Formal Methods for Timed and Probabilistic Systems

Ocan Sankur

CNRS, Université de Rennes

Habilitation Thesis Defense
October 2, 2023

Manuscript: https://people.irisa.fr/Ocan.Sankur/hdr.pdf

Model Checking

Exhaustive verification of programs or abstract models

```
module sched(input clk,
             input controllable_sched0,
             input controllable_sched1,
             input startA,
             input startB,
             input tick,
             output error,
             output _rt_startA,
             output _rt_startB,
             output _rt_tick
                     );
    reg[2:0] m1_counter;
    reg notfirst;
```

Model Checker

✓

×

Spec: output error is always 0

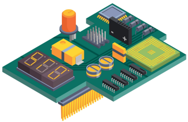
✓ Under **all** sequences of inputs, the module never raises error

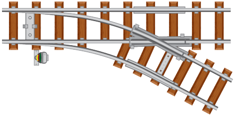
× Under **some** input sequence, the module sets error to 1

(counterexample often provided)

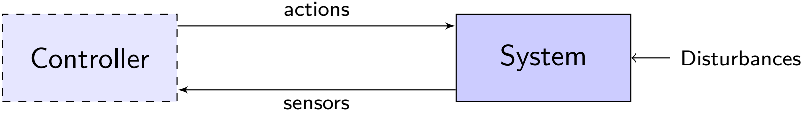
Model Checking: Some Uses

- **Hardware**: *cache coherence protocols, file transfer protocols, PCI etc.*
- **Distributed** algorithms: *consensus, leader election*
- **Railway** signaling systems



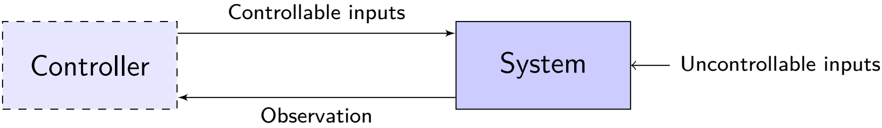


Controller Synthesis

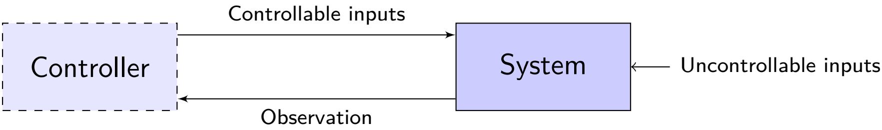


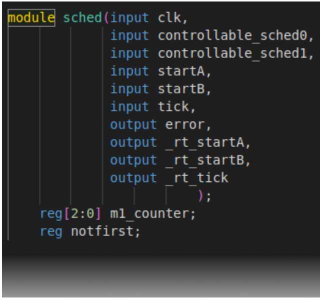
Goal: Given a system (model), automatically compute a controller to ensure the system always has desired behavior.

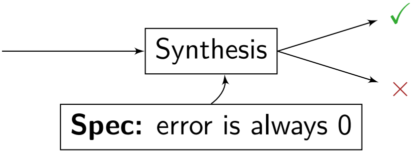
Controller Synthesis



Controller Synthesis





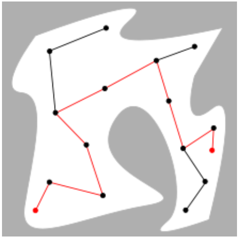


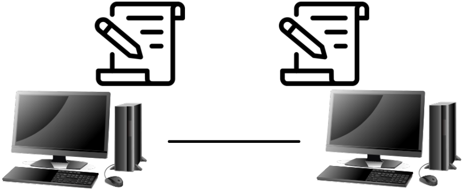
✓ There is a controller which, under **all** sequences of uncontrollable inputs, prescribes a controllable input such that the module never raises error

× For all controllers, under **some** input sequence, error is raised

Controller Synthesis: Uses

- Automatic construction / completion of **protocols**
- Solving **planning** problems under uncertainties
- **Requirements** verification





Many techniques and applications are common to model checking and synthesis

Timed Systems

Systems whose behaviors depend on the correct timings of events.

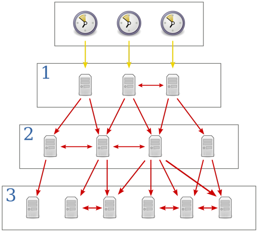
Models in which execution times and deadlines are **explicitly** represented.

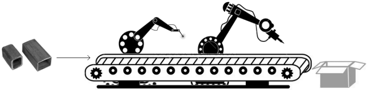
Timed Systems

Systems whose behaviors depend on the correct timings of events.

Models in which execution times and deadlines are **explicitly** represented.

- Clock synchronization protocols
- Embedded programs under a real-time scheduling policy
- Cyber physical systems: *e.g.* a program interacting with a physical environment





Probabilistic Systems

Systems whose behaviors can be understood as being generated by stochastic processes.

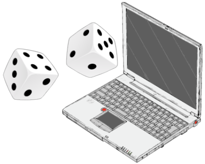
e.g. behaviors follow probability distributions.

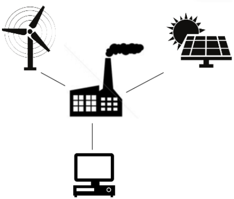
Probabilistic Systems

Systems whose behaviors can be understood as being generated by stochastic processes.

e.g. behaviors follow probability distributions.

- **Randomized** algorithms
- Deterministic programs within a **stochastic environment**:
  e.g. disturbances, random user behavior, weather conditions.





This Thesis

Quantitative features such as **time** and **probabilities** are fundamental in several applications of formal methods.

Model checking and **synthesis** algorithms for timed and probabilistic systems

This Thesis

Quantitative features such as **time** and **probabilities** are fundamental in several applications of formal methods.

Model checking and **synthesis** algorithms for timed and probabilistic systems

Overview

1. Algorithms for timed systems
2. Algorithms for probabilistic systems
3. Other Works
4. Conclusion

Outline

1. Algorithms for Timed Systems
   - Model Checking for Large Timed Automata
   - Robustness for Timed Automata

2. Probabilistic Systems
   - Stochastic Shortest Paths

Models with Explicit Timings: Timed Automata

Timed automata: Finite automaton + clock variables:

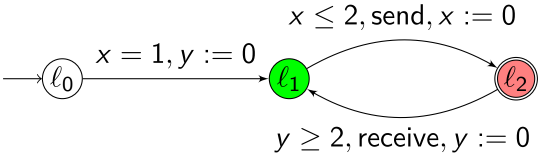
- Locations (ℓ_0, ℓ_1, ℓ_2): finite-state program
- Clocks (x, y): time constraints [Alur, Dill 1994]

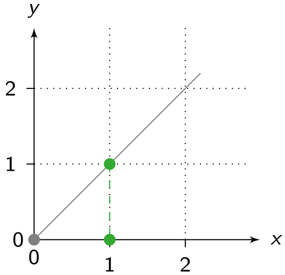
Models with Explicit Timings: Timed Automata

Timed automata: Finite automaton + clock variables:

