$$

Step 1: Modeling the Pump

- We will first assume that the pump is activated at most twice in one cycle: t_1, t_2, t_3, t_4
- $t_{i+1} - t_i \geq 2$:

$$C_2 \mathrel{\widehat{=}} \begin{array}{l} (t_1 \geq 2 \wedge t_2 - t_1 \geq 2 \wedge t_3 - t_2 \geq 2 \wedge t_4 - t_3 \geq 2 \wedge t_4 \leq 20) \\ \vee (t_1 \geq 2 \wedge t_2 - t_1 \geq 2 \wedge t_2 \leq 20 \wedge t_3 = 20 \wedge t_4 = 20) \\ \vee (t_1 = 20 \wedge t_2 = 20 \wedge t_3 = 20 \wedge t_4 = 20) \end{array}.$$

- $2.2l/s$

$$C_3 \mathrel{\widehat{=}} \begin{array}{lll} (0 \leq t \leq t_1 & \longrightarrow & V_{in} = 0) \\ \wedge (t_1 \leq t \leq t_2 & \longrightarrow & V_{in} = 2.2(t - t_1)) \\ \wedge (t_2 \leq t \leq t_3 & \longrightarrow & V_{in} = 2.2(t_2 - t_1)) \\ \wedge (t_3 \leq t \leq t_4 & \longrightarrow & V_{in} = 2.2(t_2 - t_1) + 2.2(t - t_3)) \\ \wedge (t_4 \leq t \leq 20 & \longrightarrow & V_{in} = 2.2(t_2 + t_4 - t_1 - t_3)) \end{array}.$$

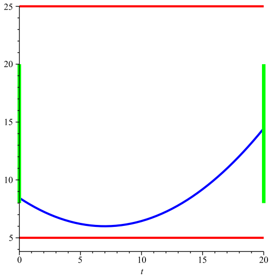
Step 1: Encoding Safety Requirements

- Oil volume in the accumulator:

$$C_4 \mathrel{\widehat{=}} v = v_0 + V_{in} - V_{out} \quad .$$

- Inductiveness and safety (considering robustness):

$$\begin{aligned} C_5 &\mathrel{\widehat{=}} \quad t = 20 \longrightarrow L{+}0.2 \leq v \leq U{-}0.2 \\ C_6 &\mathrel{\widehat{=}} \quad 0 \leq t \leq 20 \longrightarrow V_{\min}{+}0.2 \leq v \leq V_{\max}{-}0.2 \quad . \end{aligned}$$



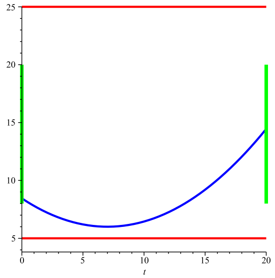
Step 1: Encoding Safety Requirements

- Oil volume in the accumulator:

$$C_4 \mathrel{\widehat{=}} v = v_0 + V_{in} - V_{out} \quad .$$

- Inductiveness and safety (considering robustness):

$$\begin{aligned} C_5 \quad &\widehat{=} \quad t = 20 \longrightarrow L{+}0.2 \leq v \leq U{-}0.2 \\ C_6 \quad &\widehat{=} \quad 0 \leq t \leq 20 \longrightarrow V_{\min}{+}0.2 \leq v \leq V_{\max}{-}0.2 \quad . \end{aligned}$$



Step 1: Encoding Safety Requirements (Cont'd)

$$\mathcal{S} \mathrel{\widehat{=}} \forall t, v, V_{in}, V_{out}.(C_1 \wedge C_3 \wedge C_4 \longrightarrow C_5 \wedge C_6)\,.$$

- C_1: oil consumed
- C_3: oil pumped
- C_4: oil in the accumulator
- C_5: inductiveness
- C_6: (local) safety

$$C_8 \mathrel{\widehat{=}} \forall v_0.\Big(C_7 \longrightarrow \exists t_1 t_2 t_3 t_4.\big(C_2 \wedge \mathcal{S}\big)\Big)\,.$$

