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 Talk 1: Preliminaries
   - Polynomials and Polynomial Ideals
   - First-order Theory of Reals
   - Continuous Dynamical Systems
   - Hybrid Automata

3 Talk 2: 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 Talk 3: 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 exhibit combinations of discrete jumps and continuous evolution.

Examples

- Automobiles
- Medical
- Aircraft
- Entertainment
- Environmental Monitoring
- Military

2011.7.23 Wenzhou Train Crash Accident
2016.3.26 Japanese Space Center ASTRO-H
Hybrid systems exhibit combinations of discrete jumps and continuous evolution.

Examples

- Automobiles
- Medical
- Airplanes
- Entertainment
- Military
- Environmental Monitoring

2011.7.23 Wenzhou Train Crash Accident
2016.3.26 Japanese Space Center ASTRO-H
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, ...
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 up to 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 (predictability), 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 up to 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 up to 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 up to 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 up to 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 up to 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, ...
**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, . . .
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
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
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]

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
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
Deduction based approach [Platzer & Clarke 2008]

- **Basic idea:** extending Floyd-Hoare-Naur inductive assertion method to hybrid systems.
- **Elements:**
  - A compositional modelling language
  - 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 [Platzer 2008], DADL [Platzer 2010], EDC [Zhou et al 1994], ...
- **Advantages:** scalability
- **Difficulties:** invariant generation
Deduction based approach [Platzer&Clarke 2008]

**Basic idea:** extending Floyd-Hoare-Naur inductive assertion method to hybrid systems.

**Elements:**
- A compositional modelling language
- 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
### Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending **Floyd-Hoare-Naur inductive assertion method** to hybrid systems.
- **Elements:**
  - A compositional modelling language
  - 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
Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending Floyd-Hoare-Naur inductive assertion method to hybrid systems.
- **Elements:**
  - A compositional modelling language
  - 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
Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending Floyd-Hoare-Naur inductive assertion method to hybrid systems.
- **Elements:**
  - A compositional modelling language
  - 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 language
  - 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
Deduction based approach [Platzer&Clarke 2008]

- **Basic idea:** extending *Floyd-Hoare-Naur inductive assertion method* to hybrid systems.
- **Elements:**
  - A compositional modelling language
  - 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
Deduction based approach [Platzer&Clarke 2008]

Basic idea: extending Floyd-Hoare-Naur inductive assertion method to hybrid systems.

Elements:
- A compositional modelling language
- 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
## Schedule and References

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

### References
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
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
Let $K$ 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 $x$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^{n} \alpha_i$ is called the **degree** of $x^\alpha$.

A **polynomial** $p(x)$ in $x$ with coefficients in $K$ is of the form $\sum_{\alpha} c_\alpha x^\alpha$, where all $c_\alpha \in K$.

- 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 $K$ form a **polynomial ring** $K[x]$.

A **parametric polynomial** is of the form $\sum_{\alpha} u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $\binom{n+d}{d}$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an **instantiation** of $p(u, x)$. 
Let $K$ 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 $x$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the **degree** of $x^\alpha$.

A **polynomial** $p(x)$ in $x$ with coefficients in $K$ is of the form $\sum_{\alpha} c_\alpha x^\alpha$, where all $c_\alpha \in K$.

- 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 $(_{n+d}^{n+d})$ many monomials.
- The set of all polynomials in $x_1, x_2, \ldots, x_n$ with coefficients in $K$ form a **polynomial ring** $K[x]$.

A **parametric polynomial** is of the form $\sum_{\alpha} u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $(_{n+d}^{n+d})$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an **instantiation** of $p(u, x)$. 
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 $x$) is a product form $x_1^{\alpha_1}x_2^{\alpha_2}\cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the *degree* of $x^\alpha$.

A *polynomial* $p(x)$ in $x$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\alpha} c_\alpha x^\alpha$, where all $c_\alpha \in \mathbb{K}$.

- The *degree* $\text{deg}(p)$ 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* $\mathbb{K}[x]$.

A *parametric polynomial* is of the form $\sum_{\alpha} u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $\binom{n+d}{d}$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an *instantiation* of $p(u, x)$. 
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 $x$) is a product form $x_1^{\alpha_1}x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the **degree** of $x^\alpha$.

A **polynomial** $p(x)$ in $x$ with coefficients in $\mathbb{K}$ is of the form $\sum_\alpha c_\alpha x^\alpha$, where all $c_\alpha \in \mathbb{K}$.

- The **degree** $\text{deg}(p)$ 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** $\mathbb{K}[x]$.

A **parametric polynomial** is of the form $\sum_\alpha u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $\binom{n+d}{d}$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an **instantiation** of $p(u, x)$.
Let $K$ 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 $x$) is a product form $x_1^{\alpha_1}x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the **degree** of $x^\alpha$.

A **polynomial** $p(x)$ in $x$ with coefficients in $K$ is of the form $\sum_\alpha c_\alpha x^\alpha$, where all $c_\alpha \in K$.

- 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 $K$ form a **polynomial ring** $K[x]$.

A **parametric polynomial** is of the form $\sum_\alpha u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $\binom{n+d}{d}$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an **instantiation** of $p(u, x)$. 
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 $x$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the **degree** of $x^\alpha$.

A **polynomial** $p(x)$ in $x$ with coefficients in $\mathbb{K}$ is of the form $\sum_\alpha c_\alpha x^\alpha$, where all $c_\alpha \in \mathbb{K}$.

- 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** $\mathbb{K}[x]$.

A **parametric polynomial** is of the form $\sum_\alpha u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $\binom{n+d}{d}$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an **instantiation** of $p(u, x)$.
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 $x$) is a product form $x_1^{\alpha_1}x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the **degree** of $x^\alpha$.

A **polynomial** $p(x)$ in $x$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\alpha} c_\alpha x^\alpha$, where all $c_\alpha \in \mathbb{K}$.

- 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** $\mathbb{K}[x]$.

A **parametric polynomial** is of the form $\sum_{\alpha} u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $\binom{n+d}{d}$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an **instantiation** of $p(u, x)$.
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 $x$ ) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$. The number $\sum_{i=1}^n \alpha_i$ is called the **degree** of $x^\alpha$.

A **polynomial** $p(x)$ in $x$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\alpha} c_\alpha x^\alpha$, where all $c_\alpha \in \mathbb{K}$.

- 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** $\mathbb{K}[x]$.

A **parametric polynomial** is of the form $\sum_{\alpha} u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $\binom{n+d}{d}$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an instantiation of $p(u, x)$.
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 $x$) is a product form $x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$, or briefly $x^\alpha$, where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$.

The number $\sum_{i=1}^{n} \alpha_i$ is called the **degree** of $x^\alpha$.

A **polynomial** $p(x)$ in $x$ with coefficients in $\mathbb{K}$ is of the form $\sum_{\alpha} c_\alpha x^\alpha$, where all $c_\alpha \in \mathbb{K}$.

- 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** $\mathbb{K}[x]$.

A **parametric polynomial** is of the form $\sum_{\alpha} u_\alpha x^\alpha$, where $u_\alpha \in \mathbb{R}$ are not constants but undetermined parameters, can be regarded as a standard polynomial $p(u, x)$ in $\mathbb{R}[u, x]$.

- A parametric polynomial with degree $d$ (in $x$) has at most $\binom{n+d}{d}$ many indeterminates.
- For any $u_0 \in \mathbb{R}^w$, $p_{u_0}(x) \in \mathbb{R}[x]$ obtained by substituting $u_0$ for $u$ in $p(u, x)$ is an **instantiation** of $p(u, x)$. 
A subset \( I \subseteq \mathbb{K}[x] \) 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 \( h \in \mathbb{K}[x] \), then \( hp \in I \).

Let \( g_1, g_2, \ldots, g_s \in \mathbb{K}[x] \), then \( \langle g_1, g_2, \ldots, g_s \rangle \) 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

A subset $I \subseteq \mathbb{K}[x]$ 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 $h \in \mathbb{K}[x]$, then $hp \in I$.

Let $g_1, g_2, \ldots, g_s \in \mathbb{K}[x]$, then $\langle g_1, g_2, \ldots, g_s \rangle \triangleq \{ \sum_{i=1}^{s} h_i g_i : h_1, h_2, \ldots, h_s \in \mathbb{K}[x] \}$ 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 $I$. 
A subset $I \subseteq K[x]$ 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 $h \in K[x]$, then $hp \in I$.

Let $g_1, g_2, \ldots, g_s \in K[x]$, then $\langle g_1, g_2, \ldots, g_s \rangle \triangleq \{ \sum_{i=1}^{s} h_i g_i : h_1, h_2, \ldots, h_s \in K[x] \}$ 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 $I \subseteq \mathbb{K}[x]$ has a finite basis, that is, $I = \langle g_1, g_2, \ldots, g_s \rangle$ for some $g_1, g_2, \ldots, g_s \in \mathbb{K}[x]$.

Ascending Chain Theorem

For any ascending chain of ideals $I_1 \subseteq I_2 \subseteq \cdots \subseteq I_k \subseteq \cdots$ in $\mathbb{K}[x]$, there exists an $N \in \mathbb{N}$ such that $I_k = I_N$ for any $k \geq N$. 
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
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: $\land, \lor, \neg, \rightarrow, \leftrightarrow, \ldots$; 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: $\land, \lor, \neg, \to, \leftrightarrow, \ldots$; 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: $\wedge, \lor, \neg, \rightarrow, \leftrightarrow, \ldots$; 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: $\land, \lor, \neg, \rightarrow, \leftrightarrow, \ldots$; 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: $\land, \lor, \neg, \rightarrow, \leftrightarrow, \ldots$; 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 $\varphi$ in $\mathcal{T}$, there exists a quantifier-free formula $\varphi_{QF}$ which only contains free variables of $\varphi$ such that $\varphi \iff \varphi_{QF}$.

- $T(\mathbb{R})$ admits quantifier elimination.

- The **decidability** of $T(\mathbb{R})$

Example

$$\exists x. ax^2 + bx + c = 0 \iff a = b = c = 0 \lor (a = 0 \land b \neq 0) \lor (a \neq 0 \land b^2 - 4ac \geq 0)$$
Quantifier Elimination

Quantifier Elimination Property

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

Example

$$\exists x. ax^2 + bx + c = 0 \iff a = b = c = 0 \lor$$
$$\quad (a = 0 \land b \neq 0) \lor$$
$$\quad (a \neq 0 \land b^2 - 4ac \geq 0)$$
Quantifier Elimination

Quantifier Elimination Property

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

Example

$$\exists x. ax^2 + bx + c = 0 \iff a = b = c = 0 \lor (a = 0 \land b \neq 0) \lor (a \neq 0 \land b^2 - 4ac \geq 0)$$
Quantifier Elimination (Cont’d)

Semi-algebraic 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 $A = \{x \in \mathbb{R}^n \mid \phi(x) \text{ is true}\}$.
  - SASs are closed under common set operations:
    - $A(\phi_1) \cap A(\phi_2) = A(\phi_1 \land \phi_2)$;
    - $A(\phi_1) \cup A(\phi_2) = A(\phi_1 \lor \phi_2)$;
    - $A(\phi_1)^c = A(\neg \phi_1)$;
    - $A(\phi_1) \setminus A(\phi_2) = A(\phi_1) \cap A(\phi_2)^c = A(\phi_1 \land \neg \phi_2)$.
  - Any SAS can be represented by a QFF in the form of
    $$\phi(x) \equiv \bigvee_{k=1}^{K} \bigwedge_{j=1}^{J_k} p_{kj}(x) \triangleright 0,$$
    where $p_{kj}(x) \in \mathbb{Q}[x]$ and $\triangleright \in \{\geq, >\}$.

Semi-algebraic Template

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

$$\phi(u, x) \equiv \bigvee_{k=1}^{K} \bigwedge_{j=1}^{J_k} p_{kj}(u_{kj}, x) > 0.$$
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 $A = \{x \in \mathbb{R}^n \mid \phi(x) \text{ is true}\}$.

SASs are closed under common set operations:

- $A(\phi_1) \cap A(\phi_2) = A(\phi_1 \land \phi_2)$;
- $A(\phi_1) \cup A(\phi_2) = A(\phi_1 \lor \phi_2)$;
- $A(\phi_1)^c = A(\neg \phi_1)$;
- $A(\phi_1) \setminus A(\phi_2) = A(\phi_1) \cap A(\phi_2)^c = A(\phi_1 \land \neg \phi_2)$.

Any SAS can be represented by a QFF in the form of

$$\phi(x) \equiv \bigvee_{k=1}^{K} \bigwedge_{j=1}^{J_k} p_{kj}(x) \triangleright 0,$$

where $p_{kj}(x) \in \mathbb{Q}[x]$ and $\triangleright \in \{\geq, >\}$.

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

$$\phi(u, x) \equiv \bigvee_{k=1}^{K} \bigwedge_{j=1}^{J_k} p_{kj}(u_{kj}, x) \triangleright 0.$$
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 $A = \{x \in \mathbb{R}^n \mid \phi(x) \text{ is true}\}$.

SASs are closed under common set operations:

- $A(\phi_1) \cap A(\phi_2) = A(\phi_1 \land \phi_2)$;
- $A(\phi_1) \cup A(\phi_2) = A(\phi_1 \lor \phi_2)$;
- $A(\phi_1)^c = A(\neg \phi_1)$;
- $A(\phi_1) \setminus A(\phi_2) = A(\phi_1) \cap A(\phi_2)^c = A(\phi_1 \land \neg \phi_2)$.

