Monitoring Arithmetic Temporal Properties on Finite Traces

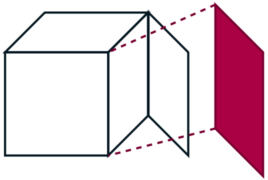
Paolo Felli,[1] Marco Montali,[2] Fabio Patrizi,[3] Sarah Winkler[2]

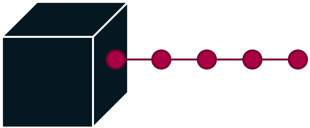
[1] University of Bologna, Italy
[2] Free University of Bozen–Bolzano, Italy
[3] Sapienza University of Rome, Italy

37th AAAI Conference on Artificial Intelligence
10 February 2023, Washington, DC

unibz

Checking properties of dynamic systems





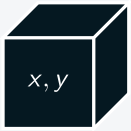
- system **fully known**, **specification** available
- analyze **all** executions, or all execution trees

analysis task:
model checking

- system **unknown**, or properties **inaccessible**
- analyze **running execution** and its possible continuations

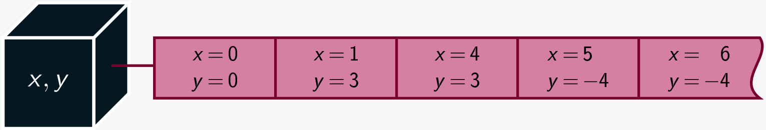
analysis task:
monitoring

Overview



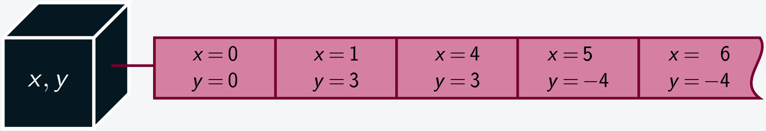
- can access finite set of numeric **process variables** V

Overview



- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V

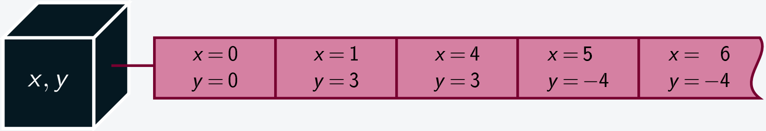
Overview



- $\boxed{\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))}$

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL**$_f$)

Overview

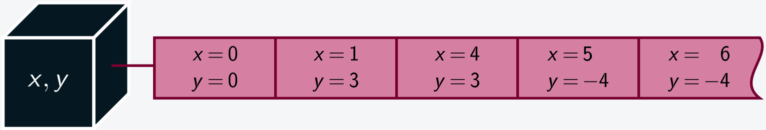


- $\boxed{\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))}$

“y is nonnegative until x is always greater than y”

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL**$_f$)

Overview

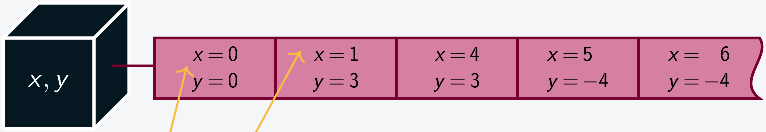


- $\boxed{\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))}$
- $\boxed{\psi_2 = \mathsf{G}(x < x') \wedge \mathsf{F}(x = 2)}$

x' is value of x looking one trace instant ahead

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL**$_f$)
  variables can have **lookahead** to refer to future values

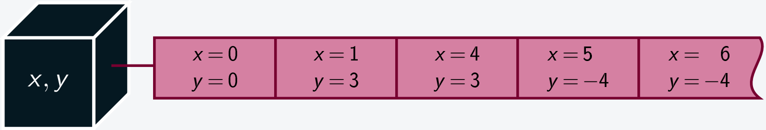
Overview



- $\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))$
- $\psi_2 = \mathsf{G}(x < x') \wedge \mathsf{F}(x = 2)$

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL**$_f$)
  variables can have **lookahead** to refer to future values

Overview

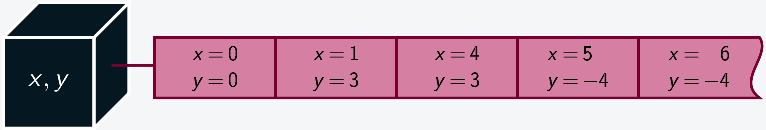


- $\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))$
- $\psi_2 = \mathsf{G}(x < x') \wedge \mathsf{F}(x = 2)$

