Robust Optimization for Hybrid MDPs with State-dependent Noise

Zahra Zamani
ANU & NICTA
Canberra, Australia
zahra.zamani@anu.edu.au

Scott Sanner
NICTA & ANU
Canberra, Australia
ssanner@nicta.com.au

Karina Valdivia Delgado
University of Sao Paulo
Sao Paulo, Brazil
kvd@ime.usp.br

Leliane Nunes de Barros
University of Sao Paulo
Sao Paulo, Brazil
leliane@ime.usp.br

Abstract

Recent advances in solutions to hybrid MDPs with discrete and continuous state and action spaces have significantly extended the class of MDPs for which exact solutions can be derived, albeit at the expense of a restricted transition noise model. In this paper, we work around limitations of previous solutions by adopting a robust optimization approach in which Nature is allowed to adversarially determine transition noise within automatically-derived confidence intervals. This allows one to derive an optimal policy with an arbitrary (user-specified) level of success probability and significantly extends the class of transition noise models for which hybrid MDPs can be solved. This work also significantly extends results for the related "chance-constrained" approach in stochastic hybrid control to accommodate state-dependent noise. We demonstrate our approach working on a variety of hybrid MDPs taken from AI planning, operations research, and control theory, noting that this is the first time optimal robust solutions have been automatically derived for such problems.

1 Introduction

[**?**]

2 Robust Hybrid MDPs

We first formally introduce the framework of Robust Hybrid (discrete and continuous) Markov decision processes (RH-MDPs) extended from CSA-MDPs [**?**]. The optimal solution for this model is then defined by a Robust Dynamic Programming (RDP) approach.

2.1 Factored Representation

A RH-MDP is modelled using state variables $(\vec{b}, \vec{x}) = (b_1, \ldots, b_a, x_1, \ldots, x_c)$ where each $b_i \in \{0, 1\}$ $(1 \leq i \leq a)$ represents discrete boolean variables and each $x_j \in \mathbb{R}$ $(1 \leq j \leq c)$ is continuous. To model uncertainty in RH-MDPs we assume intermediate noise $\vec{n} = n_1, \ldots, n_e$ where each $n_l \in \mathbb{R}$ $(1 \leq l \leq e)$ is a continuous noise variable which can be state dependent. Both discrete and continuous actions are represented by the set $A = \{a_1(\vec{y}_1), \ldots, a_p(\vec{y}_d)\}$, where $\vec{y}_k \in \mathbb{R}^{|\vec{y}_k|}$ $(1 \leq k \leq d)$ denote continuous parameters for action a_k.

Given a state $(\vec{b}, \vec{x})$ and an executed action $a(\vec{y})$ at this state, a reward function $R(\vec{b}, \vec{x}, a, \vec{y})$ specifies the immediate reward at this state. The probability of the next state $(\vec{b}', \vec{x}')$ is defined by a joint state transition model $P(\vec{b}', \vec{x}' | \vec{b}, \vec{x}, a, \vec{y}, \vec{n})$ which depends on the current state, action and noise variables. We assume stochasticity in RH-MDPs using a state dependent noise function $N(\vec{n}, \vec{b}, \vec{x})$ where each noise variable (i.e. stochastic) n_l is bounded by some convex region on state variables $(\vec{b}, \vec{x})$. Non-deterministic constraint functions define the possible values for each n_L where we assume $+\infty$ for all illegal values of noise and $-\infty$ for legal values. We show later how this general representation allows RH-MDPs to naturally determine the minimum noise in any problem.

A policy $\pi(\vec{b}, \vec{x})$ at this state specifies the action $a(\vec{y}) = \pi(\vec{b}, \vec{x})$ to take at this state. An optimal sequence of finite horizon policies $\Pi^* = (\pi^{*,1}, \ldots, \pi^{*,H})$ is desired such that given the initial state $(\vec{b}_0, \vec{x}_0)$ at $h = 0$ and a discount factor $\gamma,\ 0 \leq \gamma \leq 1$, the expected sum of discounted rewards over horizon $h \in H; H \geq 0$ is maximized:

$$V^{\Pi^*}(\vec{b}, \vec{x}) = E_{\Pi^*}\left[\sum_{h=0}^{H} \gamma^h \cdot r^h \Big| \vec{b}_0, \vec{x}_0\right]. \qquad (1)$$

where r^h is the reward obtained at horizon h following the optimal policy.