- Locations (ℓ_0, ℓ_1, ℓ_2): finite-state program
- Clocks (x, y): time constraints

[Alur, Dill 1994]



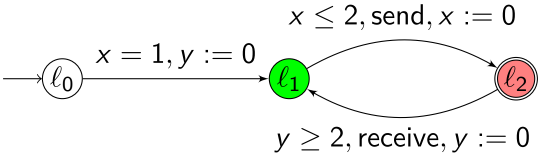


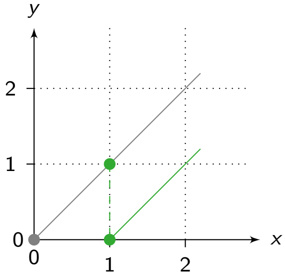
Models with Explicit Timings: Timed Automata

Timed automata: Finite automaton + clock variables:

- Locations (ℓ_0, ℓ_1, ℓ_2): finite-state program
- Clocks (x, y): time constraints

[Alur, Dill 1994]

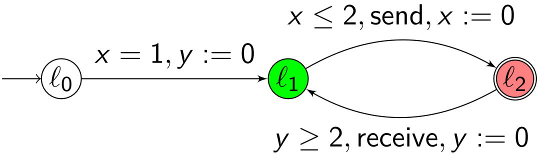


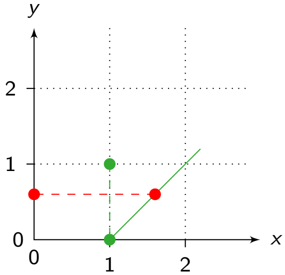


Timed automata: Finite automaton + clock variables:

- Locations (ℓ_0, ℓ_1, ℓ_2): finite-state program
- Clocks (x, y): time constraints

[Alur, Dill 1994]

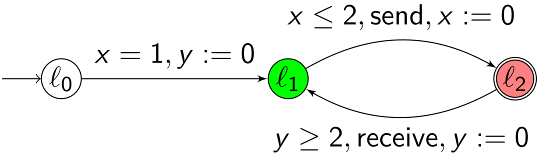


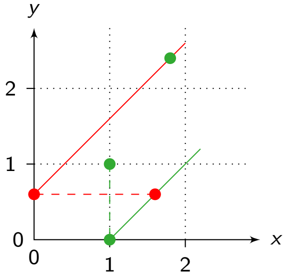


Timed automata: Finite automaton + clock variables:

- Locations (ℓ_0, ℓ_1, ℓ_2): finite-state program
- Clocks (x, y): time constraints

[Alur, Dill 1994]

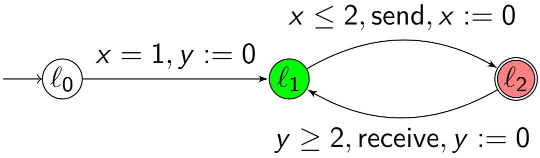


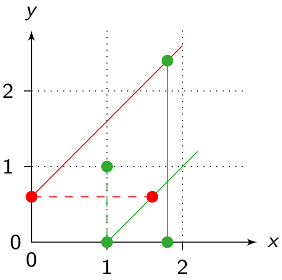


Models with Explicit Timings: Timed Automata

Timed automata: Finite automaton + clock variables:

- Locations (ℓ_0, ℓ_1, ℓ_2): finite-state program
- Clocks (x, y): time constraints [Alur, Dill 1994]





Symbolic State Space Representation: Zones

- Explicit enumeration is no longer possible

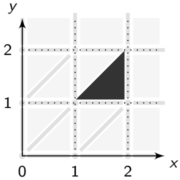
$$(\ell_2, x = 1.500, y = 0.500), (\ell_2, x = 1.501, y = 0.501), (\ell_2, x = 1.502, y = 0.502), \ldots$$

Symbolic State Space Representation: Zones

- Explicit enumeration is no longer possible

In timed automata, it suffices to enumerate **zones**:

[Henzinger, Nicollin, Sifakis, Yovine 1994]

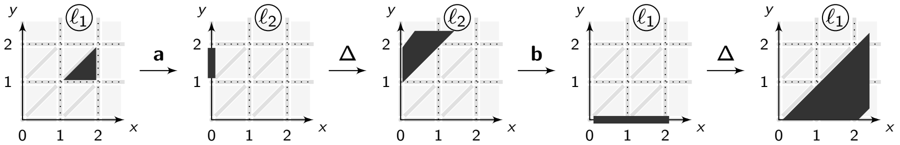


Symbolic State Space Representation: Zones

- Explicit enumeration is no longer possible

In timed automata, it suffices to enumerate **zones**:

[Henzinger, Nicollin, Sifakis, Yovine 1994]



Termination and good efficiency thanks to **sound and complete** abstraction operators.

Recent survey: **[FORMATS 2022]**

Zone-based model checking tools:

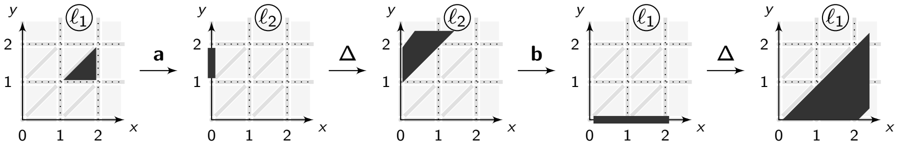
- Proprietary: Uppaal, PAT
- Free: Opaal/LTSmin, TChecker

Symbolic State Space Representation: Zones

- Explicit enumeration is no longer possible

In timed automata, it suffices to enumerate **zones**:

[Henzinger, Nicollin, Sifakis, Yovine 1994]



Termination and good efficiency thanks to **sound and complete** abstraction operators.

Recent survey: **[FORMATS 2022]**

Zone-based model checking tools:

- Proprietary: Uppaal, PAT
- Free: Opaal/LTSmin, TChecker

But they all apply explicit enumeration on discrete states

Algorithms for Timed Automata with Large Discrete State Spaces

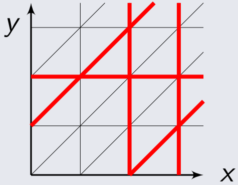
1. Algorithms based on predicate abstraction
2. Algorithm based on finite automata learning

Predicate Abstraction

- We consider an **abstract domain** defined by **predicates**:

$$x \leq k \mid x \geq k \mid x - y \leq k \qquad x, y \text{ clocks}, k \in \mathbb{Z}$$

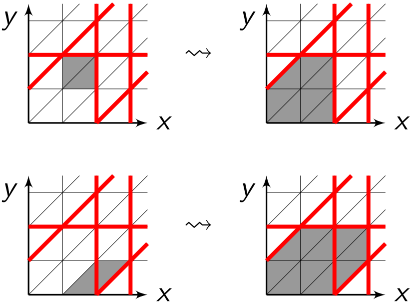
Predicates: $x - y \leq -1, \quad y \leq 2, \quad x \leq 2, \quad x \leq 3, \quad x - y \leq 2$