"the current value of x is always less than the next one, and at some point x has value 2"

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL$_f$**)
  variables can have **lookahead** to refer to future values

Overview

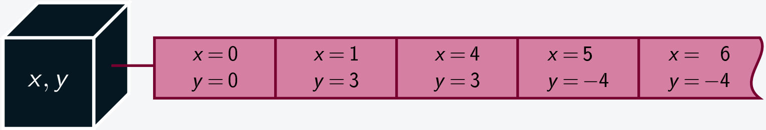


- $\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))$
- $\psi_2 = \mathsf{G}(x < x') \wedge \mathsf{F}(x = 2)$

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL$_f$**)
  variables can have **lookahead** to refer to future values
- **anticipatory monitoring**: determine current and future satisfaction

Overview

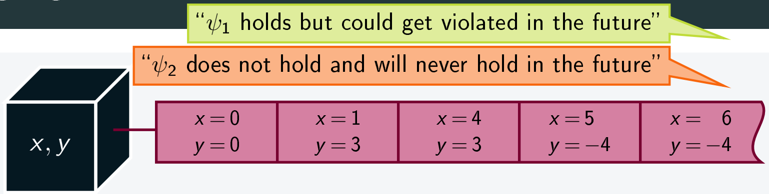
"ψ_1 holds but could get violated in the future"



- $\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))$
- $\psi_2 = \mathsf{G}(x < x') \wedge \mathsf{F}(x = 2)$

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL**$_f$)
  variables can have **lookahead** to refer to future values
- **anticipatory monitoring**: determine current and future satisfaction

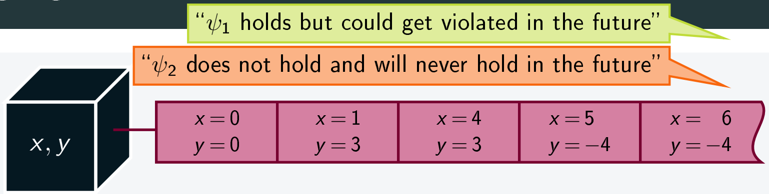
Overview



- $\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))$
- $\psi_2 = \mathsf{G}(x < x') \wedge \mathsf{F}(x = 2)$

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL**$_f$)
  variables can have **lookahead** to refer to future values
- **anticipatory monitoring**: determine current and future satisfaction

Overview



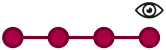
- $\psi_1 = (y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))$
- $\psi_2 = \mathsf{G}(x < x') \wedge \mathsf{F}(x = 2)$

- can access finite set of numeric **process variables** V
- **trace** is finite sequence of assignments to V
- **linear-time property** ψ with linear **arithmetic constraints** (**ALTL**$_f$)
  variables can have **lookahead** to refer to future values
- **anticipatory monitoring**: determine current and future satisfaction
- what is **decidable/solvable**? how to construct **monitors**?

Anticipatory monitoring task

given trace and ALTL_f property, determine monitoring state [BLS2010]:

ps: permanent satisfaction

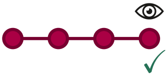


A. Bauer, M. Leucker, and C. Schallhart: Comparing LTL Semantics for Runtime Verification. J. Logic and Comput., 20(3): 651–674, 2010.

Anticipatory monitoring task

given trace and ALTL_f property, determine monitoring state [BLS2010]:

ps: permanent satisfaction

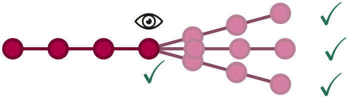


A. Bauer, M. Leucker, and C. Schallhart: Comparing LTL Semantics for Runtime Verification. J. Logic and Comput., 20(3): 651–674, 2010.

Anticipatory monitoring task

given trace and ALTL_f property, determine monitoring state [BLS2010]:

ps: permanent satisfaction

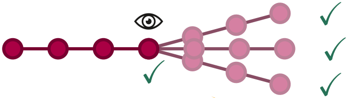


A. Bauer, M. Leucker, and C. Schallhart: Comparing LTL Semantics for Runtime Verification. J. Logic and Comput., 20(3): 651–674, 2010.

Anticipatory monitoring task

given trace and ALTL$_f$ property, determine monitoring state [BLS2010]:

ps: permanent satisfaction



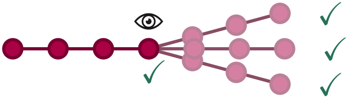
consider all finite continuations of unbounded length

A. Bauer, M. Leucker, and C. Schallhart: Comparing LTL Semantics for Runtime Verification. J. Logic and Comput., 20(3): 651–674, 2010.