Any SAS can be represented by a QFF in the form of

$$\phi(x) \equiv \bigvee_{k=1}^K \bigwedge_{j=1}^{J_k} p_{kj}(x) \triangleright 0,$$

where $p_{kj}(x) \in \mathbb{Q}[x]$ and $\triangleright \in \{\geq, >\}$.

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

$$\phi(u, x) \equiv \bigvee_{k=1}^K \bigwedge_{j=1}^{J_k} p_{kj}(u_{kj}, x) \triangleright 0.$$
Quantifier Elimination (Cont’d)

Semi-algebraic Set

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

Semi-algebraic Template

A **semi-algebraic template** with degree $d$ is of the form
\[ \phi(u, x) \equiv \bigvee_{k=1}^K \bigwedge_{j=1}^{J_k} p_{kj}(u_{kj}, x) \triangleright 0. \]
Quantifier Elimination (Cont’d)

Survey of QE Algorithms

- **Tarski’s algorithm** [Tarski 51]: the first one, but its complexity is nonelementary, impractical, 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.

Quantifier Elimination (Cont’d)

Survey of QE Algorithms

- **Tarski’s algorithm** [Tarski 51]: the first one, but its complexity is nonelementary, impractical, 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.

Quantifier Elimination (Cont’d)

Survey of QE Algorithms

- **Tarski’s algorithm** [Tarski 51]: the first one, but its complexity is nonelementary, impractical, 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.

Quantifier Elimination (Cont’d)

Survey of QE Algorithms

- **Tarski’s algorithm** [Tarski 51]: the first one, but its complexity is nonelementary, impractical, 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.

Outline

1. Background

2. Talk 1: Preliminaries
   - Polynomials and Polynomial Ideals
   - First-order Theory of Reals
   - Continuous Dynamical Systems
   - Hybrid Automata

3. Talk 2: 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. Talk 3: Controller Synthesis
   - Controller Synthesis with Safety
   - Controller Synthesis with Safety and Optimality
   - An Industrial Case Study: The Oil Pump Control Problem

5. Conclusions
Continuous Dynamical Systems

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

  \[ \dot{x} = f(x), \quad (1) \]

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

- If \( f \) in (1) satisfies **local Lipschitz condition**, then given \( x_0 \in \mathbb{R}^n \), there exists a unique solution \( x(x_0; t) : (a, b) \to \mathbb{R}^n \) such that \( x(x_0; 0) = x_0 \) and \( \forall t \in (a, b). \frac{dx(x_0; t)}{dt} = f(x(x_0; t)) \).

- If \( f \) in (1) satisfies **global Lipschitz condition**, then the existence, uniqueness and completeness of solutions to (1) can be guaranteed.

- The \( k \)-th **Lie derivatives** \( L_f^k \sigma : \mathbb{R}^n \to \mathbb{R} \) of \( \sigma \) along \( f \) is defined by:
  - \( L_f^0 \sigma(x) = \sigma(x) \),
  - \( L_f^k \sigma(x) = (\nabla L_f^{k-1} \sigma(x), f(x)) \), for \( k > 0 \),

  where \( \nabla \varphi(x) \equiv (\frac{\partial \varphi(x)}{\partial x_1}, \frac{\partial \varphi(x)}{\partial x_2}, \ldots, \frac{\partial \varphi(x)}{\partial x_n}) \) and \((\cdot, \cdot)\) is the **inner product** of two vectors.
Continuous Dynamical Systems

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

\[ \dot{x} = f(x), \quad (1) \]

where \( x \in \mathbb{R}^n \) and \( f : \mathbb{R}^n \rightarrow \mathbb{R}^n \) is a vector field.

- If \( f \) in (1) satisfies **local Lipschitz condition**, then given \( x_0 \in \mathbb{R}^n \), there exists a unique solution \( x(x_0; t) : (a, b) \rightarrow \mathbb{R}^n \) such that \( x(x_0; 0) = x_0 \) and \( \forall t \in (a, b). \frac{dx(x_0; t)}{dt} = f(x(x_0; t)) \).

- If \( f \) in (1) satisfies **global Lipschitz condition**, then the existence, uniqueness and completeness of solutions to (1) can be guaranteed.

- The \( k \)-th **Lie derivatives** \( L_f^k \sigma : \mathbb{R}^n \rightarrow \mathbb{R} \) of \( \sigma \) along \( f \) is defined by:
  - \( L_f^0 \sigma (x) = \sigma (x) \),
  - \( L_f^k \sigma (x) = ( \nabla L_f^{k-1} \sigma (x), f(x) ) \), for \( k > 0 \),

where \( \nabla \varphi (x) = \left( \frac{\partial \varphi (x)}{\partial x_1}, \frac{\partial \varphi (x)}{\partial x_2}, \ldots, \frac{\partial \varphi (x)}{\partial x_n} \right) \) and \( (\cdot, \cdot) \) is the inner product of two vectors.
Continuous Dynamical Systems

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

$$\dot{x} = f(x),$$  \hspace{1cm} (1)

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

If $f$ in (1) satisfies **local Lipschitz condition**, then given $x_0 \in \mathbb{R}^n$, there exists a unique solution $x(x_0; t) : (a, b) \to \mathbb{R}^n$ such that $x(x_0; 0) = x_0$ and $\forall t \in (a, b). \frac{dx(x_0; t)}{dt} = f(x(x_0; t))$.

If $f$ in (1) satisfies **global Lipschitz condition**, then the **existence, uniqueness and completeness** of solutions to (1) can be guaranteed.

The $k$-th **Lie derivatives** $L^k_f \sigma : \mathbb{R}^n \to \mathbb{R}$ of $\sigma$ along $f$ is defined by:

- $L^0_f \sigma(x) = \sigma(x)$,
- $L^k_f \sigma(x) = (\nabla L^{k-1}_f \sigma(x), f(x))$, for $k > 0$,

where $\nabla \varphi(x) \equiv \left( \frac{\partial \varphi(x)}{\partial x_1}, \frac{\partial \varphi(x)}{\partial x_2}, \ldots, \frac{\partial \varphi(x)}{\partial x_n} \right)$ and $(\cdot, \cdot)$ is the *inner product* of two vectors.
Continuous Dynamical Systems

- A **continuous dynamical systems (CDS)** is of the form
  \[ \dot{x} = f(x), \]  
  where \( x \in \mathbb{R}^n \) and \( f : \mathbb{R}^n \to \mathbb{R}^n \) is a **vector field**.

- If \( f \) in (1) satisfies **local Lipschitz condition**, then given \( x_0 \in \mathbb{R}^n \), there exists a unique solution \( x(x_0; t) : (a, b) \to \mathbb{R}^n \) such that \( x(x_0; 0) = x_0 \) and \( \forall t \in (a, b). \frac{dx(x_0; t)}{dt} = f(x(x_0; t)) \).

- If \( f \) in (1) satisfies **global Lipschitz condition**, then the existence, uniqueness and completeness of solutions to (1) can be guaranteed.

- The **\( k \)-th Lie derivatives** \( L_f^k \sigma : \mathbb{R}^n \to \mathbb{R} \) of \( \sigma \) along \( f \) is defined by:
  - \( L_f^0 \sigma(x) = \sigma(x) \),
  - \( L_f^k \sigma(x) = \left( \nabla L_f^{k-1} \sigma(x), f(x) \right) \), for \( k > 0 \),

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

A hybrid automaton (HA) is a system $\mathcal{H} \triangleq (Q, X, f, D, E, G, R, \Xi)$, 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 $x = (x_1, \ldots, x_n)$ 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 $f_q$;
- $D : Q \to 2^{\mathbb{R}^n}$ 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 $R_e : \mathbb{R}^n \to \mathbb{R}^n$;
- $\Xi$ 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 $\tau = \{I_i\}_{i=0}^N \ (N \text{ can be } \infty)$ 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 $I_N = [\tau_N, \tau'_N)$ is a right-closed or right-open nonempty interval ($\tau'_N$ may be $\infty$);
- $\tau_0 = 0$. 

![Diagram showing hybrid automata with intervals and transitions]
Hybrid Trajectories Accepted by HA [Tomlin et al. 00]

Definition (Hybrid Time Set)

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

- \( l_i = [\tau_i, \tau'_i] \) with \( \tau_i \leq \tau'_i = \tau_{i+1} \) for all \( i < N \);
- if \( N < \infty \), then \( l_N = [\tau_N, \tau'_N] \) is a right-closed or right-open nonempty interval (\( \tau'_N \) may be \( \infty \));
- \( \tau_0 = 0 \).
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 \( \alpha = \{\alpha_i : I_i \rightarrow Q\} \) and \( \beta = \{\beta_i : I_i \rightarrow \mathbb{R}^n\} \) are two sequences of functions satisfying

1. **Initial condition:** \( \alpha_0[0] = q_0 \) and \( \beta_0[0] = x_0 \);
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 \( \dot{x} = f_q(x) \) 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 \( \beta_0[0] = x_0 \);
2. **Discrete transition:** for all \( i < \langle \tau \rangle \),
   \[ e = (\alpha_i(\tau'_i), \alpha_{i+1}(\tau_{i+1})) \in E, \beta_i(\tau'_i) \in G_e \text{ and } \beta_{i+1}(\tau_{i+1}) = R_e(\beta_i(\tau'_i)) \] ;
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 \( \dot{x} = f_q(x) \) 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 $\alpha = \{\alpha_i : I_i \rightarrow Q\}$ and $\beta = \{\beta_i : I_i \rightarrow \mathbb{R}^n\}$ are two sequences of functions satisfying

1. **Initial condition:** $\alpha_0[0] = q_0$ and $\beta_0[0] = x_0$;
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 $\dot{x} = f_q(x)$ over $I_i$ with initial value $\beta_i(\tau_i)$, and
     - (3) for all $t \in [\tau_i, \tau'_i)$, $\beta_i(t) \in D_q$. 

---

![Diagram](image-url)
A hybrid trajectory \((\tau, \alpha, \beta)\) is called \textit{infinite} if

\[ \langle \tau \rangle = N \quad \text{is} \quad \infty, \text{ or} \]
\[ \|\tau\| = \sum_{i=0}^{N} (\tau'_i - \tau_i) \quad \text{is} \quad \infty. \]

A hybrid automaton is called \textit{non-blocking} if there is an infinite trajectory starting from any initial state \((q_0, x_0)\), and \textit{blocking} otherwise.
A hybrid trajectory \((\tau, \alpha, \beta)\) is called *infinite* if

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

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.
A hybrid trajectory \((\tau, \alpha, \beta)\) is called \textit{infinite} if

\[
\langle \tau \rangle = N \text{ is } \infty, \text{ or }
\|	au\| = \sum_{i=0}^{N} (\tau'_i - \tau_i)
\text{ is } \infty.
\]

A hybrid automaton is called \textit{non-blocking} if there is an \textit{infinite} trajectory starting from any initial state \((q_0, x_0)\), and \textit{blocking} otherwise.
Reachable Set of HA

Definition (Reachable Set)

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

$$(q_0, x_0), (q_1, x_1), \ldots, (q_l, x_l)$$

such that $(q_0, x_0) \in \Xi_H$, $(q_l, x_l) = (q, x)$, and for any $0 \leq i \leq l - 1$, one of the following two conditions holds:

- **(Discrete Jump):** $e = (q_i, q_{i+1}) \in E$, $x_i \in G_e$ and $x_{i+1} = R_e(x_i)$;
- or

- **(Continuous Evolution):** $q_i = q_{i+1}$, and there exists a $\delta \geq 0$ s.t. the solution $x(x_i; t)$ to $\dot{x} = f_{q_i}$ satisfies
  - $x(x_i; t) \in D_{q_i}$ for all $t \in [0, \delta]$; and
  - $x(x_i; \delta) = x_{i+1}$.
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
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 $(q, x) \in \mathcal{R}_\mathcal{H}$ (the reachable set), we have $x \in I_q$.

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 $x \in I_q \cap G_e$, then $x' = R_e(x) \in I_{q'}$;
3. for any $q \in Q$ and any $x_0 \in I_q$, if there exists a $\delta \geq 0$ s.t. the solution $x(x_0; t)$ to $\dot{x} = f_q$ satisfies: (i) $x(x_0; \delta) = x'$; and (ii) $x(x_0; t) \in D_q$ for all $t \in [0, \delta]$, then $x' \in I_q$. 
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 $(q, x) \in \mathcal{R}_\mathcal{H}$ (the reachable set), we have $x \in I_q$.