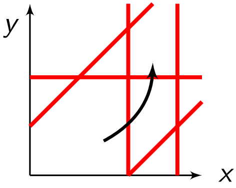
A valuation on the predicates defines a **cell** and is an **abstract state**

Abstraction Mapping

Examples



Abstract System

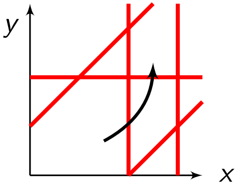


Transitions between **valuations** are replaced by transitions between cells

$$(\ell_0, x{=}3, y{=}2.7) \rightarrow (\ell_1, x{=}4.2, y{=}3.9) \rightarrow (\ell_1, x{=}4.4, y{=}4.1)$$

becomes (ℓ_0, ) $\rightarrow$ (ℓ_1, ) $\rightarrow$ (ℓ_1, )

Abstract System



Transitions between **valuations** are replaced by transitions between cells

$$(\ell_0, x{=}3, y{=}2.7) \rightarrow (\ell_1, x{=}4.2, y{=}3.9) \rightarrow (\ell_1, x{=}4.4, y{=}4.1)$$

becomes $(\ell_0,$  $) \rightarrow (\ell_1,$  $) \rightarrow (\ell_1,$  $)$

Predicate Abstraction

Algorithm

1. Select a set of **predicates** $\mathcal{P}$ e.g. constraints that appear in the guards
2. Model check abstract system with **binary decision diagrams (BDD)**
3. Refine from counterexamples using **zone interpolants**

BDDs handle large **discrete** state spaces and predicates on **clocks**

Victor Roussanaly's PhD thesis (2019), **[Roussanaly, S., Markey CAV 2019]**

Tool and experiments: https://github.com/osankur/symrob/

Using Finite Automata Learning

Cooperative Approach: combine two types of model checkers:

1. a timed automata model checker for small discrete state spaces
2. finite-state model checker for large discrete state spaces (without clocks)

Observation

Theorem

Any timed automaton can be written as

$$\mathsf{FA} \parallel \mathsf{TA}$$

where FA is a finite automaton, TA is a timed automaton.

(in such a way that, in general, FA is **large**, and TA has **1 location**)

Observation

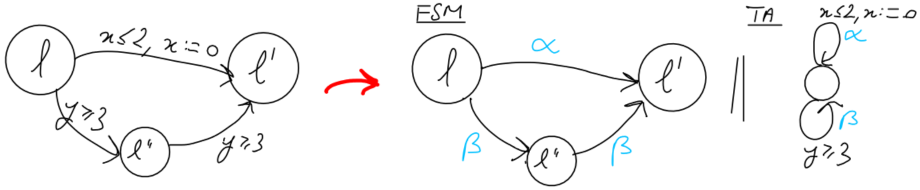
Theorem

Any timed automaton can be written as

$$\text{FA} \parallel \text{TA}$$

where FA is a finite automaton, TA is a timed automaton.

(in such a way that, in general, FA is **large**, and TA has **1 location**)



Assume-Guarantee Reasoning

Assume we want to establish

$$\mathcal{L}(\mathsf{FA} \parallel \mathsf{TA}) \subseteq \mathsf{Spec}$$

for a regular language Spec.

Assume-Guarantee Reasoning

Assume we want to establish

$$\mathcal{L}(\mathsf{FA} \parallel \mathsf{TA}) \subseteq \mathsf{Spec}$$

for a regular language Spec.

One can *guess* a finite automaton H, and apply:

$$\frac{\mathcal{L}(\mathsf{TA}) \subseteq \mathcal{L}(H) \quad \mathcal{L}(\mathsf{FA} \| H) \subseteq \mathsf{Spec}}{\mathcal{L}(\mathsf{FA} \parallel \mathsf{TA}) \subseteq \mathsf{Spec}.}\ \mathsf{Asym}$$

- First premise: timed automata model checking
- Second premise: finite-state model checking

Assume-Guarantee Reasoning

Assume we want to establish

$$\mathcal{L}(\mathsf{FA} \parallel \mathsf{TA}) \subseteq \mathsf{Spec}$$

for a regular language Spec.

One can *guess* a finite automaton H, and apply:

$$\frac{\mathcal{L}(\mathsf{TA}) \subseteq \mathcal{L}(H) \quad \mathcal{L}(\mathsf{FA} \| H) \subseteq \mathsf{Spec}}{\mathcal{L}(\mathsf{FA} \parallel \mathsf{TA}) \subseteq \mathsf{Spec}.} \text{ Asym}$$

- First premise: timed automata model checking
- Second premise: finite-state model checking
- Finding H: Use automata learning (e.g. L^*) to find appropriate H

→ application of *assume-guarantee model checking with automatic learning of assumptions* [Cobleigh, Giannakopoulou, Pasareanu, TACAS 2003]

[S. TACAS 2023]
Tool and experiments: https://github.com/osankur/compRTMC

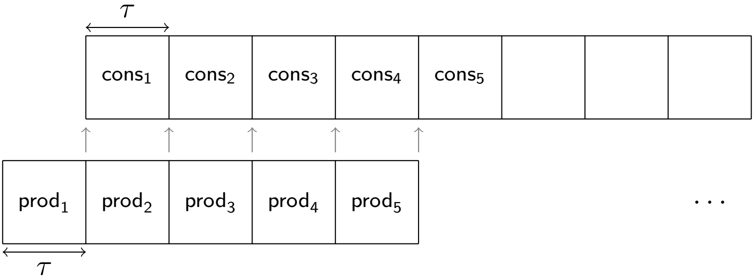
Timing Imprecisions in Timed Automata

1. Robustness Analysis
2. Robust Control

Clock Imprecisions in Timed Automata

Consider periodic tasks with specification:

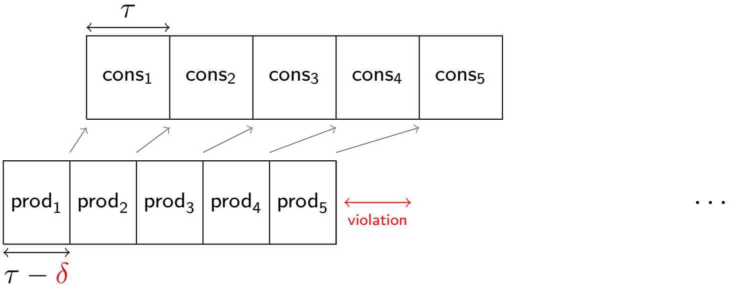
> "**cons**ume$_i$ must start at most 1 second after the end of **prod**uce$_i$."



Clock Imprecisions in Timed Automata

Consider periodic tasks with specification:

"**cons**ume$_i$ must start at most 1 second after the end of **prod**uce$_i$."

