Computable analysis and control synthesis over complex dynamical systems via formal verification

Alessandro Abate

Delft Center for Systems and Control, TU Delft

Department of Computer Science, University of Oxford

April 2013

Outline

1. Formal abstractions for verification of complex models

2. Formal verification of stochastic hybrid systems
   - Analysis and control synthesis problems
   - Computable analysis and control synthesis via abstractions

3. Formal verification of max-plus linear models
   - Analysis and control synthesis problems
   - Computable analysis and control synthesis via abstractions

4. Concluding remarks

Key references will appear here

Outline

Formal abstractions for verification of complex models

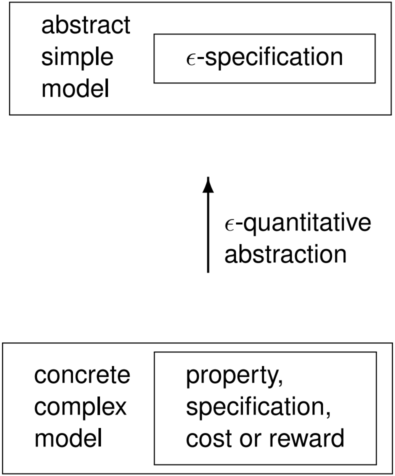
concrete complex model	property, specification, cost or reward

Formal abstractions for verification of complex models

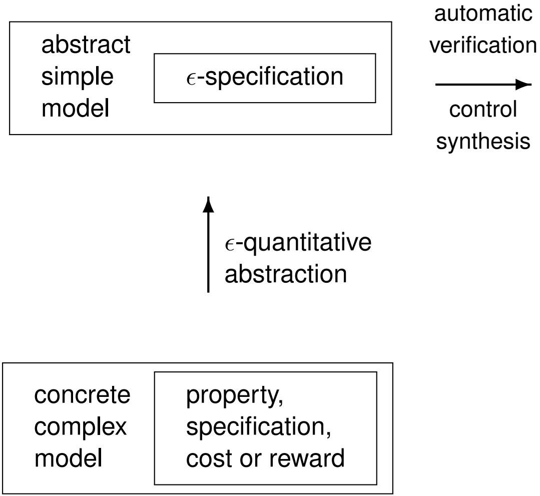
$\uparrow$ ϵ-quantitative abstraction

concrete complex model	property, specification, cost or reward

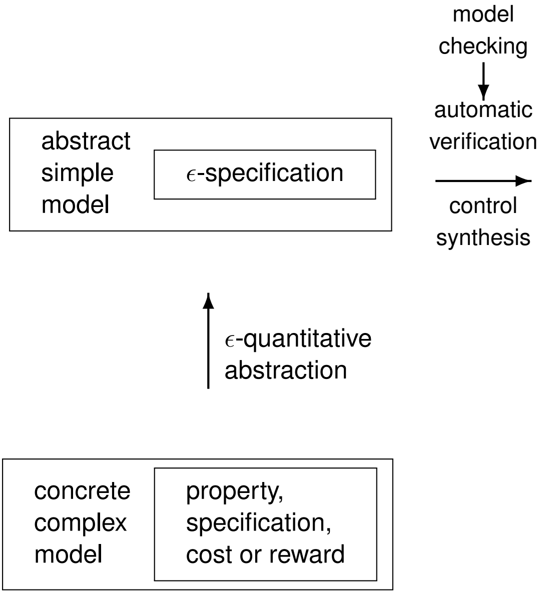
Formal abstractions for verification of complex models



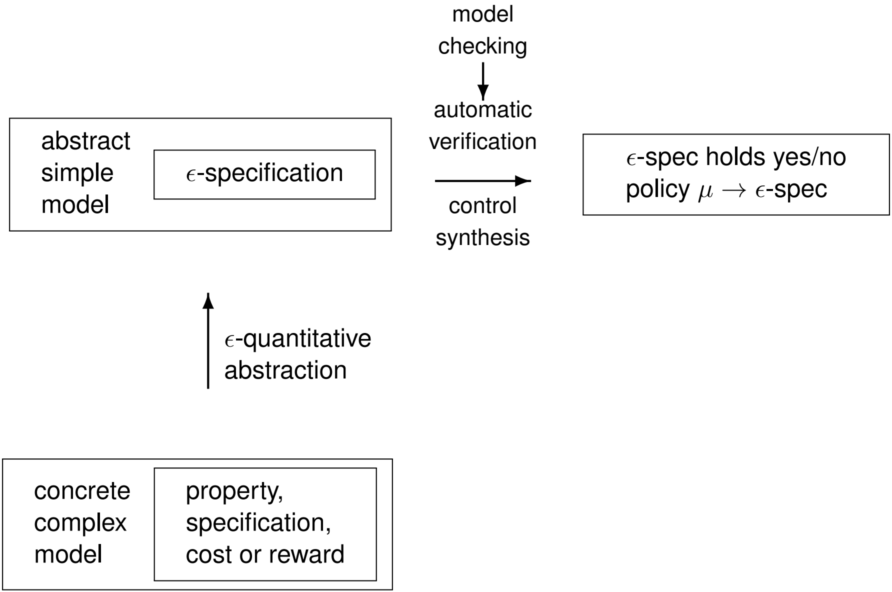
Formal abstractions for verification of complex models



Formal abstractions for verification of complex models



Formal abstractions for verification of complex models



Formal abstractions for verification of complex models

abstract simple model | ϵ-specification

model checking → automatic verification → control synthesis

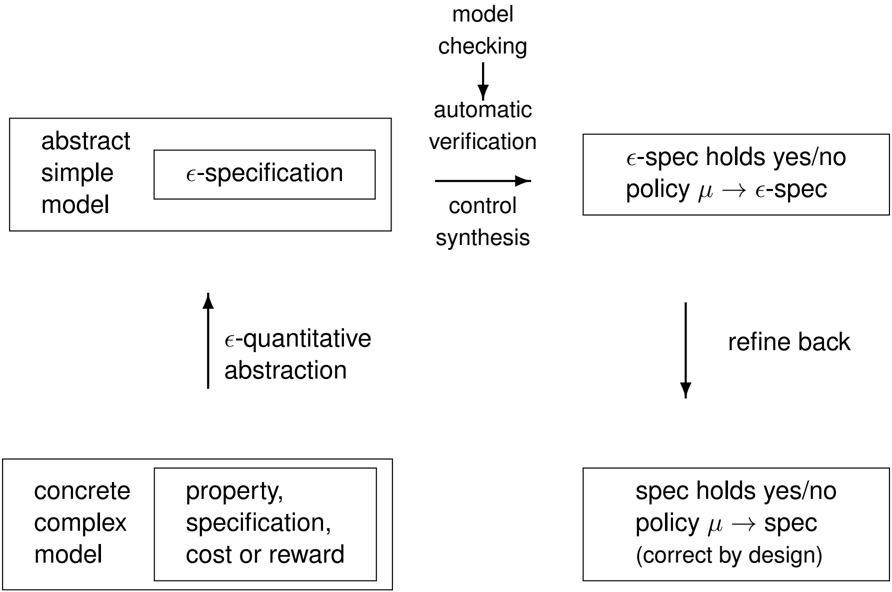
ϵ-spec holds yes/no
policy $\mu \rightarrow \epsilon$-spec

ϵ-quantitative abstraction ↑

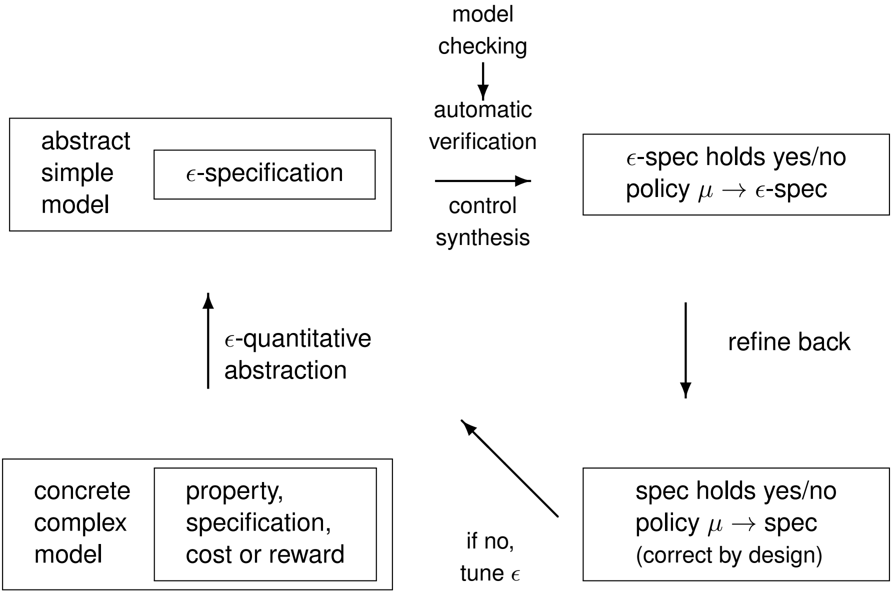
refine back ↓

concrete complex model | property, specification, cost or reward

Formal abstractions for verification of complex models



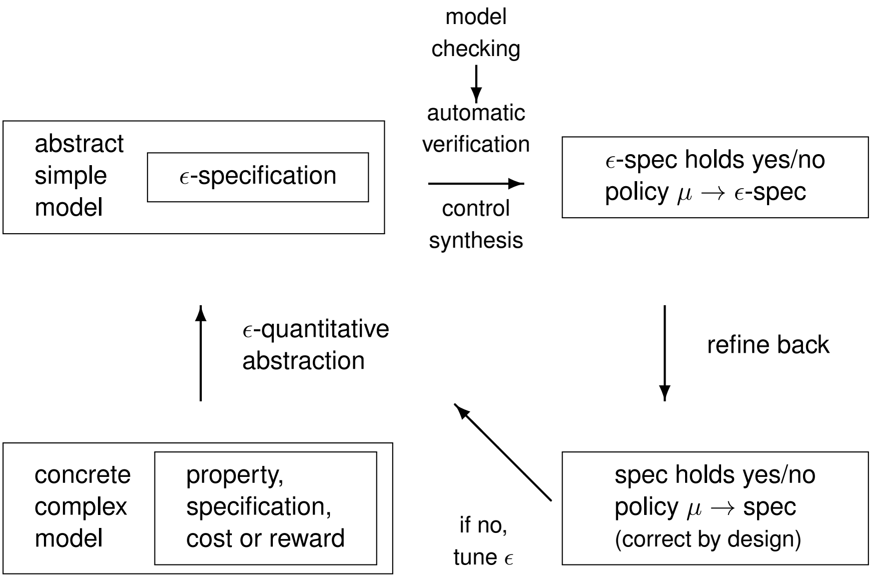
Formal abstractions for verification of complex models



