Uncertainty-aware Exploration in Model-based Testing

Matteo Camilli*, Angelo Gargantini †, Patrizia Scandurra †, Catia Trubiani ‡
* Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano, Italy
Email: matteo.camilli@unibz.it
† DIGIP, Universitá degli Studi di Bergamo, Bergamo, Italy
Email: {angelo.gargantini,patrizia.scandurra}@unibg.it
‡ Gran Sasso Science Institute, L'Aquila, Italy
Email: catia.trubiani@gssi.it

***Abstract*—Modern software systems operate in complex and changing environments and are exposed to multiple sources of uncertainty. Testing methods shall be tailored to uncertainty as a first-class concern in order to quantify it and deliver increased confidence in the level of assurance of the final product. In this paper, we introduce novel model-based exploration strategies that generate test cases targeting uncertain components of the system under test. Our testing framework leverages Markov Decision Processes as modeling formalism of choice. The tester explicitly specifies uncertainty by means of beliefs attached to transition probabilities. The structural properties of the model and the uncertainty specification are then exploited to drive the test case generation process. Bayesian inference is used to achieve this objective by updating the initial beliefs through the evidence collected by testing. The proposed uncertainty-aware test selection strategies have been systematically evaluated on three realistic benchmarks and nine synthetic systems exhibiting up to $10k$ model transitions. We demonstrate the effectiveness of the novel strategies with well-established metrics. Results show they outperform existing testing methods with a gain up to $2.65\times$ in terms of accuracy of the inference process.**

***Index Terms*—Model-based testing, Probabilistic systems, Uncertainty quantification, Bayesian inference**

I. Introduction

Model-based testing (MBT) relies on explicit models that encode the intended behaviors of a system under test (SUT) and/or the behavior of its environment [1], [2], [3]. However, modern software systems are exposed to sources of uncertainty that can arise from an ambiguous specification of the system, and execution environments' characteristics that are unknown before the system is running [4]. To deal with this challenge [5], testing techniques have been tailored not only to detect failures, but also to actively learn the SUT dynamics and its surroundings in order to verify initial hypothesis [6], [7]. In particular, endowing conventional software testing with techniques and practices able to model, quantify, and mitigate uncertainty is becoming crucial [8], [9], [10].

The research community is recently investigating the possibility of endowing MBT approaches with awareness of possible sources of uncertainty [11], [12]. Existing approaches focus on spotting unknown occurrences of environmental uncertainties in Cyber-Physical Systems (CPS) [13], [14]. An initial attempt to explicitly model and quantify uncertainty with Markov Decision Processes (MDP) is shown in [15], [16], however, fixed reward values are used to generate tests. To the best of our knowledge, there is no approach leveraging fine-grained characteristics of existing uncertainties to drive MBT. Thus, further investigation on this topic is required.

The goal of our research is to introduce and compare novel MBT strategies that are tailored to uncertainty quantification and incremental refinement of an initial underspecified MDP. To achieve this objective, the testing process embeds awareness on the sources of uncertainty and quantifies it by applying Bayesian inference [17]. The uncertainty-aware MBT strategies proposed in this paper are: (i) *History*, that tracks information about visited model regions to select those test cases that increase the probability of testing unexplored uncertain components of the SUT; (ii) *Distance*, that uses information on the SUT branching points that are more likely to execute components associated with a higher level of uncertainty; and (iii) *Frequency*, that considers the likelihood of using the different components, hence the actual usage of the SUT is exploited for the selection of tests. This paper provides the following main contributions:

- novel MBT strategies that take into account uncertainty-related characteristics of the SUT;
- extensive evaluation of the effectiveness of these strategies under bounded effort, by comparing the updated beliefs after testing and the accuracy of the inference process to quantify existing uncertainties.

As running example, we adopt a CPS benchmark called SafeHome [9]. The empirical evaluation has been performed on three realistic systems from literature [9], [18], [19] and nine synthetic systems generated from pseudorandom MDP models with the goal of increasing structural complexity (from 250 to $10k$ model transitions). The empirical evaluation shows that the *Distance* strategy yields the smallest relative error and the highest fault detection rate when testing the selected realistic systems. The uncertainty quantification capability increases when increasing structural complexity of the synthetic systems. With complex models, the *Distance* strategy is likely to be the best choice, with few exceptions when the overall level of uncertainty is either very low or very high and when the number of possible inputs is high. In this case, the *Frequency* strategy is likely to be superior.