Similar to the dynamic Bayes net (DBN) structure of CSA-MDPs [**?**] we assume *synchronic arcs* (variables that condition on each other in the same time slice) from $\vec{b}$ to $\vec{x}$ and from the noise variables $\vec{n}$ to both $\vec{b}$ and $\vec{x}$ but not within the binary $\vec{b}$ or continuous variables $\vec{x}$.Thus the factorized joint transition model is defined as

$$P(\vec{b}', \vec{x}' | \vec{b}, \vec{x}, a, \vec{y}, \vec{n}) = \prod_{i=1}^{a} P(b_i' | \vec{b}, \vec{x}, a, \vec{y}, \vec{n}) \prod_{j=1}^{c} P(x_j' | \vec{b}, \vec{b}', \vec{x}, a, \vec{y}, \vec{n}) \qquad (2)$$

For binary variables b_i $(1 \leq i \leq a)$ the conditional probability $P(b_i' | \vec{b}, \vec{x}, a, \vec{y}, \vec{n})$ is defined as conditional probability

functions (CPFs) which are not tabular. For continuous variables x_j ($1 \leq j \leq c$), the CPFs $P(x'_j|\vec{b}, \vec{b}', \vec{x}, a, \vec{y}, \vec{n})$ are represented with *piecewise linear equations* (PLEs) that condition on the action, noise, current state, and previous state variables with piecewise conditions that may be arbitrary logical combinations of $\vec{b}$, $\vec{b}'$ and linear inequalities over $\vec{x}$.

We allow the reward function $R(\vec{b}, \vec{x}, a, \vec{y})$ to be a general piecewise linear function (boolean or linear conditions and linear values) or a piecewise quadratic function of univariate state (current state or the next state) and a linear function of univariate action parameters. These constraints ensure piecewise linear boundaries that can be checked for consistency using a linear constraint feasibility checker, which we will later see is crucial for efficiency.

As a simple example, we consider the following rover problem to demonstrate the conditional probability forms for discrete and continuous variables as well as the noise function:

Example (Rover)**.** *A Rover state consists of its continuous position x along a given route. In a given time step, the rover may move a continuous distance $a \in [-10, 10]$. The rover receives its greatest reward for taking a picture in the region of $x \in [-40, 40]$. Sensory noise n on the rover results in a noisy transition to the next state. This noise is bounded $n \in [-5, 5]$ on the current location of the rover. Taking a picture $b = 1$ where $b \in \{0, 1\}$ occurs only if it has not been performed previously.*

We define the rover as a RH-MDP using the following piecewise dynamics and reward:

$$P(b' = 1|x, b, n) = \begin{cases} b \vee (x \geq -10 \wedge x \leq 10) : & 1.0 \\ \neg b \wedge (x < -10 \vee x > 10) : & 0.0 \end{cases} \quad (3)$$

$$P(x'|x, a, n) = \delta\left(x' - (x + n + a)\right) \quad (4)$$

$$R(x, b) = \begin{cases} x > 40 \vee x < -40 : & -\infty \\ b' \wedge \neg b \wedge x \geq -2 \wedge x \leq 2 : & 10 \\ b' \wedge b \wedge x \geq -2 \wedge x \leq 2 : & -1 \\ \neg b' \wedge x \geq -2 \wedge x \leq 2 : & -1 \end{cases} \quad (5)$$

Where the $\delta[\cdot]$ function ensures that the continuous CPF over x' integrates to 1.

2.2 Robust Dynamic Programming

Given the general dynamic programming approach for obtaining the optimal policy, we extend the value iteration algorithm [**?**] to a Robust dynamic programming algorithm for continuous states and actions. Initializing $V^0(\vec{b}, \vec{x}) = 0$) the algorithm builds the h-stage-to-go value functions $V^h(\vec{b}, \vec{x})$.