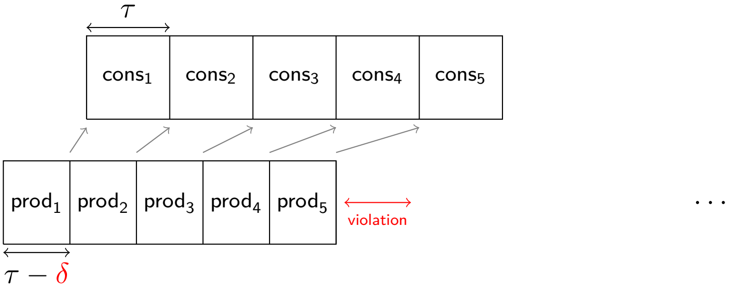


- Assume that in reality, the period of produce is shorter by $\delta = 0.1$
- **Buffer overflow** not caught by a standard analysis

Clock Imprecisions in Timed Automata

Consider periodic tasks with specification:

"**cons**ume$_i$ must start at most 1 second after the end of **prod**uce$_i$."



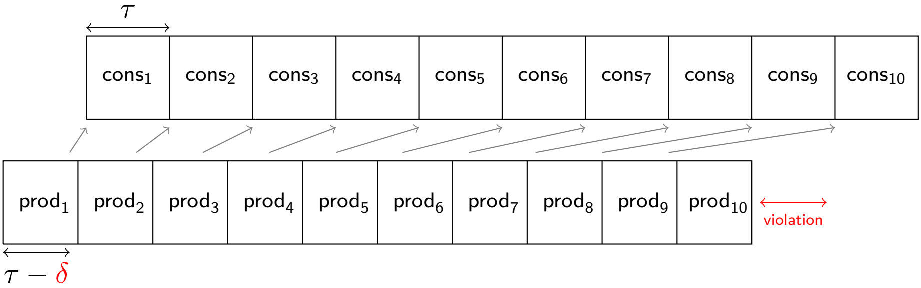
- Assume that in reality, the period of produce is shorter by $\delta = 0.1$
- **Buffer overflow** not caught by a standard analysis

Error is inevitable for all possible bounds δ

Clock Imprecisions in Timed Automata

Consider periodic tasks with specification:

"**cons**ume$_i$ must start at most 1 second after the end of **prod**uce$_i$."



- Assume that in reality, the period of produce is shorter by $\delta = 0.1$
- **Buffer overflow** not caught by a standard analysis

Error is inevitable for all possible bounds δ

Robust model checking

Imprecisions are modeled by **guard enlargement**:

e.g. $1 \leq x \leq 2 \rightsquigarrow 1 - \delta \leq x \leq 2 + \delta$.

[Puri 2000], [Doyen, De Wulf, Markey, Raskin 2008]

Robust model checking

Given timed automaton TA and specification ϕ, decide if there exists $\delta > 0$ such that $\mathrm{TA}_{+\delta} \models \phi$.

Contribution: Semi-algorithm for checking robust safety

often small overhead over exact model checking

computes bound δ

Tool and experiments: https://github.com/osankur/symrob/

[S. TACAS 2015]

Robust Controller Synthesis

Real-time planning:

- Stay at each station $[20, 40]$ sec
- Move between stations in $[60, 90]$ sec
- Complete a tour in $[420, 520]$ sec

And repeat all day!

Robust Controller Synthesis

Real-time planning:

- Stay at each station $[20, 40]$ sec
- Move between stations in $[60, 90]$ sec
- Complete a tour in $[420, 520]$ sec

And repeat all day!

Adversarial perturbations: Passenger behavior, weather conditions, etc.

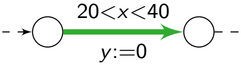
Real-time planning:

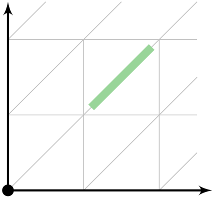
- Stay at each station [20, 40] sec
- Move between stations in [60, 90] sec
- Complete a tour in [420, 520] sec

And repeat all day!

Adversarial perturbations: Passenger behavior, weather conditions, etc.

Time Perturbation model given bound δ:





Robust Controller Synthesis

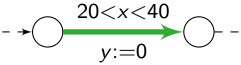
Real-time planning:

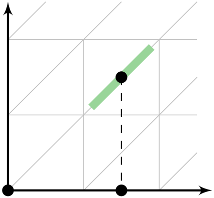
- Stay at each station [20, 40] sec
- Move between stations in [60, 90] sec
- Complete a tour in [420, 520] sec

And repeat all day!

Adversarial perturbations: Passenger behavior, weather conditions, etc.

Time Perturbation model given bound δ:





Robust Controller Synthesis

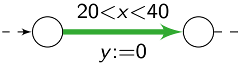
Real-time planning:

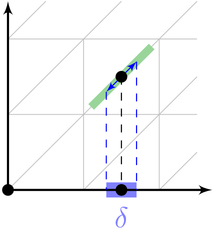
- Stay at each station $[20, 40]$ sec
- Move between stations in $[60, 90]$ sec
- Complete a tour in $[420, 520]$ sec

And repeat all day!

Adversarial perturbations: Passenger behavior, weather conditions, etc.

Time Perturbation model given bound δ:





Robust Controller Synthesis

Problem

Given timed automaton TA, and Buchi property ϕ, is there $\delta > 0$ and a control strategy σ such that σ guarantees ϕ.

[Chatterjee, Henzinger, Prabhu 2011], **[S., Bouyer, Markey, Reynier CONCUR 2013]**

Can the airtrain run for an indefinitely?

Robust Controller Synthesis

Problem

Given timed automaton TA, and Buchi property ϕ, is there $\delta > 0$ and a control strategy σ such that σ guarantees ϕ.

[Chatterjee, Henzinger, Prabhu 2011], **[S., Bouyer, Markey, Reynier CONCUR 2013]**

Can the airtrain run for an indefinitely?

A zone-based algorithm based on **double DFS** with **parametric zones** for checking robust Buchi:

This also applies to when perturbations can be stochastic and independent

[Oualhadj, Reynier, S. CONCUR 2014]

[Busatto-Gaston, Monmege, Reynier, S. CAV 2019]

Tool and experiments: https://verif.ulb.ac.be/dbusatto/

Perspectives

Target: Scalability

- Combine the knowledge developed on the **clock space** with model checking techniques for **large finite-state systems**
  - Symbolic verification techniques
  - Program verification
- Applications and benchmarks: target synchronous systems or real-time programs

Perspectives

Target: Scalability

- Combine the knowledge developed on the **clock space** with model checking techniques for **large finite-state systems**
  - Symbolic verification techniques
  - Program verification
- Applications and benchmarks: target synchronous systems or real-time programs

