Time-Bounded Resilience

Tajana Ban Kirigin, Jesse Comer, Max Kanovich,
Andre Scedrov, and Carolyn Talcott

LAP 2024

Resilience

ESSAY

America Has a Resilience Problem

The chair of the Federal Trade Commission makes the case for competition in an increasingly consolidated world.

FORBES > INNOVATION

Cyber Resilience Is Your Business: How To Improve It

Administration Priorities The Re

NOVEMBER 30, 2023

Issue Brief: Supply Chain Resilience

Ecosystems as Infrastructure: A New Way of Looking at Climate Resilience

World Health Organization

Home / Publications / Overview /

Operational framework for building climate resilient and low carbon health systems

Resilience

What is resilience?

"[Resilience emphasizes] the ability of a system to adapt and respond to change (both environmental and internal)." Bloomfield et. al., [2].

Why resilience?

"We must recognize the trade-off between efficiency and resilience. It is time to develop the discipline of resilient algorithms." Moshe Vardi, [3].

Overview

- Timed multiset rewriting (MSR) systems are an expressive formalism for modeling planning scenarios with discrete time.
- Expository example:
  1. **Example**: a researcher is planning travel to a conference.
  2. The researcher wants a **resilient** travel plan which achieves his goal despite issues such as flight delays.
- We will formalize resilience for planning scenarios based on timed MSR systems.
- At the end, we will discuss our Maude implementation of this example.

Resilience via Timed Multiset Rewriting Systems

- We want to model a planning scenario.
- High level idea:
  1. We represent states of the scenario via **configurations**.
  2. **Rewrite rules**, representing "actions" in the scenario, modify configurations.
     - **System rules** represent actions of our "protagonist."
     - **Update rules** can be seen as actions of an "adversary."
  3. **Planning** corresponds to finding **compliant** traces to a **goal** configuration.
  4. n-**Resilience** is a decision problem: can we find a compliant trace to a goal configuration which is resilient to n adversarial disruptions?
- There is an intuitive game-theoretic interpretation to this formalism: its complexity lands naturally within the polynomial hierarchy (PH).

First-order Formulas and Facts

- We fix a first-order alphabet Σ.
- **Atomic formulas** are of the form $R(t_1, \ldots, t_n)$, where
  1. R is an n-ary relation symbol in Σ, and
  2. the t_i are Σ-terms which may contain variables.
- **Facts** are atomic formulas without variables.
- **Timestamped atomic formulas** are of the form $F@(T + D)$, where F is an atomic formula, T is a **time variable**, and D is a natural number.
- **Timestamped facts** are of the form $F@t$, where F is a fact and t is a natural number.

Configurations

- **Configurations** are multisets of timestamped facts.

- The **global time** of a configuration is given by the timestamp of a (unique) timestamped fact of the form $\text{Time}@t$.

$$\{\text{Time}@(3d\ 14{:}42),\ \underline{\text{Attended}}(\text{main}, \text{no})@0,\ \text{At}(\text{FRA}, \text{airport})@(3d\ 14{:}05),$$
$$\underline{\text{Event}}(\text{main})@(5d\ 12{:}00),\ \text{Flight}_2(\text{FRA}, \text{DBV})@(3d\ 15{:}25)\}$$

- Note – configurations contain only **ground terms** (i.e., no variables).

Rewrite Rules

- Configurations are modified by **rewrite rules**.
- There is a special rule Tick which increments the global time by one:

$$\text{Time@}T \longrightarrow \text{Time@}(T+1)$$

- All other rewrite rules are **instantaneous**, unable to modify the global time.

