IMITATOR II

A Tool for Solving the Good Parameters Problem in Timed Automata

Étienne André
Laboratoire Spécification et Vérification
LSV, ENS de Cachan & CNRS, France
The Good Parameters Problem

- **Context:** Verification of timed systems
  - Use of timing parameters (unknown constants)
  - Model of Parametric Timed Automata (PTA)
The Good Parameters Problem

- **Context:** Verification of timed systems
  - Use of timing parameters (unknown constants)
  - Model of Parametric Timed Automata (PTA)

- **The good parameters problem:** [Frehse et al., 2008]
  - “Given a bounded parameter domain $V_0$, find a dense set of points (timing parameters) of good behavior in $V_0$ (ideally the largest one)”
The Good Parameters Problem

- **Context:** Verification of timed systems
  - Use of timing parameters (unknown constants)
  - Model of Parametric Timed Automata (PTA)

- **The good parameters problem:** [Frehse et al., 2008]
  - “Given a bounded parameter domain $V_0$, find a dense set of points (timing parameters) of good behavior in $V_0$ (ideally the largest one)”
Classical approaches

- Verification of the property on a set of *discrete points*
  - Drawback: would need an infinite number of verifications to obtain a *dense* set of points

- **Computation of all the reachable states** of a PTA, and intersection with the set of bad states [Alur et al., 1995]
  - Drawback: too costly in practice

- Approach based on **CEGAR** [Clarke et al., 2000, Frehse et al., 2008]
  - Drawback: underapproximation
Classical approaches

- Verification of the property on a set of **discrete points**
  - Drawback: would need an infinite number of verifications to obtain a **dense** set of points

- **Computation of all the reachable states** of a PTA, and intersection with the set of bad states [Alur et al., 1995]
  - Drawback: too costly in practice

- **Approach based on CEGAR** [Clarke et al., 2000, Frehse et al., 2008]
  - Drawback: underapproximation

- **New approach implemented in Imitator II**
  - Method of **behavioral cartography**
Good and Bad Traces

- **Trace** over a PTA: finite alternating sequence of locations and actions (time-abstract run)

- A trace is said to be **good** if it verifies a given property
  - Example of property $\phi$: “$b$ always occurs before $c$”

  - Example of **good** trace w.r.t. $\phi$

  - Example of **bad** trace w.r.t. $\phi$
Outline

1. The Inverse Method Algorithm
2. The Behavioral Cartography Algorithm
3. Implementation and Case Studies
4. Final Remarks
Outline

1. The Inverse Method Algorithm
2. The Behavioral Cartography Algorithm
3. Implementation and Case Studies
4. Final Remarks
The Inverse Method (1/2)

- PTA $\mathcal{A}$
- Reference point $\pi_0$
- IMITATOR II
- Inverse Method
- Constraint $K_0$ on the parameters
The Inverse Method (2/2)

- **Input**
  - A PTA $\mathcal{A}$
  - A reference valuation $\pi_0$ of all the parameters of $\mathcal{A}$

Output: convex constraint on the parameters such that $\forall \pi \mid \pi = \pi_0 \Rightarrow \mathcal{A}$ under $\pi$ has the same trace set as for $\pi_0$ [André et al., 2009]
The Inverse Method (2/2)

• **Input**
  - A PTA $\mathcal{A}$
  - A **reference valuation** $\pi_0$ of all the parameters of $\mathcal{A}$

• **Output:** tile $K_0$
  - Convex **constraint** on the parameters such that
    - $\pi_0 \models K_0$
    - For all point $\pi \models K_0$, $\mathcal{A}$ under $\pi$ has the same trace set as for $\pi_0$

[André et al., 2009]
Application to the Root Contention Protocol

- Root contention protocol of the IEEE 1394 ("FireWire") High Performance Serial Bus [Hune et al., 2002]

- Input: IEEE reference valuation
  \[ \text{rc}_{\text{slow}}_{\text{min}} = 159\text{ns} \]
  \[ \text{delay} = 30\text{ns} \]

- Output:
  \[ K_0 : 2^{\text{delay}} < 76 \land 2^{\text{delay}} + 85 < \text{rc}_{\text{slow}}_{\text{min}} \]