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 $x \in I_q \cap G_e$, then $x' = R_e(x) \in I_{q'}$;
3. for any $q \in Q$ and any $x_0 \in I_q$, if there exists a $\delta \geq 0$ s.t. the solution $x(x_0; t)$ to $\dot{x} = f_q$ satisfies: (i) $x(x_0; \delta) = x'$; and (ii) $x(x_0; t) \in D_q$ for all $t \in [0, \delta]$, then $x' \in I_q$. 
Continuous Invariant

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

Given \((D_q, f_q)\), we call \(P \subseteq \mathbb{R}^n\) a **continuous invariant** of \((D_q, f_q)\) if for all \(x_0 \in P\) and all \(T \geq 0\),

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

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 \((D_q, f_q)\), we call \(P \subseteq \mathbb{R}^n\) a **continuous invariant** of \((D_q, f_q)\) if for all \(x_0 \in P\) and all \(T \geq 0\),

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

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&Jadbabaie 2004, Plazer&Clarke 2008]
  - Basic idea: Let $D = \{ \dot{x} = f(x) \}$ and $H = \{ h(x) \geq 0 \}$. A function $B : \mathbb{R}^n \rightarrow \mathbb{R}$ is a barrier certificate if it is differentiable and satisfying
    \[ \forall x \in H. \quad \frac{\partial B}{\partial x} f(x) \leq 0. \]
    or
    \[ \forall x \in H(B(x) = 0 \Rightarrow \frac{\partial B}{\partial x} f(x) < 0). \]
  - Let $P := \{ x \mid B(x) \leq 0 \}$. Then $P$ is an invariant of $(D, H)$. 
Related Work (Cont’d)

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

\[
\forall y : (p(y) = 0 \land \|y - x\| < \delta \Rightarrow L_f p(y) \geq 0 \land \frac{\partial p}{\partial y} \neq 0),
\]

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 $I \subseteq \mathcal{R}[x]$ has the property:
  1. $(\forall p \in I, x \in H) p(x) = 0$,
  2. $(\forall p \in I), L_f p \in I$;

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

  \[
  \forall y : (p(y) = 0 \land \|y - x\| < \delta \implies L_f p(y) \geq 0 \land \frac{\partial p}{\partial y} \neq 0),
  \]

  then \( P \) is an invariant of \( (D, H) \).

  It imposes a strong assumption on the boundary.

- **Ideal fixed point method** [Sankaranarayanan, HSCC 2010]

  Basic idea: If an ideal \( I \subseteq \mathbb{R}[x] \) has the property:

  1. \((\forall p \in I, x \in H) p(x) = 0,\)
  2. \((\forall p \in I, L_f p \in I,\)

  then \( P := \{ x \mid p(x) = 0, \forall p \in I \} \) is an invariant of \( (D, H) \).

  It cannot cope with invariants as general semi-algebraic sets.
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

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**
Basic Idea

Let \((D, f)\) be a PDS, \(x(t)\) is a trajectory of \((D, f)\) from \(x_0\), and \(P \triangleq p(x) \geq 0\). Then \(P\) be a differential invariant of \((D, f)\) iff
\[
\forall x_0 \in \partial P \cap D, \exists \epsilon > 0, \forall t \in [0, \epsilon]. p(x(t)) \geq 0 \quad (2)
\]

1. \(p(x(t))\)'s Taylor's expansion at \(t = 0\)
\[
p(x(t)) = L_1^f p(x_0).t + L_2^f p(x_0).\frac{t^2}{2!} + \cdots L_i^f p(x_0).\frac{t^i}{i!} + \cdots
\]

2. (2) holds iff
   1. either for all \(i \geq 0, L_i^f p(x_0) = 0\)
   2. or there is some \(k > i \geq 0\), such that \(L_i^f p(x_0) = 0\) and \(L_k^f p(x_0) > 0\).

The pointwise rank of \(p\) with respect to \(f\) as the function
\[
\gamma_{p,f} : \mathbb{R}^n \to \mathbb{N} \cup \{\infty\} \text{ defined by}
\]
\[
\gamma_{p,f}(x) = \min\{k \in \mathbb{N} \mid L_k^f p(x) \neq 0\}
\]
if such \(k\) exists, and \(\gamma_{p,f}(x) = \infty\) otherwise.
Basic Idea

- Let \((D, f)\) be a PDS, \(x(t)\) is a trajectory of \((D, f)\) from \(x_0\), and \(P \equiv p(x) \geq 0\). Then \(P\) be a differential invariant of \((D, f)\) iff
  \[
  \forall x_0 \in \partial P \cap D, \exists \epsilon > 0, \forall t \in [0, \epsilon], p(x(t)) \geq 0 \tag{2}
  \]
- \(p(x(t))\)'s Taylor’s expansion at \(t = 0\)
  \[
p(x(t)) = L^1_f p(x_0) \cdot t + L^2_f p(x_0) \cdot \frac{t^2}{2!} + \cdots L^i_f p(x_0) \cdot \frac{t^i}{i!} + \cdots
  \]
- (2) holds iff
  1. either for all \(i \geq 0\), \(L^i_f p(x_0) = 0\)
  2. or there is some \(k > i \geq 0\), such that \(L^i_f p(x_0) = 0\) and \(L^k_f p(x_0) > 0\).
- The pointwise rank of \(p\) with respect to \(f\) as the function \(\gamma_{p,f} : \mathbb{R}^n \rightarrow \mathbb{N} \cup \{\infty\}\) defined by
  \[
  \gamma_{p,f}(x) = \min\{k \in \mathbb{N} \mid L^k_f p(x) \neq 0\}
  \]
  if such \(k\) exists, and \(\gamma_{p,f}(x) = \infty\) otherwise.
Basic Idea

- Let \((D, f)\) be a PDS, \(x(t)\) is a trajectory of \((D, f)\) from \(x_0\), and \(P = p(x) \geq 0\). Then \(P\) be a differential invariant of \((D, f)\) iff
  \[
  \forall x_0 \in \partial P \cap D, \exists \epsilon > 0, \forall t \in [0, \epsilon]. p(x(t)) \geq 0 \tag{2}
  \]
- \(p(x(t))\)'s Taylor’s expansion at \(t = 0\)
  \[
p(x(t)) = L^1_f p(x_0).t + L^2_f p(x_0).\frac{t^2}{2!} + \cdots L^i_f p(x_0).\frac{t^i}{i!} + \cdots
  \]
- (2) holds iff
  1. either for all \(i \geq 0, L^i_f p(x_0) = 0\)
  2. or there is some \(k > i \geq 0\), such that \(L^i_f p(x_0) = 0\) and \(L^k_f p(x_0) > 0\).
- The pointwise rank of \(p\) with respect to \(f\) as the function \(\gamma_{p,f} : \mathbb{R}^n \to \mathbb{N} \cup \{\infty\}\) defined by
  \[
  \gamma_{p,f}(x) = \min\{k \in \mathbb{N} | L^k_f p(x) \neq 0\}
  \]
  if such \(k\) exists, and \(\gamma_{p,f}(x) = \infty\) otherwise.
Let $f = (-x, y)$ and $p(x, y) = x + y^2$. Then

\[
L^0_f p(x, y) = x + y^2 \\
L^1_f p(x, y) = -x + 2y^2 \\
L^2_f p(x, y) = x + 4y^2 \\
\vdots
\]

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 $f$ at the point.
- $A$ is the gradient $\nabla p|_{(-1,1)}$ of $p(x, y)$.
- $L^1_f p|_{(-1,1)} = 3$ predicts that the trajectory starting at $(-1, 1)$ will enter the white area.
Example

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

\[
\begin{align*}
L^0_f p(x, y) &= x + y^2 \\
L^1_f p(x, y) &= -x + 2y^2 \\
L^2_f p(x, y) &= x + 4y^2 \\
&\vdots
\end{align*}
\]

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 \( f \) at the point.
- \( A \) is the gradient \( \nabla p|_{(-1,1)} \) of \( p(x, y) \).
- \( L^1_f p|_{(-1,1)} = 3 \) predicts that the trajectory starting at \((-1, 1)\) will enter the white area.
Example

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

\[
\begin{align*}
L_f^0 p(x, y) &= x + y^2 \\
L_f^1 p(x, y) &= -x + 2y^2 \\
L_f^2 p(x, y) &= x + 4y^2 \\
&
\end{align*}
\]

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 \( f \) at the point.
- \( A \) is the gradient \( \nabla p|_{(-1,1)} \) of \( p(x, y) \).
- \( L_f^1 p|_{(-1,1)} = 3 \) predicts that the trajectory starting at \((-1, 1)\) will enter the white area.
Example

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

\[
\begin{align*}
L^0_f h(x, y) &= x + y^2 \\
L^1_f h(x, y) &= -2y + 2x^2y \\
L^2_f h(x, y) &= -8y^2x - (2 - 2x^2)x^2
\end{align*}
\]

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., $L^1_f h(-1, 1) = 0$, 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, $L^2_f h(-1, 1) = 8$. Hence $\gamma_{h,f}^{-1,1} = 2$. 

II: Demand for Higher Order Lie Derivative
Example

Let \( f(x, y) = (-2y, x^2) \) and \( h(x, y) = x + y^2 \). Then
\[
L_0^0 f h(x, y) = x + y^2 \\
L_0^1 f h(x, y) = -2y + 2x^2y \\
L_0^2 f h(x, y) = -8y^2x - (2 - 2x^2)x^2 \\
\]

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., \( L_0^1 f h(-1, 1) = 0 \), 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, \( L_0^2 f h(-1, 1) = 8 \). Hence \( \gamma_{h,f}(-1, 1) = 2 \).
### Theoretical Results

**Theorem (Rank Theorem)**

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

**Theorem (Parametric Rank Theorem)**

Given a parametric polynomial \( p(u, x) \) and a PVF \( f \), there is an integer \( N_{p,f} \in \mathbb{N} \) such that \( \gamma_{pu_0,f}(x) < \infty \) implies \( \gamma_{pu_0,f}(x) \leq N_{p,f} \) for all \( x \in \mathbb{R}^n \) and all \( u_0 \in \mathbb{R}^w \).

**Theorem (Criterion Theorem)**

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

\[
\theta(h, p, f, x) \equiv (p(x) = 0 \land \pi(p, f, x)) \rightarrow \pi(h, f, x),
\]

holds for all \( x \in \mathbb{R}^n \), where

\[
\pi^{(i)}(p, f, x) \equiv \left( \bigwedge_{0 \leq j < i} L_f^i p(x) = 0 \right) \land L_f^i p(x) < 0,
\]

\[
\pi(p, f, x) \equiv \bigvee_{0 \leq i \leq N_{p,f}} \pi^{(i)}(p, f, x).
\]
Theoretical Results

Theorem (Rank Theorem)

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

Theorem (Parametric Rank Theorem)

Given a parametric polynomial $p(u, x)$ and a PVF $f$, there is an integer $N_{p,f} \in \mathbb{N}$ such that $\gamma_{pu_0,f}(x) < \infty$ implies $\gamma_{pu_0,f}(x) \leq N_{p,f}$ for all $x \in \mathbb{R}^n$ and all $u_0 \in \mathbb{R}^w$.

Theorem (Criterion Theorem)

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

$$\theta(h, p, f, x) \equiv (p(x) = 0 \land \pi(p, f, x)) \rightarrow \pi(h, f, x),$$

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

$$\pi^{(i)}(p, f, x) \equiv \left( \bigwedge_{0 \leq j < i} L^j_f p(x) = 0 \right) \land L^i_f p(x) < 0,$$

$$\pi(p, f, x) \equiv \bigvee_{0 \leq i \leq N_{p,f}} \pi^{(i)}(p, f, x).$$
Theoretical Results

**Theorem (Rank Theorem)**

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

**Theorem (Parametric Rank Theorem)**

Given a parametric polynomial $p(u, x)$ and a PVF $f$, there is an integer $N_{p,f} \in \mathbb{N}$ such that $\gamma_{pu_0,f}(x) < \infty$ implies $\gamma_{pu_0,f}(x) \leq N_{p,f}$ for all $x \in \mathbb{R}^n$ and all $u_0 \in \mathbb{R}^w$.

**Theorem (Criterion Theorem)**

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

$$\theta(h, p, f, x) \equiv (p(x) = 0 \land \pi(p, f, x)) \rightarrow \pi(h, f, x),$$

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

$$\pi^{(i)}(p, f, x) \equiv \left(\bigwedge_{0 \leq j < i} L_f^j p(x) = 0\right) \land L_f^i p(x) < 0,$$

$$\pi(p, f, x) \equiv \bigvee_{0 \leq i \leq N_{p,f}} \pi^{(i)}(p, f, x).$$
**Algorithm**

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

II. Then apply QE to the formula $\forall x. \theta(h, p, f, x)$. In practice, QE may be applied to a formula $\forall x. (\theta \land \phi)$, where $\phi$ 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 $u$, denoted by $R(u)$, will be returned.

III. Now, use an SMT solver like Z3 to pick a $u_0 \in R(u)$ and then $p_{u_0}(x) \geq 0$ is an SCI of $(h(x) \geq 0, f)$. 
Algorithm

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

II. Then apply QE to the formula $\forall x. \theta(h, p, f, x)$. In practice, QE may be applied to a formula $\forall x.(\theta \land \phi)$, where $\phi$ 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 $u$, denoted by $R(u)$, will be returned.

III. Now, use an SMT solver like Z3 to pick a $u_0 \in R(u)$ and then $p_{u_0}(x) \geq 0$ is an SCI of $(h(x) \geq 0, f)$. 
Algorithm

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

II. Then apply QE to the formula $\forall x. \theta(h, p, f, x)$. In practice, QE may be applied to a formula $\forall x. (\theta \land \phi)$, where $\phi$ 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 $u$, denoted by $R(u)$, will be returned.

III. Now, use an SMT solver like Z3 to pick a $u_0 \in R(u)$ and then $p_{u_0}(x) \geq 0$ is an SCI of $(h(x) \geq 0, f)$. 
Running Example

Consider a PDS \((D = -x - y^2 \geq 0, f(x, y) = (-2y, x^2))\).