Outline

1. Formal abstractions for verification of complex models
2. Formal verification of stochastic hybrid systems
   - Analysis and control synthesis problems
   - Computable analysis and control synthesis via abstractions
3. Formal verification of max-plus linear models
   - Analysis and control synthesis problems
   - Computable analysis and control synthesis via abstractions
4. Concluding remarks

Formal abstractions for verification of complex models



Formal abstractions for verification of dtSHS

PRISM
MRMC

dtMC dtMDP	relax'd/strenght'd PCTL inflated LTL – ϵ-spec

prob. model checking

dynamic programming

ϵ-spec holds policy max/min ϵ-spec

adaptive, sequential abstractions

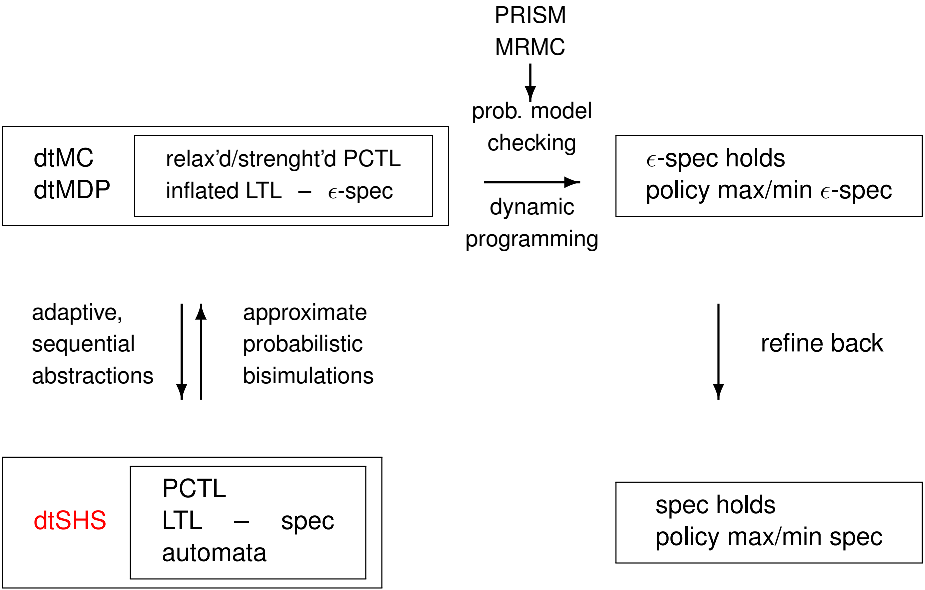
approximate probabilistic bisimulations

refine back

dtSHS	PCTL LTL – spec automata

spec holds policy max/min spec

Stochastic hybrid (discrete/continuous) systems



Stochastic hybrid (discrete/continuous) systems

- discrete-time models

finite-space Markov chain

$(\mathcal{Z}, \mathcal{T})$

$\mathcal{Z} = (z_1, z_2, z_3)$

$$\mathcal{T} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & \cdots & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix}$$

$P(z_1, \{z_2, z_3\}) = p_{12} + p_{13}$

uncountable-space Markov process

$(\mathcal{S}, T_s)$

$\mathcal{S} = \mathbb{R}^2$

$$T_s(x|s) = \frac{e^{-\frac{1}{2}(x-m(s))^T \Sigma^{-1}(s)(x-m(s))}}{\sqrt{2\pi}|\Sigma(s)|^{1/2}}$$

$P(s, A) = \int_A T_s(dx|s), \quad A \in \mathcal{B}(\mathcal{S})$

Stochastic hybrid (discrete/continuous) systems

- discrete-time models

finite-space Markov chain	uncountable-space Markov process
$(\mathcal{Z}, \mathcal{T})$	$(\mathcal{S}, T_s)$
$\mathcal{Z} = (z_1, z_2, z_3)$	$\mathcal{S} = \mathbb{R}^2$
$\mathcal{T} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & \cdots & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix}$	$T_s(x\|s) = \frac{e^{-\frac{1}{2}(x-m(s))^T \Sigma^{-1}(s)(x-m(s))}}{\sqrt{2\pi}\|\Sigma(s)\|^{1/2}}$
$P(z_1, \{z_2, z_3\}) = p_{12} + p_{13}$	$P(s, A) = \int_A T_s(dx\|s), \quad A \in \mathcal{B}(\mathcal{S})$

$\Rightarrow$ discrete-time, stochastic hybrid systems

Stochastic hybrid (discrete/continuous) systems

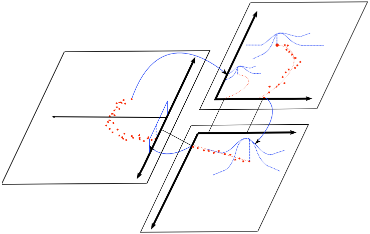
Definition

A discrete-time stochastic hybrid system is a pair $(\mathcal{S}, T_s)$, where

- $\mathcal{S} = \cup_{q \in \mathcal{Q}} (\{q\} \times \mathbb{R}^{n(q)})$, $\mathcal{Q}$ a discrete set of modes, $n : \mathcal{Q} \to \mathbb{N}$
- $T_s : \mathcal{S} \times \mathcal{S} \to [0, 1]$ specifies the dynamics of process at point $s = (q, x)$:

$$T_s(ds' | s) = \begin{cases} T_x(dx' | (q, x)) T_q(q | (q, x)), & \text{if } q' = q \text{ (no transition)} \\ T_r(dx' | (q, x), q') T_q(q' | (q, x)), & \text{if } q' \neq q \text{ (transition)} \end{cases}$$

- initial state $\pi : \mathcal{S} \to [0, 1]$



[AA et al - Automatica 08]

Stochastic hybrid (discrete/continuous) systems

Definition

A discrete-time stochastic hybrid system is a pair $(\mathcal{S}, T_s)$, where

- $\mathcal{S} = \cup_{q \in \mathcal{Q}} (\{q\} \times \mathbb{R}^{n(q)})$, $\mathcal{Q}$ a discrete set of modes, $n : \mathcal{Q} \to \mathbb{N}$
- $T_s : \mathcal{S} \times \mathcal{S} \to [0, 1]$ specifies the dynamics of process at point $s = (q, x)$:

$$T_s(ds' | s) = \begin{cases} T_x(dx'|(q,x)) T_q(q|(q,x)), & \text{if } q' = q \text{ (no transition)} \\ T_r(dx'|(q,x), q') T_q(q'|(q,x)), & \text{if } q' \neq q \text{ (transition)} \end{cases}$$

- initial state $\pi : \mathcal{S} \to [0, 1]$

- can be control dependent ($u \in \mathcal{U}$):

$$T_s(ds' | s, u) = \begin{cases} T_x(dx'|(q,x), u) T_q(q|(q,x), u), & \text{if } q' = q \text{ (no transition)} \\ T_r(dx'|(q,x), u, q') T_q(q'|(q,x), u), & \text{if } q' \neq q \text{ (transition)} \end{cases}$$

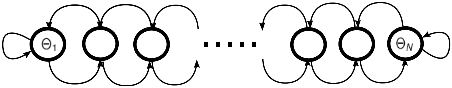
- policy μ: “string” of controls
- equivalent dynamical representation: $s_{k+1} = f(s_k, \xi_k, u_k)$

[AA et al - Automatica 08]

Stochastic hybrid systems in risk analysis

$$\begin{cases} Z_{n+1} = g(Z_n, \theta_n) & Z_n \in \mathbb{R}, & \leftarrow \text{ capital} \\ \theta_{n+1} = h(Z_n, \theta_n, \xi_n) & \theta_n \in \{\Theta_1, \ldots, \Theta_N\}, & \leftarrow \text{ interest} \end{cases}$$

where ξ_n i.i.d. random variables; g, h measurable; (Z_0, θ_0) given

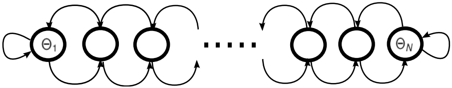


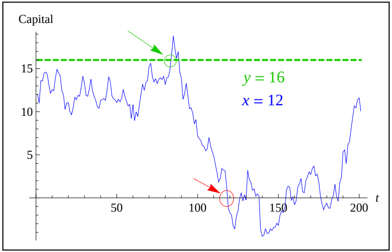
[I. Tkachev, AA - CDC 11]