Instantaneous Rules

- **Instantaneous rules** have the form

$$\overbrace{Time@T, \mathcal{W}, F_1@T_1, \ldots, F_n@T_n}^{\textbf{Precondition}} \mid \mathcal{C} \longrightarrow \underbrace{Time@T, \mathcal{W}, Q_1@(T + D_1), \ldots, Q_m@(T + D_m)}_{\textbf{Postcondition}}$$

$\mathcal{W}$	—	multiset of timestamped atomic formulas
		(the side condition)
$F_i@T_i$ & $Q_j@T_j$	—	timestamped atomic formulas
$\mathcal{C}$	—	a set of **time constraints** of the form
		$T_1 > T_2 \pm N$ or $T_1 = T_2 \pm N$

Rule Application: Travel Example

Modeling "taking a (two-hour) flight" with an instantaneous rule:

$$\{\mathsf{Time}@(3d\ 14{:}42), \underline{\mathsf{Attended}}(\mathsf{main}, \mathsf{no})@0,\ \mathsf{At}(\mathsf{FRA}, \mathsf{airport})@(3d\ 14{:}05), \underline{\mathsf{Event}}(\mathsf{main})@(5d\ 12{:}00), \mathsf{Flight}_2(\mathsf{FRA}, \mathsf{DBV})@(3d\ 15{:}25)\}$$

Rule Application: Travel Example

Modeling "taking a (two-hour) flight" with an instantaneous rule:

$$\{Time@(3d\ 14{:}42), \underline{Attended}(main, no)@0,\ At(FRA, airport)@(3d\ 14{:}05), \underline{Event}(main)@(5d\ 12{:}00), Flight_2(FRA, DBV)@(3d\ 15{:}25)\}$$

$$Time@T, Flight_2(x_1, x_2)@T_1, At(x_1, airport)@T_2, \mid \{T = T_1, T_2 + 30 \leq T\} \longrightarrow Time@T, Flight_2(x_1, x_2)@T_1, At(x_2, airport)@(T + 120),$$

Rule Application: Travel Example

Modeling "taking a (two-hour) flight" with an instantaneous rule:

$$\{Time@(3d\ 14{:}42), \underline{Attended}(main, no)@0,\ At(FRA, airport)@(3d\ 14{:}05), \underline{Event}(main)@(5d\ 12{:}00), Flight_2(FRA, DBV)@(3d\ 15{:}25)\}$$

$$Time@T, Flight_2(x_1, x_2)@T_1, At(x_1, airport)@T_2, \mid \{T = T_1, T_2 + 30 \leq T\} \longrightarrow Time@T, Flight_2(x_1, x_2)@T_1, At(x_2, airport)@(T + 120),$$

Not applicable! $T \neq T_1$.

Rule Application: Travel Example

Modeling "taking a (two-hour) flight" with an instantaneous rule:

$$\{\text{Time}@(3d\ 14{:}42), \underline{\text{Attended}}(\text{main}, \text{no})@0,\ \text{At}(\text{FRA}, \text{airport})@(3d\ 14{:}05), \underline{\text{Event}}(\text{main})@(5d\ 12{:}00), \text{Flight}_2(\text{FRA}, \text{DBV})@(3d\ 15{:}25)\}$$

$$\text{Time}@T, \text{Flight}_2(x_1, x_2)@T_1, \text{At}(x_1, \text{airport})@T_2, \mid \{T = T_1, T_2 + 30 \leq T\} \longrightarrow \text{Time}@T, \text{Flight}_2(x_1, x_2)@T_1, \text{At}(x_2, \text{airport})@(T + 120),$$

Not applicable! $T \neq T_1$.

After 43 applications of Tick:

$$\{\text{Time}@(3d\ 13{:}25), \underline{\text{Attended}}(\text{main}, \text{no})@0,\ \text{At}(\text{FRA}, \text{airport})@(3d\ 14{:}05), \underline{\text{Event}}(\text{main})@(5d\ 12{:}00), \text{Flight}_2(\text{FRA}, \text{DBV})@(3d\ 15{:}25)\}$$

Rule Application: Travel Example

$$\{\text{Time}@(3d\ 15{:}25), \underline{\text{Attended}}(\text{main}, \text{no})@0,\ \text{At}(\text{FRA}, \text{airport})@(3d\ 14{:}05), \underline{\text{Event}}(\text{main})@(5d\ 12{:}00), \text{Flight}_2(\text{FRA}, \text{DBV})@(3d\ 15{:}25)\}$$

$$\text{Time}@T, \text{Flight}_2(x_1, x_2)@T_1, \text{At}(x_1, \text{airport})@T_2, \mid \{T = T_1, T_2 + 30 \leq T\} \longrightarrow \text{Time}@T, \text{Flight}_2(x_1, x_2)@T_1, \text{At}(x_2, \text{airport})@(T + 120),$$