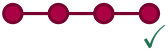
Anticipatory monitoring task

given trace and ALTL_f property, determine monitoring state [BLS2010]:

ps: permanent satisfaction



cs: current satisfaction

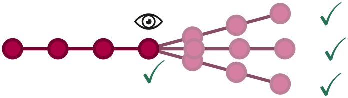


A. Bauer, M. Leucker, and C. Schallhart: Comparing LTL Semantics for Runtime Verification. J. Logic and Comput., 20(3): 651–674, 2010.

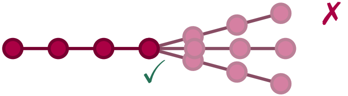
Anticipatory monitoring task

given trace and ALTL_f property, determine monitoring state [BLS2010]:

ps: permanent satisfaction



cs: current satisfaction



A. Bauer, M. Leucker, and C. Schallhart: Comparing LTL Semantics for Runtime Verification. J. Logic and Comput., 20(3): 651–674, 2010.

Anticipatory monitoring task

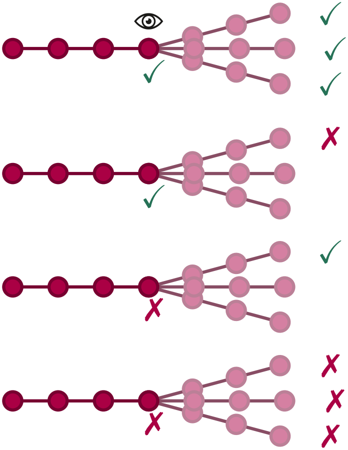
given trace and ALTL_f property, determine monitoring state [BLS2010]:

ps: permanent satisfaction

cs: current satisfaction

cv: current violation

pv: permanent violation



A. Bauer, M. Leucker, and C. Schallhart: Comparing LTL Semantics for Runtime Verification. J. Logic and Comput., 20(3): 651–674, 2010.

Anticipatory monitoring task

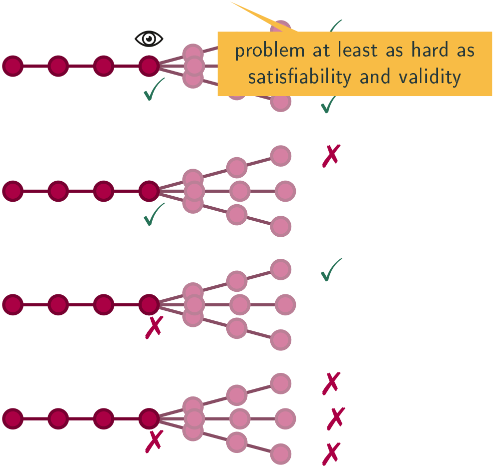
given trace and ALTL$_f$ property, determine monitoring state [BLS2010]:

ps: permanent satisfaction

cs: current satisfaction

cv: current violation

pv: permanent violation



A. Bauer, M. Leucker, and C. Schallhart: Comparing LTL Semantics for Runtime Verification. J. Logic and Comput., 20(3): 651–674, 2010.

Monitoring without lookahead

Theorem

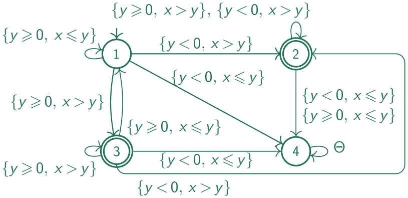
monitoring of lookahead-free properties is solvable

Theorem

monitoring of lookahead-free properties is solvable

Example

- construct DFA for $\boxed{(y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))}$, treating constraints as propositions



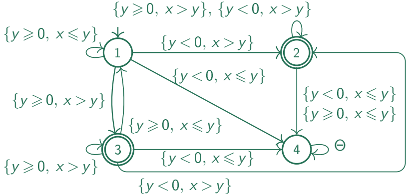
Monitoring without lookahead

Theorem

monitoring of lookahead-free properties is solvable

Example

- construct DFA for $\boxed{(y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))}$, treating constraints as propositions