- $C_7 \mathrel{\widehat{=}} L \leq v_0 \leq U$
- C_2: 2-second latency

Step 1: Encoding Safety Requirements (Cont'd)

$$\mathcal{S} \mathrel{\widehat{=}} \forall t, v, V_{in}, V_{out}.(C_1 \wedge C_3 \wedge C_4 \longrightarrow C_5 \wedge C_6)\,.$$

- C_1: oil consumed
- C_3: oil pumped
- C_4: oil in the accumulator
- C_5: inductiveness
- C_6: (local) safety

$$C_8 \mathrel{\widehat{=}} \forall v_0.\Big(C_7 \longrightarrow \exists t_1 t_2 t_3 t_4.\big(C_2 \wedge \mathcal{S}\big)\Big)\,.$$

- $C_7 \mathrel{\widehat{=}} L \leq v_0 \leq U$
- C_2: 2-second latency

Step 1: Encoding Safety Requirements (Cont'd)

$$\mathcal{S} \mathrel{\widehat{=}} \forall t, v, V_{in}, V_{out}.(C_1 \wedge C_3 \wedge C_4 \longrightarrow C_5 \wedge C_6)\,.$$

- C_1: oil consumed
- C_3: oil pumped
- C_4: oil in the accumulator
- C_5: inductiveness
- C_6: (local) safety

$$C_8 \mathrel{\widehat{=}} \forall v_0.\Big(C_7 \longrightarrow \exists t_1 t_2 t_3 t_4.\big(C_2 \wedge \mathcal{S}\big)\Big)\quad.$$

- $C_7 \mathrel{\widehat{=}} L \leq v_0 \leq U$
- C_2: 2-second latency

Deriving Constraints

Applying QE to

$$C_8 \mathrel{\widehat{=}} \forall v_0. \Big(C_7 \longrightarrow \exists t_1 t_2 t_3 t_4. \big(C_2 \wedge \mathcal{S} \big) \Big) \quad ,$$

we get

$$C_9 \mathrel{\widehat{=}} L \geq 5.1 \wedge U \leq 24.9 \wedge U - L \geq 2.4 \quad .$$

Deriving Constraints

Applying QE to

$$C_8 \mathrel{\widehat{=}} \forall v_0 . \Big(C_7 \longrightarrow \exists t_1 t_2 t_3 t_4 . \big(C_2 \land \mathcal{S} \big) \Big) \quad ,$$

we get

$$C_9 \mathrel{\widehat{=}} L \geq 5.1 \land U \leq 24.9 \land U - L \geq 2.4 \quad .$$

Deriving Constraints (Cont'd)

$$C_{10} \mathrel{\widehat{=}} C_2 \wedge C_7 \wedge C_9 \wedge \mathcal{S}\,.$$

- C_2: 2-second latency
- $C_7 : L \leq v_0 \leq U$
- C_9: constraint on L, U
- $\mathcal{S}$: safety and inductiveness

After QE:

$$\mathcal{D}(L, U, v_0, t_1, t_2, t_3, t_4) \mathrel{\widehat{=}} \bigvee_{i=1}^{92} D_i$$

Deriving Constraints (Cont'd)

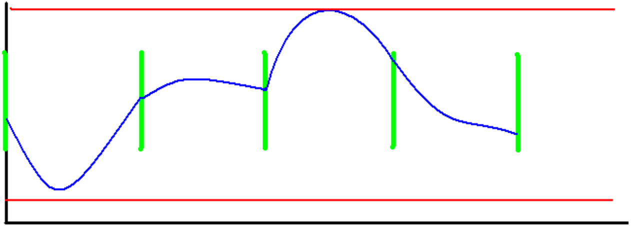
$$C_{10} \mathrel{\widehat{=}} C_2 \wedge C_7 \wedge C_9 \wedge \mathcal{S}\,.$$