Rule Application: Travel Example

$$\{\text{Time}@(3d\ 15{:}25), \underline{\text{Attended}}(\text{main}, \text{no})@0,\ \text{At}(\text{FRA}, \text{airport})@(3d\ 14{:}05),$$
$$\underline{\text{Event}}(\text{main})@(5d\ 12{:}00), \text{Flight}_2(\text{FRA}, \text{DBV})@(3d\ 15{:}25)\}$$

$$\text{Time}@T, \text{Flight}_2(x_1, x_2)@T_1, \text{At}(x_1, \text{airport})@T_2, \mid \{T = T_1, T_2 + 30 \leq T\}$$
$$\longrightarrow \text{Time}@T, \text{Flight}_2(x_1, x_2)@T_1, \text{At}(x_2, \text{airport})@(T + 120),$$

$$x_1 \mapsto \text{FRA}$$
$$x_2 \mapsto \text{DBV}$$
$$T \mapsto 3d\ 15 : 25$$
$$T_1 \mapsto 3d\ 15 : 25$$
$$T_2 \mapsto 3d\ 14 : 05$$

Rule Application: Travel Example

$$\{\text{Time}@(3d\ 15{:}25), \underline{\text{Attended}}(\text{main}, \text{no})@0,\ \text{At}(\text{FRA}, \text{airport})@(3d\ 14{:}05),$$
$$\underline{\text{Event}}(\text{main})@(5d\ 12{:}00), \text{Flight}_2(\text{FRA}, \text{DBV})@(3d\ 15{:}25)\}$$

$$\text{Time}@T, \text{Flight}_2(x_1, x_2)@T_1, \text{At}(x_1, \text{airport})@T_2, \mid \{T = T_1, T_2 + 30 \leq T\}$$
$$\longrightarrow \text{Time}@T, \text{Flight}_2(x_1, x_2)@T_1, \text{At}(x_2, \text{airport})@(T + 120),$$

$x_1 \mapsto \text{FRA}$
$x_2 \mapsto \text{DBV}$
$T \mapsto 3d\ 15:25$
$T_1 \mapsto 3d\ 15:25$
$T_2 \mapsto 3d\ 14:05$

Rule instance :

$$\text{Time}@3d\ 15:25, \text{Flight}_2(\text{FRA}, \text{DBV})@3d\ 15:25,$$
$$\text{At}(\text{FRA}, \text{airport})@3d\ 14:05,$$
$$\longrightarrow \text{Time}@3d\ 15:25, \text{Flight}_2(\text{FRA}, \text{DBV})@3d\ 15:25,$$
$$\text{At}(\text{DBV}, \text{airport})@(3d\ 15:25 + 120)$$

Rule Application: Travel Example

$$\{\text{Time}@(3d\ 15{:}25), \underline{\text{Attended}}(\text{main},\text{no})@0,\ \text{At}(\text{FRA},\text{airport})@(3d\ 14{:}05),\ \underline{\text{Event}}(\text{main})@(5d\ 12{:}00), \text{Flight}_2(\text{FRA},\text{DBV})@(3d\ 15{:}25)\}$$

$$\text{Time}@T, \text{Flight}_2(x_1,x_2)@T_1, \text{At}(x_1,\text{airport})@T_2, \mid \{T = T_1,\ T_2 + 30 \leq T\} \longrightarrow \text{Time}@T, \text{Flight}_2(x_1,x_2)@T_1, \text{At}(x_2,\text{airport})@(T+120),$$

$x_1 \mapsto \text{FRA}$
$x_2 \mapsto \text{DBV}$
$T \mapsto 3d\ 15:25$
$T_1 \mapsto 3d\ 15:25$
$T_2 \mapsto 3d\ 14:05$