- every trace prefix leads to unique DFA state

$x = 0$ $y = 0$	$x = 1$ $y = 3$	$x = 4$ $y = 3$	$x = 5$ $y = 4$	$x = 6$ $y = -4$

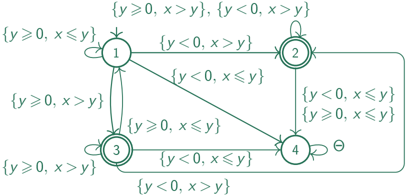
A A C C B

Theorem

monitoring of lookahead-free properties is solvable

Example

- construct DFA for $\boxed{(y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))}$, treating constraints as propositions



- every trace prefix leads to unique DFA state

$x=0$, $y=0$	$x=1$, $y=3$	$x=4$, $y=3$	$x=5$, $y=4$	$x=6$, $y=-4$
A **cv**	A **cv**	C **cs**	C **cs**	B **cs**

Monitoring without lookahead

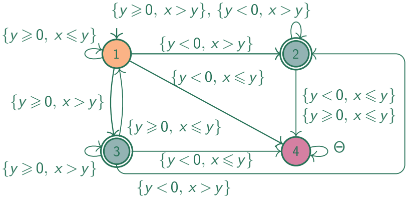
every DFA state q corresponds to unique monitoring state

Theorem

monitoring of lookahead-free properties is solvable: DFAs serve as monitors

Example

- construct DFA for $(y \geqslant 0)\ \mathsf{U}\ (\mathsf{G}(x > y))$, treating constraints as propositions



- every trace prefix leads to unique DFA state

$x=0$ $y=0$	$x=1$ $y=3$	$x=4$ $y=3$	$x=5$ $y=4$	$x=6$ $y=-4$
A **cv**	A **cv**	C **cs**	C **cs**	B **cs**

Monitoring without lookahead

every DFA state q corresponds to unique monitoring state

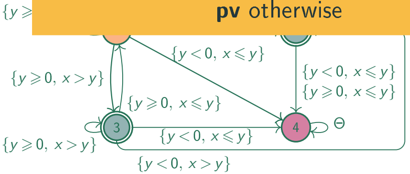
Theorem

monitoring of lookahead-free properties is solvable: DFAs serve as monitors

Example

- construct DFA for $(y$ … s

if q final:	**cs** if non-final state reachable from q	
	ps otherwise	
if q not final:	**cv** if final state reachable from q	
	pv otherwise	

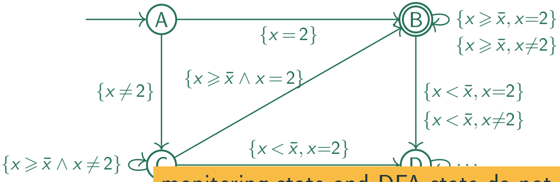


- every trace prefix leads to unique DFA state

$x=0$, $y=0$	$x=1$, $y=3$	$x=4$, $y=3$	$x=5$, $y=4$	$x=6$, $y=-4$
A **cv**	A **cv**	C **cs**	C **cs**	B **cs**

Example (DFAs are not monitors)

- DFAs construction for $\boxed{\mathsf{G}(x' > x) \wedge \mathsf{F}(x=2)}$



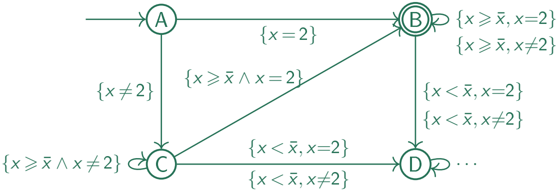
monitoring state and DFA state do not correspond

- sequence of monitoring states and DFA states

$x=0$, $y=0$	$x=1$, $y=3$	$x=4$, $y=3$	$x=5$, $y=4$	$x=6$, $y=-4$
cv C	**cv** C	**pv** C	**pv** C	**pv** C

Example (DFAs are not monitors)

- DFAs construction for $\boxed{\mathsf{G}(x' > x) \wedge \mathsf{F}(x = 2)}$



- sequence of monitoring states and DFA states

$x=0$	$x=1$	$x=4$	$x=5$	$x=6$
$y=0$	$y=3$	$y=3$	$y=4$	$y=-4$
cv C	**cv** C	**pv** C	**pv** C	**pv** C

Fact

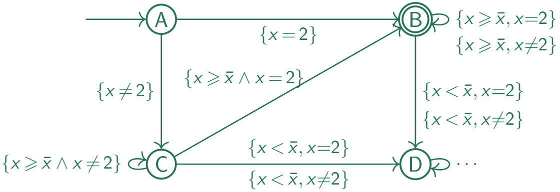
Monitoring with lookahead is not solvable: reduction from reachability in 2CM