The quality $Q_a^h(\vec{b}, \vec{x}, \vec{y}, \vec{n})$ of taking action $a(\vec{y})$ in state $(\vec{b}, \vec{x})$ with noise parameters $\vec{n}$ and acting so as to obtain $V^{h-1}(\vec{b}, \vec{x})$ is defined as the following:

$$Q_a^h(\vec{b}, \vec{x}, \vec{y}, \vec{n}) = \max_{n_1, \ldots, n_l} N(\vec{n}, \vec{b}, \vec{x}) \cdot \left[R(\vec{b}, \vec{x}, a, \vec{y}) + \gamma \cdot \sum_{\vec{b}'} \int \left(\prod_{i=1}^{a} P(b'_i|\vec{b}, \vec{x}, a, \vec{y}, \vec{n}) \prod_{j=1}^{c} P(x'_j|\vec{b}, \vec{b}', \vec{x}, a, \vec{y}, \vec{n}) \right) V^{h-1}(\vec{b}', \vec{x}') d\vec{x}' \right] \quad (6)$$

Here the noise function $N(\vec{n}, \vec{b}, \vec{x})$ is is used to in-cooperate the noise variable $n_1, \ldots, n_l$ maximally in the original Q-value definition of value iteration. Given $Q_a^h(\vec{b}, \vec{x}, \vec{y}, \vec{n})$ for each $a \in A$, we can proceed to define the h-stage-to-go value function so as to minimize the noise as follows:

$$V^h(\vec{b}, \vec{x}) = \max_{a \in A} \max_{\vec{y} \in \mathbb{R}^{|\vec{y}|}} \min_{\vec{n} \mathbb{R}^{|\vec{n}|}} \left\{ Q_a^h(\vec{b}, \vec{x}, \vec{y}, \vec{n}) \right\} \quad (7)$$

The optimal policy in horizon h is also determined using the Q-function as below:

$$\pi^{*,h}(\vec{b}, \vec{x}) = \arg\max_a \arg\max_{\vec{y}} \arg\min_{\vec{n}} Q_a^h(\vec{b}, \vec{x}, \vec{y}, \vec{n}) \quad (8)$$

For finite-horizon RH-MDPs the optimal value function and policy are obtained up to horizon H. For infinite horizons, the optimal policy has finitely bounded value and value iteration terminates when two values are equal in consequent horizons ($V^h = V^{h-1}$). In this case $V^\infty = V^h$ and $\pi^{*,\infty} = \pi^{*,h}$.

Next we compute (6) and (7) using symbolic methods.

3 Symbolic Dynamic Programming (SDP)

In order to compute the equations above, we propose a *Robust symbolic dynamic programming* (RSDP) approach similar to [**?**]. This requires a value iteration algorithm proposed in 1 (`VI`) and a regression subroutine in Algorithm 2 In general we define symbolic functions to be represented in *case* form [**?**]:

$$f = \begin{cases} \phi_1 : & f_1 \\ \vdots & \vdots \\ \phi_k : & f_k \end{cases} \quad (9)$$

where ϕ_i are logical formulae defined over the state $(\vec{b}, \vec{x})$ and can include arbitrary logical $(\wedge, \vee, \neg)$ combinations of boolean variables and *linear* inequalities $(\geq, >, \leq, <)$ over continuous variables. The f_i may be either linear or quadratic in the continuous parameters with no discontinuities at partition boundaries. The transition and reward functions of the Rover example are all symbolic function examples.

While explaining the steps of the `VI` algorithm we show that all operations required to compute the optimal policy is supported by case representation and defined symbolically.

Initially the value function V^h is assigned to 0. For every horizon the h-stage-to-go value functions $V^h(\vec{b}, \vec{x})$ is computed. To follow the steps, we use the second iteration of the Rover example here. For simplification, we omit the boolean variable b of taking a picture and only use one noisy continuous variable x. We now perform steps 1–4 for $h = 2$.