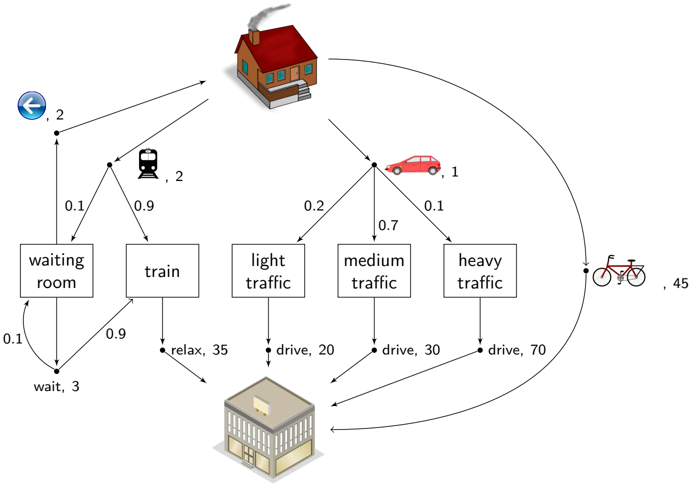
Target: Applications of Robustness

- Target new applications, case studies where a **low-level** analysis is needed
- Computing maximal perturbation bounds: slack-time analysis

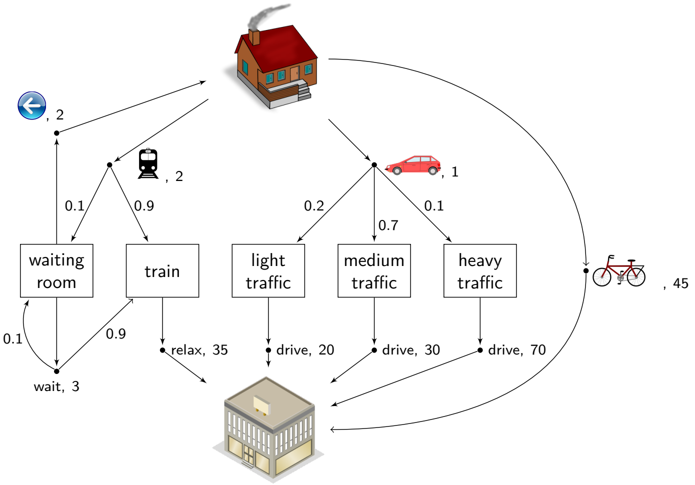
Outline

1. Algorithms for Timed Systems
   - Model Checking for Large Timed Automata
   - Robustness for Timed Automata
2. Probabilistic Systems
   - Stochastic Shortest Paths

Weighted Markov Decision Processes (MDP)



Weighted Markov Decision Processes (MDP)



What is the **shortest** path?

Strategies in MDPs

Strategy

Function that assigns an action to every history

 $\mapsto$ 

 (waiting room) $\mapsto$ wait

 (waiting room) (waiting room) $\mapsto$ 

...

Strategies in MDPs

Strategy

Function that assigns an action to every history

 $\mapsto$ 

 (waiting room) $\mapsto$ wait

 (waiting room) (waiting room) $\mapsto$ 

...

Total Payoff of a History

Sum of the weights until reaching target state T =  (or ∞)

total-payoff(  ) $= 45$

total-payoff(  (waiting room) (wait) (train) (relax) ) $= 2 + 3 + 35 = 40$

Strategies in MDPs

Strategy

Function that assigns an action to every history

Total Payoff of a History

Sum of the weights until reaching target state T =  (or ∞)

Expected Total Payoff of a strategy σ: $\mathbb{E}^{\sigma}_{M}[\mathrm{TP}(T)]$

The **stochastic shortest path** to T is the strategy with minimal expected total payoff:

$$\inf_{\sigma} \mathbb{E}^{\sigma}_{M}[\mathrm{TP}(T)]$$

Strategies in MDPs

Strategy

Function that assigns an action to every history

Total Payoff of a History

Sum of the weights until reaching target state T =  (or ∞)

Expected Total Payoff of a strategy σ: $\mathbb{E}_M^\sigma[\mathrm{TP}(T)]$

The **stochastic shortest path** to T is the strategy with minimal expected total payoff:

$$\inf_\sigma \mathbb{E}_M^\sigma[\mathrm{TP}(T)]$$

- Here,  is the "shortest path":

$\mathbb{E}_M$  $[\mathrm{TP}(T)] = 33.$

Stochastic Shortest Path with **Arbitrary** Weights

Litterature

Only for nonnegative weights [de Alfaro 1999]

Or under following assumptions: [Bertsekas, Tsitsiklis 1991]

- there exists a **proper** strategy (σ is proper if $\mathbb{P}^{\sigma}_{M}(\diamond T) = 1$)
- total payoff goes to $+\infty$ under all **non-proper** strategies

Stochastic Shortest Path with **Arbitrary** Weights

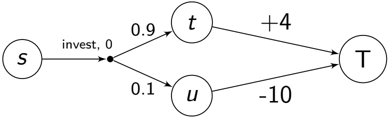
Litterature

Only for nonnegative weights [de Alfaro 1999]

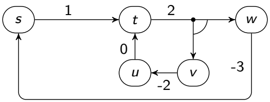
Or under following assumptions: [Bertsekas, Tsitsiklis 1991]

- there exists a **proper** strategy (σ is proper if $\mathbb{P}^{\sigma}_{M}(\diamond T) = 1$)
- total payoff goes to $+\infty$ under all **non-proper** strategies

- General weights:



- Total payoff $\not\to \infty$:



Stochastic Shortest Path with **Arbitrary** Weights

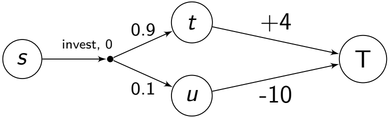
Litterature

Only for nonnegative weights [de Alfaro 1999]

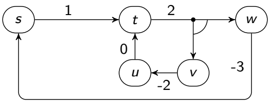
Or under following assumptions: [Bertsekas, Tsitsiklis 1991]

- there exists a **proper** strategy (σ is proper if $\mathbb{P}^{\sigma}_{M}(\diamond T) = 1$)
- total payoff goes to $+\infty$ under all **non-proper** strategies

- General weights:



- Total payoff $\not\to \infty$:

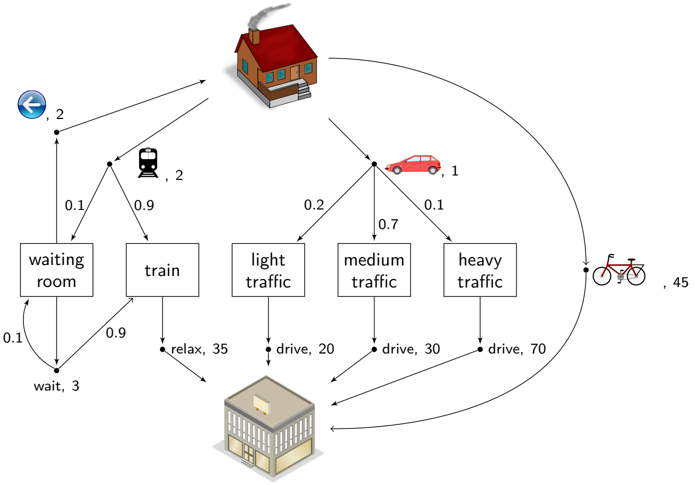


Theorem

Stochastic shortest paths in general weighted MDPs can be computed in polynomial time.