Monitoring with lookahead is not solvable

problem: state reachability depends on assignment

Example (DFAs are not monitors)

- DFAs construction for $\boxed{\mathsf{G}(x' > x) \wedge \mathsf{F}(x = 2)}$



- sequence of monitoring states and DFA states

$x = 0$, $y = 0$	$x = 1$, $y = 3$	$x = 4$, $y = 3$	$x = 5$, $y = 4$	$x = 6$, $y = -4$
cv C	**cv** C	**pv** C	**pv** C	**pv** C

Fact

Monitoring with lookahead is not solvable: reduction from reachability in 2CM

Aim

given DFA state q reached by trace τ, find **condition** whether final DFA state is reachable from q after τ

Aim

given DFA state q reached by trace τ, find **condition** whether final DFA state is reachable from q after τ

Approach: Symbolic finite state abstraction

- **history constraints** are constraints accumulated along paths in DFA:

$$h(\mathsf{A} \to \mathsf{C}) = (x = x_0 \wedge x \neq 2)$$
$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C}) = \exists x_1.\ (x_1 = x_0 \wedge x_1 \neq 2) \wedge (x \geqslant x_1 \wedge x \neq 2)$$
$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C} \to \mathsf{C}) = \exists x_1\, x_2.\ \cdots \wedge (x \geqslant x_2 \wedge x \neq 2)$$

Aim

given DFA state q reached by trace τ, find **condition** whether final DFA state is reachable from q after τ

Approach: Symbolic finite state abstraction

- **history constraints** are constraints accumulated along paths in DFA:

$$h(\mathsf{A} \to \mathsf{C}) = (x = x_0 \wedge x \neq 2)$$

$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C}) = \exists x_1.\ (x_1 = x_0 \wedge x_1 \neq 2) \wedge (x \geqslant x_1 \wedge x \neq 2)$$

$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C} \to \mathsf{C}) = \exists x_1\, x_2.\ \cdots \wedge (x \geqslant x_2 \wedge x \neq 2)$$

Aim

given DFA state q reached by trace τ, find **condition** whether final DFA state is reachable from q after τ

Approach: Symbolic finite state abstraction

by quantifier elimination

- **history constraints** are constraints accumulated along paths in DFA:

$$h(\mathsf{A} \to \mathsf{C}) = (x = x_0 \wedge x \neq 2)$$

$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C}) = \exists x_1.\ (x_1 = x_0 \wedge x_1 \neq 2) \wedge (x \geqslant x_1 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C} \to \mathsf{C}) = \exists x_1\, x_2.\ \cdots \wedge (x \geqslant x_2 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

Aim

given DFA state q reached by trace τ, find **condition** whether final DFA state is reachable from q after τ

Approach: Symbolic finite state abstraction

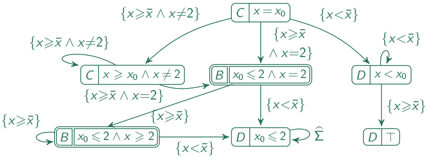
- **history constraints** are constraints accumulated along paths in DFA:

$$h(\mathsf{A} \to \mathsf{C}) = (x = x_0 \wedge x \neq 2)$$

$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C}) = \exists x_1.\ (x_1 = x_0 \wedge x_1 \neq 2) \wedge (x \geqslant x_1 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C} \to \mathsf{C}) = \exists x_1\, x_2.\ \cdots \wedge (x \geqslant x_2 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

- **constraint graph** CG(q) represents history constraints for all paths from q



Aim

given DFA state q reached by trace τ, find **condition** whether final DFA state is reachable from q after τ

Approach: Symbolic finite state abstraction

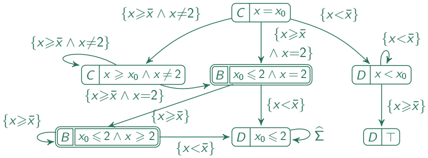
- **history constraints** are constraints accumulated along paths in DFA:

$$h(\mathrm{A} \to \mathrm{C}) = (x = x_0 \wedge x \neq 2)$$

$$h(\mathrm{A} \to \mathrm{C} \to \mathrm{C}) = \exists x_1.\ (x_1 = x_0 \wedge x_1 \neq 2) \wedge (x \geqslant x_1 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

$$h(\mathrm{A} \to \mathrm{C} \to \mathrm{C} \to \mathrm{C}) = \exists x_1\, x_2.\ \cdots \wedge (x \geqslant x_2 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