Prop 3: The minimum probability that a leader is elected after three rounds or less is greater or equal to 0.

\[ \forall \pi | \pi = K_0, \text{Prop 3 is satisfied} \]
Application to the Root Contention Protocol

- Root contention protocol of the IEEE 1394 ("FireWire") High Performance Serial Bus [Hune et al., 2002]

- **Input**: IEEE reference valuation
  \[ rc_{\text{slow min}} = 159\text{ns} \]
  \[ \text{delay} = 30\text{ns} \]

- **Output**:
  \[ K_0 : 2\text{delay} < 76 \]
  \[ 2\text{delay} + 85 < rc_{\text{slow min}} \]
Application to the Root Contention Protocol

- Root contention protocol of the IEEE 1394 ("FireWire") High Performance Serial Bus [Hune et al., 2002]
- Input: IEEE reference valuation
  \[ rc_{\text{slow min}} = 159 \text{ns} \]
  \[ \text{delay} = 30 \text{ns} \]
- Output:
  \[ K_0 : 2\text{delay} < 76 \]
  \[ \land 2\text{delay} + 85 < rc_{\text{slow min}} \]
- Prop\textsubscript{3}: The minimum probability that a leader is elected after three rounds or less is greater or equal to 0.75
  - For all \( \pi \models K_0 \), Prop\textsubscript{3} is satisfied
Outline

1. The Inverse Method Algorithm
2. The Behavioral Cartography Algorithm
3. Implementation and Case Studies
4. Final Remarks
The Behavioral Cartography Algorithm

- Goal: Find the maximal set of points corresponding to a good behavior
The Behavioral Cartography Algorithm

- Goal: Find the maximal set of points corresponding to a good behavior

- Method: Iterate the inverse method for all the integer points of a given rectangle $V_0$
The Behavioral Cartography Algorithm

- **Goal:** Find the *maximal set of points* corresponding to a *good behavior*

- **Method:** Iterate the inverse method for all the integer points of a given rectangle $V_0$

- **Output:** set of tiles for all the integer points of $V_0$
  - $\leadsto$ *behavioral cartography* of the parameter space
    - [André and Fribourg, 2010]
The Root Contention Protocol: Cartography

We consider the following $V_0$:

- $rc_{\text{slow}}_{\text{min}} \in [140; 200]$ and
- $\text{delay} \in [1; 50]$
We consider the following $V_0: rc_{\text{slow \_ min}} \in [140; 200]$ and $\text{delay} \in [1; 50]$.

Remarks

▶ Tiles 1 and 6 are infinite towards one dimension
▶ The cartography does not cover the whole real-valued space within $V_0$ (holes in the lower right corner of $V_0$)
We consider the following $V_0$:

$$rc_{\text{slow} \text{ min}} \in [140; 200] \text{ and } \text{delay} \in [1; 50]$$
We consider the following $V_0$:

$$rc_{\text{slow\_min}} \in [140; 200] \text{ and } delay \in [1; 50]$$
We consider the following $V_0$: $rc_{\text{slow min}} \in [140; 200]$ and $delay \in [1; 50]$. 

Remarks

- Tiles 1 and 6 are infinite towards one dimension.
- The cartography does not cover the whole real-valued space within $V_0$ (holes in the lower right corner of $V_0$).
We consider the following $V_0$: $rc_{\text{slow}_{\text{min}}} \in [140; 200]$ and $delay \in [1; 50]$
We consider the following $V_0$:

- $rc_{\text{slow min}} \in [140; 200]$ and $delay \in [1; 50]$
We consider the following $V_0$:

- $rc_{\text{slow min}} \in [140; 200]$
- $delay \in [1; 50]$
The Root Contention Protocol: Cartography

We consider the following $V_0$:

- $rc_{\text{slow\_min}} \in [140; 200]$ and $delay \in [1; 50]$
We consider the following $V_0$:

- $rc_{\text{slow}_{\text{min}}} \in [140; 200]$ and
- $delay \in [1; 50]$
We consider the following $V_0$:

$rc_{\text{slow\_min}} \in [140; 200]$ and $delay \in [1; 50]$
The Root Contention Protocol: Cartography