Apply procedure (I-III), we have:

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

II. Compute the corresponding formula

\[
\theta(h, p, f, x) \triangleq p = 0 \land (\pi_{p,f,x}^{(0)} \lor \pi_{p,f,x}^{(1)} \lor \pi_{p,f,x}^{(2)}) \rightarrow \\
(\pi_{h,f,x}^{(0)} \lor \pi_{h,f,x}^{(1)} \lor \pi_{h,f,x}^{(2)})
\]

where

\[
\begin{align*}
\pi_{h,f,x}^{(0)} & \triangleq -x - y^2 < 0, \\
\pi_{h,f,x}^{(1)} & \triangleq -x - y^2 = 0 \land 2y - 2x^2y < 0, \\
\pi_{h,f,x}^{(2)} & \triangleq -x - y^2 = 0 \land 2y - 2x^2y = 0 \land 8xy^2 + 2x^2 - 2x^4 < 0, \\
\pi_{p,f,x}^{(0)} & \triangleq ay(x - y) < 0, \\
\pi_{p,f,x}^{(1)} & \triangleq ay(x - y) = 0 \land -2ay^2 + ax^3 - 2yax^2 < 0, \\
\pi_{p,f,x}^{(2)} & \triangleq ay(x - y) = 0 \land -2ay^2 + ax^3 - 2yax^2 = 0 \\
& \land 40axy^2 - 16ay^3 + 32ax^3y - 10ax^4 < 0.
\end{align*}
\]
**Running Example**

Consider a PDS \((D = -x - y^2 \geq 0, f(x, y) = (-2y, x^2))\).

**Apply procedure (I-III), we have:**

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

II. Compute the corresponding formula

\[
\theta(h, p, f, x) \triangleq p = 0 \land (\pi_{p,f,x}^{(0)} \lor \pi_{p,f,x}^{(1)} \lor \pi_{p,f,x}^{(2)}) \longrightarrow \\
(\pi_{h,f,x}^{(0)} \lor \pi_{h,f,x}^{(1)} \lor \pi_{h,f,x}^{(2)})
\]

where

\[
\begin{align*}
\pi_{h,f,x}^{(0)} & \triangleq -x - y^2 < 0, \\
\pi_{h,f,x}^{(1)} & \triangleq -x - y^2 = 0 \land 2y - 2x^2y < 0, \\
\pi_{h,f,x}^{(2)} & \triangleq -x - y^2 = 0 \land 2y - 2x^2y = 0 \land 8xy^2 + 2x^2 - 2x^4 < 0, \\
\pi_{p,f,x}^{(0)} & \triangleq ay(x - y) < 0, \\
\pi_{p,f,x}^{(1)} & \triangleq ay(x - y) = 0 \land -2ay^2 + ax^3 - 2yax^2 < 0, \\
\pi_{p,f,x}^{(2)} & \triangleq ay(x - y) = 0 \land -2ay^2 + ax^3 - 2yax^2 = 0 \land 40axy^2 - 16ay^3 + 32ax^3 y - 10ax^4 < 0.
\end{align*}
\]
Running Example

Consider a PDS \((D = -x - y^2 \geq 0, f(x, y) = (-2y, x^2))\).

Apply procedure (I-III), we have:

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

II Compute the corresponding formula

\[
\theta(h, p, f, x) \doteq p = 0 \land (\pi_{p,f,x}^{(0)} \lor \pi_{p,f,x}^{(1)} \lor \pi_{p,f,x}^{(2)}) \rightarrow \\
(\pi_{h,f,x}^{(0)} \lor \pi_{h,f,x}^{(1)} \lor \pi_{h,f,x}^{(2)})
\]

where

\[
\begin{align*}
\pi_{h,f,x}^{(0)} & \doteq -x - y^2 < 0, \\
\pi_{h,f,x}^{(1)} & \doteq -x - y^2 = 0 \land 2y - 2x^2y < 0, \\
\pi_{h,f,x}^{(2)} & \doteq -x - y^2 = 0 \land 2y - 2x^2y = 0 \land 8xy^2 + 2x^2 - 2x^4 < 0, \\
\pi_{p,f,x}^{(0)} & \doteq ay(x - y) < 0, \\
\pi_{p,f,x}^{(1)} & \doteq ay(x - y) = 0 \land -2ay^2 + ax^3 - 2yax^2 < 0, \\
\pi_{p,f,x}^{(2)} & \doteq ay(x - y) = 0 \land -2ay^2 + ax^3 - 2yax^2 = 0 \\
& \land 40axy^2 - 16ay^3 + 32ax^3y - 10ax^4 < 0.
\end{align*}
\]
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 \(\text{QE}\) to the formula

\[
\forall x \forall y. (\theta(h, p, f, x) \land 0.5a(-1 - 0.5) \geq 0 \land -0.6a(-0.5 + 0.6) \geq 0).
\]

The result is \(a \leq 0\).

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

\[\forall x \forall y. (\theta(h, p, f, x) \land 0.5a(-1 - 0.5) \geq 0 \land -0.6a(-0.5 + 0.6) \geq 0).\]

The result is \(a \leq 0\).

IV Just pick \(a = -1\), and then \(-xy + y^2 \geq 0\) is an SCI of \((D, f)\). The grey part of Picture III is the intersection of the invariant \(P\) and domain \(D\).
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
**Problem:** Consider a PDS \((D, f)\) with

\[
D = \bigvee_{i=1}^{l} \bigwedge_{j=1}^{J_i} p_{ij}(x) \triangleright 0,
\]

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

\[
P = \bigvee_{k=1}^{K} \bigwedge_{l=1}^{L_k} p_{kl}(u_{kl}, x) \triangleright 0, \quad \triangleright \in \{\geq, >\}
\]

**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 \((D, f)\) with

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

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

\[
P = \bigvee_{k=1}^{K} \bigwedge_{l=1}^{L_k} p_{kl}(u_{kl}, x) \triangleright 0, \quad \triangleright \in \{\geq, >\}
\]

**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 $P(u,x)$ defined by

$$K \bigvee_{k=1}^{K} \left( \bigwedge_{j=1}^{j_k} p_{kj}(u_{kj}, x) \geq 0 \land \bigwedge_{j=j_k+1}^{J_k} p_{kj}(u_{kj}, x) > 0 \right)$$

is a CI of the PCCDS $(D,f)$ with

$$D \triangleq M \bigvee_{m=1}^{M} \left( \bigwedge_{l=1}^{l_m} p_{ml}(x) \geq 0 \land \bigwedge_{l=l_m+1}^{L_m} p_{ml}(x) > 0 \right),$$

iff $u$ satisfies

$$\forall x. \left( (P \land D \land \Phi_D \rightarrow \Phi_P) \land (\neg P \land D \land \Phi_D^{lv} \rightarrow \neg \Phi_P^{lv}) \right),$$

where
Theorem (Main Result (Cont’d))

\[\Phi_D \equiv \bigvee_{m=1}^{M} \left( \bigwedge_{l=1}^{l_m} \psi_0^+(p_{ml}, f) \land \bigwedge_{l=l_m+1}^{L_m} \psi^+(p_{ml}, f) \right),\]

\[\Phi_P \equiv \bigvee_{k=1}^{K} \left( \bigwedge_{j=1}^{j_k} \psi_0^+(p_{kj}, f) \land \bigwedge_{j=j_k+1}^{J_k} \psi^+(p_{kj}, f) \right),\]

\[\Phi_{IV}^D \equiv \bigvee_{m=1}^{M} \left( \bigwedge_{l=1}^{l_m} \varphi_0^+(p_{ml}, f) \land \bigwedge_{l=l_m+1}^{L_m} \varphi^+(p_{ml}, f) \right),\]

\[\Phi_{IV}^P \equiv \bigvee_{k=1}^{K} \left( \bigwedge_{j=1}^{j_k} \varphi_0^+(p_{kj}, f) \land \bigwedge_{j=j_k+1}^{J_k} \varphi^+(p_{kj}, f) \right),\]

\[\psi^+(p, f) \equiv \bigvee_{0 \leq i \leq N_{p,f}} \psi^{(i)}(p, f) \text{ with } \psi^{(i)}(p, f) \equiv \left( \bigwedge_{0 \leq j < i} L^j_{fp} = 0 \right) \land L^i_{fp} > 0, \text{ and}\]

\[\psi_0^+(p, f) \equiv \psi^+(p, f) \lor \left( \bigwedge_{0 \leq j \leq N_{p,f}} L^j_{fp} = 0 \right)\]

\[\varphi^+(p, f) \equiv \bigvee_{0 \leq i \leq N_{p,f}} \varphi^{(i)}(p, f) \text{ with } \varphi^{(i)}(p, f) \equiv \left( \bigwedge_{0 \leq j < i} L^j_{fp} = 0 \right) \land (-1)^i \cdot L^i_{fp} > 0, \text{ and}\]

\[\varphi_0^+(p, f) \equiv \varphi^+(p, f) \lor \left( \bigwedge_{0 \leq j \leq N_{p,f}} L^j_{fp} = 0 \right).\]
Running Example

Let \( \mathbf{f}(x, y) = (-2y, x^2) \) and \( D \cong \mathbb{R}^2 \).

- Take a template: \( P(u, x) \cong x - a \geq 0 \lor y - b > 0 \) with \( u = (a, b) \).
- So, \( P \) is an SCI of \( (D, \mathbf{f}) \) iff \( a, b \) satisfy
  \[
  \forall x \forall y. (P \rightarrow \zeta) \land (\neg P \rightarrow \neg \xi),
  \]

where

\[
\zeta \cong (x - a > 0) \lor (x - a = 0 \land -2y > 0) \\
\lor (x - a = 0 \land -2y = 0 \land -2x^2 \geq 0) \\
\lor (y - b > 0) \lor (y - b = 0 \land x^2 > 0) \\
\lor (y - b = 0 \land x^2 = 0 \land -4yx > 0) \\
\lor (y - b = 0 \land x^2 = 0 \land -4yx = 0 \land 8y^2 - 4x^3 > 0)
\]

\[
\xi \cong (x - a > 0) \lor (x - a = 0 \land -2y < 0) \\
\lor (x - a = 0 \land -2y = 0 \land -2x^2 \geq 0) \\
\lor (y - b > 0) \lor (y - b = 0 \land x^2 < 0) \\
\lor (y - b = 0 \land x^2 = 0 \land -4yx > 0) \\
\lor (y - b = 0 \land x^2 = 0 \land -4yx = 0 \land 8y^2 - 4x^3 < 0)
\]
Running Example

- Let $f(x, y) = (-2y, x^2)$ and $D \cong \mathbb{R}^2$.
- Take a template: $P(u, x) \cong x - a \geq 0 \lor y - b > 0$ with $u = (a, b)$.
- So, $P$ is an SCI of $(D, f)$ iff $a, b$ satisfy
  \[ \forall x \forall y. (P \rightarrow \zeta) \land (\neg P \rightarrow \neg \xi), \]

  where
  \[
  \zeta \cong (x - a > 0) \lor (x - a = 0 \land -2y > 0) \\
  \lor (x - a = 0 \land -2y = 0 \land -2x^2 \geq 0) \\
  \lor (y - b > 0) \lor (y - b = 0 \land x^2 > 0) \\
  \lor (y - b = 0 \land x^2 = 0 \land -4yx > 0) \\
  \lor (y - b = 0 \land x^2 = 0 \land -4yx = 0 \land 8y^2 - 4x^3 > 0)
  \]

  \[
  \xi \cong (x - a > 0) \lor (x - a = 0 \land -2y < 0) \\
  \lor (x - a = 0 \land -2y = 0 \land -2x^2 \geq 0) \\
  \lor (y - b > 0) \lor (y - b = 0 \land x^2 < 0) \\
  \lor (y - b = 0 \land x^2 = 0 \land -4yx > 0) \\
  \lor (y - b = 0 \land x^2 = 0 \land -4yx = 0 \land 8y^2 - 4x^3 < 0)
  \]
Running Example

- Let $f(x, y) = (-2y, x^2)$ and $D \cong \mathbb{R}^2$.
- Take a template: $P(u, x) \cong x - a \geq 0 \lor y - b > 0$ with $u = (a, b)$.
- So, $P$ is an SCI of $(D, f)$ iff $a, b$ satisfy
  $$\forall x \forall y. (P \rightarrow \zeta) \land (\neg P \rightarrow \neg \xi),$$

where

$$\zeta \cong (x - a > 0) \lor (x - a = 0 \land -2y > 0)$$
$$\lor (x - a = 0 \land -2y = 0 \land -2x^2 \geq 0)$$
$$\lor (y - b > 0) \lor (y - b = 0 \land x^2 > 0)$$
$$\lor (y - b = 0 \land x^2 = 0 \land -4yx > 0)$$
$$\lor (y - b = 0 \land x^2 = 0 \land -4yx = 0 \land 8y^2 - 4x^3 > 0)$$

$$\xi \cong (x - a > 0) \lor (x - a = 0 \land -2y < 0)$$
$$\lor (x - a = 0 \land -2y = 0 \land -2x^2 \geq 0)$$
$$\lor (y - b > 0) \lor (y - b = 0 \land x^2 < 0)$$
$$\lor (y - b = 0 \land x^2 = 0 \land -4yx > 0)$$
$$\lor (y - b = 0 \land x^2 = 0 \land -4yx = 0 \land 8y^2 - 4x^3 < 0)$$
Running Example (Cont’d)

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

![Diagram](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 \( a + b \leq 0 \land b \leq 0 \).
- Let \( a = -1 \) and \( b = -0.5 \), and we obtain an SCI \( P \cong x + 1 \geq 0 \lor y + 0.5 > 0 \).
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
Algorithm

I. Predefine a family of semi-algebraic templates $l_q(u, x)$ 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 $l_q(u, x)$ to be a GI of $H$, i.e.