- C_2: 2-second latency
- $C_7 : L \leq v_0 \leq U$
- C_9: constraint on L, U
- $\mathcal{S}$: safety and inductiveness

After QE:

$$\mathcal{D}(L, U, v_0, t_1, t_2, t_3, t_4) \mathrel{\widehat{=}} \bigvee_{i=1}^{92} D_i$$

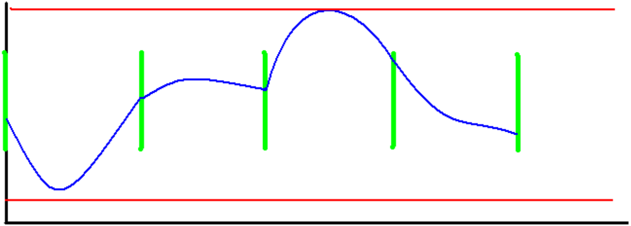
Step 2: Optimization Criterion



$\mathrm{R_o}$ (*optimality*): minimize the average accumulated oil volume in the limit, i.e. minimize

$$\lim_{T\to\infty} \frac{1}{T} \int_{t=0}^{T} v(t)\mathrm{d}t$$

Optimization Criterion (Contd.)



- R'_{o} : $\min\limits_{[L,U]} \max\limits_{v_0 \in [L,U]} \min\limits_{\mathbf{t}} \frac{1}{20} \int_{t=0}^{20} v(t)\mathrm{d}t$.

Step 2: Encoding the Optimization Criterion

Cost function:

$$
\begin{aligned}
g(v_0, t_1, t_2, t_3, t_4) &\mathrel{\widehat{=}} \frac{1}{20}\int_{t=0}^{20} v(t)\mathrm{d}t \\
&= \frac{20v_0+1.1(t_1^2-t_2^2+t_3^2-t_4^2-40t_1+40t_2-40t_3+40t_4)-132.2}{20}
\end{aligned}
$$

R'_{o} can be encoded into

$$\exists L, U.\Big(C_9 \wedge \forall v_0.\big(C_7 \longrightarrow \exists t_1 t_2 t_3 t_4.(\mathcal{D} \wedge g \leq z)\big)\Big),$$

which is equivalent to $z \geq z^*$ or $z > z^*$

Step 2: Encoding the Optimization Criterion

Cost function:

$$g(v_0, t_1, t_2, t_3, t_4) \;\widehat{=}\; \frac{1}{20}\int_{t=0}^{20} v(t)\mathrm{d}t$$
$$= \frac{20v_0 + 1.1(t_1^2 - t_2^2 + t_3^2 - t_4^2 - 40t_1 + 40t_2 - 40t_3 + 40t_4) - 132.2}{20}$$

R'_{o} can be encoded into

$$\exists L, U.\Big(C_9 \wedge \forall v_0.\big(C_7 \longrightarrow \exists t_1 t_2 t_3 t_4.(\mathcal{D} \wedge g \leq z)\big)\Big),$$

which is equivalent to $z \geq z^*$ or $z > z^*$

Step 3: Performing QE

$$\exists L, U. \Big(C_9 \wedge \forall v_0. \big(C_7 \longrightarrow \exists t_1 t_2 t_3 t_4. (\mathcal{D} \wedge g \leq z) \big) \Big)$$

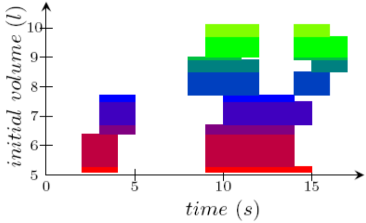
- the inner $\exists$: *qudratic programming*
- the outer $\exists$: discretization

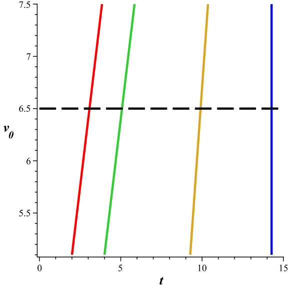
$$L \geq 5.1 \wedge U \leq 24.9 \wedge U - L \geq 2.4$$