(1) For every action, the function Q_a^h is computed. Line 6 refers to Algorithm 2 which has the main steps below:

(1a) Priming the current state variables (b_i, x_j) to build the next states (b'_i, x'_j) in the value function. This indicates that any occurrence of the current state in V^h is *symbolically substituted* with the next state variables that is $V'^h = V^h\sigma$ where $\sigma = \{b_i \setminus b'_i, x_j \setminus x'_j\}$ for all values of i and j. For the second iteration this step is equal to the following:

Algorithm 1: VI(CSA-MDP, H) $\longrightarrow (V^h, \pi^{*,h})$

1 **begin**
2 $V^0 := 0, h := 0$
3 **while** $h < H$ **do**
4 $h := h + 1$
5 **foreach** $a(\vec{y}) \in A$ **do**
6 $Q_a^h(\vec{y}, \vec{n}) :=$ Regress$(V^{h-1}, a, \vec{y})$
7 $Q_a^h(\vec{y}) := \min_{\vec{n}} Q_a^h(\vec{y}, \vec{n})$ *//Stochastic* min
8 $Q_a^h := \max_{\vec{y}} Q_a^h(\vec{y})$ *// Continuous* max
9 $V^h := \text{casemax}_a\, Q_a^h$ *// casemax all* Q_a
10 $\pi^{*,h} := \arg\max_{(a,\vec{y})} Q_a^h(\vec{y})$
11 **if** $V^h = V^{h-1}$ **then**
12 break *// Terminate if early convergence*
13
14 **return** $(V^h, \pi^{*,h})$
15 **end**

Algorithm 2: Regress$(V, a, \vec{y}) \longrightarrow Q$

1 **begin**
2 $Q =$ Prime(V) *// All* $b_i \to b_i'$ *and all* $x_i \to x_i'$
3 *// Continuous regression marginal integration*
4 **for** *all* x_j' *in* Q **do**
5 $Q := \int Q \otimes P(x_j' | \vec{b}, \vec{b}', \vec{x}, a, \vec{y}, \vec{n})\, d_{x_j'}$
6 *// Discrete regression marginal summation*
7 **for** *all* b_i' *in* Q **do**
8 $Q := \left[Q \otimes P(b_i' | \vec{b}, \vec{x}, a, \vec{y}, \vec{n}) \right] |_{b_i'=1}$
9 $\oplus \left[Q \otimes P(b_i' | \vec{b}, \vec{x}, a, \vec{y}, \vec{n}) \right] |_{b_i'=0}$
10 $Q := R(\vec{b}, \vec{x}, a, \vec{y}) \oplus (\gamma \cdot Q)$
11 *//* max*-in noise variables*
12 **for** *all* n_k *in* Q **do**
13 $Q_a^h(\vec{y}, \vec{n}) := \text{casemax}_{n_k}(Q, N(n_k, b_i', x_j'))$
14 **return** Q
15 **end**

$$Q = V^{'1} = \begin{cases} -40 \le x' \le 40 : & 10 \\ x > 40 \vee x' < -40 : & -\infty \end{cases}$$

(1b) Performing Regression for continuous variable x in line 5 and boolean variable b in lines 8–9. *Boolean restriction* $f|_{b=v}$ assigns the value $v \in \{0, 1\}$ to any occurrence of b in f.

The Binary operators of $\oplus$ and $\otimes$ (also $\ominus$ not defined here) on two case statements are done by taking the cross-product of the logical partitions of the two case statement and performing the corresponding operation on the resulting paired partitions. The cross-sum is defined as below:

$$\begin{cases} \phi_1 : & f_1 \\ \phi_2 : & f_2 \end{cases} \oplus \begin{cases} \psi_1 : & g_1 \\ \psi_2 : & g_2 \end{cases} = \begin{cases} \phi_1 \wedge \psi_1 : & f_1 + g_1 \\ \phi_1 \wedge \psi_2 : & f_1 + g_2 \\ \phi_2 \wedge \psi_1 : & f_2 + g_1 \\ \phi_2 \wedge \psi_2 : & f_2 + g_2 \end{cases}$$

The other operations of $\ominus$ and $\otimes$ are performed by subtracting or multiplying partition values. Note that some partitions may become inconsistent (infeasible) which are removed from the final result.