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

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

1. $\forall x. (\Xi_q \rightarrow l_q(u, x))$ for all $q \in Q$;
2. $\forall x, x'. (l_q(u, x) \land G_e \land x' = R_e(x) \rightarrow l_{q'}(u, x'))$ for all $q \in Q$ and all $e = (q, q') \in E$;
3. $\forall x. ((l_q(u, x) \land D_q \land \Phi_{D_q} \rightarrow \Phi_{l_q}) \land (\neg l_q(u, x) \land D_q \land \Phi_{D_q}}^{I_q} \rightarrow \neg \Phi_{l_q}))$, for each $q \in Q$.

For safety property $S$, there may be a fourth set of formulas:

4. $\forall x. (l_q(u, x) \rightarrow S_q)$ for all $q \in Q$.

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

I. Predefine a family of semi-algebraic templates $l_q(u, x)$ 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 $l_q(u, x)$ to be a GI of $H$, i.e.

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

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

1. $\forall x. (\Xi_q \rightarrow l_q(u, x))$ for all $q \in Q$;
2. $\forall x, x'. (l_q(u, x) \land G_e \land x' = R_e(x) \rightarrow l_{q'}(u, x'))$ for all $q \in Q$ and all $e = (q, q') \in E$;
3. $\forall x. ((l_q(u, x) \land D_q \land \Phi_{D_q} \rightarrow \Phi_{l_q}) \land (\neg l_q(u, x) \land D_q \land \Phi_{D_q}^{\text{IV}} \rightarrow \neg \Phi_{l_q}^{\text{IV}}))$, for each $q \in Q$.

For safety property $S$, there may be a fourth set of formulas:

4. $\forall x. (l_q(u, x) \rightarrow S_q)$ for all $q \in Q$.

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

I. Predefine a family of semi-algebraic templates \( l_q(u, x) \) 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 \( l_q(u, x) \) to be a GI of \( \mathcal{H} \), i.e.

- \( \exists_q \subseteq l_q \) for all \( q \in Q \);
- for any \( e = (q, q') \in E \), if \( x \in l_q \cap G_e \), then \( x' = R_e(x) \in l_{q'} \);
- for any \( q \in Q \), \( l_q \) is a CI of \((D_q, f_q)\)

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

\[
(1) \quad \forall x. (\exists_q \rightarrow l_q(u, x)) \text{ for all } q \in Q;
(2) \quad \forall x, x'. (l_q(u, x) \land G_e \land x' = R_e(x) \rightarrow l_{q'}(u, x')) \text{ for all } q \in Q \text{ and all } e = (q, q') \in E;
(3) \quad \forall x. ((l_q(u, x) \land D_q \land \Phi_{D_q} \rightarrow \Phi_{l_q}) \land (\neg l_q(u, x) \land D_q \land \Phi_{l_q}^{iv} \rightarrow \neg \Phi_{l_q}^{iv})), \text{ for each } q \in Q.
\]

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

\[
(4) \quad \forall x. (l_q(u, x) \rightarrow S_q) \text{ for all } q \in Q.
\]

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

I. Predefine a family of semi-algebraic templates \( I_q(u, x) \) 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 \( I_q(u, x) \) 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 \( x \in I_q \cap G_e \), then \( x' = R_e(x) \in I_{q'} \);
- for any \( q \in Q \), \( I_q \) is a CI of \((D_q, f_q)\)

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

\[
\begin{align*}
(1) & \quad \forall x. (\Xi_q \rightarrow I_q(u, x)) \text{ for all } q \in Q; \\
(2) & \quad \forall x, x'. (I_q(u, x) \land G_e \land x' = R_e(x) \rightarrow I_{q'}(u, x')) \text{ for all } q \in Q \text{ and all } e = (q, q') \in E; \\
(3) & \quad \forall x. ((I_q(u, x) \land D_q \land \Phi_{D_q} \rightarrow \Phi_{I_q}) \land (\neg I_q(u, x) \land D_q \land \Phi_{I_q} \rightarrow \neg \Phi_{I_q})) ,
\end{align*}
\]

for each \( q \in Q \).

For safety property \( S \), there may be a fourth set of formulas:

\[
(4) \quad \forall x. (I_q(u, x) \rightarrow S_q) \text{ for all } q \in Q.
\]

III. Take the conjunction of all the formulas in Step 2 and apply QE to get a QFF \( \phi(u) \). Then choose a specific \( u_0 \) from \( \phi(u) \) with a tool like \( Z3 \), and the set of instantiations \( I_{q,u_0}(x) \) 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 $\Xi_H \triangleq \{ q_{ht} \} \times X_0$ with $X_0 \triangleq c = 0 \land 5 \leq T \leq 10$, $S \triangleq T \geq 4.5$ 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 $\Xi_H \equiv \{q_{ht}\} \times X_0$ with $X_0 \equiv c = 0 \land 5 \leq T \leq 10$, $S \equiv T \geq 4.5$ is satisfied at all modes.
Running Example (Cont’d)

- Firstly, predefine the following set of templates:
  - \( I_{q_{\text{ht}}} \triangleq T + a_1 c + a_0 \geq 0 \land c \geq 0; \)
  - \( I_{q_{\text{cl}}} \triangleq T + a_2 \geq 0; \)
  - \( I_{q_{\text{ck}}} \triangleq T \geq a_3 c^2 - 4.5c + 9 \land c \geq 0 \land c \leq 1 \)

- By the second step, we get
  
  \[
  10a_3 - 9 \leq 0 \land 2a_3 - 1 \geq 0 \land a_1 + 2 = 0 \land a_0 + 2a_1 + 9 = 0 \land a_2 - a_0 = 0 .
  \]

- By choosing \( a_0 = -5, a_1 = -2, a_2 = -5, a_3 = \frac{1}{2} \), obtain the following SGI
  - \( I_{q_{\text{ht}}} \triangleq T \geq 2c + 5 \land c \geq 0; \)
  - \( I_{q_{\text{cl}}} \triangleq T \geq 5; \)
  - \( I_{q_{\text{ck}}} \triangleq 2T \geq c^2 - 9c + 18 \land c \geq 0 \land c \leq 1. \)

- The safety property is successfully verified by the SGI.
Firstly, predefine the following set of templates:

- \( I_{q_{ht}} \triangleq T + a_1 c + a_0 \geq 0 \land c \geq 0; \)
- \( I_{q_{cl}} \triangleq T + a_2 \geq 0; \)
- \( I_{q_{ck}} \triangleq T \geq a_3 c^2 - 4.5c + 9 \land c \geq 0 \land c \leq 1 \)

By the second step, we get

\[
10a_3 - 9 \leq 0 \land 2a_3 - 1 \geq 0 \land a_1 + 2 = 0 \land a_0 + 2a_1 + 9 = 0 \land a_2 - a_0 = 0.
\]

By choosing \( a_0 = -5, a_1 = -2, a_2 = -5, a_3 = \frac{1}{2} \), obtain the following SGI

- \( I_{q_{ht}} \triangleq T \geq 2c + 5 \land c \geq 0; \)
- \( I_{q_{cl}} \triangleq T \geq 5; \)
- \( I_{q_{ck}} \triangleq 2T \geq c^2 - 9c + 18 \land c \geq 0 \land c \leq 1. \)

The safety property is successfully verified by the SGI.
Running Example (Cont’d)

- Firstly, predefine the following set of templates:
  - \( I_{q_{ht}} \triangleq T + a_1 c + a_0 \geq 0 \land c \geq 0 \);
  - \( I_{q_{cl}} \triangleq T + a_2 \geq 0 \);
  - \( I_{q_{ck}} \triangleq T \geq a_3 c^2 - 4.5c + 9 \land c \geq 0 \land c \leq 1 \)

- By the second step, we get
  \[
  10a_3 - 9 \leq 0 \land 2a_3 - 1 \geq 0 \land a_1 + 2 = 0 \land a_0 + 2a_1 + 9 = 0 \land a_2 - a_0 = 0 .
  \]

- By choosing \( a_0 = -5, a_1 = -2, a_2 = -5, a_3 = \frac{1}{2} \), obtain the following SGI
  - \( I_{q_{ht}} \triangleq T \geq 2c + 5 \land c \geq 0 \);
  - \( I_{q_{cl}} \triangleq T \geq 5 \);
  - \( I_{q_{ck}} \triangleq 2T \geq c^2 - 9c + 18 \land c \geq 0 \land c \leq 1 \).

- The safety property is successfully verified by the SGI.
Running Example (Cont’d)

Firstly, predefine the following set of templates:

- \( I_{q_{ht}} \triangleq T + a_1 c + a_0 \geq 0 \land c \geq 0; \)
- \( I_{q_{cl}} \triangleq T + a_2 \geq 0; \)
- \( I_{q_{ck}} \triangleq T \geq a_3 c^2 - 4.5c + 9 \land c \geq 0 \land c \leq 1 \)

By the second step, we get

\[
10a_3 - 9 \leq 0 \land 2a_3 - 1 \geq 0 \land a_1 + 2 = 0 \land a_0 + 2a_1 + 9 = 0 \land a_2 - a_0 = 0.
\]

By choosing \( a_0 = -5, a_1 = -2, a_2 = -5, a_3 = \frac{1}{2} \), obtain the following SGI

- \( I_{q_{ht}} \triangleq T \geq 2c + 5 \land c \geq 0; \)
- \( I_{q_{cl}} \triangleq T \geq 5; \)
- \( I_{q_{ck}} \triangleq 2T \geq c^2 - 9c + 18 \land c \geq 0 \land c \leq 1. \)

The safety property is successfully verified by the SGI.
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
Elementary Functions

\[ f, g \ ::= \ c \mid x \mid f + g \mid f - g \mid f \times g \mid \frac{f}{g} \mid f^a \mid e^f \mid \ln(f) \mid \sin(f) \mid \cos(f) \]

- \( c \in \mathbb{R}, \ a \in \mathbb{Q}, \ x \in \{x, y, z, \ldots, x_1, x_2, \ldots, x_n\}\)

- elementary (or polynomial) hybrid system (or CDS), EHS/PHS/EDS/PDS:
Elementary Functions

\[ f, g \ ::= \ c \mid x \mid f + g \mid f - g \mid f \times g \mid \frac{f}{g} \mid f^a \mid e^f \mid \ln(f) \mid \sin(f) \mid \cos(f) \]

- \( c \in \mathbb{R}, \ a \in \mathbb{Q}, \ x \in \{x, y, z, \ldots, x_1, x_2, \ldots, x_n\} \)

- elementary (or polynomial) hybrid system (or CDS),
  \( \text{EHS/PHS/EDS/PDS}: \)
Univariate Basic Elementary Functions: \( \dot{x} = f(x) \)

- \( f(x) = \frac{1}{x} \): let \( v = \frac{1}{x} \), and thus \( \dot{v} = -\frac{x}{x^2} \), 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 \( \dot{v} = \frac{x}{2\sqrt{x}} \), 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

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= uv \\
\dot{u} &= -u^2v
\end{align*}
\]

- \( f(x) = \sin x \): 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

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= uv \\
\dot{u} &= -v^2
\end{align*}
\]

- \( f(x) = \cos x \): 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

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= uv \\
\dot{u} &= -u^2v
\end{align*}
\]

- \( f(x) = \sin x \): 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

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= uv \\
\dot{u} &= -v^2
\end{align*}
\]

- \( f(x) = \cos x \): 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

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= uv \\
\dot{u} &= -u^2 v
\end{align*}
\]

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

\[
\begin{align*}
\dot{x} &= v \\
\dot{v} &= uv \\
\dot{u} &= -v^2
\end{align*}
\]

- $f(x) = \cos x$: 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
  
  \[
  \begin{align*}
  v &= \sin x \\
  u &= \cos x \\
  w &= \ln (2 + v) = \ln (2 + \sin x) \\
  z &= \frac{1}{2+v} = \frac{1}{2+\sin x}
  \end{align*}
  \]

  so (1) is transformed to
  
  \[
  \begin{align*}
  \dot{x} &= w \\
  \dot{v} &= uw \\
  \dot{u} &= -vw \\
  \dot{w} &= zuw \\
  \dot{z} &= -z^2uw
  \end{align*}
  \]

- **Multivariate**: analogous.
Abstracting EDSs

Abstracting EDS $C_x \equiv (\Xi_x, f_x, D_x)$ to PDS $C_y \equiv (\Xi_y, f_y, D_y)$

(S1) Introduce new variables to replace all non-polynomial terms in $f_x$, $\Xi_x$ and $D_x$, and obtain a collection of replacement equations $v = \Gamma(x)$.

(S2) Differentiate both sides of $v = \Gamma(x)$ 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 $v = \Gamma(x)$.

(S4) Define the simulation map as $\Theta(x) = (x, \Gamma(x))$. Then use $v = \Gamma(x)$ to construct $\Xi_y$ and $D_y$ as illustrated by the following example.
Abstracting EDSs

Abstracting EDS $C_x \equiv (\Xi_x, f_x, D_x)$ to PDS $C_y \equiv (\Xi_y, f_y, D_y)$

(S1) Introduce new variables to replace all non-polynomial terms in $f_x$, $\Xi_x$ and $D_x$, and obtain a collection of replacement equations $v = \Gamma(x)$.

(S2) Differentiate both sides of $v = \Gamma(x)$ 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 $v = \Gamma(x)$.