Rule instance :

$$\text{Time}@3d\ 15:25, \text{Flight}_2(\text{FRA},\text{DBV})@3d\ 15:25,\ \text{At}(\text{FRA},\text{airport})@3d\ 14:05, \longrightarrow \text{Time}@3d\ 15:25, \text{Flight}_2(\text{FRA},\text{DBV})@3d\ 15:25,\ \text{At}(\text{DBV},\text{airport})@(3d\ 15:25+120)$$

$$\{\text{Time}@(3d\ 15{:}25), \underline{\text{Attended}}(\text{main},\text{no})@0,\ \text{At}(\text{DBV},\text{airport})@(3d\ 17{:}25),\ \underline{\text{Event}}(\text{main})@(5d\ 12{:}00), \text{Flight}_2(\text{FRA},\text{DBV})@(3d\ 15{:}25)\}$$

Timed MSR Systems

- A **timed MSR system** is a set $\mathcal{R}$ containing the Tick rule and some finite number of instantaneous rules.

- A **trace** of $\mathcal{R}$ rules from an "initial" configuration $\mathcal{S}_0$ is a sequence $\mathcal{S}_0 \longrightarrow \cdots \longrightarrow \mathcal{S}_n$ of configurations, where some instance of a rule $r \in \mathcal{R}$ applied to $\mathcal{S}_i$ yields $\mathcal{S}_{i+1}$.

- A **goal configuration specification** designates conditions for a configuration to be a **goal configuration**. It contains pairs of the form $\langle \mathcal{S}, \mathcal{C} \rangle$, where $\mathcal{S}$ is a multiset of timestamped atomic formulas and $\mathcal{C}$ is a set of time contraints.

- For example:

$$\{\langle \{\underline{\text{Attended}}(\text{main}, \text{yes})@T_1\}, \emptyset \rangle\}$$

Critical Configurations and Compliance

- A **critical configuration specification** describes when a configuration is "critical."

$$\{\langle \text{Time}@T, \underline{\text{Attended}}(\text{main}, \text{no})@T_1, \underline{\text{Event}}(\text{main})@T_2\}, \{T > T_2\}\rangle\}$$

- A trace is **compliant** if it does not contain any critical configurations.

- Critical configurations can be thought of as **safety violations**, while compliant traces are analogous to **safe traces**.

Toward Resilience

- To model resilience, we need a notion of actions which are under the control of the system, and disruptions which are imposed on the system.

- We model the former via **system rules**, and the latter via **update rules**.

- **Example (update rule)** — A flight is delayed by 30 minutes:

$$\mathsf{Time}@T, \mathsf{Flight}_D(x_1, x_2)@T_1 \mid \{T = T_1\} \longrightarrow \mathsf{Time}@T, \mathsf{Flight}_D(x_1, x_2)@(T + 30).$$

Definition (Planning Scenario, [1])

If $\mathcal{R}$ and $\mathcal{E}$ are sets of system and update rules, $\mathcal{GS}$ and $\mathcal{CS}$ are a goal and critical configuration specifications, and $\mathcal{S}_0$ is an initial configuration, then the tuple $(\mathcal{R}, \mathcal{GS}, \mathcal{CS}, \mathcal{E}, \mathcal{S}_0)$ is a *planning scenario*.

Simplifying Assumptions

For our complexity results, we assume

1. Bounded depth of function applications in terms in facts occurring in traces;
2. η-**simplicity**: there is a fixed bound η on the number of (first-order and time) variables allowed to occur in a pair $\langle \mathcal{S}_i, \mathcal{C}_i \rangle$ in $\mathcal{CS}$; and
3. All planning scenarios are progressing.

Definition (Progressing Planning Scenarios (PPSs))

A planning scenario is **progressing** if, for each rule $r \in \mathcal{R} \cup \mathcal{E}$,

1. r is balanced (i.e., the precondition and postcondition have equal cardinality),
2. r consumes only facts with timestamps in the past or at the current time, and
3. r creates *at least one* fact with timestamp greater than the global time.