Stochastic hybrid systems in risk analysis

$$\begin{cases} Z_{n+1} = g(Z_n, \theta_n) & Z_n \in \mathbb{R}, & \leftarrow \text{ capital} \\ \theta_{n+1} = h(Z_n, \theta_n, \xi_n) & \theta_n \in \{\Theta_1, \dots, \Theta_N\}, & \leftarrow \text{ interest} \end{cases}$$

where ξ_n i.i.d. random variables; g, h measurable; (Z_0, θ_0) given



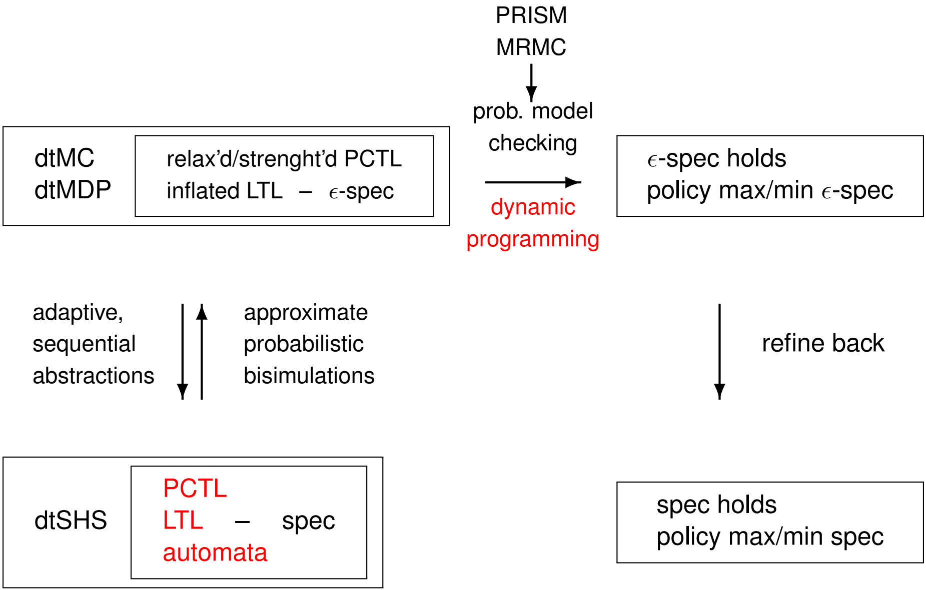


- **objective:** what is the probability that, starting from initial capital $Z_0 = x$, high capitalization y is reached, while company's bankruptcy is avoided

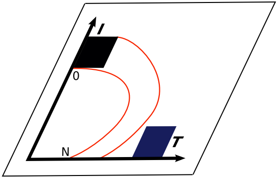
[I. Tkachev, AA - CDC 11]

Outline

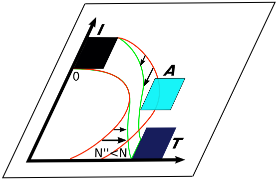
Analysis and control synthesis problems



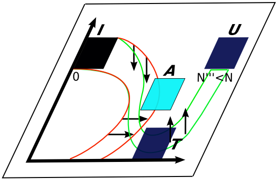
Analysis and control synthesis problems



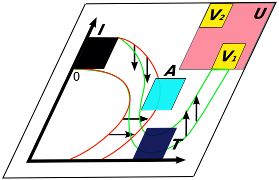
reachability
(safety/invariance)



reach-avoid
(constrained reachability)



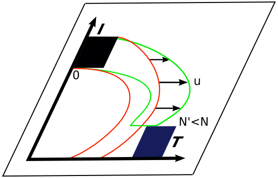
sequential reachability
(trajectory planning)



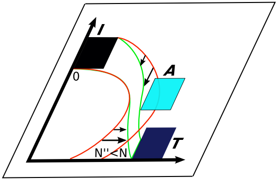
∞-horizon objectives
(i.o., eventually always)

- properties expressed via PCTL, LTL (DFA or Büchi automata)

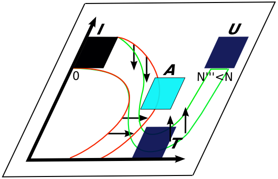
Analysis and control synthesis problems



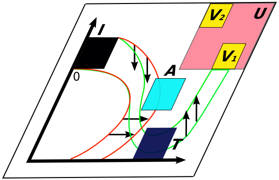
synthesis for reachability games ($2 - 1/2$ players)



synthesis for reach-avoid (pursuit evasion games)



sequential reachability (trajectory planning)



∞-horizon objectives (i.o., eventually always)

- properties expressed via PCTL, LTL (DFA or Büchi automata)

Probabilistic safety/invariance: characterization

- probabilistic invariance is *the probability that the execution associated with an initial distribution π stays in S (safe set) during the time horizon $[0, N]$*:

$$\mathcal{P}_{\pi}(S) := P_{\pi}(s_k \in S, \forall k \in [0, N])$$

[AA et al. - Automatica 08]

Probabilistic safety/invariance: characterization

- probabilistic invariance is *the probability that the execution associated with an initial distribution* π *stays in* S *(safe set) during the time horizon* $[0, N]$:

$$\mathcal{P}_{\pi}(S) := P_{\pi}(s_k \in S, \forall k \in [0, N])$$

- consider realization $s_k \in \mathcal{S}$, $k \in [0, N]$ – then

$$\prod_{k=0}^{N} \mathbf{1}_S(s_k) = \begin{cases} 1, & \text{if } \forall k \in [0, N] : s_k \in S \\ 0, & \text{otherwise} \end{cases}$$

$$\Rightarrow \mathcal{P}_{\pi}(S) = P_{\pi}\left(\prod_{k=0}^{N} \mathbf{1}_S(s_k) = 1\right) = E_{\pi}\left[\prod_{k=0}^{N} \mathbf{1}_S(s_k)\right]$$

[AA et al. - Automatica 08]

Probabilistic safety/invariance: characterization

- probabilistic invariance is *the probability that the execution associated with an initial distribution* π *stays in* S *(safe set) during the time horizon* $[0, N]$:

$$\mathcal{P}_\pi(S) := P_\pi(s_k \in S, \forall k \in [0, N])$$

- consider realization $s_k \in \mathcal{S}$, $k \in [0, N]$ – then

$$\prod_{k=0}^{N} \mathbf{1}_S(s_k) = \begin{cases} 1, & \text{if } \forall k \in [0, N] : s_k \in S \\ 0, & \text{otherwise} \end{cases}$$

$$\Rightarrow \mathcal{P}_\pi(S) = P_\pi\left(\prod_{k=0}^{N} \mathbf{1}_S(s_k) = 1\right) = E_\pi\left[\prod_{k=0}^{N} \mathbf{1}_S(s_k)\right]$$

- select $\epsilon \in [0, 1]$ – probabilistic safe/invariant set with safety level ϵ is

$$S(\epsilon) \doteq \{s \in \mathcal{S} : \mathcal{P}_s(S) \geq \epsilon\} \quad (\text{here } \pi = \delta_s)$$

[AA et al. - Automatica 08]

Probabilistic invariance: computation

- computation of $\mathcal{P}_s(S)$ (and thus of $S(\epsilon)$) via dynamic programming: sequential update, backward in time, of multi-stage value function

$$V_k(s) : [0, N] \times \mathcal{S} \to \mathbb{R}^+,$$

accounting for current and expected future rewards – in particular

$$V_N(s) = \mathbf{1}_S(s), \quad V_k(s) = \int_S V_{k+1}(x) T_s(dx|s)$$

$$\boxed{V_0(s) = \mathcal{P}_s(S)}$$

[AA et al. - Automatica 08]

Probabilistic invariance: computation

- computation of $\mathcal{P}_s(S)$ (and thus of $S(\epsilon)$) via dynamic programming: sequential update, backward in time, of multi-stage value function

$$V_k(s) : [0, N] \times \mathcal{S} \to \mathbb{R}^+,$$

accounting for current and expected future rewards – in particular

$$V_N(s) = \mathbf{1}_S(s), \quad V_k(s) = \int_S V_{k+1}(x) T_s(dx|s)$$

$$\boxed{V_0(s) = \mathcal{P}_s(S)}$$

- control dependent models: find optimal policy μ, optimizing recursively over

$$V_k(s, u) : [0, N] \times \mathcal{S} \times \mathcal{U} \to \mathbb{R}^+$$

[AA et al. - Automatica 08]

Computing probabilistic invariance

- issues
  1. non-standard (max, multiplicative) value functions
  2. continuous control space
  3. hybrid state space

$\Rightarrow$ solution of DP is seldom analytical

Computing probabilistic invariance

- issues
  1. non-standard (max, multiplicative) value functions
  2. continuous control space
  3. hybrid state space

$\Rightarrow$ solution of DP is seldom analytical

- numerical solutions are needed

$\Rightarrow$ problem # 1: difference between real solution and computed solution

$\Rightarrow$ problem # 2: Bellman's *curse of dimensionality*

Outline

1. Formal abstractions for verification of complex models
2. Formal verification of stochastic hybrid systems
   - Analysis and control synthesis problems
   - Computable analysis and control synthesis via abstractions
3. Formal verification of max-plus linear models
   - Analysis and control synthesis problems
   - Computable analysis and control synthesis via abstractions
4. Concluding remarks

Dynamical properties as temporal specifications

PRISM
MRMC

dtMC
dtMDP

relax'd/strenght'd PCTL
inflated LTL – ϵ-spec

prob. model checking

dynamic programming

ϵ-spec holds
policy max/min ϵ-spec

adaptive, sequential abstractions

approximate probabilistic bisimulations

refine back

dtSHS

PCTL
LTL – spec
automata

spec holds
policy max/min spec

Approximate model checking of probabilistic invariance

- model $(\mathcal{S}, T_s)$, invariance set $S \in \mathcal{S}$, finite time horizon N, safety level ϵ

[AA et al. - EJC 11]

Approximate model checking of probabilistic invariance

- model $(\mathcal{S}, T_s)$, invariance set $S \in \mathcal{S}$, finite time horizon N, safety level ϵ
- δ-approximate $(\mathcal{S}, T_s)$ with finite-state dt-MC $(\mathcal{Z}, \mathcal{T})$

$\star$ compute approximation error $f(\delta, N)$

- $S \to S_\delta$: define formula Φ_{S_δ} characterizing set S_δ, label states in $\mathcal{Z}$

[AA et al. - EJC 11]