(S4) Define the simulation map as $\Theta(x) = (x, \Gamma(x))$. Then use $v = \Gamma(x)$ to construct $\Xi_y$ and $D_y$ as illustrated by the following example.
Abstracting EDSs

Abstracting EDS $C_x \equiv (\Xi_x, f_x, D_x)$ to PDS $C_y \equiv (\Xi_y, f_y, D_y)$

(S1) Introduce new variables to replace all non-polynomial terms in $f_x$, $\Xi_x$ and $D_x$, and obtain a collection of replacement equations $v = \Gamma(x)$.

(S2) Differentiate both sides of $v = \Gamma(x)$ 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 $v = \Gamma(x)$.

(S4) Define the simulation map as $\Theta(x) = (x, \Gamma(x))$. Then use $v = \Gamma(x)$ to construct $\Xi_y$ and $D_y$ as illustrated by the following example.
Abstracting EDSs

Abstracting EDS $C_x \equiv (\Xi_x, f_x, D_x)$ to PDS $C_y \equiv (\Xi_y, f_y, D_y)$

(S1) Introduce new variables to replace all non-polynomial terms in $f_x, \Xi_x$ and $D_x$, and obtain a collection of replacement equations $v = \Gamma(x)$.

(S2) Differentiate both sides of $v = \Gamma(x)$ 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 $v = \Gamma(x)$.

(S4) Define the simulation map as $\Theta(x) = (x, \Gamma(x))$. Then use $v = \Gamma(x)$ to construct $\Xi_y$ and $D_y$ as illustrated by the following example.
Abstracting EDSs: An Example

Consider the EDS $C_x \triangleq (\Xi_x, f_x, D_x)$, where

- $\Xi_x \triangleq (x + 0.5)^2 + (y - 0.5)^2 - 0.16 \leq 0$;
- $D_x \triangleq -2 \leq x \leq 2 \land -2 \leq y \leq 2$; and
- $f_x$ 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 \( v = \Gamma(x) \)

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

we get the transformed polynomial ODE (i.e. \( f_y \))

\[
\begin{pmatrix}
\dot{x} \\
\dot{y} \\
\dot{v}_1 \\
\dot{v}_2 \\
\dot{v}_3
\end{pmatrix}
=
\begin{pmatrix}
v_2 + y - 1 \\
-v_1^2 \\
v_3(v_2 + y - 1) \\
-v_2(v_2 + y - 1) \\
-v_1(v_2 + y - 1)
\end{pmatrix},
\]
Abstracting EDSs: An Example

• (S4): the simulation map is $\Theta(x, y) = (x, y, \sin x, e^{-x}, \cos x)$
  • $\Theta(\Xi_x) \equiv \Xi_x \land v_1 = \sin x \land v_2 = e^{-x} \land v_3 = \cos x$
  • $\Theta(D_x) \equiv D_x \land v_1 = \sin x \land v_2 = e^{-x} \land v_3 = \cos x$
  • abstracting $v_1 = \sin x \land v_2 = e^{-x} \land v_3 = \cos x$ by polynomial expressions
Abstracting EDSs: An Example

- (S4): the simulation map is \( \Theta(x, y) = (x, y, \sin x, e^{-x}, \cos x) \)
- \( \Theta(\Xi_x) \triangleq \Xi_x \wedge v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x \)
- \( \Theta(D_x) \triangleq D_x \wedge v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x \)
- abstracting \( v_1 = \sin x \wedge v_2 = e^{-x} \wedge v_3 = \cos x \) by polynomial expressions
Polynomial Approximation via Taylor Model

- $D_x \triangleq -2 \leq x \leq 2 \land -2 \leq y \leq 2$
- $D_x \land v_1 = \sin x$, expand up to degree 6
- $D_x \land v_2 = e^{-x}$, expand up to degree 6

In this way we can obtain $\Xi_y$, $D_y$
Polynomial Approximation via Taylor Model

- \( D_x \triangleq -2 \leq x \leq 2 \land -2 \leq y \leq 2 \)
- \( D_x \land v_1 = \sin x \), expand up to degree 6
- \( D_x \land v_2 = e^{-x} \), expand up to degree 6

In this way we can obtain \( \Xi_y, D_y \)
Abstracting EHSs

- abstracting EHS $\mathcal{H}_x \equiv (Q, X, f_x, D_x, E, G_x, R_x, \Xi_x)$ by PHS $\mathcal{H}_y \equiv (Q, Y, f_y, D_y, E, G_y, R_y, \Xi_y)$
- 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 \( e = (q, q); \)
- \( D_{x,q} \models y \geq \sin x; \ G_{x,e} \models y = \sin x; \)
- \( \Xi_{x,q} \models y \geq 4.9 \land y \leq 5.1 \land x = 0 \land v_x = -1 \land v_y = 0; \)
- \( f_{x,q} = \begin{cases} 
\dot{x} &= v_x \\
\dot{y} &= v_y \\
\dot{v}_x &= 0 \\
\dot{v}_y &= -9.8 
\end{cases} \)
- \( R_{x,e}(x, y, v_x, v_y) \models \{(x, y, v'_x, v'_y)\} \) with
  \[
  \begin{align*}
  v'_x &= \frac{(\sin x)^2 \cdot v_x + 2(\cos x) \cdot v_y}{1 + (\cos x)^2} \\
  v'_y &= \frac{2(\cos x) \cdot v_x - (\sin x)^2 \cdot v_y}{1 + (\cos x)^2}
  \end{align*}
  \]
Abstracting EHSs: An Example

- Bouncing ball on a sine-waved surface
- \( Q = \{q\} \); \( X = \{x, y, v_x, v_y\} \);
- \( E = \{e\} \) with \( e = (q, q) \);
- \( D_{x,q} \equiv y \geq \sin x \); \( G_{x,e} \equiv y = \sin x \);
- \( \Xi_{x,q} \equiv y \geq 4.9 \land y \leq 5.1 \land x = 0 \land v_x = -1 \land v_y = 0 \);
- \( f_{x,q} = \begin{cases} 
\dot{x} & = v_x \\
\dot{y} & = v_y \\
\dot{v}_x & = 0 \\
\dot{v}_y & = -9.8 
\end{cases} \)
- \( R_{x,e}(x, y, v_x, v_y) \equiv \{(x, y, v'_x, v'_y)\} \) with
  \[
  \begin{align*}
  v'_x &= \frac{(\sin x)^2 \cdot v_x + 2(\cos x) \cdot v_y}{1 + (\cos x)^2} \\
  v'_y &= \frac{2(\cos x) \cdot v_x - (\sin x)^2 \cdot v_y}{1 + (\cos x)^2}
  \end{align*}
  \]
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 $C_x \equiv (\Xi_x, f_x, D_x)$, where

1. $\Xi_x \equiv (x + 0.5)^2 + (y - 0.5)^2 - 0.16 \leq 0$;
2. $D_x \equiv -2 \leq x \leq 2 \land -2 \leq y \leq 2$; and
3. $f_x$ defines the ODE

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

4. verify the safety of $C_x$ w.r.t. an unsafe region $\bar{S}_x \equiv (x - 0.7)^2 + (y + 0.7)^2 - 0.09 \leq 0$. 
The Verification Problem

Consider the EDS $C_x \equiv (\Xi_x, f_x, D_x)$, where

- $\Xi_x \equiv (x + 0.5)^2 + (y - 0.5)^2 - 0.16 \leq 0$;
- $D_x \equiv -2 \leq x \leq 2 \land -2 \leq y \leq 2$; and
- $f_x$ 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 $C_x$ w.r.t. an unsafe region $\bar{S}_x \equiv (x - 0.7)^2 + (y + 0.7)^2 - 0.09 \leq 0$
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

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
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
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
Problem Description

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

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 $\omega$ that $\mathcal{H}'$ accepts, if $(q, x)$ is on $\omega$, then $x \in S_q$;
(r3) Non-blocking: $\mathcal{H}'$ is non-blocking.
Problem Description

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

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

1. **Refinement**: for any $q \in Q$, $D'_q \subseteq D_q$, and for any $e \in E$, $G'_e \subseteq G_e$;
2. **Safety**: for any trajectory $\omega$ that $\mathcal{H}'$ accepts, if $(q, x)$ is on $\omega$, then $x \in S_q$;
3. **Non-blocking**: $\mathcal{H}'$ is non-blocking.
Problem Description

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

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 $\omega$ that $\mathcal{H}'$ accepts, if $(q, x)$ is on $\omega$, then $x \in S_q$;

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

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

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

1. Refinement: for any $q \in Q$, $D'_q \subseteq D_q$, and for any $e \in E$, $G'_e \subseteq G_e$;
2. Safety: for any trajectory $\omega$ that $\mathcal{H}'$ accepts, if $(q, x)$ is on $\omega$, then $x \in S_q$;
3. Non-blocking: $\mathcal{H}'$ is non-blocking.
Problem Description

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

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 $\omega$ that $\mathcal{H}'$ accepts, if $(q, x)$ is on $\omega$, then $x \in S_q$;
- (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

$q_1$: no rod

$q_2$: being immersed

$q_3$: immersed

$q_4$: being removed
**A Nuclear Reactor Example (Cont’d)**

- $x$: temperature;
- $p$: proportion immersed

\[
\begin{align*}
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 0
\end{align*}
\]

- $q_1$: no rod
- $q_2$: being immersed
- $q_3$: immersed
- $q_4$: being removed

\[
\begin{align*}
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 1
\end{align*}
\]

\[
\begin{align*}
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= -1
\end{align*}
\]

\[
\begin{align*}
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 0
\end{align*}
\]
A Nuclear Reactor Example (Cont’d)

- $x$: temperature;
- $p$: proportion immersed

\[ \dot{x} = \frac{x}{10} - 6p - 50 \]

- $q_1$: no rod
  - $\dot{x} = \frac{x}{10} - 6p - 50$
  - $\dot{p} = 0$
  - $D_1 \equiv p = 0$

- $q_2$: being immersed
  - $\dot{x} = \frac{x}{10} - 6p - 50$
  - $\dot{p} = 1$
  - $D_2 \equiv 0 \leq p \leq 1$

- $q_3$: immersed
  - $\dot{x} = \frac{x}{10} - 6p - 50$
  - $\dot{p} = 0$
  - $D_3 \equiv p = 1$

- $q_4$: being removed
  - $\dot{x} = \frac{x}{10} - 6p - 50$
  - $\dot{p} = -1$
  - $D_4 \equiv 0 \leq p \leq 1$
A Nuclear Reactor Example (Cont’d)

- $x$: temperature;
- $p$: proportion immersed

$q_1$: no rod

\[
\begin{align*}
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 0 \\
D_1 &\equiv p = 0
\end{align*}
\]

$q_2$: being immersed

\[
\begin{align*}
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 1 \\
D_2 &\equiv 0 \leq p \leq 1
\end{align*}
\]

$q_3$: immersed

\[
\begin{align*}
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 0 \\
D_3 &\equiv p = 1
\end{align*}
\]

$q_4$: being removed

\[
\begin{align*}
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= -1 \\
D_4 &\equiv 0 \leq p \leq 1
\end{align*}
\]
Switching Controller Synthesis for the Reactor

\( S \triangleq 510 \leq x \leq 550 \) for all modes

\[ \begin{align*}
q_1 &: \text{no rod} \\
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 0 \\
D_1 &\triangleq p = 0 \\
G_{41} &\triangleq p = 0 \\
q_2 &: \text{being immersed} \\
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 1 \\
D_2 &\triangleq 0 \leq p \leq 1 \\
G_{23} &\triangleq p = 1 \\
q_4 &: \text{being removed} \\
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= -1 \\
D_4 &\triangleq 0 \leq p \leq 1 \\
q_3 &: \text{immersed} \\
\dot{x} &= x/10 - 6p - 50 \\
\dot{p} &= 0 \\
D_3 &\triangleq p = 1
\end{align*} \]
Switching Controller Synthesis for the Reactor

\( S \triangleq 510 \leq x \leq 550 \) for all modes

- **q₁**: no rod
  - \( \dot{x} = \frac{x}{10} - 6p - 50 \)
  - \( \dot{p} = 0 \)
  - \( D₁ \triangleq p = 0 \)
  - \( G₄₁ \triangleq p = 0 \)

- **q₂**: being immersed
  - \( \dot{x} = \frac{x}{10} - 6p - 50 \)
  - \( \dot{p} = 1 \)
  - \( D₂ \triangleq 0 \leq p \leq 1 \)
  - \( G₂₃ \triangleq p = 1 \)

- **q₃**: immersed
  - \( \dot{x} = \frac{x}{10} - 6p - 50 \)
  - \( \dot{p} = 0 \)
  - \( D₃ \triangleq p = 1 \)

- **q₄**: being removed
  - \( \dot{x} = \frac{x}{10} - 6p - 50 \)
  - \( \dot{p} = -1 \)
  - \( D₄ \triangleq 0 \leq p \leq 1 \)
Bad Switching Violates Safety Property

Transition from mode $q_1$ to $q_2$
Solution to the Controller Synthesis Problem

Abstract Solution

Let $H$ be a hybrid system and $S$ 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

$$H_q \triangleq \left( \bigcup_{e=(q,q') \in E} G'_e \right)^c,$$

where $G'_e \triangleq G_e \cap D'_q$, 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 \( S \) be a safety property. If we can find a family of \( D'_q \subseteq \mathbb{R}^n \) such that