Formalizing Resilience

Definition (The (n, a, b)-resilience problem (by recursion on n))

Let $a \in \mathbb{Z}^+$ and $b \in \mathbb{N}$. Inputs: planning scenarios $A = (\mathcal{R}, \mathcal{GS}, \mathcal{CS}, \mathcal{E}, \mathcal{S}_0)$.
A trace is $(0, a, b)$-**resilient with respect to** A if it is a compliant trace of $\mathcal{R}$ rules from $\mathcal{S}_0 \ni Time@t_0$ to a goal configuration and contains at most $a + b$ applications of the Tick rule. For $n > 0$, a trace τ is (n, a, b)-**resilient with respect to** A if

1. τ is $(0, a, b)$-resilient with respect to A, and
2. for any system or goal update rule $r \in \mathcal{E}$ applied to a configuration $\mathcal{S}_i$ in τ, with $\mathcal{S}_i \longrightarrow_r \mathcal{S}'_{i+1}$, where global time t_i in $\mathcal{S}_i$ satisfies $d_i = t_i - t_0 \leq a$, there exists a reaction trace τ' of $\mathcal{R}$ rules from $\mathcal{S}'_{i+1}$ to a goal configuration $\mathcal{S}'$ such that τ' is $(n - 1, a - d_i, b)$-resilient with respect to $A' = (\mathcal{R}, \mathcal{GS}, \mathcal{CS}, \mathcal{E}, \mathcal{S}'_{i+1})$.

A planning scenario $A = (\mathcal{R}, \mathcal{GS}, \mathcal{CS}, \mathcal{E}, \mathcal{S}_0)$ is (n, a, b)-*resilient* if an (n, a, b)-resilient trace with respect to A exists. The (n, a, b)-resilience problem is to determine if a given planning scenario A is (n, a, b)-resilient.

Formalizing Resilience

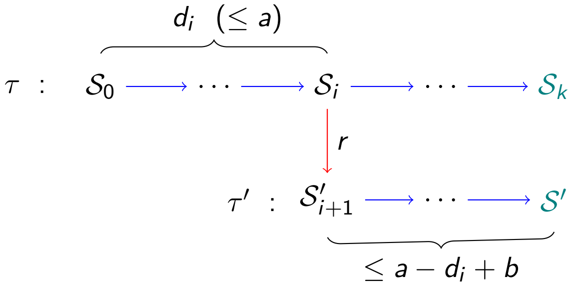

Figure: An (n, a, b)-resilient trace τ and an $(n-1, a-d_i, b)$-resilient reaction trace τ'. The horizontal arrows correspond to system rule applications, while the downward-facing arrow represents an update rule application. The configurations $\mathcal{S}_k$ and $\mathcal{S}'$ on the far right are goal configurations.

Complexity Results

Definition

A decision problem is in Σ_n^P (for n odd) if and only if there exists a polynomial-time algorithm M such that an input x is a *yes* instance of the problem if and only if

$$\exists u_1 \forall u_2 \exists u_3 \ldots \forall u_{n-1} \exists u_n \; M(x, u_1, \ldots, u_n) \text{ accepts},$$

where the u_i are polynomially-bounded in the size of x.

In our case, the existentially-quantified variables represent compliant goal traces, while the universally-quantified variables represent update rule applications.

Complexity Results

Theorem

The (n, a, b)-resilience problem for η-simple PPSs with traces containing only facts of bounded size is $\Sigma^{\mathsf{P}}_{2n+1}$-complete.

Upper bound — by the quantifier-alternation characterization of $\Sigma^{\mathsf{P}}_{2n+1}$.

Lower bound — by a reduction from $\Sigma^{\mathsf{P}}_{2n+1}$-SAT.

Travel Planning in Maude

- The goal is to attend a set of events in different places, with some required and some optional.
- There is a knowledge base of flights to choose from.
- Updates include
  1. flight delay, cancellation, or diversion; and
  2. change of event start or duration.
- A critical configuration is one where the current time is later than the start time of a required event and the event has not been attended.

RWL vs MSR

- System state is represented by data structures rather than facts.
- There is just one time, the current time, represented as an element of the state
- Timers, delays, durations control the passing of time.
- Here, the passage of time is modeled using event/action duration:
  1. taking a flight takes time, and
  2. searching for flight is instantaneous.
- A variant on the usual RTMaude tick and instantaneous rules.
- An optimization of the uniform one time unit ticks model.