The remainder of the paper is as follows. Sect. II provides background concepts. Sect. III describes the SafeHome running example. Sect. IV provides an overview of our testing framework and Sect. V presents our novel MBT strategies. Sect. VI reports an extensive empirical evaluation, all experiments and replication data is publicly available[1]. Sect. VII discusses related work, and Sect. VIII concludes the paper.

II. Preliminaries

This section introduces basic concepts and techniques used throughout the paper: Markov Decision Processes (MDPs) with rewards, Bayesian inference, and online MBT of probabilistic systems.

A. Markov Decision Processes and Rewards

MDPs [20], [21] represent a widely used formalism for modeling systems exhibiting both probabilistic and nondeterministic behavior. Formally, a MDP is defined as a tuple $\mathcal{M} = (S, s_0, A, \delta)$, where:

- S is a finite set of states ($s_0 \in S$ initial state);
- A is a finite alphabet of actions;
- $\delta : S \times A \rightarrow Dist(S)$ is a partial probabilistic transition function. *Dist(S)* represents the set of discrete probability distributions over a countable set S.

State transitions occur in two steps: i) a nondeterministic choice among the actions from state s: $A(s) = \{a \in A : \exists \delta(s, a)\}$; ii) a stochastic choice of the successor state s', according to the probability distribution δ, such that $\delta(s, a)(s')$ represents the probability that a transition from s to s' occurs when a happens. The function δ satisfies $\sum_{s'} \delta(s, a)(s') = 1$, for each source state s, action a and target state s'.

MDPs can be augmented with *reward*s to quantify a benefit (or loss) due to the sojourn in a specific state or to the occurrence of a certain state transition. A reward is a non-negative value assigned to states and/or transitions that can represent information such as average execution time, power consumption or usability. A reward structure associated with a MDP $\mathcal{M}$ is defined as a pair $r = (r_s, r_a)$ composed of a *state* reward function $r_s : S \rightarrow \mathbb{R}_{\geq 0}$ and an *action* reward function $r_a : S \times A \times S \rightarrow \mathbb{R}_{\geq 0}$ that assigns rewards to states and transitions, respectively. Given a reward structure, a common problem is to find a policy function π that specifies the action $\pi(s)$ chosen by a decision maker when state s holds. The best policy π^* maximizes some function of the cumulated rewards, typically the expected discounted sum over a potentially infinite path. Namely, given a reward structure r, π^* can be computed solving a *dynamic decision problem* [20]. The best policy π^* returns for each state s the action that allows the cumulated reward to be maximized.

[1]The dataset containing experimental results is available at https://doi.org/10.5281/zenodo.4095279. The software used to obtain raw data is an open source project available at https://github.com/SELab-unimi/mbt-module.

B. Bayesian Inference

A very common goal in statistics is to learn about one (or more) uncertain parameter(s) θ describing some details of a stochastic phenomenon of interest. To learn about θ, we observe the phenomenon and collect a data sample $y = (y_1, y_2, ..., y_n)$ to compute the conditional density $f(y|\theta)$ of the observed data given θ, i.e., the *likelihood* function. The Bayesian inference approach consists of taking into account the hypothesis (or assumptions) about θ. This information is often available from external sources, such as expert information based on past experience or previous studies [22]. The hypothesis is given in probabilistic terms distribution $f(\theta)$, so called *prior*. The Bayes' theorem formulation given below defines how the prior and the likelihood can be combined to obtain the *posterior* distribution:

$$\textit{Posterior} \propto \textit{Likelihood} \cdot \textit{Prior} \tag{1}$$

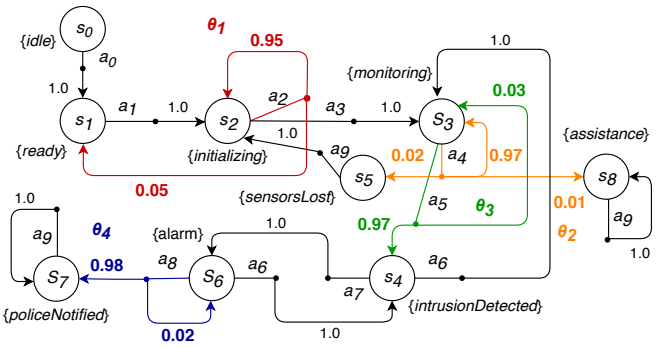
The *posterior* $f(\theta|y)$ describes the best knowledge of the true value of θ, given the data sample y. It can be used in turn to perform point and interval estimation of the uncertain parameters. The estimation yields the notion of *updated beliefs*. As described in [17], this is typically addressed by summarizing the distribution through the posterior *mean* and the smallest possible credible region of 0.95 probability, called *Highest Density Region* (HDR). This region is defined as the set of θ values, such that $\text{HDR}_\theta = \{\theta : f(\theta|y) \geq 0.95\}$. The HDR contains the values considered most likely a posteriori (i.e., credible values having the highest density). The magnitude of the region, denoted as $\|\text{HDR}_\theta\|$, is traditionally used in Bayesian statistics as a measure of the highest possible accuracy in the estimation [22]. Namely, it represents the confidence of the inference process, i.e., the smaller the magnitude, the higher the confidence.