Continuous integration of $\int Q(x_j') \otimes P(x_j' | \cdots) dx_j'$ where results in the following according to the rules of integration:

$$\int f(x_j') \otimes \delta[x_j' - h(\vec{z})] dx_j' = f(x_j')\{x_j'/h(\vec{z})\}$$

Here $P(x_j' | \cdots)$ is in the form of $\delta[x_j' - h(\vec{z})]$ ($h(\vec{z})$ which is a case statement and $\vec{z}$ does not contain x_j'). The latter operation in the result indicates that any occurrence of x_j' in $f(x_j')$ is substituted with the case statement $h(\vec{z})$. For our example this step results in the following intermediate Q-value:

$$\begin{cases} -40 \le x' \le 40 : & 10 \\ x > 40 \vee x' < -40 : & -\infty \end{cases} \otimes \delta\left(x' - (x + n + a)\right) =$$
$$\begin{cases} -40 \le (x + a + n) \le 40 : & 10 \\ (x + n + a) > 40 \vee (x + n + a) < -40 : & -\infty \end{cases}$$

(1c) Multiplying the regression by the discount factor and adding the reward function in line 10. *Unary operations* such as scalar multiplication $\gamma \cdot Q$ (and also negation $-Q$) on case statements Q is performed by applying the operation to each Q_i ($1 \le i \le k$) while adding the reward is a binary $\oplus$:

$$Q = V^{'1} = \begin{cases} -40 \le (x + n + a) \le 40 : & 20 \\ (x + n + a) > 40 \vee (x + n + a) < -40 : & -\infty \\ x > 40 \vee x < -40 : & -\infty \end{cases}$$

(1d) Maximizing this result with the noise function in line 13. This step incorporates noise into the regressed Q-function consequently for each noise variable. Each noise variable assigns -∞ for legal values inside the boundary range +∞ for illegal values defined by the noise model $N(\vec{n}, \vec{b}, \vec{x})$. By maximizing in n_k all illegal values will remain +∞ since this is the maximum value compared to any other value and all legal values will be replaced by the regressed Q-value defined in step (1c) -∞ is less than any other Q-value so it is omitted in the maximization. The Rover example is redefined with this noise variable as below:

$$Q = \begin{cases} -5 \le n \le 5 : & +\infty \\ (-40 \le (x + n + a) \le 40) \wedge \neg(-5 \le n \le 5) : & 20 \\ \neg(-40 \le (x + n + a) \le 40) \wedge \neg(-5 \le n \le 5) : & -\infty \\ (x > 40 \vee x < -40) \wedge (-5 \le n \le 5) : & +\infty \\ (x > 40 \vee x < -40) \wedge \neg(-5 \le n \le 5) : & -\infty \end{cases}$$

(2) Naturally a noisy process aims to minimize the noise to reach robustness Thus the regressed stochastic $Q_a^h(\vec{y}, \vec{n})$ from Algorithm 2 is now minimized over the noise variables $\vec{n}$ in line 7. Intuitively this continuous minimization will never choose +∞ as there is always some value smaller which insures that the transitioned model never chooses illegal values. All legal Q-values are considered in the minimization step to find the value corresponding

to the minimum noise. Each partition i of this intermediate Q is considered for a continuous minimization separately with the final result a $\mathrm{casemin}$ on all the individual minimum results $\mathrm{casemin}_i \min_n \phi_i(\vec{b},\vec{x},\vec{n}) f_i(\vec{b},\vec{x},\vec{n})$. We demonstrate the steps of this algorithm for the second partition of Q defined as:

$$\min_n(-40 \leq (x+n+a) \leq 40) \wedge \neg(-5 \leq n \leq 5) : 20$$

For each partition the logical constraints are used to derive the (a) lower bound on n (LB = -5, $-40-a-x$); (b) upper bound on n (UB = 5, $40-a-x$) and (c) constraints independent of n (IND). In case of several bounds on n the maximum of all lower bounds and the minimum of all upper bounds is desired. A *symbolic case maximization* on two case statements of $(\phi_i : f_i)$ and (ψ_i, g_i) where $(i \in \{1,2\})$ is performed below.