1. (c1) for all \( q \in Q \), \( D'_q \subseteq D_q \cap S_q \);
2. (c2) for all \( q \in Q \), \( D'_q \) is a *continuous invariant* of \((H_q, f_q)\) with

\[
H_q \triangleq \bigcup_{e = (q, q') \in E} G'_e^c,
\]

where \( G'_e \triangleq G_e \cap D'_q \) 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 $S$ 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

$$H_q \doteq \left( \bigcup_{e=(q, q') \in E} G'_e \right)^c,$$

where $G'_e \doteq G_e \cap D'_q$ 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 $G'_e \equiv G_e \cap D'_{q'}$;

(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 $G'_e \equiv G_e \cap D'_q$;

(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 \( G'_e = G_e \cap D'_q \);

(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 $G'_e \supseteq G_e \cap D'_{q'}$;

(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 $G'_e \equiv G_e \cap D'_{q'}$;

(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
For the running example,

- At $D_{q_2}$, temperature $x$ achieves maximal value when crossing $l_1 \equiv x/10 - 6p - 50 = 0$.
- $E(5/6, 550)$ 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 $l_1 \equiv x/10 - 6p - 50 = 0$.
- $E(5/6, 550)$ 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 $D'_{q_2}$.
For the running example,

- At $D_{q_2}$, temperature $x$ achieves maximal value when crossing $l_1 \triangleq x/10 - 6p - 50 = 0$.
- $E(5/6, 550)$ 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 $D'_{q_2}$.
For the running example,

- At $D_{q_2}$, temperature $x$ achieves maximal value when crossing $l_1 \triangleq x/10 - 6p - 50 = 0$.
- $E(5/6, 550)$ 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 $D'_{q_2}$.
The set of parameters: \( a, b, c, d \)

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

- \( G'_{12} \triangleq p = 0 \land b \leq x \leq a \)
- \( G'_{23} \triangleq p = 1 \land d \leq x \leq 550 \)
- \( G'_{34} \triangleq p = 1 \land d \leq x \leq c \)
- \( G'_{41} \triangleq p = 0 \land 510 \leq x \leq a \)
Revisiting the Running Example (Cont’d)

The set of parameters: \( a, b, c, d \)

- \( D'_1 \triangleq p = 0 \land 510 \leq x \leq a \)
- \( D'_2 \triangleq 0 \leq p \leq 1 \land x - b \geq p(d - b) \land \)
  \( x - 550 - \frac{36}{25}(a - 550)(p - \frac{5}{6})^2 \leq 0 \)
- \( D'_3 \triangleq p = 1 \land d \leq x \leq 550 \)
- \( D'_4 \triangleq 0 \leq p \leq 1 \land x - a \leq p(c - a) \land \)
  \( x - 510 - \frac{36}{25}(d - 510)(p - \frac{1}{6})^2 \geq 0 \)
- \( G'_{12} \triangleq p = 0 \land b \leq x \leq a \)
- \( G'_{23} \triangleq p = 1 \land d \leq x \leq 550 \)
- \( G'_{34} \triangleq p = 1 \land d \leq x \leq c \)
- \( G'_{41} \triangleq p = 0 \land 510 \leq x \leq a \)
Revisiting the Running Example (Cont’d)

- \( a = \frac{6575}{12} \land b = \frac{4135}{8} \land c = \frac{4345}{8} \land d = \frac{6145}{12} \).

- From this result we get that the cooling rod should be immersed before temperature rises to \( \frac{6575}{12} \approx 547.92 \), and removed before temperature drops to \( \frac{6145}{12} \approx 512.08 \).

- By solving differential equations explicitly, the corresponding exact bounds are 547.97 and 512.03.
Revisiting the Running Example (Cont’d)

- \(a = \frac{6575}{12} \land b = \frac{4135}{8} \land c = \frac{4345}{8} \land d = \frac{6145}{12}\).

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)

- \( a = \frac{6575}{12} \land b = \frac{4135}{8} \land c = \frac{4345}{8} \land d = \frac{6145}{12} \).

- 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

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
Problem Description

- Given a hybrid system $\mathcal{H}$ in which transition conditions $h_{ij}$ are not determined but parameterized by $u$, a vector of control parameters.
- Our task is to determine $u$ such that $\mathcal{H}$ can make discrete jumps at desired points, thus guaranteeing that
  - a safety property $S$ is satisfied, i.e. $\mathbf{x} \in S$ at any time
  - an optimization goal, e.g. $\min_u g(\mathbf{u})$, is achieved
Our Approach – Step 1

Derive constraint $D(u)$ on $u$ from the safety requirements $S$

- Compute
  - the exact reachable set $\text{Reach}_H(x, u)$ of $H$, or
  - an inductive invariant $\text{Inv}_H(x, u)$

as polynomial formulas

- Suppose $S$ is also modeled by polynomial formulas, then $D(u)$ can be obtained by applying QE to

\[
\forall x. \left( \text{Reach}_H(x, u) \rightarrow S \right)
\]

or

\[
\forall x. \left( \text{Inv}_H(x, u) \rightarrow S \right)
\]
Our Approach – Step 1

Derive constraint $D(u)$ on $u$ from the safety requirements $S$

- Compute
  - the exact reachable set $\text{Reach}_H(x, u)$ of $H$, or
  - an inductive invariant $\text{Inv}_H(x, u)$

as polynomial formulas

- Suppose $S$ is also modeled by polynomial formulas, then $D(u)$ can be obtained by applying QE to

$$\forall x. \left( \text{Reach}_H(x, u) \rightarrow S \right)$$

or

$$\forall x. \left( \text{Inv}_H(x, u) \rightarrow S \right)$$
Our Approach – Step 1

Derive constraint $D(u)$ on $u$ from the safety requirements $S$

- Compute
  - the exact reachable set $\text{Reach}_H(x, u)$ of $H$, or
  - an inductive invariant $\text{Inv}_H(x, u)$

  as polynomial formulas

- Suppose $S$ is also modeled by polynomial formulas, then $D(u)$ can be obtained by applying QE to

$$\forall x. \left( \text{Reach}_H(x, u) \rightarrow S \right)$$

or

$$\forall x. \left( \text{Inv}_H(x, u) \rightarrow S \right)$$
Our Approach – Step 1

Derive constraint $D(u)$ on $u$ from the safety requirements $S$

- Compute
  - the exact reachable set $\text{Reach}_H(x, u)$ of $H$, or
  - an inductive invariant $\text{Inv}_H(x, u)$ as polynomial formulas

- Suppose $S$ is also modeled by polynomial formulas, then $D(u)$ can be obtained by applying QE to

$$\forall x. \left( \text{Reach}_H(x, u) \rightarrow S \right)$$

or

$$\forall x. \left( \text{Inv}_H(x, u) \rightarrow S \right)$$
Our Approach – Step 2

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

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

$$\min_{u \in [-1,1]} u^2 = 0$$
Encode the optimization problem (suppose the objective function $g$ is a polynomial) over constraint $D(u)$ into a quantified first-order polynomial formula $Qu.\varphi(u, z)$ by introducing a fresh variable $z$

- Minimize $u^2$ on $[-1, 1]$
- Introduce a fresh variable $z$: $u \geq -1 \land u \leq 1 \land u^2 \leq z$
- Projection to the $z$-axis: $\exists u. (u \geq -1 \land u \leq 1 \land u^2 \leq z)$
- 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 $D(u)$ into a quantified first-order polynomial formula $Qu.\varphi(u, z)$ by introducing a fresh variable $z$

- Minimize $u^2$ on $[-1, 1]$
- Introduce a fresh variable $z$: $u \geq -1 \land u \leq 1 \land u^2 \leq z$
- Projection to the $z$-axis: $\exists u. (u \geq -1 \land u \leq 1 \land u^2 \leq z)$
- 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 $D(u)$ into a quantified first-order polynomial formula $Qu.\varphi(u, z)$ by introducing a fresh variable $z$

- Minimize $u^2$ on $[-1, 1]$
- Introduce a fresh variable $z$:
  $u \geq -1 \land u \leq 1 \land u^2 \leq z$
- Projection to the $z$-axis:
  $\exists u. (u \geq -1 \land u \leq 1 \land u^2 \leq z)$
- 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 \( D(u) \) into a quantified first-order polynomial formula \( Q_u.\varphi(u, z) \) by introducing a fresh variable \( z \)

- Minimize \( u^2 \) on \([-1, 1]\)
- Introduce a fresh variable \( z \):
  \[
  u \geq -1 \land u \leq 1 \land u^2 \leq z
  \]
- Projection to the \( z \)-axis:
  \[
  \exists u. (u \geq -1 \land u \leq 1 \land u^2 \leq z)
  \]
- After QE: \( z \geq 0 \), which means
  \[
  \min_{u \in [-1, 1]} u^2 = 0
  \]
Encoding Optimization Criteria

Lemma

Suppose \( g_1(u_1), g_2(u_1, u_2), g_3(u_1, u_2, u_3) \) are polynomials, and \( D_1(u_1), D_2(u_1, u_2), D_3(u_1, u_2, u_3) \) are nonempty compact semi-algebraic sets. Then there exist \( c_1, c_2, c_3 \in \mathbb{R} \) s.t.

\[
\exists u_1. (D_1 \land g_1 \leq z) \iff z \geq c_1
\]

\[
\forall u_2. (\exists u_1. D_2 \implies \exists u_1. (D_2 \land g_2 \leq z)) \iff z \geq c_2
\]

\[
\exists u_3. ((\exists u_1 u_2. D_3) \land \forall u_2. (\exists u_1. D_3 \implies \exists u_1. (D_3 \land g_3 \leq z))) \iff z \triangleright c_3
\]

where \( \triangleright \in \{>, \geq\} \), and \( c_1, c_2, c_3 \) satisfy

\[
c_1 = \min_{u_1} g_1(u_1) \quad \text{over } D_1(u_1), \quad \text{(7)}
\]

\[
c_2 = \sup_{u_2} \min_{u_1} g_2(u_1, u_2) \quad \text{over } D_2(u_1, u_2), \quad \text{(8)}
\]

\[
c_3 = \inf_{u_3} \sup_{u_2} \min_{u_1} g_3(u_1, u_2, u_3) \quad \text{over } D_3(u_1, u_2, u_3). \quad \text{(9)}
\]
Our Approach – Step 3

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

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

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

where $F_A$ is a finite subset of $A$
Our Approach – Step 3

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

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

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

where $F_A$ is a finite subset of $A$
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
A Reported Case Study


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


- Provided by the HYDAC ELECTRONIC GMBH company within the European project Quasimodo

- An oil pump control problem
  - safety
  - robustness
  - optimality
A Reported Case Study


- 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

![Graph showing oil consumption rate over time](image-url)
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

![Diagram showing consumption rate over time](image-url)
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 $2.2 \text{ l/s (liter/second)}$
- 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

- $R_s$ (safety): maintain

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

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

and considering the energy cost and wear of the system,

- $R_o$ (optimality): minimize the average accumulated oil volume in the limit, i.e. minimize

\[ \lim_{T \to \infty} \frac{1}{T} \int_{t=0}^{T} v(t) \, dt \]
Control Objectives

Determine the $t_i$’s in order to

- $R_s$ (safety): maintain

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

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

and considering the energy cost and wear of the system,

- $R_o$ (optimality): minimize the average accumulated oil volume in the limit, i.e. minimize

$$\lim_{T \to \infty} \frac{1}{T} \int_{t=0}^{T} v(t) \, dt$$
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 $f = 0.1/\text{s}$
  - imprecision in the measurement of oil volume, up to $\epsilon = 0.06/\text{l}$
  - imprecision in the measurement of time, up to $\delta = 0.015/\text{s}$. 
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 $[L, U] \subseteq [V_{\text{min}}, V_{\text{max}}]$
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 $[L, U] \subseteq [V_{\text{min}}, V_{\text{max}}]$
Repeated Cycles
Step 1: Modeling Oil Consumption

- Fluctuation of consumption rate: $f = 0.1$

<table>
<thead>
<tr>
<th>time</th>
<th>[2,4]</th>
<th>[8,10]</th>
<th>[10,12]</th>
<th>[14,16]</th>
<th>[16,18]</th>
</tr>
</thead>
<tbody>
<tr>
<td>rate</td>
<td>1.2</td>
<td>1.2</td>
<td>2.5</td>
<td>1.7</td>
<td>0.5</td>
</tr>
</tbody>
</table>
Step 1: Modeling Oil Consumption

<table>
<thead>
<tr>
<th>time</th>
<th>[2,4]</th>
<th>[8,10]</th>
<th>[10,12]</th>
<th>[14,16]</th>
<th>[16,18]</th>
</tr>
</thead>
<tbody>
<tr>
<td>rate</td>
<td>1.2</td>
<td>1.2</td>
<td>2.5</td>
<td>1.7</td>
<td>0.5</td>
</tr>
</tbody>
</table>

fluctuation of consumption rate: $f = 0.1$

\[ C_1 \equiv \begin{align*}
\&(0 \leq t \leq 2) \quad \rightarrow \quad V_{out} = 0) \\
\& (2 \leq t \leq 4) \quad \rightarrow \quad 1.1(t-2) \leq V_{out} \leq 1.3(t-2)) \\
\& (4 \leq t \leq 8) \quad \rightarrow \quad 2.2 \leq V_{out} \leq 2.6) \\
\& (8 \leq t \leq 10) \quad \rightarrow \quad 2.2+1.1(t-8) \leq V_{out} \leq 2.6+1.3(t-8)) \\
\& (10 \leq t \leq 12) \quad \rightarrow \quad 4.4+2.4(t-10) \leq V_{out} \leq 5.2+2.6(t-10)) \\
\& (12 \leq t \leq 14) \quad \rightarrow \quad 9.2 \leq V_{out} \leq 10.4) \\
\& (14 \leq t \leq 16) \quad \rightarrow \quad 9.2+1.6(t-14) \leq V_{out} \leq 10.4+1.8(t-14)) \\
\& (16 \leq t \leq 18) \quad \rightarrow \quad 12.4+0.4(t-16) \leq V_{out} \leq 14+0.6(t-16)) \\
\& (18 \leq t \leq 20) \quad \rightarrow \quad 13.2 \leq V_{out} \leq 15.2) 
\end{align*}\]
Step 1: Modeling the Pump