III. A Running Example

To illustrate our novel testing methods, we adopt the *SafeHome* case study, i.e., an open-source security system borrowed from [9]. It is in charge of controlling and configuring alarms and sensors that implement some safety features, e.g., the intrusion detection.

Fig. 1 shows the high-level behavior of the system modeled with MDP. After the setup phase, the system exhibits three main phases: *initializing*, *monitoring* and *alarm*, in charge of sensor initialization, detection, and alarm handling, respectively. Annotations follow the standard notation "[pre-condition] trigger / post-condition" and provide guidance on the interpretation of the MDP model. As an example, from state s_2 (during *monitor initialization*), the SafeHome system tries to initialize all the available sensors by executing the a_2 action, i.e., the trigger of *initSensors*. If the task succeeds, the sensors are correctly registered and the a_3 action can be executed (i.e., the pre-condition *initialized* holds) to proceed towards the *monitoring* and *alarm* phases.

According to [23], sources of uncertainty in CPSs affect the behavior of the SUT at different levels: i) *application* level, due to events/data originating from software components



source	target	[pre-condition] trigger / post-condition
s_0	s_1	[systemOk] a_0: activate / ready
s_1	s_2	[ready] a_1: startInit / monitor
s_2	s_2	[!initialized] a_2: initSensors / registered
s_2	s_1	[!initialized] a_2: initSensors / failed
s_2	s_3	[initialized] a_3: startMonitoring / monitoring
s_3	s_5	[timeout] a_4: sensorsCheck / !sensorsOk
s_3	s_3	[timeout] a_4: sensorsCheck / sensorsOk
s_3	s_8	[timeout and tooManyFailures] a_4: sensorsCheck / !sensorsOk
s_5	s_2	[sensorsLost] a_9: resetSensors / !registered
s_8	s_8	[assistance] a_9: wait / assistance
s_3	s_3	[sensorsOk] a_5: intrusionOccurred / !sensed
s_3	s_4	[sensorsOk] a_5: intrusionOccurred / sensed
s_4	s_3	[!verified] a_6: cancel / setTimerOn
s_4	s_6	[verified] a_7: turnAlarmOn / setTimerOn
s_6	s_4	[!timeout] a_6: cancel / setTimerOff
s_6	s_6	[timeout or tooManyDetections] a_8: notify / success
s_6	s_7	[timeout or tooManyDetections] a_8: notify / networkError
s_7	s_7	[notificationSent] a_9: wait / notificationSent

Fig. 1: MDP model of the SafeHome system

running upon physical units of the CPS; *ii*) *infrastructure* level, due to data transmission through networking and/or cloud infrastructure; *iii*) *integration* level, due to interactions among physical units at either application level or infrastructure levels. To exemplify some of these uncertainties, let us consider the following scenario. When the system is in state s_3, it means that *monitoring* holds, sensors can send the a_5: *intrusionOccurred* trigger to the security system that makes the alarm ring via the effect of a_7: *turnAlarmOn* attached to the outgoing transition of the state s_4 *intrusionDetected*. Nevertheless, the intrusion detection capability is affected by uncertainty at integration level. This capability is influenced by the interaction of sensors and their individual ability of correctly sensing the physical environment. Thus, the a_5 action leads to either state s_4 (i.e., the intrusion has been sensed) or state s_3 (i.e., the intrusion has not been sensed) with a substantial degree of uncertainty. This uncertain outcome is explicitly represented by uncertain probability values (i.e., 0.97 and 0.03, respectively), as shown in the arcs of the MDP model. We refer to a set of uncertain probability values associated with a state-action pair in the model as *uncertain region* and we denote it as θ_i. Note that the disjoint union of all θ_i is θ, i.e., the set of uncertain model parameters. The full list of uncertain regions (and affected levels) of the SafeHome example is reported in Table I.