Approximate model checking of probabilistic invariance

- model $(\mathcal{S}, T_s)$, invariance set $S \in \mathcal{S}$, finite time horizon N, safety level ϵ
- δ-approximate $(\mathcal{S}, T_s)$ with finite-state dt-MC $(\mathcal{Z}, \mathcal{T})$

$\star$ compute approximation error $f(\delta, N)$

- $S \to S_\delta$: define formula Φ_{S_δ} characterizing set S_δ, label states in $\mathcal{Z}$

$\Rightarrow$ probabilistic safe set

$$S(\epsilon) = \{s \in \mathcal{S} : \mathcal{P}_s(S) \geq \epsilon\}$$
$$= \{s \in \mathcal{S} : (1 - \mathcal{P}_s(S)) \leq 1 - \epsilon\}$$

[AA et al. - EJC 11]

Approximate model checking of probabilistic invariance

- model $(\mathcal{S}, T_s)$, invariance set $S \in \mathcal{S}$, finite time horizon N, safety level ϵ
- δ-approximate $(\mathcal{S}, T_s)$ with finite-state dt-MC $(\mathcal{Z}, \mathcal{T})$
- ⋆ compute approximation error $f(\delta, N)$
- $S \to S_\delta$: define formula Φ_{S_δ} characterizing set S_δ, label states in $\mathcal{Z}$

⇒ probabilistic safe set

$$\begin{aligned} S(\epsilon) &= \{s \in \mathcal{S} : \mathcal{P}_s(S) \geq \epsilon\} \\ &= \{s \in \mathcal{S} : (1 - \mathcal{P}_s(S)) \leq 1 - \epsilon\} \end{aligned}$$

can be related to

$$\begin{aligned} Z_\delta(\epsilon) &\doteq \mathrm{Sat}\left(\mathbb{P}_{\leq 1-\epsilon}\left(\mathtt{true}\ \mathcal{U}^{\leq N} \neg\Phi_{S_\delta}\right)\right) \\ &= \left\{z \in \mathcal{Z} : z \models \mathbb{P}_{\leq 1-\epsilon}\left(\mathtt{true}\ \mathcal{U}^{\leq N} \neg\Phi_{S_\delta}\right)\right\} \end{aligned}$$

[AA et al. - EJC 11]

Approximate model checking of probabilistic invariance

- model $(\mathcal{S}, T_s)$, invariance set $S \in \mathcal{S}$, finite time horizon N, safety level ϵ
- δ-approximate $(\mathcal{S}, T_s)$ with finite-state dt-MC $(\mathcal{Z}, \mathcal{T})$
- ⋆ compute approximation error $f(\delta, N)$
- $S \to S_\delta$: define formula Φ_{S_δ} characterizing set S_δ, label states in $\mathcal{Z}$

1. define

$$S(\epsilon) = \{s \in \mathcal{S} : \mathcal{P}_s(S) \geq \epsilon\}$$
$$Z_\delta(\epsilon) = \mathrm{Sat}\left(\mathbb{P}_{\leq 1-\epsilon}\left(\mathtt{true}\ \mathcal{U}^{\leq N} \neg\Phi_{S_\delta}\right)\right)$$

2. select $\eta > 0 : \eta/2 \in (0, 1-\epsilon)$
3. pick $\delta : f(\delta, N) \leq \eta/2$
4. compute $Z_\delta(\epsilon + \eta/2)$
5. define $\hat{S}_\eta(\epsilon) \doteq \{s \in \mathcal{S} \leftrightarrow z \in Z_\delta(\epsilon + \eta/2)\}$

$\Rightarrow$

$$S(\epsilon + \eta) \subseteq \hat{S}_\eta(\epsilon) \subseteq S(\epsilon)$$

[AA et al. - EJC 11]

Verification of over- or under-specifications in PCTL

- any PCTL formula can be expressed via equivalent DP recursions

- consider PCTL formula $\mathbb{P}_{\sim\epsilon}(\Psi)$ on SHS $(\mathcal{S}, T_s)$
- δ-approximate SHS $(\mathcal{S}, T_s)$ as a dt-MC $(\mathcal{Z}, \mathcal{T})$
- compute approximation error $f(\delta, N)$

[D'Innocenzo, AA, J.-P. Katoen - HSCC 12]

Verification of over- or under-specifications in PCTL

- any PCTL formula can be expressed via equivalent DP recursions

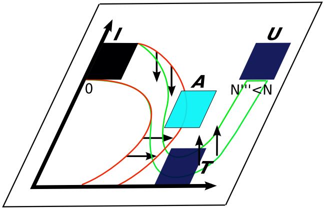
- consider PCTL formula $\mathbb{P}_{\sim\epsilon}(\Psi)$ on SHS $(\mathcal{S}, T_s)$
- δ-approximate SHS $(\mathcal{S}, T_s)$ as a dt-MC $(\mathcal{Z}, \mathcal{T})$
- compute approximation error $f(\delta, N)$

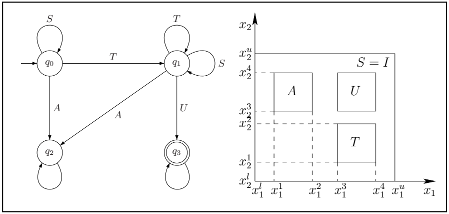
- compute $g(\Psi, f)$, a function based on formula & error
- model check $\mathbb{P}_{\sim\epsilon\pm g(\Psi,f)}(\Psi)$ on $(\mathcal{Z}, \mathcal{T})$

1. if PCTL formula is "robust", then conclusion holds for $\mathbb{P}_{\sim\epsilon}(\Psi)$ on SHS
2. else refine $\delta \to$ reduce $f(\delta, N) \to$ decrease $g(\Psi, f)$

[D'Innocenzo, AA, J.-P. Katoen - HSCC 12]

Approximate model checking of automata specifications

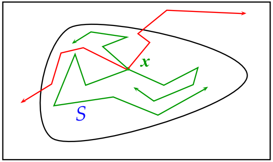




- generalization to “richer” set of properties over dtSHS
- specifications expressed as a DFA or a Büchi automata
- probabilistic reachability-like computation over product construction

[AA et al. - HSCC 11; I. Tkachev et al. - HSCC13]

Characterization & computation of ∞-horizon properties

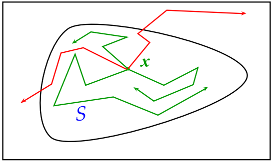


- consider target set T; invariant set $S = T^c = \mathcal{S} \setminus T$; $\forall s \in \mathcal{S}$:

$$P_s(\forall n \geq 0 : s_n \in S) \quad \leftrightarrow \quad 1 - P_s(\texttt{true}\,\mathcal{U}\,T)$$

[I. Tkachev, AA - CDC 11, HSCC 12, CDC12]

Characterization & computation of ∞-horizon properties



- consider target set T; invariant set $S = T^c = \mathcal{S} \setminus T$; $\forall s \in \mathcal{S}$:

$$P_s(\forall n \geq 0 : s_n \in S) \quad \leftrightarrow \quad 1 - P_s(\mathtt{true}\,\mathcal{U}\,T)$$

- existence and computation of absorbing set B : $\forall x \in B$, $T_s(B|x) = 1$

- characterization – study of existence/uniqueness of (non-trivial) solutions of Bellman equations

  convergence of Bellman recursions, contractivity of operators

- computation – formal reduction to finite-horizon problems

[I. Tkachev, AA - CDC 11, HSCC 12, CDC12]

On the approximation error $f(\delta, N)$

PRISM
MRMC

prob. model checking

dynamic programming

dtMC dtMDP	relax'd/strenght'd PCTL inflated LTL – ϵ-spec

ϵ-spec holds policy max/min ϵ-spec

adaptive, sequential abstractions

approximate probabilistic bisimulations

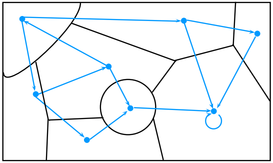
refine back

dtSHS	PCTL LTL – spec automata

spec holds policy max/min spec

On the approximation error $f(\delta, N)$

- approximation via δ-partitioning: $S = \cup_{q \in \mathcal{Q}} \{q\} \times S_q = \cup_{q \in \mathcal{Q}, i=1,\ldots,m_q} \{q\} \times S_q^i$



- under Lip-continuity assumptions on density of kernel T_s,

$$h(i,j), \qquad i,j = 1, \ldots, m_q, q \in \mathcal{Q}$$

- for any $z_q^i \in S_\delta, \forall s : s \wedge z_q^i \in S_q^i$, error is

$$f(\delta, N) \doteq \left| \mathcal{P}_s(S) - \mathcal{P}_{z_q^i}(S_\delta) \right| \leq \max_{i=1,\ldots,m_q, q \in \mathcal{Q}} N\delta_{q,i} \sum_{j=1,\ldots,m_r, r \in \mathcal{Q}} h(i,j),$$

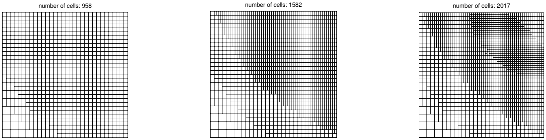
$\delta = \max_{i=1,\ldots,m_q, q \in \mathcal{Q}} \delta_{q,i}, \ \delta_{q,i} = \text{diam}\,(S_q^i)$

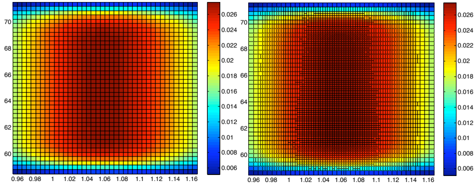
error is linear in N, $\delta_{q,i}$ and depends on local constants $h(i,j)$ $\rightarrow$ local tuning

[AA et al. - EJC 11, S. Soudjani, AA - QEST 11]

On the approximation error $f(\delta, N)$

- software (in the making) for sequential, adaptive grid generation based on approximation error
- formula-based abstractions





[S. Soudjani, AA - QEST 11, HSCC 12, ATVA12, SIAM 13]