- **constraint graph** $\mathrm{CG}(q)$ represents history constraints for all paths from q



can be infinite

Aim

given DFA state q reached by trace τ, find **condition** whether final DFA state is reachable from q after τ

Approach: Symbolic finite state abstraction

- **history constraints** are constraints accumulated along paths in DFA:

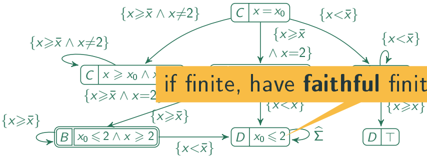
$$h(\mathsf{A} \to \mathsf{C}) = (x = x_0 \wedge x \neq 2)$$

$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C}) = \exists x_1.\ (x_1 = x_0 \wedge x_1 \neq 2) \wedge (x \geqslant x_1 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C} \to \mathsf{C}) = \exists x_1\, x_2.\ \cdots \wedge (x \geqslant x_2 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

- **constraint graph** $\mathsf{CG}(q)$ represents history constraints for all paths from q

can be infinite



if finite, have **faithful** finite state abstraction

Aim

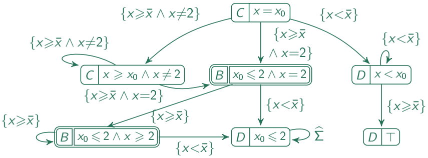
given DFA state q reached by trace τ, find **condition** whether final DFA state is reachable from q after τ

Approach: Symbolic finite state abstraction

- **history constraints** are constraints accumulated along paths in DFA:

$$h(\mathsf{A} \to \mathsf{C}) = (x = x_0 \wedge x \neq 2)$$
$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C}) = \exists x_1.\ (x_1 = x_0 \wedge x_1 \neq 2) \wedge (x \geqslant x_1 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$
$$h(\mathsf{A} \to \mathsf{C} \to \mathsf{C} \to \mathsf{C}) = \exists x_1\, x_2.\ \cdots \wedge (x \geqslant x_2 \wedge x \neq 2) \qquad \equiv x_0 \neq 2 \wedge x \geqslant x_0$$

- **constraint graph** $\mathsf{CG}(q)$ represents history constraints for all paths from q



- formulas in final nodes of CG give **condition** for reachability of final DFA states: captured by $\mathsf{FSat}(\mathsf{CG}(q))$ (similarly $\mathsf{FUns}(\mathsf{CG}(q))$)

Monitoring procedure

all monitoring structures can be computed upfront
(DFA, CGs, FSat, FUns)

procedure $\textsc{Monitor}(\psi, \tau)$
  compute DFA for ψ
  $w \leftarrow$ word over constraints consistent with τ
  $q \leftarrow$ DFA state in such that $\{q_0\} \rightarrow^*_w q$
  $\alpha \leftarrow$ last assignment in τ
  if q accepting in DFA **then**
    return (**cs if** $\alpha \models$ FUns(CG(q)) **else ps**)
  else return (**cv if** $\alpha \models$ FSat(CG(q)) **else pv**)

Monitoring procedure

all monitoring structures can be computed upfront (DFA, CGs, FSat, FUns)

```
1: procedure MONITOR(ψ, τ)
2:    compute DFA for ψ
3:    w ← word over constraints consistent with τ
4:    q ← DFA state in such that {q0} →*_w q
5:    α ← last assignment in τ
6:    if q accepting in DFA then
7:       return (cs if α ⊨ FUns(CG(q)) else ps)
8:    else return (cv if α ⊨ FSat(CG(q)) else pv)
```

Theorem (Correctness)

if $\textsc{monitor}(\psi, \tau) = s$ *then* s *is monitoring state for* ψ *and* τ

all monitoring structures can be computed upfront
(DFA, CGs, FSat, FUns)

procedure $\textsc{Monitor}(\psi, \tau)$
  compute DFA for ψ
  $w \leftarrow$ word over constraints consistent with τ
  $q \leftarrow$ DFA state in such that $\{q_0\} \rightarrow^*_w q$
  $\alpha \leftarrow$ last assignment in τ
  if q accepting in DFA **then**
    return (**cs if** $\alpha \models$ FUns(CG(q)) **else ps**)
  else return (**cv if** $\alpha \models$ FSat(CG(q)) **else pv**)

does not terminate if CGs infinite

Theorem (Correctness)

if $\textsc{Monitor}(\psi, \tau) = s$ *then* s *is monitoring state for* ψ *and* τ

previously used in context of model checking [FMW22]