$$\mathrm{casemax} = \begin{cases} \phi_1 \wedge \psi_1 \wedge f_1 > g_1 : f_1 \\ \phi_1 \wedge \psi_1 \wedge f_1 \leq g_1 : g_1 \\ \phi_1 \wedge \psi_2 \wedge f_1 > g_2 : f_1 \\ \phi_1 \wedge \psi_2 \wedge f_1 \leq g_2 : g_2 \\ \phi_2 \wedge \psi_1 \wedge f_2 > g_1 : f_2 \\ \phi_2 \wedge \psi_1 \wedge f_2 \leq g_1 : g_1 \\ \phi_2 \wedge \psi_2 \wedge f_2 > g_2 : f_2 \\ \phi_2 \wedge \psi_2 \wedge f_2 \leq g_2 : g_2 \end{cases}$$

Thus the bounds are defined as below:

$$LB = \begin{cases} a+x < -35 : & -40-x-a \\ a+x \geq -35 : & -5 \end{cases}$$

$$UB = \begin{cases} a+x < 35 : & 5 \\ a+x \geq -35 : & 40-a-x \end{cases}$$

The minima points of upper and lower bounds are evaluated for the leaf value which equals to substituting the bounds instead of the noise variable n in the leaf function. The minimum of these two evaluations are then stored, note that in our example the leaf is a constant 20 value which is not effected by this step.
Natural constraints on bounds $LB \leq UB$ and the IND constraints are also considered for the minimization on a single partition to obtain:

$$Q = \begin{cases} (-40 \leq x \leq 40) \wedge (-45 \leq (x+a) \leq 45) : & 20 \\ (-40 \leq x \leq 40) \wedge \neg(-45 \leq (x+a) \leq 45) : & +\infty \\ (x > 40 \vee x < -40) : & +\infty \end{cases}$$

The final result of a continuous minimization is a $\mathrm{casemin}$ over all partitions which results in the following Q-value:

$$Q = \begin{cases} (-40 \leq x \leq 40) \wedge (-35 \leq (x+a) \leq 35) : & 20 \\ (-40 \leq x \leq 40) \wedge \neg(-35 \leq (x+a) \leq 35) : & -\infty \\ (x > 40 \vee x < -40) : & -\infty \end{cases}$$

3 The resulting Q-value with minimal noise is maximized over the continuous action parameter in line 8; a symbolic continuous maximization operation, the major contribution of [**?**]. The resulting Q_a^h can be determined as the following:

$$Q = \begin{cases} (-40 \leq x \leq 40) : & 20 \\ (x > 40 \vee x < -40) : & -\infty \end{cases} \quad (10)$$

4 A discrete $\mathrm{casemax}$ on the set of discrete actions for all Q-functions defines the final V and the optimal policy is defined as the $\arg\max$ over the set of discrete and continuous actions on Q. In our example the final value $V^h = Q^h$ because there is only one single discrete action.

To implement the case statements efficiently with continuous variables, extended Algebraic Decision diagrams (XADDs) are used from [**?**] which is extended from ADDs [**?**]. Unreachable paths can be pruned in XADDs using LP solvers and all operations including the continuous minimization can be defined using XADDs by treating each path from root to leaf node as a single case partition with conjunctive constraints, $\min_n$ is performed at each leaf subject to these constraints and all path $\min_n$'s are then accumulated via the $\mathrm{casemin}$ operation to obtain the final result.