- the middle $\forall$: divide and conquer

Optimal Controllers with 2 Activations

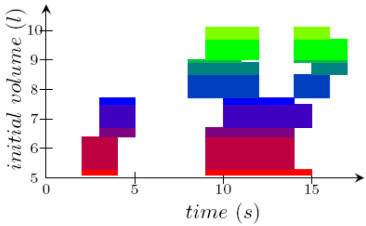
- In [Cassez et al hscc09], the optimal value 7.95 is obtained at interval [5.1,8.3]
- Using our approach, the optimal value is 7.53 (a 5% improvement) and the corresponding interval is $[5.1, 7.5]$
- Comparison of local optimal controllers: (the left one comes from [Cassez et al hscc09])

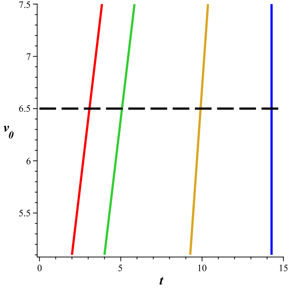




Optimal Controllers with 2 Activations

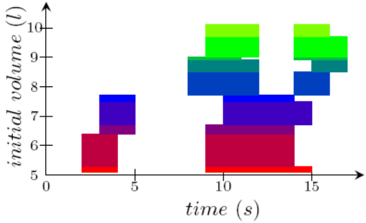
- In [Cassez et al hscc09], the optimal value 7.95 is obtained at interval [5.1,8.3]
- Using our approach, the optimal value is 7.53 (a 5% improvement) and the corresponding interval is $[5.1, 7.5]$
- Comparison of local optimal controllers: (the left one comes from [Cassez et al hscc09])

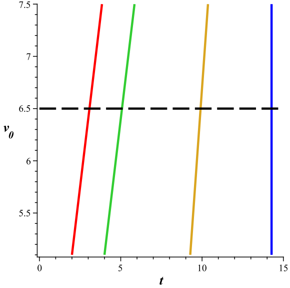




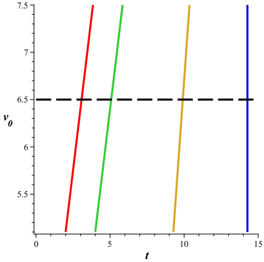
Optimal Controllers with 2 Activations

- In [Cassez et al hscc09], the optimal value 7.95 is obtained at interval [5.1,8.3]
- Using our approach, the optimal value is 7.53 (a 5% improvement) and the corresponding interval is $[5.1, 7.5]$
- Comparison of local optimal controllers: (the left one comes from [Cassez et al hscc09])





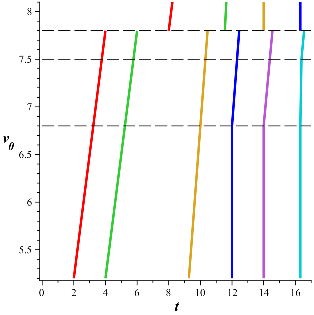
Local Optimal Controllers — 2 Activations



$$t_1 = \frac{10v_0 - 25}{13} \wedge t_2 = \frac{10v_0 + 1}{13} \wedge t_3 = \frac{10v_0 + 153}{22} \wedge t_4 = \frac{157}{11}$$

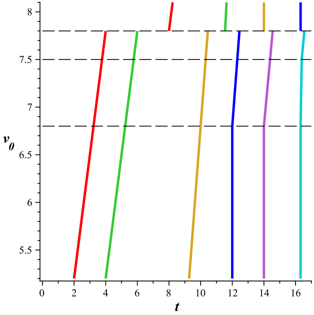
Improvement by Increasing Activations

- The pump is allowed to be switched on at most 3 times in one cycle
- The optimal average accumulated oil volume 7.35 (a 7.5% improvement) is obtained at interval $[5.2, 8.1]$
- The local optimal controllers corresponding to $v_0 \in [5.2, 8.1]$:



