Symbolic Monitoring against Specification Parametric in Time and Data

Masaki Waga[1,2], Étienne André[3,4], Ichiro Hasuo[1,2]
SOKENDAI[1] / NII[2] / Université de Lorraine, CNRS, Inria, LORIA[3] / JFLI, UMI CNRS[4]

Background

Runtime Verification (RV)

Runtime verification (or monitoring) is not exhaustive but

- it does **not require system model**
  - i.e., Blackbox systems are OK
  - e.g., System w/
    - Machine learning components
    - 3rd-party components
    - Unknown environment

and

- It tends to be **scalable** for complex systems

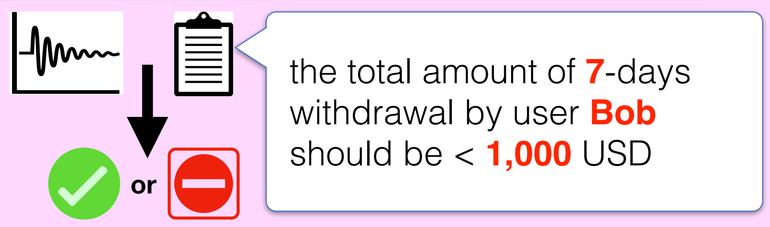
Issues in RV

- It is **difficult** to determine the best threshold in the spec.
  - e.g., What is the best threshold of "too large acceleration"?
- We want **quantitative** results rather than **Boolean** results
  - Clearly satisfied vs. Satisfied but near the borderline

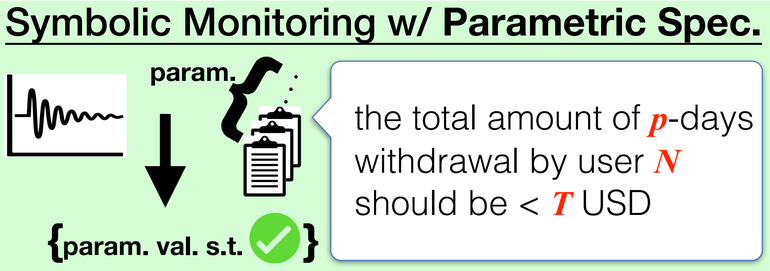
Our Solution

Symbolic Monitoring w/ Parametric Specifications

Boolean Monitoring w/ Concrete Spec.



Symbolic Monitoring w/ Parametric Spec.



Symbolic Monitoring

Input [Contribution]

- **Time-series data**
  - System ***log*** (event + *data* + timestamp)
  - e.g., withdraw(**Alice**, ***100***) withdraw(**Bob**, ***10***) withdraw(**Alice**, ***30***) at 0.7, 1.2, 2.7, t
- **Parameterized real-time spec. with data**
  - ***Spec.*** to be monitored
  - e.g., the total amount of p-days withdrawal by user N should be < T USD

Output

- **All** of the param. val. such that the ***log*** satisfies the ***spec.***
  - e.g., (N, T, p) = (**Alice**, ***140***, ***3.0***), (**Alice**, ***135***, ***4.0***), (**Bob**, ***20***, ***1.0***), …
  - **Infinitely** many → **Symbolic** representation

Contribution

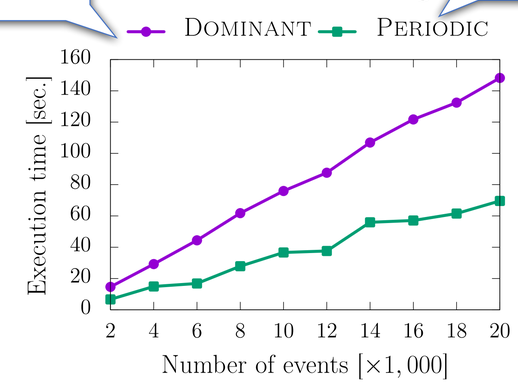
- Introduced **parametric timed data automata** (**PTDA**)
  - **PTDA**: NFA + clock/data variables + time/data parameters
- Gave **symbolic monitoring** algorithm over a PTDA spec.
  - Idea: follow trans. using symbolic representation
  - (Potentially) **infinitely** many param. val. → **symbolic** representation/operations
  - Experiments → **Scalable**!!
- Demo is available on Google Colab

Official Version | arXiv Version | Tool Demo

Scalability

Detects dominant withdrawals by a user

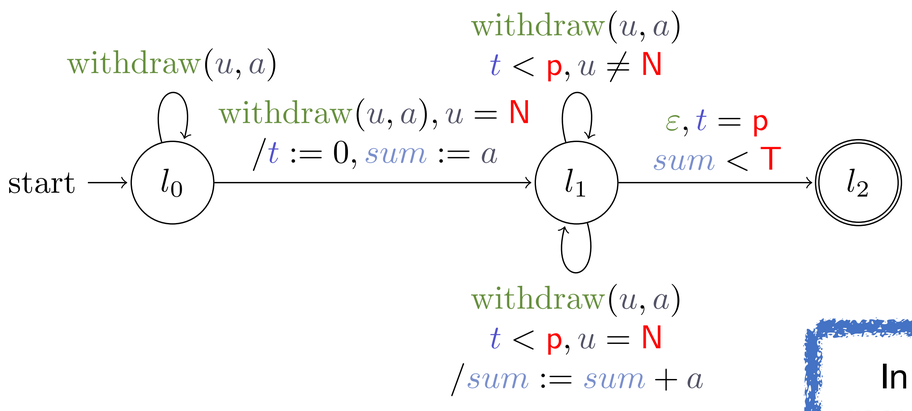
Synthesizes periods of periodic withdrawals



- 20,000 entries in 1 - 2 min.
- Execution time is **linear** in these two benchmarks
- Much more efficient than the worst case!!
- **Important lesson**: these useful spec. can be monitored efficiently

Detail on PTDA

Example of a PTDA



Q: What is "Data"?
A: Any triple $(\mathbb{D}, \mathcal{DE}, \mathcal{DU})$

$\mathbb{D}$: infinite domain

$\mathcal{DE}$: Boolean expression (for guards)

$\mathcal{DU}$: updates (for variable updates/assignments)

In the algorithm, we require ***a data structure*** with some operations

Symbolic Monitoring Algorithm

Idea of the Algorithm

See also an illustration!!

follow the transitions of PTDA
+
abstraction of clock/data/param. val.
(e.g., by convex polyhedra or lists of forbidden strings)
+
(Non-deterministic branching by breadth first search)

Termination

Thm.
Our symbolic monitoring algorithm terminates for any data types $(\mathbb{D}, \mathcal{DE}, \mathcal{DU})$ such that we can compute

restriction, update, emptiness checking, and projection.

Examples

- Strings ($\mathbb{S}$) with ***lists (of forbidden strings)***
- Rationals ($\mathbb{Q}$) with ***convex polyhedra***

This work is partially supported by JST ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), by JSPS Grants-in-Aid No. 15KT0012 & 18J22498, and by the ANR national research program PACS (ANR-14-CE28-0002).