On the approximation error $f(\delta, N)$

- error generalization
  - discontinuous and partially degenerate kernels
  - ill-conditioned kernels (different time scales, e.g. biology)

- error refinement by higher-order approximations
  - δ: faster convergence upon tuning
  - N: possibly bounded in time (allows considering ∞-horizon properties)

[S. Soudjani, AA - QEST 11, HSCC 12, ATVA12, SIAM 13; I. Tkachev, AA - HSCC 13]

On the approximation error $f(\delta, N)$

- error generalization
  - discontinuous and partially degenerate kernels
  - ill-conditioned kernels (different time scales, e.g. biology)

- error refinement by higher-order approximations
  - δ: faster convergence upon tuning
  - N: possibly bounded in time (allows considering ∞-horizon properties)

- alternative: formula-free abstractions

[S. Soudjani, AA - QEST 11, HSCC 12, ATVA12, SIAM 13; I. Tkachev, AA - HSCC 13]

Approximate probabilistic bisimulations

PRISM
MRMC

dtMC
dtMDP

relax'd/strenght'd PCTL
inflated LTL – ϵ-spec

prob. model checking

dynamic programming

ϵ-spec holds
policy max/min ϵ-spec

adaptive, sequential abstractions

approximate probabilistic bisimulations

refine back

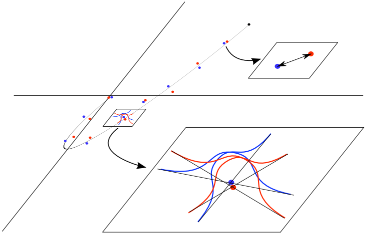
dtSHS

PCTL
LTL – spec
automata

spec holds
policy max/min spec

Approximate probabilistic bisimulations

- above abstraction leads to approximate probabilistic bisimulation *[Larsen & Skou, 91]* - alternatively ...



- consider models $(T_{s,i}, \mathcal{S}_i)$ with solution processes $s_i(k), i = 1, 2, k \geq 0$
- parallel composition of models with output $s_{1,2}(k) = s_1(k) - s_2(k)$

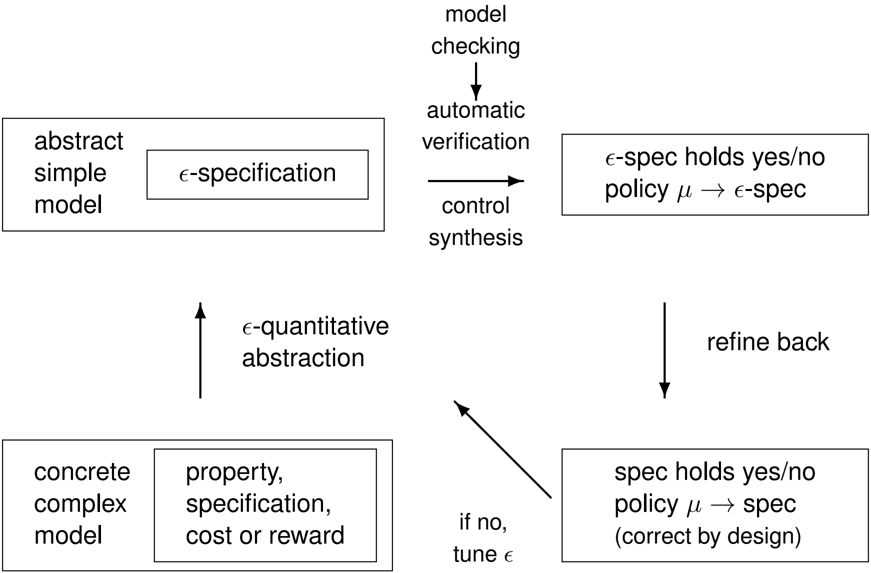
Definition

A function $\psi : \mathcal{S}_1 \times \mathcal{S}_2 \to \mathbb{R}^+$ is a probabilistic bisimulation function if $\psi(s_{1,2}) \geq \|s_1 - s_2\|^2$ and if $\psi_{s_0}(s_{1,2}(k))$ is a *supermartingale*.

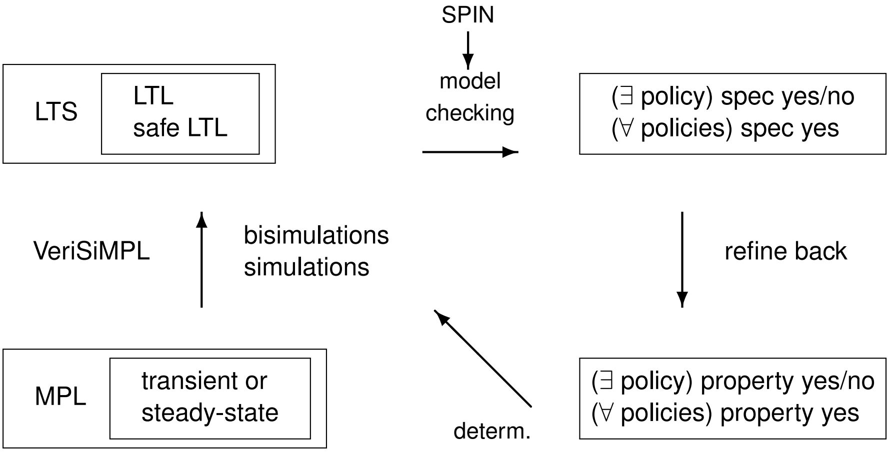
- ψ is an upper bound on the distance btw solutions of two models:
  $P_{s_0}\left(\sup_{k \geq 0} \|s_1(k) - s_2(k)\|^2 \geq \epsilon\right) \leq \psi_{s_0}(s_{1,2}(0))/\epsilon$ *[AA - ENTCS 13]*

Outline

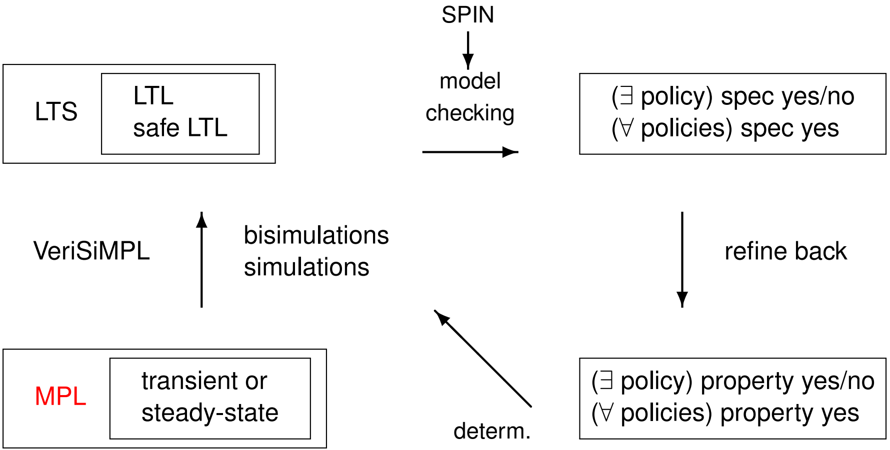
Formal abstractions for verification of complex models



Formal abstractions for verification of MPL models



Introduction to MPL systems



Introduction to MPL systems

- Max-Plus-Linear (MPL) systems are event-driven models
- applications: railway scheduling, planning of production lines, network calculus

- $x(k)$ is the time of k-th event, $k \in \mathbb{N} \cup \{0\}$
- timing updates: maximization ($\oplus$) and addition ($\otimes$) operations

$\rightarrow$ max-plus algebra

Max-plus algebra

- $\epsilon = -\infty, \quad \mathbb{R}_\epsilon = \mathbb{R} \cup \{\epsilon\}$
- $\alpha, \beta \in \mathbb{R}_\epsilon, \quad A, B \in \mathbb{R}_\epsilon^{m \times p}, \quad C \in \mathbb{R}_\epsilon^{p \times n}$
- $\alpha \oplus \beta \stackrel{\text{def}}{=} \max(\alpha, \beta)$
- $\alpha \otimes \beta \stackrel{\text{def}}{=} \alpha + \beta$

Max-plus algebra

- $\epsilon = -\infty, \quad \mathbb{R}_\epsilon = \mathbb{R} \cup \{\epsilon\}$
- $\alpha, \beta \in \mathbb{R}_\epsilon, \quad A, B \in \mathbb{R}_\epsilon^{m \times p}, \quad C \in \mathbb{R}_\epsilon^{p \times n}$

- $\alpha \oplus \beta \stackrel{\text{def}}{=} \max(\alpha, \beta)$

- $\alpha \otimes \beta \stackrel{\text{def}}{=} \alpha + \beta$

- $[A \oplus B]_{i,j} \stackrel{\text{def}}{=} [A]_{i,j} \oplus [B]_{i,j}$, for $i = 1, \ldots, m$ and $j = 1, \ldots, p$

- $[A \otimes C]_{i,j} \stackrel{\text{def}}{=} \bigoplus_{k=1}^{p} [A]_{i,k} \otimes [C]_{k,j}$, for $i = 1, \ldots, m$ and $j = 1, \ldots, n$

Max-plus-linear models

Definition (Autonomous MPL model)

$$x(k+1) = A \otimes x(k),$$

where $A \in \mathbb{R}_{\epsilon}^{n \times n}$ and $k \in \mathbb{N} \cup \{0\}$

Example

A simple railway model *[Heidergott, 06]*

$$x(k+1) = \begin{bmatrix} 2 & 5 \\ 3 & 3 \end{bmatrix} \otimes x(k), \quad \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} \max\{2 + x_1(k), 5 + x_2(k)\} \\ \max\{3 + x_1(k), 3 + x_2(k)\} \end{bmatrix}$$

[Baccelli et al., 92]

Max-plus-linear models

Definition (Autonomous MPL model)

$$x(k+1) = A \otimes x(k),$$

where $A \in \mathbb{R}_{\epsilon}^{n\times n}$ and $k \in \mathbb{N} \cup \{0\}$