We consider the following $V_0$: $\text{rc\_slow\_min} \in [140; 200]$ and $\text{delay} \in [1; 50]$

Remarks

- Tiles 1 and 6 are infinite towards one dimension
- The cartography does not cover the whole real-valued space within $V_0$ (holes in the lower right corner of $V_0$)
Partition into Good and Bad Tiles

- A tile is said to be **good** if all its corresponding traces are good.
- According to the nature of the trace sets, we can partition the tiles into **good** and **bad** ones.
Prop$_3$: “The minimum probability that a leader is elected after three rounds or less is greater or equal to 0.75”

- Good tile: 1
- Bad tiles: 2 to 19
**Prop**$_5$: “The minimum probability that a leader is elected after five rounds or less is greater or equal to 0.75”

- Good tiles: 1, 2, 3
- Bad tiles: 4 to 19
Outline

1. The Inverse Method Algorithm
2. The Behavioral Cartography Algorithm
3. Implementation and Case Studies
4. Final Remarks
Implementation

- In short
  - Imitator II: new improved version of Imitator
  - “Inverse Method for Inferring Time Abstract Behavior”
  - 8000 lines of code
  - Program written in OCaml
  - Makes use of the PPL library for handling polyhedra [Bagnara et al., 2008]

- Some features
  - List of tiles with their corresponding trace set under a graphical form
  - Cartography under a graphical form (for 2 parameter dimensions)

- Imitator II is available on its Web page
  - http://www.lsv.ens-cachan.fr/~andre/IMITATOR2
Case Studies

- Application to various case studies
  - Asynchronous circuits
  - Communication protocols

- Computation time for the cartography algorithm
  - Experiments conducted on an Intel Core2 Duo 2.4 GHz with 2 Gb

| Example       | PTAs | loc./PTA | $|X|$ | $|P|$ | $|V_0|$ | tiles | states | trans. | Time (s) |
|---------------|------|----------|-----|-----|------|-------|--------|--------|---------|
| SR-latch      | 3    | [3, 8]   | 3   | 3   | 1331 | 6     | 5      | 4      | 0.3     |
| Flip-flop     | 5    | [4, 16]  | 5   | 2   | 644  | 8     | 15     | 14     | 3       |
| Latch circuit | 7    | [2, 5]   | 8   | 4   | 73062| 5     | 21     | 20     | 96.3    |
| And–Or        | 3    | [4, 8]   | 4   | 6   | 75600| 4     | 64     | 72     | 118     |
| CSMA/CD       | 3    | [3, 8]   | 3   | 3   | 2000 | 140   | 349    | 545    | 269     |
| SPSMALL       | 10   | [3, 8]   | 10  | 2   | 3149 | 259   | 60     | 61     | 1194    |
| RCP           | 5    | [6, 11]  | 6   | 3   | 186050| 19    | 5688   | 9312   | 7018    |
Outline

1. The Inverse Method Algorithm
2. The Behavioral Cartography Algorithm
3. Implementation and Case Studies
4. Final Remarks
Summary

- Implementation of the **Inverse Method**
  - Modeling of a system with parametric timed automata
  - Starting with a valuation $\pi_0$ of the system, we synthesize a constraint $K_0$ with the same trace set as $\pi_0$
  - Gives a criterion of robustness by guaranteeing the same behavior around $\pi_0$

- Implementation of the **Behavioral cartography**
  - Solves the good parameters problem: synthesizes the largest set of points within a rectangle $V_0$ corresponding to a given good behavior
  - Independent from the property: only the partition does
Future Work

- Automatize the partition into good and bad tiles
  - Make use of the UPPAAL model checker [Larsen et al., 1997]

- Extend the behavioral cartography to hybrid automata
  - Allow to consider different clock rates

- Consider a dynamic cartography
  - Refine the scale in order to fill the whole real-valued $V_0$

- Consider a weaker property than equality of trace sets
  - Reference trace with partial orders
The algorithmic analysis of hybrid systems.

An inverse method for parametric timed automata.

André, É. and Fribourg, L. (2010).
Behavioral cartography of timed automata.
In *RP’10*, volume 6227 of *LNCS*, pages 76–90. Springer.

The Parma Polyhedra Library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems.

Counterexample-guided abstraction refinement.