[Baier, Bertrand, Dubslaff, Gburek, S. LICS 2018]

Variance Weighted Markov Decision Processes

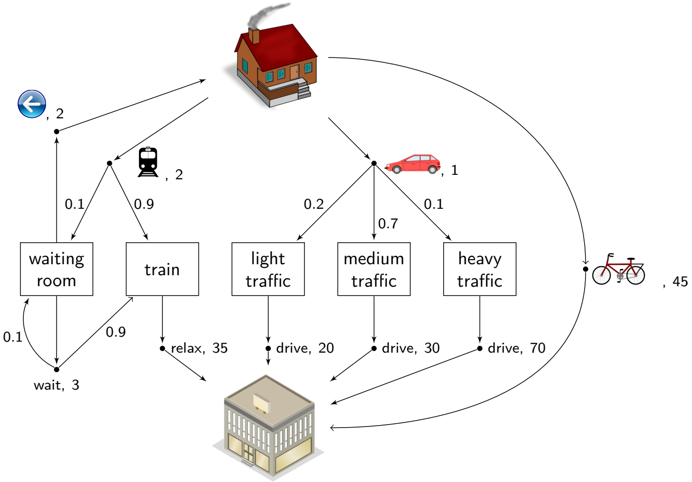


$$\mathbb{E}_{M}[\mathrm{TP}(T)] = 33$$

$$\mathbb{V}_{M}[\mathrm{TP}(T)] = 0.2(20-33)^2 + 0.2 \times 0.7(30-33)^2 + 0.1(70-33)^2$$

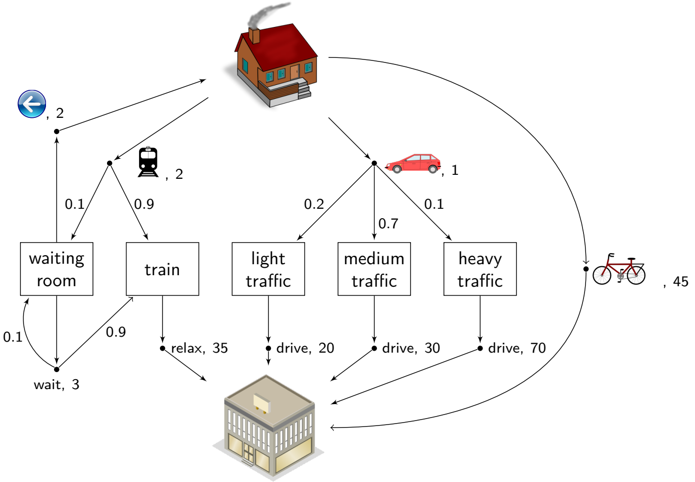
$$= 177 \qquad \approx 13.30^2$$

Variance Weighted Markov Decision Processes



$$\mathbb{E}_M^{🚆}[\mathrm{TP}(T)] = 37.33, \qquad \mathbb{V}_M^{🚆}[\mathrm{TP}(T)] \approx 1.02$$

Variance Weighted Markov Decision Processes



$\mathbb{E}_M[\mathrm{TP}(T)] = 45,$ $\qquad$ $\mathbb{V}_M[\mathrm{TP}(T)] = 0$

Variance in Weighted Markov Decision Processes

Minimizing variance is difficult because it is a quadratic expression

Open: Given bounds α, β, compute a strategy σ with $\mathbb{E}^\sigma[\mathsf{TP}(T)] \leq \alpha, \mathbb{V}^\sigma[\mathsf{TP}(T)] \leq \beta$.

Variance in Weighted Markov Decision Processes

Minimizing variance is difficult because it is a quadratic expression

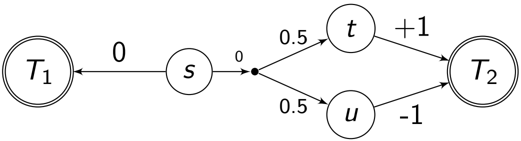
Open: Given bounds α, β, compute a strategy σ with $\mathbb{E}^\sigma[\mathsf{TP}(T)] \leq \alpha, \mathbb{V}^\sigma[\mathsf{TP}(T)] \leq \beta$.

Lexicographic Optimality

1) $\alpha = \inf_\sigma \mathbb{E}^\sigma[\mathsf{TP}(T)]$. 2) Minimize variance among strategies with expectation α

in polynomial time.

[Piribauer, S., Baier CONCUR 2022]



Variance in Weighted Markov Decision Processes

Minimizing variance is difficult because it is a quadratic expression

Open: Given bounds α, β, compute a strategy σ with $\mathbb{E}^\sigma[\mathsf{TP}(T)] \leq \alpha, \mathbb{V}^\sigma[\mathsf{TP}(T)] \leq \beta$.

Lexicographic Optimality

1) $\alpha = \inf_\sigma \mathbb{E}^\sigma[\mathsf{TP}(T)]$. 2) Minimize variance among strategies with expectation α

in polynomial time.

Variance-Penalized Expectation (VPE)

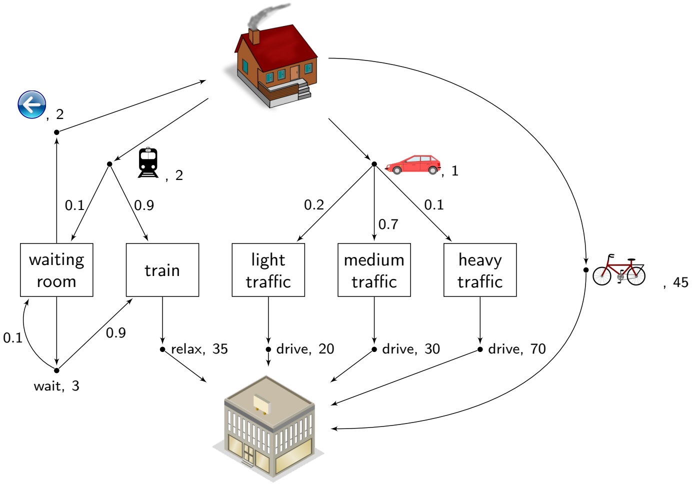
Maximize: $\mathbb{E}^\sigma[\mathsf{TP}(T)] - \lambda\mathbb{V}^\sigma[\mathsf{TP}(T)]$

linear combination of expectation and variance

The problem is EXPTIME-hard, and is in EXPSPACE.