Definition (Finite summary)

property ψ has finite summary if paths in DFA for ψ are covered by finitely many history constraints

[FMW22] P. Felli, M. Montali, S. Winkler. Linear-time verification of data-aware dynamic systems with arithmetic. AAAI-36(5), 5642-5650, 2022

Abstract solvability criterion

previously used in context of model checking [FMW22]

Definition (Finite summary)

property ψ has finite summary if paths in DFA for ψ
are covered by finitely many history constraints

Observation

for properties with finite summary, constraint graphs are finite

[FMW22] P. Felli, M. Montali, S. Winkler. Linear-time verification of data-aware dynamic systems with arithmetic. AAAI-36(5), 5642-5650, 2022

Abstract solvability criterion

previously used in context of model checking [FMW22]

Definition (Finite summary)
property ψ has finite summary if paths in DFA for ψ are covered by finitely many history constraints

Observation
for properties with finite summary, constraint graphs are finite

Theorem
monitoring task is solvable for any ψ that has finite summary, and MONITOR *is monitoring procedure*

[FMW22] P. Felli, M. Montali, S. Winkler. Linear-time verification of data-aware dynamic systems with arithmetic. AAAI-36(5), 5642-5650, 2022

Concrete solvable property classes

Property classes that enjoy finite summary

- **monotonicity constraint** properties over $\mathbb{Q}$ or $\mathbb{Z}$ $\mathsf{G}(x' > x) \wedge \mathsf{F}(x = 2)$
  (all constraints are variable-to-variable/constant comparisons)

S. Demri and D. D'Souza: An automata-theoretic approach to constraint LTL. Inform. Comput., 205(3): 380-415, 2007.

Concrete solvable property classes

Property classes that enjoy finite summary

- **monotonicity constraint** properties over $\mathbb{Q}$ or $\mathbb{Z}$ $\mathsf{G}(x' > x) \land \mathsf{F}(x = 2)$
  (all constraints are variable-to-variable/constant comparisons)
- **integer periodicity constraint** properties $\mathsf{F}(x' > 3) \land \mathsf{G}(x \equiv_7 2)$
  (variable-to-variable/constant comparisons with modulo operator)

S. Demri: LTL over integer periodicity constraints. Theor. Comput. Sci., 360(1-3): 96–123, 2006.

Concrete solvable property classes

Property classes that enjoy finite summary

- **monotonicity constraint** properties over $\mathbb{Q}$ or $\mathbb{Z}$ $\mathsf{G}(x' > x) \wedge \mathsf{F}(x = 2)$
  (all constraints are variable-to-variable/constant comparisons)
- **integer periodicity constraint** properties $\mathsf{F}(x' > 3) \wedge \mathsf{G}(x \equiv_7 2)$
  (variable-to-variable/constant comparisons with modulo operator)
- **bounded lookback** properties $\mathsf{F}(x' > y) \wedge \mathsf{G}(x + z = 7)$
  (restriction on interaction of constraints via lookahead, generalizes feedback freedom)

E. Damaggio, A. Deutsch and V. Vianu: Artifact systems with data dependencies and arithmetic. ACM Trans. Database Syst., 37(3): 22:1–22:36, 2012

Concrete solvable property classes

Property classes that enjoy finite summary

- **monotonicity constraint** properties over $\mathbb{Q}$ or $\mathbb{Z}$ $\quad \mathsf{G}(x' > x) \wedge \mathsf{F}(x = 2)$
  (all constraints are variable-to-variable/constant comparisons)
- **integer periodicity constraint** properties $\quad \mathsf{F}(x' > 3) \wedge \mathsf{G}(x \equiv_7 2)$
  (variable-to-variable/constant comparisons with modulo operator)
- **bounded lookback** properties $\quad \mathsf{F}(x' > y) \wedge \mathsf{G}(x + z = 7)$
  (restriction on interaction of constraints via lookahead, generalizes feedback freedom)

Non-solvable class

- **gap-order** properties $\quad \mathsf{G}(x' - y \geqslant 3) \wedge \mathsf{F}(x - z' \geqslant 2)$
  (all constraints are gap-order comparisons)

L. Bozzelli and S. Pinchinat: Verification of gap- order constraint abstractions of counter systems. Theor. Comput. Sci., 523: 1–36, 2014