Example

A simple railway model *[Heidergott, 06]*

$$x(k+1) = \begin{bmatrix} 2 & 5 \\ 3 & 3 \end{bmatrix} \otimes x(k), \qquad \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} \max\{2 + x_1(k), 5 + x_2(k)\} \\ \max\{3 + x_1(k), 3 + x_2(k)\} \end{bmatrix}$$

Definition (Non-autonomous MPL model)

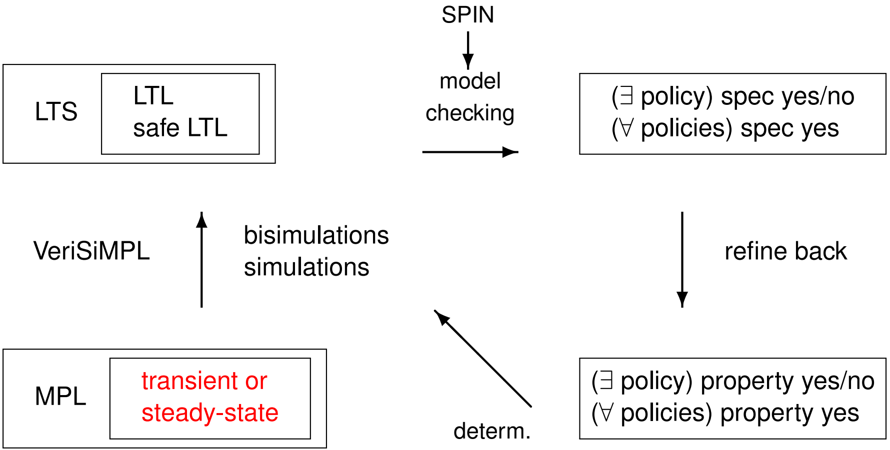
$$x(k+1) = A \otimes x(k) \oplus B \otimes u(k),$$

where $B \in \mathbb{R}_{\epsilon}^{n\times m}$ and $u \in \mathbb{R}^{m}$ (synthesis = scheduling)

[Baccelli et al., 92]

Outline

Classical analysis of MPL models



Classical analysis of MPL models

- study of transient and periodic regimes, of asymptotics
- classical analysis based on algebraic or geometric properties

Definition

1. max-plus eigenvector $x \in \mathbb{R}^n$: $A \otimes x = \lambda \otimes x \Rightarrow x(k+1) = \lambda \otimes x(k)$
2. cycles on precedence graph $\Rightarrow$ periodic regime with period c: $\forall k \geq k_0, x(k+c) = \lambda^{\otimes^c} \otimes x(k)$

Example

1. eigenspace (periodic regime with period 1 and $\lambda = 4$):

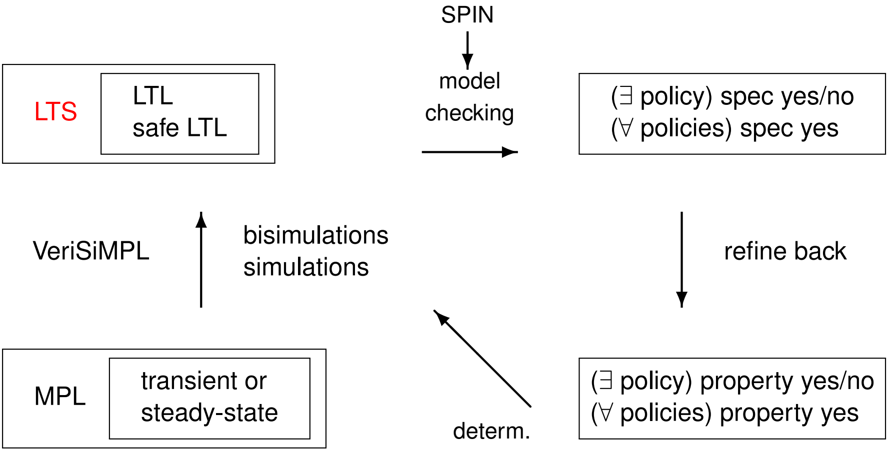
$$\begin{bmatrix}1\\0\end{bmatrix}, \begin{bmatrix}5\\4\end{bmatrix}, \begin{bmatrix}9\\8\end{bmatrix}, \begin{bmatrix}13\\12\end{bmatrix}, \begin{bmatrix}17\\16\end{bmatrix}, \begin{bmatrix}21\\20\end{bmatrix}, \begin{bmatrix}25\\24\end{bmatrix}, \begin{bmatrix}29\\28\end{bmatrix}, \begin{bmatrix}33\\32\end{bmatrix}, \begin{bmatrix}37\\36\end{bmatrix}, \begin{bmatrix}41\\40\end{bmatrix}, \begin{bmatrix}45\\44\end{bmatrix}, \ldots$$

2. periodic regime with period $c = 2$ (transient $k_0 = 3$):

$$\begin{bmatrix}4\\0\end{bmatrix}, \begin{bmatrix}6\\7\end{bmatrix}, \begin{bmatrix}12\\10\end{bmatrix}, \begin{bmatrix}15\\15\end{bmatrix}, \begin{bmatrix}20\\18\end{bmatrix}, \begin{bmatrix}23\\23\end{bmatrix}, \begin{bmatrix}28\\26\end{bmatrix}, \begin{bmatrix}31\\31\end{bmatrix}, \begin{bmatrix}36\\34\end{bmatrix}, \begin{bmatrix}39\\39\end{bmatrix}, \begin{bmatrix}44\\42\end{bmatrix}, \begin{bmatrix}47\\47\end{bmatrix}, \ldots$$

Outline

Labeled transition system (LTS)



Labeled transition system (LTS)

- consider AP, a set of atomic propositions

Definition

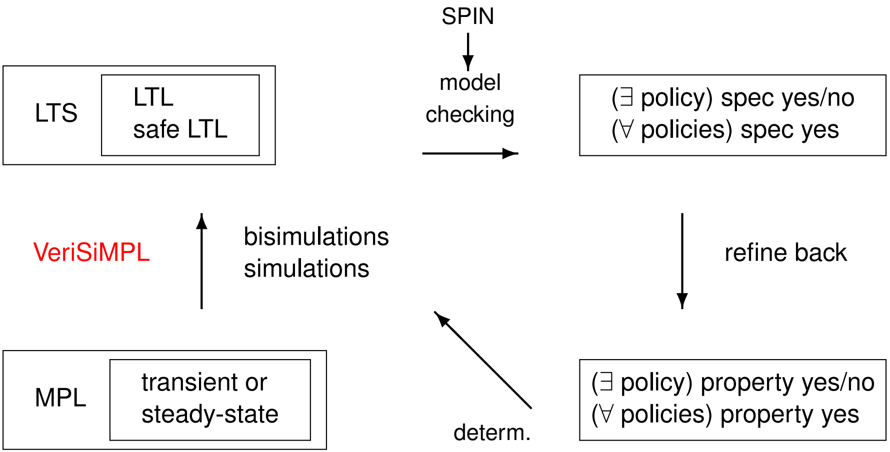
A labeled transition system (L, S, δ) consists of

- S: a set of states
- L: a set of labels in 2^{AP}
- $\delta \subseteq S \times L \times S$: a transition relation

- labels can be defined over states
- LTS can be deterministic vs non-deterministic
- LTS can be infinite vs finite

[Baier & Katoen, 08]

Finite LTS as abstractions of MPL models



Finite LTS as abstractions of MPL models

- procedure
  1. S: construct collection of LTS states from partitions of MPL state space
  2. δ: determine LTS transitions via one-step reach over MPL
  3. L: compute labels related to MPL timing $\rightarrow$ induce set of AP

Finite LTS as abstractions of MPL models

- procedure
  1. S: construct collection of LTS states from partitions of MPL state space
  2. δ: determine LTS transitions via one-step reach over MPL
  3. L: compute labels related to MPL timing $\rightarrow$ induce set of AP

Definition (Regular matrix)

A matrix $A \in \mathbb{R}_{\epsilon}^{m \times n}$ is called regular (row-finite) if it contains at least one element different from ϵ in each row (in practice, no instantaneous events)

LTS states: state-space partitioning

- autonomous MPL model can be expressed as PWA system
- PWA dynamics are associated to polytopic regions

- collection of regions is a cover of $\mathbb{R}^n$ (in general not a partition)
- partition constructed via further refinement

LTS states: state-space partitioning

- autonomous MPL model can be expressed as PWA system
- PWA dynamics are associated to polytopic regions

- collection of regions is a cover of $\mathbb{R}^n$ (in general not a partition)
- partition constructed via further refinement

- obtained state-space partition defines states of LTS
- partition is not arbitrary: it is adapted to underlying dynamics

State-space partitioning, an example

Example

- after refinement, total of 5 regions:

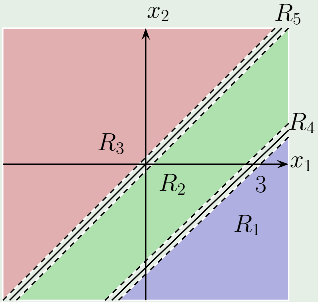
$$R_1 = \{x \in \mathbb{R}^2 : x_1 - x_2 < 0\}$$

$$R_2 = \{x \in \mathbb{R}^2 : x_1 - x_2 = 0\}$$

$$R_3 = \{x \in \mathbb{R}^2 : x_1 - x_2 > 3\}$$

$$R_4 = \{x \in \mathbb{R}^2 : x_1 - x_2 = 3\}$$

$$R_5 = \{x \in \mathbb{R}^2 : 0 < x_1 - x_2 < 3\}$$



Difference-bound matrices (DBM)

Definition (DBM)