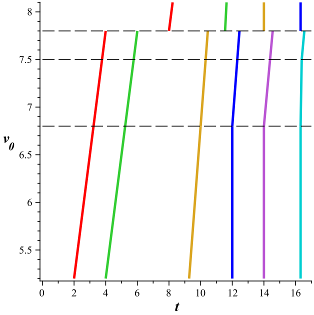
Improvement by Increasing Activations

- The pump is allowed to be switched on at most 3 times in one cycle
- The optimal average accumulated oil volume 7.35 (a 7.5% improvement) is obtained at interval $[5.2, 8.1]$
- The local optimal controllers corresponding to $v_0 \in [5.2, 8.1]$:

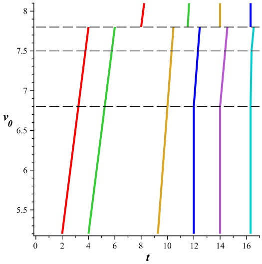


Improvement by Increasing Activations

- The pump is allowed to be switched on at most 3 times in one cycle
- The optimal average accumulated oil volume 7.35 (a 7.5% improvement) is obtained at interval [5.2, 8.1]
- The local optimal controllers corresponding to $v_0 \in [5.2, 8.1]$:



Local Optimal Controllers — 3 Activations



$$\begin{cases} t_1=\frac{10v_0-26}{13} \wedge t_2=\frac{10v_0}{13} \wedge t_3=\frac{5v_0+76}{11} \wedge t_4=12 \wedge t_5=14 \wedge t_6=\frac{359}{22} & v_0\in[5.2,6.8) \\ t_1=\frac{10v_0-26}{13} \wedge t_2=\frac{10v_0}{13} \wedge t_3=\frac{5v_0+76}{11} \wedge t_4=\frac{5v_0+98}{11} \wedge t_5=\frac{5v_0+92}{9} \wedge t_6=\frac{20v_0+3095}{198} & v_0\in[6.8,7.5) \\ t_1=\frac{10v_0-26}{13} \wedge t_2=\frac{10v_0}{13} \wedge t_3=\frac{5v_0+76}{11} \wedge t_4=\frac{5v_0+98}{11} \wedge t_5=\frac{5v_0+92}{9} \wedge t_6=\frac{5v_0+110}{9} & v_0\in[7.5,7.8) \\ t_1=\frac{10v_0+26}{13} \wedge t_2=\frac{45v_0+1300}{143} \wedge t_3=14 \wedge t_4=\frac{359}{22} \wedge t_5=20 \wedge t_6=20 & v_0\in[7.8,8.1] \end{cases}$$

Three Activations are Enough

Proposition

For each admissible $[L, U]$, each $v_0 \in [L, U]$, and any local control strategy s_4 with at least 4 activations subject to R_{lu}, R_i and R_{ls}, there exists a local control strategy s_3 subject to R_{lu}, R_i and R_{ls} with 3 activations such that

$$\frac{1}{20}\int_{t=0}^{20} v_{s_3}(t)\mathrm{d}t < \frac{1}{20}\int_{t=0}^{20} v_{s_4}(t)\mathrm{d}t$$

where $v_{s_3}(t)$ (resp. $v_{s_4}(t)$) is the oil volume in the accumulator at t with s_3 (resp. s_4).