4 Empirical Results

5 Related Work

Hybrid systems are a class of dynamical systems that involve both continuous and discrete dynamics. The dynamics of the continuous variables are defined typically through differential equations and the evolution of the discrete variables through finite state machines, Petri nets or other abstract computational machines. One accepted manner to model hybrid systems is using hybrid automata that represents, in a single formalism, the discrete changes by automata transitions and the continuous changes by differential equations [**?**; **?**]. One special class of hybrid systems are the switched linear systems that have a collection of subsystems defined by linear dynamics (differential equations) and a switching rule that specifies the switching between the subsystems [**?**].

One problem that has been studied in the area of hybrid systems is the verification of the safety property, that tries to proof that the system does not enter in unsafe configurations from an initial configuration [**?**]. Then, we say that the system satisfies the safety property if all reachable states are safe [**?**]. There are many tools for the automatic verification of hybrid systems such as HyTech [**?**], KRONOS [**?**], PHAVer [**?**] and HSOLVER [**?**]. All the techniques rely on the ability to compute reachable sets of hybrid systems. For example, HyTech, a symbolic model checker, automatically computes reachable sets for linear hybrid automata, a subclass of hybrid automata. HyTech can also return the values of design parameters for which this automata satisfies a temporal-logic requirement [**?**]. Some examples of verification of hybrid systems can be found in [**?**; **?**].

Another challenging topic in hybrid systems is to evaluate the effect of the hybrid controller on the systems operation, i.e., to solve the controllability problem for hybrid systems [**?**]. A hybrid system is called hybrid controllable if, for any pair of valid states, there exists at least one permitted control sequence (correct control-laws) between them [**?**; **?**]. The general controllability problem of hybrid systems is NP hard [**?**]. However, for special classes of hybrid systems, some necessary and sufficient conditions for controllability

were obtained in [**?**; **?**; **?**; **?**; **?**]. For example, by employing algebraic manipulation of system matrices, a sufficient and necessary condition for the controllability analysis of a class of piecewise linear hybrid systems is given in Zhenyu:2003. This class is called controlled switching linear hybrid system and have the following properties: all mode switches are controllable, the dynamical subsystems within each mode has a LTI form, the admissible operating regions within each mode is the whole state space, and there are no discontinuous state jumps. The controllability test for this class of hybrid system can be determined based on the system matrices. In [**?**] is proposed an approach for controllability analysis of a class of more complex hybrid systems. This approach uses a discrete-path searching algorithm that integrates global reachability analysis at the discrete event system level and a local reachability analysis at the continuous level. This method cannot guarantee the existence of a solution for an arbitrary hybrid system [**?**].

Much of the work on hybrid systems has focused on deterministic models without allowing any uncertainty. In practice, there are real world applications where the environment is inherent uncertainty. To cope with this, the stochastic hybrid systems was proposed. Stochastic hybrid systems allow uncertainty (1) replacing deterministic jumps between discrete states by random jumps or (2) replacing the deterministic dynamics inside the discrete state by a stochastic differential equation or (3) combinations of 1 and 2 [**?**]. A critical problem in this type of systems is the verification of reachability properties because it is necessary to cope with the interaction between the discrete and continuous stochastic dynamics, in this case it is computed the probability that the system satisfies the property [**?**]. Related with the concept of verification of safety property, in stochastic hybrid systems, the system tries to maximize the probability that the execution will remain in safe states as long as possible [**?**].

Chance-constrained predictive stochastic control of dynamic systems characterizes uncertainty in a probabilistic manner, and finds the optimal sequence of control inputs subject to the constraint that the probability of failure must be below a user-specified threshold [**?**]. This constraint is known as a chance constraint [**?**] and is used to define stochastic robustness.

A great deal of work has taken place in recent years relating to chance-constrained optimal control of linear systems subject to Gaussian uncertainty in convex regions [**?**; **?**; **?**] and linear systems in nonconvex regions [**?**; **?**]. The approach in [**?**] uses samples, or particles, to approximate the chance constraint, and hence does not guarantee satisfaction of the constraint. It applies to arbitrary uncertainty distributions and is significantly more computationally intensive than others. The approach proposed in [**?**] uses analytic bound to ensure satisfaction of the constraint and applies for linear-Gaussian systems.

6 Concluding Remarks