A difference-bound matrix in $\mathbb{R}^n$ is the finite intersection of sets defined by

$$x_i - x_j \simeq_{i,j} \alpha_{i,j},$$

where $\simeq_{i,j} \in \{<, \leq\}$, $\alpha_{i,j} \in \mathbb{R} \cup \{+\infty\}$, for $1 \leq i \neq j \leq n$

- DBM allow compact matrix representation
- DBM are easy to manipulate (projections, emptiness and inclusion check)

[Dill, 90]

Difference-bound matrices (DBM)

Definition (DBM)

A difference-bound matrix in $\mathbb{R}^n$ is the finite intersection of sets defined by

$$x_i - x_j \simeq_{i,j} \alpha_{i,j},$$

where $\simeq_{i,j} \in \{<, \leq\}$, $\alpha_{i,j} \in \mathbb{R} \cup \{+\infty\}$, for $1 \leq i \neq j \leq n$

- DBM allow compact matrix representation
- DBM are easy to manipulate (projections, emptiness and inclusion check)

- image/inverse image of DBM over MPL dynamics is again a DBM

[Dill, 90]

LTS transitions: one-step reachability

- consider any two TS states (partitioning regions) R, R'
- $R \to R'$ iff there exists a $x(k) \in R$ such that $x(k+1) \in R'$:
  check whether $R' \cap \{x(k+1) : x(k) \in R\} \neq \emptyset$

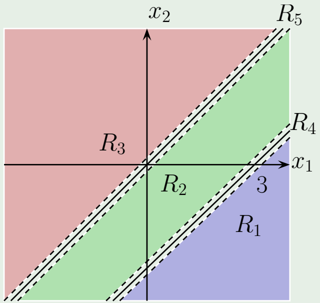
LTS transitions: one-step reachability

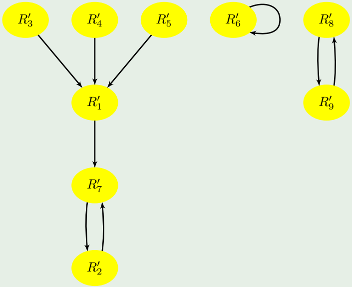
- consider any two TS states (partitioning regions) R, R'
- $R \to R'$ iff there exists a $x(k) \in R$ such that $x(k+1) \in R'$:
  check whether $R' \cap \{x(k+1) : x(k) \in R\} \neq \emptyset$

- use DBM representation, DBM forward-mapping via PWA dynamics, DBM emptiness check

LTS transitions, an example

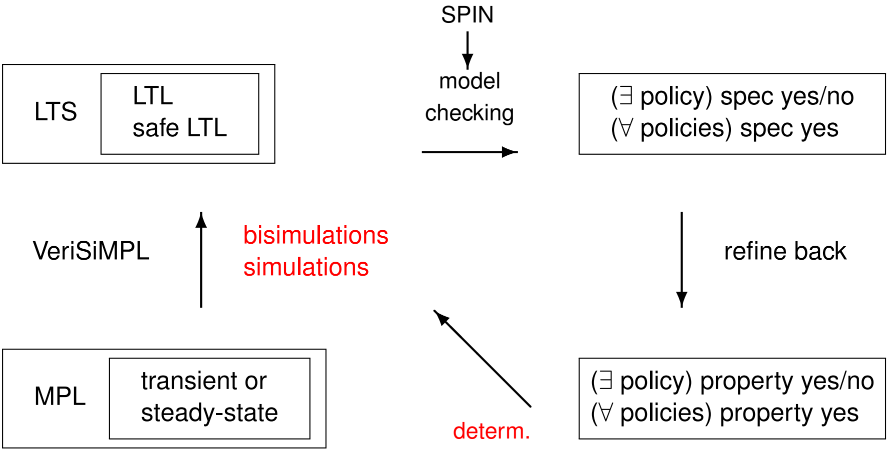
Example





- determinism vs non-determinism of obtained TS
- above R_i - original partitions, R'_i - refined partitions (determinization)

Relationship between LTS and MPL



Relationship between LTS and MPL

Theorem

- *TS simulates the original MPL model*
- *TS bisimulates the MPL model if and only if it is deterministic*

- non-deterministic TS can be “determinized” by refining partitioning regions
- however, refinement procedure may not terminate

Theorem

- *if TS is deterministic over the periodic regime, then TS is globally deterministic*
- *every irreducible MPL model admits finite deterministic TS abstraction*

LTS labels

Definition

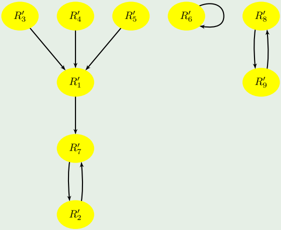
- state labels:
  all possible values of $x_i(k) - x_j(k)$, for $1 \leq i < j \leq n$
  time difference of same-event variables
- transition labels:
  all possible values of $x_i(k+1) - x_i(k)$, for $1 \leq i \leq n$
  time difference of successive events

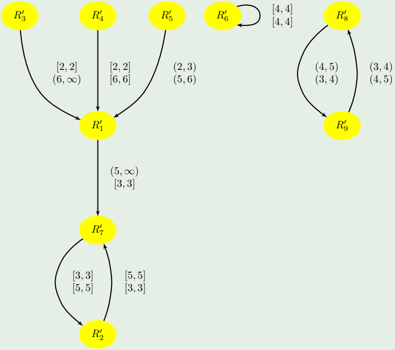
- labels are vectors of intervals, can be represented as DBM

LTS labels, an example

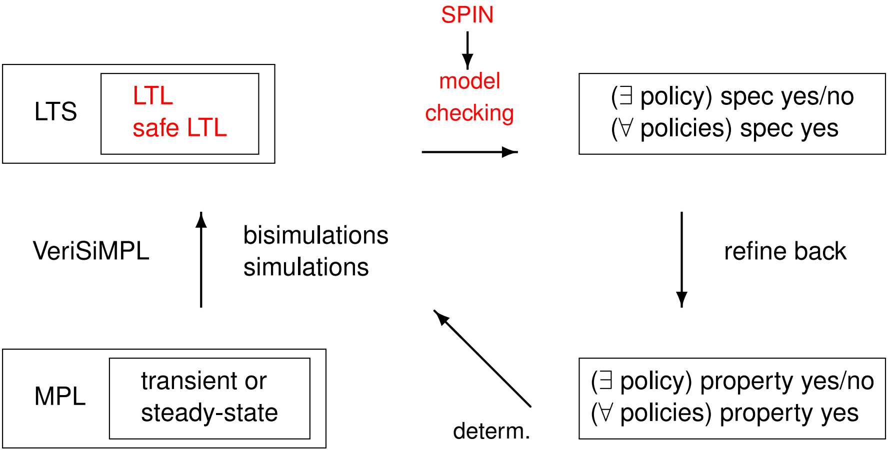
Example

- LTS transition labels





Computational benchmark for abstraction



Computational benchmark for abstraction

- A randomly generated with elements taking values between 1 and 100
- row-finite matrix A: with 2 finite elements placed randomly in each row
- 10 independent experiments per dimension – mean values are displayed:

size of MPL model	time for generation of states	time for generation of transitions	time for generation of labels	total number of LTS states	total number of LTS transitions
3	0.1 [s]	0.4 [s]	0.1 [s]	3.6	4.3
5	0.2 [s]	0.4 [s]	0.1 [s]	8.6	13.8
7	0.9 [s]	0.5 [s]	0.3 [s]	37.2	289.3
9	4.1 [s]	0.8 [s]	1.6 [s]	120.0	$1.7 \cdot 10^3$
11	24.8 [s]	15.2 [s]	16.1 [s]	613.2	$1.9 \cdot 10^4$
13	3.5 [m]	5.5 [m]	2.8 [m]	$1.9 \cdot 10^3$	$1.9 \cdot 10^5$
15	53.6 [m]	2.0 [h]	39.4 [m]	$7.4 \cdot 10^3$	$2.0 \cdot 10^6$

- coded in MATLAB, run over 12-core Intel Xeon, 3.47 GHz, 24 GB
- bottleneck: generation of transitions

Computational benchmark for reachability analysis

- *A* randomly generated with elements taking values between 1 and 100
- row-finite matrix *A*: with 2 finite elements placed randomly in each row
- 10 independent experiments per dimension – mean values are displayed:
- set of initial conditions is selected as the unit hypercube

size of MPL model	time for generation of PWA system	number of regions of PWA system	time for generation of reach tube
3	0.09 [s]	5	0.09 [s]
10	4.73 [s]	700	8.23 [s]
18	29.13 [m]	$1.58 \cdot 10^5$	5.82 [h]

- comparison MPL vs MPT
- generation time for reach tube of 10-dimensional MPL model, different time horizons

time horizon	20	40	60	80	100
MPL	11.02 [s]	17.94 [s]	37.40 [s]	51.21 [s]	64.59 [s]
MPT	47.61 [m]	1.19 [h]	2.32 [h]	3.03 [h]	3.73 [h]

Formal analysis of MPL models is now "very simple"

VeriSiMPL – Verification via biSimulation of MPL models

- abstract MPL model as LTS (in MATLAB)
- export LTS abstraction (as PROMELA script) into SPIN model checker
- consider properties in LTL logic
- verify property via SPIN over LTS and export outcome back to MPL model


A. Abate
Home
Contact Info
Bio Sketch
Research
Interests
Publications
Group
Teaching
Courses

VeriSiMPL ("very simple")
Verification via biSimulations of Max-Plus Linear Models

VeriSiMPL
- is a software tool for concrete MPL models implemented in Matlab, which exports abstract LTS models to SPIN in Promela language

Documentation
- comes as a text file: txt

Download
- the toolbox as a compressed folder: zip

Contacts
- for questions and queries, please send an email to
  - D. Adzkiya, d dot adzkiya at tudelft dot nl
  - A. Abate, a dot abate at tudelft dot nl