- We will first assume that the pump is activated at most twice in one cycle: \( t_1, t_2, t_3, t_4 \)

\( t_{i+1} - t_i \geq 2: \)

\[
C_2 \doteq (t_1 \geq 2 \land t_2 - t_1 \geq 2 \land t_3 - t_2 \geq 2 \land t_4 - t_3 \geq 2 \land t_4 \leq 20) \\
\lor (t_1 \geq 2 \land t_2 - t_1 \geq 2 \land t_2 \leq 20 \land t_3 = 20 \land t_4 = 20) \\
\lor (t_1 = 20 \land t_2 = 20 \land t_3 = 20 \land t_4 = 20)
\]

- 2.2l/s

\[
C_3 \doteq (0 \leq t \leq t_1) \rightarrow V_{in} = 0 \\
\land (t_1 \leq t \leq t_2) \rightarrow V_{in} = 2.2(t - t_1) \\
\land (t_2 \leq t \leq t_3) \rightarrow V_{in} = 2.2(t_2 - t_1) \\
\land (t_3 \leq t \leq t_4) \rightarrow V_{in} = 2.2(t_2 - t_1) + 2.2(t - t_3) \\
\land (t_4 \leq t \leq 20) \rightarrow V_{in} = 2.2(t_2 + t_4 - t_1 - t_3)
\]
Step 1: Modeling the Pump

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

\[
C_2 \equiv \begin{cases} 
(t_1 \geq 2 \land t_2 - t_1 \geq 2 \land t_3 - t_2 \geq 2 \land t_4 - t_3 \geq 2 \land t_4 \leq 20) \\
\lor (t_1 \geq 2 \land t_2 - t_1 \geq 2 \land t_2 \leq 20 \land t_3 = 20 \land t_4 = 20) \\
\lor (t_1 = 20 \land t_2 = 20 \land t_3 = 20 \land t_4 = 20)
\end{cases}
\]

- 2.2l/s

\[
C_3 \equiv \begin{cases} 
(0 \leq t \leq t_1) \rightarrow V_{in} = 0 \\
\land (t_1 \leq t \leq t_2) \rightarrow V_{in} = 2.2(t - t_1) \\
\land (t_2 \leq t \leq t_3) \rightarrow V_{in} = 2.2(t_2 - t_1) \\
\land (t_3 \leq t \leq t_4) \rightarrow V_{in} = 2.2(t_2 - t_1) + 2.2(t - t_3) \\
\land (t_4 \leq t \leq 20) \rightarrow V_{in} = 2.2(t_2 + t_4 - t_1 - t_3)
\end{cases}
\]
Step 1: Modeling the Pump

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

\[
C_2 \equiv \begin{cases} 
(t_1 \geq 2 \land t_2 - t_1 \geq 2 \land t_3 - t_2 \geq 2 \land t_4 - t_3 \geq 2 \land t_4 \leq 20) \\
\vee (t_1 \geq 2 \land t_2 - t_1 \geq 2 \land t_2 \leq 20 \land t_3 = 20 \land t_4 = 20) \\
\vee (t_1 = 20 \land t_2 = 20 \land t_3 = 20 \land t_4 = 20)
\end{cases}
\]

- \( 2.2l/s \)

\[
C_3 \equiv \begin{cases} 
(0 \leq t \leq t_1) \rightarrow V_{in} = 0 \\
\land (t_1 \leq t \leq t_2) \rightarrow V_{in} = 2.2(t - t_1) \\
\land (t_2 \leq t \leq t_3) \rightarrow V_{in} = 2.2(t_2 - t_1) \\
\land (t_3 \leq t \leq t_4) \rightarrow V_{in} = 2.2(t_2 - t_1) + 2.2(t - t_3) \\
\land (t_4 \leq t \leq 20) \rightarrow V_{in} = 2.2(t_2 + t_4 - t_1 - t_3)
\end{cases}
\]
Step 1: Encoding Safety Requirements

- Oil volume in the accumulator:

\[ C_4 \triangleq v = v_0 + V_{in} - V_{out} \, . \]

- Inductiveness and safety (considering robustness):

\[ C_5 \triangleq t = 20 \rightarrow L + 0.2 \leq v \leq U - 0.2 \]
\[ C_6 \triangleq 0 \leq t \leq 20 \rightarrow V_{min} + 0.2 \leq v \leq V_{max} - 0.2 \, . \]
Step 1: Encoding Safety Requirements

- Oil volume in the accumulator:
  \[ C_4 \triangleq \nu = v_0 + V_{in} - V_{out} \, . \]

- Inductiveness and safety (considering robustness):
  \[ C_5 \triangleq t = 20 \implies L + 0.2 \leq \nu \leq U - 0.2 \]
  \[ C_6 \triangleq 0 \leq t \leq 20 \implies V_{min} + 0.2 \leq \nu \leq V_{max} - 0.2 \, . \]
Step 1: Encoding Safety Requirements (Cont’d)

\[ S \equiv \forall t, v, V_{in}, V_{out}. (C_1 \land C_3 \land C_4 \rightarrow C_5 \land C_6). \]

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

\[ C_8 \equiv \forall v_0. (C_7 \rightarrow \exists t_1 t_2 t_3 t_4. (C_2 \land S)) \]

- \( C_7 \equiv L \leq v_0 \leq U \)
- \( C_2 \): 2-second latency
Step 1: Encoding Safety Requirements (Cont’d)

\[ S \equiv \forall t, v, V_{in}, V_{out}. (C_1 \land C_3 \land C_4 \longrightarrow C_5 \land C_6). \]

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

\[ C_8 \equiv \forall v_0. \left( C_7 \longrightarrow \exists t_1 t_2 t_3 t_4. (C_2 \land S) \right). \]

- \( C_7 \equiv L \leq v_0 \leq U \)
- \( C_2 \): 2-second latency
Step 1: Encoding Safety Requirements (Cont’d)

\[ S \triangleq \forall t, v, V_{\text{in}}, V_{\text{out}}. (C_1 \land C_3 \land C_4 \rightarrow C_5 \land C_6) . \]

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

\[ C_8 \triangleq \forall v_0. \left( C_7 \rightarrow \exists t_1 t_2 t_3 t_4. (C_2 \land S) \right) . \]

- \( C_7 \triangleq L \leq v_0 \leq U \)
- \( C_2 \): 2-second latency
Deriving Constraints

Applying QE to

\[ C_8 \equiv \forall v_0. \left( C_7 \rightarrow \exists t_1 t_2 t_3 t_4. (C_2 \land S) \right) , \]

we get

\[ C_9 \equiv L \geq 5.1 \land U \leq 24.9 \land U - L \geq 2.4 . \]
Deriving Constraints

Applying QE to

\[ C_8 \equiv \forall \nu_0. \left( C_7 \rightarrow \exists t_1 t_2 t_3 t_4. (C_2 \land S) \right) , \]

we get

\[ C_9 \equiv L \geq 5.1 \land U \leq 24.9 \land U - L \geq 2.4 . \]
Deriving Constraints (Cont’d)

\[ C_{10} \equiv C_2 \land C_7 \land C_9 \land S. \]

- \( C_2 \): 2-second latency
- \( C_7 \): \( L \leq v_0 \leq U \)
- \( C_9 \): constraint on \( L, U \)
- \( S \): safety and inductiveness

After **QE**:

\[ D(L, U, v_0, t_1, t_2, t_3, t_4) \equiv \bigvee_{i=1}^{92} D_i \]
Deriving Constraints (Cont’d)

\[ C_{10} \equiv C_2 \land C_7 \land C_9 \land S. \]

- \( C_2 \): 2-second latency
- \( C_7 \): \( L \leq v_0 \leq U \)
- \( C_9 \): constraint on \( L, U \)
- \( S \): safety and inductiveness

After \( \text{QE} \):

\[ \mathcal{D}(L, U, v_0, t_1, t_2, t_3, t_4) \equiv \bigvee_{i=1}^{92} D_i \]
Step 2: Optimization Criterion

**R_o (optimality):** minimize the average accumulated oil volume in the limit, i.e. minimize

\[
\lim_{T \to \infty} \frac{1}{T} \int_{t=0}^{T} \nu(t) \, dt
\]
Optimization Criterion (Contd.)

- $R'_o : \min_{[L,U]} \max_{v_0 \in [L,U]} \min_t \frac{1}{20} \int_{t=0}^{20} v(t) dt$. 

\[
R'_o : \min_{[L,U]} \max_{v_0 \in [L,U]} \min_t \frac{1}{20} \int_{t=0}^{20} v(t) dt. 
\]
Step 2: Encoding the Optimization Criterion

Cost function:

\[
g(v_0, t_1, t_2, t_3, t_4) = \frac{1}{20} \int_{t=0}^{20} v(t) \, dt
\]

\[
= \frac{20 v_0 + 1.1(t_1^2 - t_2^2 + t_3^2 - t_4^2 - 40 t_1 + 40 t_2 - 40 t_3 + 40 t_4) - 132.2}{20}
\]

\( R'_o \) can be encoded into

\[
\exists L, U. \left( C_9 \land \forall v_0. \left( C_7 \rightarrow \exists t_1 t_2 t_3 t_4. (D \land g \leq z) \right) \right),
\]

which is equivalent to \( z \geq z^* \) or \( z > z^* \)
Step 2: Encoding the OptimizationCriterion

Cost function:

\[ g(v_0, t_1, t_2, t_3, t_4) \triangleq \frac{1}{20} \int_{t=0}^{20} v(t) \, dt \]

\[ = \frac{20v_0 + 1.1(t_1^2 - t_2^2 + t_3^2 - t_4^2 - 40t_1 + 40t_2 - 40t_3 + 40t_4)}{20} - 132.2 \]

R' can be encoded into

\[ \exists L, U. \left( C_9 \land \forall v_0. \left( C_7 \rightarrow \exists t_1 t_2 t_3 t_4. (D \land g \leq z) \right) \right), \]

which is equivalent to \( z \geq z^* \) or \( z > z^* \)
Step 3: Performing QE

\[ \exists L, U. \left( C_9 \land \forall v_0. \left( C_7 \rightarrow \exists t_1 t_2 t_3 t_4. (D \land g \leq z) \right) \right) \]

- the inner \( \exists \): quadratic programming
- the outer \( \exists \): discretization

\[ L \geq 5.1 \land U \leq 24.9 \land U - L \geq 2.4 \]

- 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 hsc09], 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 hsc09])
Optimal Controllers with 2 Activations

- In [Cassez et al hsc09], 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 hsc09])
Local Optimal Controllers — 2 Activations

\[ t_1 = \frac{10v_0 - 25}{13} \quad \land \quad t_2 = \frac{10v_0 + 1}{13} \quad \land \quad t_3 = \frac{10v_0 + 153}{22} \quad \land \quad t_4 = \frac{157}{11} \]
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 $v_0 \in [5.2, 8.1]$: 

![Graph showing optimal controllers and oil volume over time](chart.png)
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 $v_0 \in [5.2, 8.1]$: 

![Diagram showing oil pump control](image-url)
**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 $v_0 \in [5.2, 8.1]$: 

![Graph showing oil pump control problem](image-url)
Local Optimal Controllers — 3 Activations

\[
\begin{align*}
    t_1 &= \frac{10v_0 - 26}{13} \land t_2 = \frac{10v_0}{13} \land t_3 = \frac{5v_0 + 76}{11} \land t_4 = 12 \land t_5 = 14 \land t_6 = \frac{359}{22} \\
    t_1 &= \frac{10v_0 - 26}{13} \land t_2 = \frac{10v_0}{13} \land t_3 = \frac{5v_0 + 76}{11} \land t_4 = \frac{5v_0 + 98}{9} \land t_5 = \frac{5v_0 + 92}{9} \land t_6 = \frac{20v_0 + 3095}{198} \\
    t_1 &= \frac{10v_0 - 26}{13} \land t_2 = \frac{10v_0}{13} \land t_3 = \frac{5v_0 + 76}{11} \land t_4 = \frac{5v_0 + 98}{9} \land t_5 = \frac{5v_0 + 92}{9} \land t_6 = \frac{5v_0 + 110}{9} \\
    t_1 &= \frac{10v_0 + 26}{13} \land t_2 = \frac{45v_0 + 1300}{143} \land t_3 = 14 \land t_4 = \frac{359}{22} \land t_5 = 20 \land t_6 = 20
\end{align*}
\]

\( v_0 \in [5.2, 6.8) \land v_0 \in [6.8, 7.5) \land v_0 \in [7.5, 7.8) \land v_0 \in [7.8, 8.1] \)
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

\[
\frac{1}{20} \int_{t=0}^{20} v_{s_3}(t) dt < \frac{1}{20} \int_{t=0}^{20} v_{s_4}(t) dt
\]

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

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
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.
**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?