Outline

Conclusions

- Hybrid systems attracts more and more interests with the development of **safety critical** embedded systems
- Invariant plays an important role in the study **(formal verification, controller synthesis)** of hybrid systems
- Semi-algebraic inductive invariant checking for polynomial continuous/hybrid systems is **decidable**
- Use parametric polynomials and symbolic computation to automatically discover invariants, and to perform optimization
  - rigorous
  - high complexity (may be combined with numeric computation)
  - Non-polynomial systems transformed to polynomials ones
- Case studies show good prospect of proposed methods

Conclusions

- Hybrid systems attracts more and more interests with the development of **safety critical** embedded systems
- Invariant plays an important role in the study **(formal verification, controller synthesis)** of hybrid systems
- Semi-algebraic inductive invariant checking for polynomial continuous/hybrid systems is **decidable**
- Use parametric polynomials and symbolic computation to automatically discover invariants, and to perform optimization
  - rigorous
  - high complexity (may be combined with numeric computation)
  - Non-polynomial systems transformed to polynomials ones
- Case studies show good prospect of proposed methods

Conclusions

- Hybrid systems attracts more and more interests with the development of **safety critical** embedded systems
- Invariant plays an important role in the study **(formal verification, controller synthesis)** of hybrid systems
- Semi-algebraic inductive invariant checking for polynomial continuous/hybrid systems is **decidable**
- Use parametric polynomials and symbolic computation to automatically discover invariants, and to perform optimization
  - rigorous
  - high complexity (may be combined with numeric computation)
  - Non-polynomial systems transformed to polynomials ones
- Case studies show good prospect of proposed methods

Conclusions

- Hybrid systems attracts more and more interests with the development of **safety critical** embedded systems
- Invariant plays an important role in the study **(formal verification, controller synthesis)** of hybrid systems
- Semi-algebraic inductive invariant checking for polynomial continuous/hybrid systems is **decidable**
- Use parametric polynomials and symbolic computation to automatically discover invariants, and to perform optimization
  - rigorous
  - high complexity (may be combined with numeric computation)
  - Non-polynomial systems transformed to polynomials ones
- Case studies show good prospect of proposed methods

Conclusions

- Hybrid systems attracts more and more interests with the development of **safety critical** embedded systems
- Invariant plays an important role in the study **(formal verification, controller synthesis)** of hybrid systems
- Semi-algebraic inductive invariant checking for polynomial continuous/hybrid systems is **decidable**
- Use parametric polynomials and symbolic computation to automatically discover invariants, and to perform optimization
  - rigorous
  - high complexity (may be combined with numeric computation)
  - Non-polynomial systems transformed to polynomials ones
- Case studies show good prospect of proposed methods

Conclusions

- Hybrid systems attracts more and more interests with the development of **safety critical** embedded systems
- Invariant plays an important role in the study **(formal verification, controller synthesis)** of hybrid systems
- Semi-algebraic inductive invariant checking for polynomial continuous/hybrid systems is **decidable**
- Use parametric polynomials and symbolic computation to automatically discover invariants, and to perform optimization
  - rigorous
  - high complexity (may be combined with numeric computation)
  - Non-polynomial systems transformed to polynomials ones
- Case studies show good prospect of proposed methods

Conclusions

- Hybrid systems attracts more and more interests with the development of **safety critical** embedded systems
- Invariant plays an important role in the study **(formal verification, controller synthesis)** of hybrid systems
- Semi-algebraic inductive invariant checking for polynomial continuous/hybrid systems is **decidable**
- Use parametric polynomials and symbolic computation to automatically discover invariants, and to perform optimization
  - rigorous
  - high complexity (may be combined with numeric computation)
  - Non-polynomial systems transformed to polynomials ones
- Case studies show good prospect of proposed methods

Conclusions

- Hybrid systems attracts more and more interests with the development of **safety critical** embedded systems
- Invariant plays an important role in the study **(formal verification, controller synthesis)** of hybrid systems
- Semi-algebraic inductive invariant checking for polynomial continuous/hybrid systems is **decidable**
- Use parametric polynomials and symbolic computation to automatically discover invariants, and to perform optimization
  - rigorous
  - high complexity (may be combined with numeric computation)
  - Non-polynomial systems transformed to polynomials ones
- Case studies show good prospect of proposed methods

Thank you!

Questions?