[Piribauer, S., Baier CONCUR 2022]

Percentiles in Weighted Markov Decision Processes

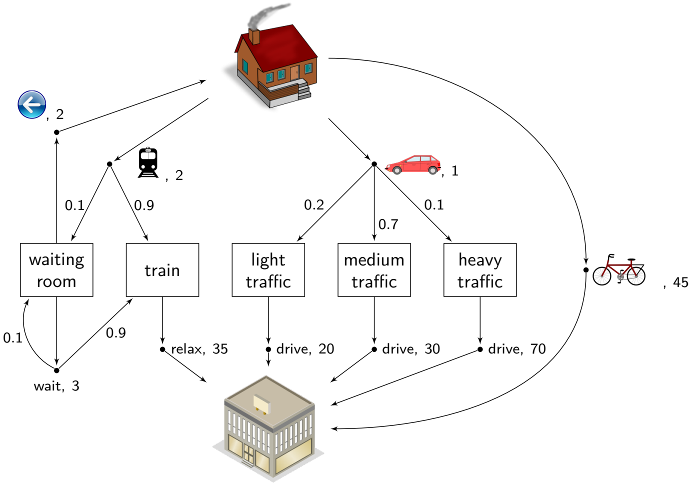


Another way of controlling the risks:

$$\mathbb{P}^\sigma[\mathrm{TP}(T) \leq 37] \geq 0.9$$

[Filar, Krass, Ross 1995]

Percentiles in Weighted Markov Decision Processes

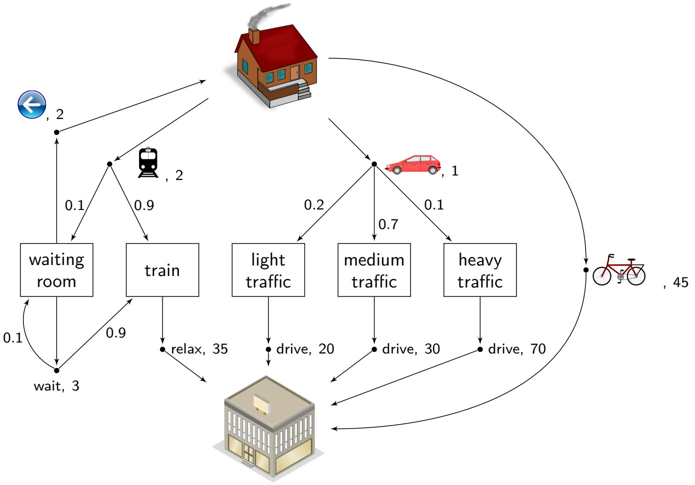


Another way of controlling the risks:

$$\mathbb{P}^{\sigma}[\mathrm{TP}(T) \leq 37] \geq 0.9 \ \wedge \ \mathbb{P}^{\sigma}[\mathrm{TP}(T) \leq 40] \geq 0.99$$

[Filar, Krass, Ross 1995]

Percentiles in Weighted Markov Decision Processes



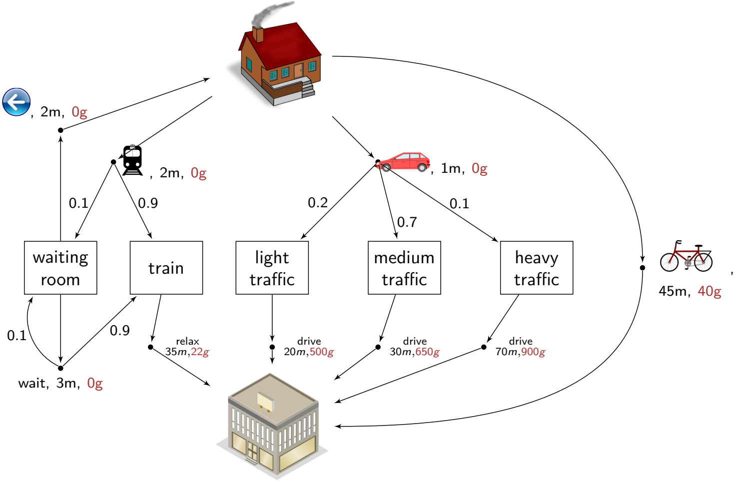
Another way of controlling the risks:

$$\mathbb{P}^{\sigma}[\mathrm{TP}(T) \leq 37] \geq 0.9 \;\wedge\; \mathbb{P}^{\sigma}[\mathrm{TP}(T) \leq 40] \geq 0.99 \;\wedge\; \mathbb{P}^{\sigma}[\mathrm{TP}(T) \leq 43] \geq 0.999$$

[Filar, Krass, Ross 1995]

Multiple Weights in Markov Decision Processes

Weights: Duration (minutes), Carbon footprint (grams of CO_2 per km)



$$\mathbb{P}^\sigma[\mathrm{TP}_{⏲}(T) \leq 50] \geq 0.99 \;\wedge\; \mathbb{P}^\sigma[\mathrm{TP}_{⏲}(T) \leq 60] \geq 1$$

$$\wedge\; \mathbb{P}^\sigma[\mathrm{TP}_{🍃}(T) \leq 110] \geq 0.9 \;\wedge\; \mathbb{P}^\sigma[\mathrm{TP}_{🍃}(T) \leq 140] \geq 0.99$$

Percentiles on Multiple Weights

Multiple Percentile Queries

One can compute strategy σ that satisfies a given Boolean combination of queries of the form $\mathbb{P}^{\sigma}[\mathsf{TP}_i(T) \leq \beta] \geq \alpha$, if such a strategy exists, in pseudo-polynomial time.

[Randour, Raskin, S. CAV 2015, VMCAI 2015, FMSD 2017]

Percentiles on Multiple Weights

Multiple Percentile Queries

One can compute strategy σ that satisfies a given Boolean combination of queries of the form $\mathbb{P}^\sigma[\mathrm{TP}_i(T) \leq \beta] \geq \alpha$, if such a strategy exists, in pseudo-polynomial time.

[Randour, Raskin, S. CAV 2015, VMCAI 2015, FMSD 2017]

Similar results hold for other objectives:

- (LimSup) Mean-payoff: In polynomial-time for

$$\overline{\mathrm{MP}}(w) = \lim \sup_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} w_i.$$

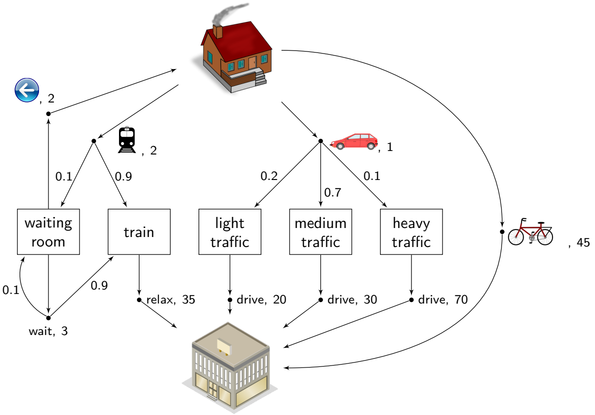
- (LimInf) MeanPayoff: Exponential in the dimensions:

$$\underline{\mathrm{MP}}(w) = \lim \inf_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} w_i.$$

- Discounted sum: pseudo-polynomial time

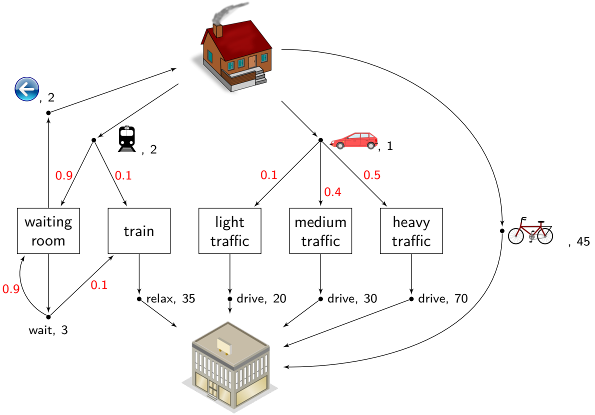
Multiple Environments in Markov Decision Processes