Some details — state representation

`tc(dateTime,city,location,events)` — planning
`tc(dateTime,city,location,events, event, flightLists)` —traveling

- Maude Example

```
dateTime = dt(yd(23, 247), hm(12, 42)),
city = FRA, location = airport
event = ev("id215", DBV, center, yd(23,249), hm(14,0), hm(120,0),
false) attendance optional
flightList(s) = fi(fl(FRA,DBV,"id14",hm(15,25),hm(2,0))) –abstract flight
dt(yd(23,247),hm(15,25)), — departure date time
dt(yd(23,247),hm(17,25)) — arrival date time
```

- MSR example

$$\{Time@(3d\ 14{:}42), \underline{Attended}(main, no)@0,\ At(FRA, airport)@(3d\ 14{:}05), \underline{Event}(main, id_{215})@(5d\ 12{:}00), Flight_2(id_{14}, FRA, DBV)@(3d\ 15{:}25)\}$$

Some details — rules

- `plan` – find flight lists from to next event location
- `flt` – take flight duration = arrival - current time
- `event` – attend event duration = event dur + time to airport
- `fltDigress` — apply a flight update
- `replan` – when current time is too late for event or flight

Checking n,a,b-resilience — input

initial state — a planning state with the full set of events to attend
tc(dateTime,city,location,events)
critical state – tcCrit(....)
goal state – tc(dateTime,city,location,mtE) – a terminal state

Checking (n, a, b)-resilience — search algorithm

- Search algorithm
  1. Use Maude search to find a compliant trace. If $n = 0$ return true.
  2. Step through this trace. At each point make a branch for each enabled update, decrement n and go to 1.
- How to do (2):
  1. Convert the trace found by Maude search into a sequence of rule instances.
  2. For each prefix, a maude strategy, and each update, append the update rule and use srewrite to find all updates for this point in the trace.

Experimental results

N:	1		2		3	
2ev	**R?**	**time**	**R?**	**time**	**R?**	**time**
247	N	86ms	-	-	-	-
246	Y	81ms	Y	147ms	N	7476ms
3ev	**R?**	**time**	**R?**	**time**	**R?**	**time**
247	N	1400ms	-	-	-	-
246	Y	325ms	Y	685ms	NF	-

(a) flight/system update rules

N:	1		2		3	
2ev	**R?**	**time**	**R?**	**time**	**R?**	**time**
247	Y	78ms	N	77ms	-	-
246	Y	98ms	N	34800ms	-	-
3ev	**R?**	**time**	**R?**	**time**	**R?**	**time**
247	Y	143ms	N	2627ms	-	-
246	Y	220ms	Y	633ms	Y	2634ms

(b) event/goal update rules

Figure: Summary of (n, a, b)-resilience experiments

Summary

We have:

1. Described timed MSR systems for modeling planning scenarios.
2. Given a formal definition of resilience and analyzed its complexity.
3. Implemented this formalism in Maude and run experiments on resilience of our travel example.

Questions?

References

M. A. Alturki, T. Ban Kirigin, M. Kanovich, V. Nigam, A. Scedrov, and C. Talcott.
On the formalization and computational complexity of resilience problems for cyber-physical systems.
In *Theoretical Aspects of Computing–ICTAC 2022: 19th International Colloquium, Tbilisi, Georgia, September 27–29, 2022, Proceedings*, pages 96–113. Springer, 2022.

R. Bloomfield, G. Fletcher, H. Khlaaf, P. Ryan, S. Kinoshita, Y. Kinoshit, M. Takeyama, Y. Matsubara, P. Popov, K. Imai, et al.
Towards identifying and closing gaps in assurance of autonomous road vehicles–a collection of technical notes part 1.
arXiv preprint arXiv:2003.00789, 2020.

M. Vardi.
Efficiency vs. resilience: What covid-19 teaches computing.
Communications of the ACM, 63(5):9–9, 2020.