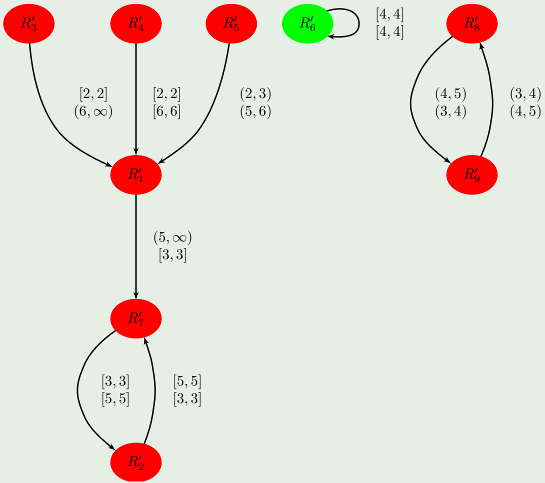
Page generated 2012-06-21 22:33:18 CEST, by jemdoc.

http://sourceforge.net/projects/verisimpl

MPL verification in practice

Example

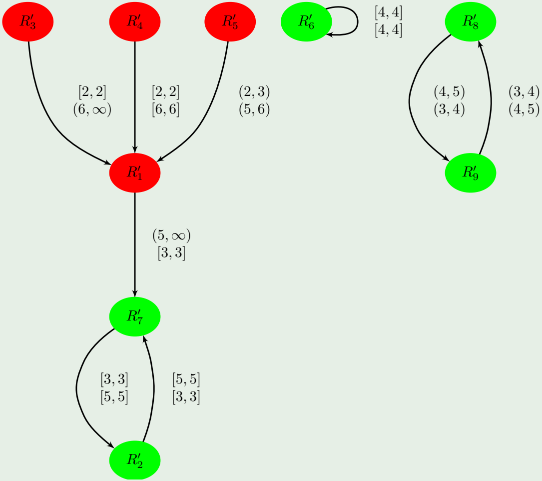
- automatically identify MPL eigenspace: $\bigvee_{\varphi \in L = AP} (\Box\varphi \ \wedge \ |\varphi| = 0)$



MPL verification in practice

Example

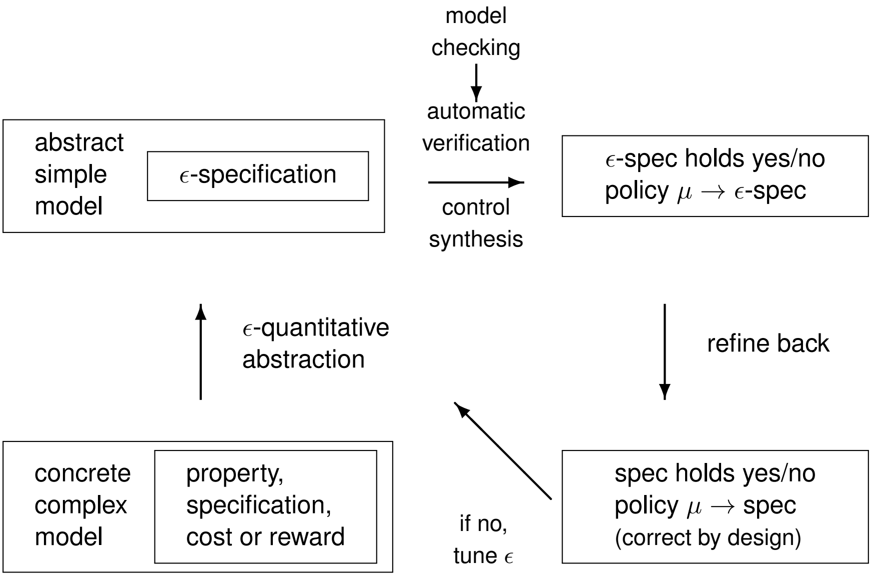
- automatically identify MPL periodic regime: $\Psi = \bigvee_{\varphi \in L = AP} \Box(\varphi \wedge \bigcirc^c \varphi)$



Outline

1. Formal abstractions for verification of complex models
2. Formal verification of stochastic hybrid systems
   - Analysis and control synthesis problems
   - Computable analysis and control synthesis via abstractions
3. Formal verification of max-plus linear models
   - Analysis and control synthesis problems
   - Computable analysis and control synthesis via abstractions
4. Concluding remarks

Formal abstractions for verification of complex models



Computable analysis and synthesis via formal verification

- theory: correct-by-design controller synthesis
- computations: coupling abstraction techniques with existing model checking software

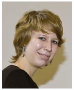
- SHS: composition, concurrency, continuous-time
- MPL models: probabilistic delays $\rightarrow$ SHS techniques

- applications: energy, biology, networked control systems

Acknowledgments

- main collaborators: J. Lygeros, M. Prandini, J.-P. Katoen, C. Tomlin, B. De Schutter
- students:

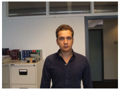
D. Adzkiya



S. Haesaert

S.E.Z. Soudjani

K. Staňková



I. Tkachev

M. Zamani

topics: stochastic hybrid systems, max-plus linear models

Thanks for your attention!

For more info:

`www.dcsc.tudelft.nl/∼aabate`
`a.abate@tudelft.nl`

Selected key references

– A. Abate, "Approximation Metrics based on Probabilistic Bisimulations for General State-Space Markov Processes: a Survey," Electronic Notes in Theoretical Computer Sciences, 2012, In Press.
– A. Abate, A. D'Innocenzo, and M.D. Di Benedetto, "Approximate Abstractions of Stochastic Hybrid systems," IEEE Transactions on Automatic Control, vol. 56, nr. 11, pp. 2688-2694, 2011.
– A. Abate, J.P Katoen, J. Lygeros, and M. Prandini, "Approximate Model Checking of Stochastic Hybrid Systems," European Journal of Control, nr. 6, pp. 624-641, 2010.
– A. Abate, J. Lygeros, and S. Sastry, "Probabilistic Safety and Optimal Control for Survival Analysis of *Bacillus Subtilis*," Systems and Control Letters, vol. 59, nr. 1, pp. 79-85, 2010.
– A. Abate, M. Prandini, J. Lygeros, and S. Sastry: "Probabilistic Reachability and Safety Analysis of Controlled Discrete-Time Stochastic Hybrid Systems," Automatica, vol. 44, nr. 11, pp. 2724-2734, Nov. 2008.
– I. Tkachev and A. Abate, "Computation of ruin probabilities for general discrete-time Markov models," Journal of Applied Probability. 2011, Under Review.
– S. Soudjani and A. Abate, "Adaptive and Sequential Gridding for Abstraction and Verification of Stochastic Processes," SIAM Journal on Applied Dynamical Systems. 2012, Under Review.
– I. Tkachev and A. Abate, "Characterization and computation of infinite horizon specifications over Markov processes," IEEE Transactions on Automatic Control. 2011, Under Review.
– I. Tkachev and A. Abate, "Regularization of Bellman equations for infinite-horizon probabilistic properties," Hybrid Systems: Computation and Control (HSCC 12), Beijing, PRC, Apr 2012.
– S. Soudjani and A. Abate, "Probabilistic Invariance of Mixed Deterministic-Stochastic Dynamical Systems," Hybrid Systems: Computation and Control (HSCC 12), Beijing, PRC, Apr 2012.
– A. D'Innocenzo, A. Abate and J.-P. Katoen, "Robust PCTL model checking," Hybrid Systems: Computation and Control (HSCC 12), Beijing, PRC, Apr 2012.
– I. Tkachev and A. Abate, "On infinite-horizon probabilistic properties and stochastic bisimulation functions," 50th IEEE Conference on Decision and Control and European Control Conference (CDC 11), Orlando, FL, December 2011, pp. 526–531.
– S. Soudjani and A. Abate, "Adaptive Gridding for Abstraction and Verification of Stochastic Hybrid Systems," Quantitative Evaluation of SysTems (QEST 11), Aachen (DE), Sept. 2011, pp. 59–69.
– A. Abate, J.-P. Katoen, and A. Mereacre, "Quantitative Automata Model Checking of Autonomous Stochastic Hybrid Systems," Hybrid Systems: Computation and Control (HSCC 11), Chicago, IL, April 2011, pp. 83 - 92.

Additional references

– J. Ding, M. Kamgarpour, S. Summers, A. Abate, J. Lygeros and C.J. Tomlin, “A dynamic game framework for verification and control of stochastic hybrid systems,” Automatica. 2011, Under Review.
– A. Abate and M. Prandini, “Approximate abstractions of stochastic systems: a randomized method,” Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, December 2011, pp. 4861–4866.
– A. Abate, A. D'Innocenzo, M.D. Di Benedetto and S. Sastry, “Markov Set-Chains as abstractions of Stochastic Hybrid Systems,” Hybrid Systems: Computation and Control (HSCC 08), Saint Louis (MS), April 2008.
– A. Abate, M. Prandini, J. Lygeros, and S. Sastry, “Approximation of General Stochastic Hybrid Systems by Switching Diffusions with Random Hybrid Jumps,” Hybrid Systems: Computation and Control, Saint Louis (MS), April 2008.
– A. Abate, S. Amin, M. Prandini, J. Lygeros, and S. Sastry, “Computational Approaches to Reachability Analysis of Stochastic Hybrid Systems,” Hybrid Systems: Computation and Control, Pisa (IT), April 2007.
– A. Abate, “Probabilistic Bisimulations of Switching and Resetting Diffusions,” 49th IEEE Conference of Decision and Control, Atlanta, GA, Dec. 2010, pp. 5918 - 5923.
– A. Abate, “A Contractivity Approach for Probabilistic Bisimulations of Diffusion Processes,” 48th IEEE Conference of Decision and Control, Shanghai, CN, Dec. 2009, pp. 2230-2235.
– A. Abate, M. Prandini, J. Lygeros, and S. Sastry, “An approximate dynamic programming approach to probabilistic reachability for stochastic hybrid systems, ” 47th IEEE Conference of Decision and Control, Cancun, MX, Dec. 2008, pp. 4018-4023.