Regular day



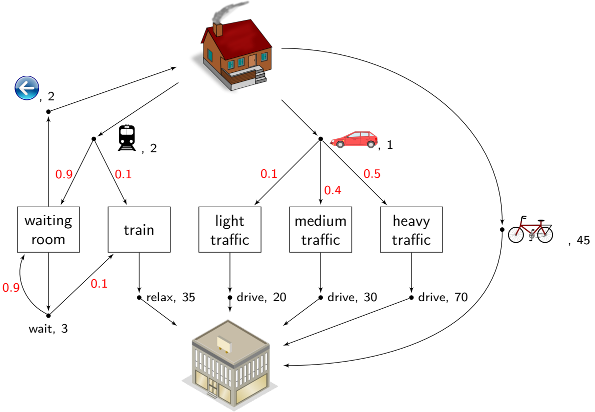
Multiple Environments in Markov Decision Processes

National strike



Multiple Environments in Markov Decision Processes

National strike



Is there a single strategy σ satisfying

$$\mathbb{P}^{\sigma}_{\text{regular}}[\phi] \geq 0.9 \ \wedge \ \mathbb{P}^{\sigma}_{\text{strike}}[\phi] \geq 0.7$$

without a prior knowledge of which case it is.

- Limited form of partial observation

Multiple Environments in Markov Decision Processes

Given a finite family of MDPs $M_i = (S, A, \delta_i), i \in I$, objective ϕ, and probability thresholds α_i, compute σ such that

$$\forall i \in I, \mathbb{P}^{\sigma}_{M_i}[\phi] \geq \alpha_i.$$

Multiple Environments in Markov Decision Processes

Given a finite family of MDPs $M_i = (S, A, \delta_i), i \in I$, objective ϕ, and probability thresholds α_i, compute σ such that

$$\forall i \in I, \mathbb{P}^{\sigma}_{M_i}[\phi] \geq \alpha_i.$$

Qualitative Case $(\alpha_i = 1)$

- Almost-sure: PSPACE-complete for reachability, safety, and parity
  Complexity poly. in $|M|$, exp. in $|I|$
- Limit-Sure: PSPACE-complete $\forall \epsilon, \exists \sigma, \forall i \in I, \mathbb{P}^{\sigma}_{M_i}[\phi] \geq 1 - \epsilon$

Multiple Environments in Markov Decision Processes

Given a finite family of MDPs $M_i = (S, A, \delta_i), i \in I$, objective ϕ, and probability thresholds α_i, compute σ such that

$$\forall i \in I, \mathbb{P}^{\sigma}_{M_i}[\phi] \geq \alpha_i.$$

Qualitative Case $(\alpha_i = 1)$

- Almost-sure: PSPACE-complete for reachability, safety, and parity
  Complexity poly. in $|M|$, exp. in $|I|$
- Limit-Sure: PSPACE-complete $\forall \epsilon, \exists \sigma, \forall i \in I, \mathbb{P}^{\sigma}_{M_i}[\phi] \geq 1 - \epsilon$

Quantitative Case for $|I| = 2$

- The ϵ-gap problem is decidable:
  Answer **Yes** if $\forall i \in I, \mathbb{P}^{\sigma}_{M_i}[\phi] \geq \alpha_i$.
  Answer **No** if $\exists i \in I, \mathbb{P}^{\sigma}_{M_i}[\phi] < \alpha_i - \epsilon$.
  Answer arbitrarily otherwise.

[Raskin, S. FSTTCS 2014]

Perspectives

We want **more control** over distributions induced by synthesis

Simply optimizing expectation may not be sufficient

Target: Feasible Cases

- for bounding expectation and variance
- for the quantitative case for multi-environment MDPs

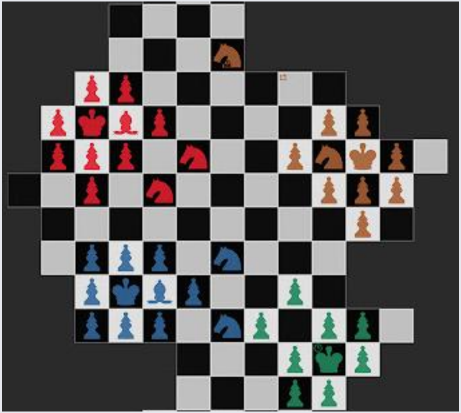
Target: Rich specifications

- Can we mix guarantees on expectation, variance, percentiles
- Worst-case guarantees

Other Works

Other Topics and Students

- Non-Zero Sum Games, Synthesis

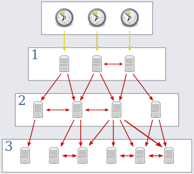


PhD: Suman Sadhukhan (2021)

Other Works

Other Topics and Students

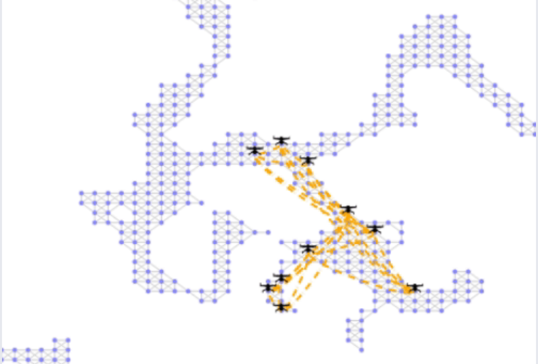
- Non-Zero Sum Games, Synthesis — PhD: Suman Sadhukhan (2021)
- Parameterized Verification — PhD: Nicolas Waldburger (Ongoing)



Other Works

Other Topics and Students

- Non-Zero Sum Games, Synthesis — PhD: Suman Sadhukhan (2021)
- Parameterized Verification — PhD: Nicolas Waldburger (Ongoing)
- Multi-Agent Path Finding — PhD: Arthur Queffelec (2021)



Other Works

Other Topics and Students

- Non-Zero Sum Games, Synthesis — PhD: Suman Sadhukhan (2021)
- Parameterized Verification — PhD: Nicolas Waldburger (Ongoing)
- Multi-Agent Path Finding — PhD: Arthur Queffelec (2021)

Some Projects

- PI of ANR Ticktac (2018-2023)
- Academic and industrial projects: other ANR and European projects, Nokia Bell Labs, Mitsubishi Electric, NewLogUp (upcoming)

Conclusions

Model checking and **Synthesis** with quantitative features:

- real-time constraints
  - large discrete state spaces
  - robustness verification and robust synthesis
- probabilities
  - expectation, variance, percentile, multiple environments
  - stochastic shortest path, mean payoff, discounted sum, parity