Concrete solvable property classes

Property classes that enjoy finite summary

- **monotonicity constraint** properties over $\mathbb{Q}$ or $\mathbb{Z}$ $\quad$ $\mathsf{G}(x' > x) \wedge \mathsf{F}(x = 2)$
  (all constraints are variable-to-variable/constant comparisons)
- **integer periodicity constraint** properties $\quad$ $\mathsf{F}(x' > 3) \wedge \mathsf{G}(x \equiv_7 2)$
  (variable-to-variable/constant comparisons with modulo operator)
- **bounded lookback** properties $\quad$ $\mathsf{F}(x' > y) \wedge \mathsf{G}(x + z = 7)$
  (restriction on interaction of constraints via lookahead, generalizes feedback freedom)

Non-solvable class

model checking is decidable

- **gap-order** properties $\quad$ $\mathsf{G}(x' - y \geqslant 3) \wedge \mathsf{F}(x - z' \geqslant 2)$
  (all constraints are gap-order comparisons)

L. Bozzelli and S. Pinchinat: Verification of gap- order constraint abstractions of counter systems. Theor. Comput. Sci., 523: 1–36, 2014

Summary

1. ALTL$_f$ monitoring with linear arithmetic constraints:

   without lookahead: solvable (DFA construction for monitors)

   with lookahead: not solvable

Summary

1. ALTL$_f$ monitoring with linear arithmetic constraints:
   without lookahead: solvable (DFA construction for monitors)
   with lookahead: not solvable
2. general monitoring procedure for lookahead properties:
   terminates for finite summary properties

Summary

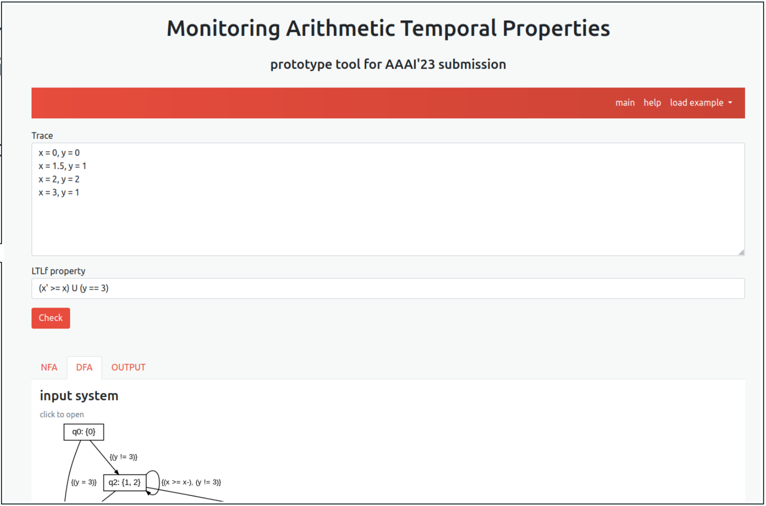
1. ALTL_f monitoring with linear arithmetic constraints:
   without lookahead: solvable (DFA construction for monitors)
   with lookahead: not solvable
2. general monitoring procedure for lookahead properties:
   terminates for finite summary properties
3. solvability for several practical classes of formulae:
   monotonicity and integer periodicity constraints, bounded lookback

Summary

1. ALTL_f monitoring with linear arithmetic constraints:
   without lookahead: solvable (DFA construction for monitors)
   with lookahead: not solvable
2. general monitoring procedure for lookahead properties:
   terminates for finite summary properties
3. solvability for several practical classes of formulae:
   monotonicity and integer periodicity constraints, bounded lookback
4. SMT-based prototype ada witnesses feasibility of approach

Summary

1. ALTL_f monitoring with linear arithmetic constraints:
   without lookahead: solvable (DFA construction for monitors)
   with lookahead: not solvable
2. gener
   termi
3. solva
   monc
4. SMT



Summary

1. ALTL_f monitoring with linear arithmetic constraints:
   without lookahead: solvable (DFA construction for monitors)
   with lookahead: not solvable
2. general monitoring procedure for lookahead properties:
   terminates for finite summary properties
3. solvability for several practical classes of formulae:
   monotonicity and integer periodicity constraints, bounded lookback
4. SMT-based prototype ada witnesses feasibility of approach

Future work

- lift approach to richer properties equipped with full-fledged relations
- possibly study more general, controlled first-order quantification across time