IV. Approach Overview

Our approach adopts online (or on-the-fly) MBT to drive the selection of tests from an MDP model by stochastically sampling its state space. The *functional* evaluation procedure adopted in our framework is based on a *conformance game* approach [24]. Beside the conformance game, our focus is the application of statistical inference during testing to incrementally refine uncertain beliefs of an initial underspecified MDP model. We make use of Bayesian inference (while gathering evidence from test executions) to compute the posterior density function of uncertain/unknown θ parameters of the MDP.

TABLE I: Uncertain regions

region	state-action	affected level	target states	probability values
θ_1	s_2-a_2	integration	s_2, s_1	$0.95, 0.05$
θ_2	s_3-a_4	integration	s_3, s_4	$0.03, 0.97$
θ_3	s_3-a_5	application	s_3, s_5, s_8	$0.01, 0.97, 0.02$
θ_4	s_6-a_8	infrastructure	s_6, s_7	$0.02, 0.98$

Our approach relies on the assumption that a partial specification of the SUT is available. Namely, the state-action space is known while transition probabilities can be unknown/uncertain. Thus, design-time uncertainty affects a subset of model parameters. Furthermore, we assume that we can anticipate the location (i.e., which model parameters are uncertain/unknown). These assumptions are valid in many practical cases as described in [19]. The advantages (and costs) of modeling in testing are discussed in many existing papers [5]. We consider this latter point outside the scope of this paper.

Fig. 2 provides an overview of our approach, where numbered labels refer to the major components detailed in the following. The starting point is a *Modeling module* (1) that allows the SUT behavior to be specified as a *MDP model* (2) through a simple textual Domain Specific Language (DSL). This language is also used to define the uncertain parameters by annotating MDP transitions with initial (a priori) hypothesis, given in terms of prior density functions. The *priors* (3) describe the modeler's beliefs on the uncertain transition probabilities. In addition to that, the DSL permits the declaration of a number of controllable APIs and observable outcomes. We adopt the approach introduced in [25] to distinguish between controllable behavior from the tester (i.e., the environment, such as user requests) and observable behavior from the running software system.

The DSL allows the modeler to map model elements and software components. More precisely, the modeler uses the DSL to define a model-system *binding* that provides the framework with a high-level view of the SUT behavior at the abstraction level of the MDP model as follows. Arbitrary input data for the system is associated with each MDP action a denoted as $\mathcal{I}(a)$. Input data is a vector $\vec{v}_{in}$ of parameters provided to a controllable API associated with each MDP state and denoted by $\mathcal{H}(s)$. Arbitrary pre- and post- conditions are then associated with MDP transitions. Namely, $\mathcal{P}re(s,a)$ must hold for $\mathcal{I}(a)$ and $\mathcal{P}ost(s,a)$ must hold for $\vec{v}_{out}$, i.e., the output obtained by executing $\mathcal{H}(s)$ with input $\mathcal{I}(a)$. The binding is then used by the framework to automatically generate a *test harness* (4) used by the *MBT module* to carry out the conformance game upon the SUT (5).

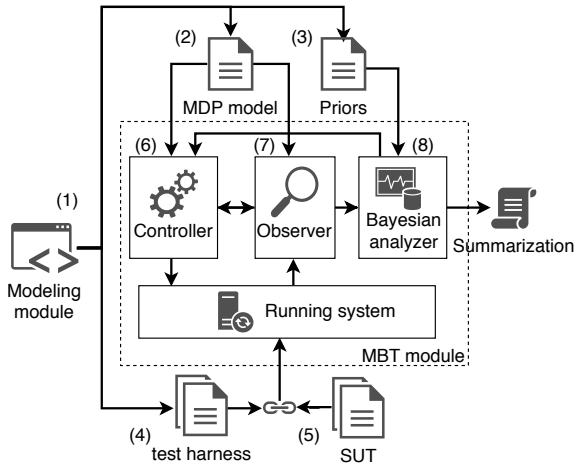


Fig. 2: Uncertainty-aware MBT framework

The software tools implementing our approach have been released as publicly available open-source software using the Java language for the MBT module and the Xtext/Xtend [26] framework to develop the Modeling module. However, the approach is general and does not refer to any specific feature of our programming language of choice. Therefore, it can be applied (with limited technological modifications) to other languages. In the following we provide further details on the major components of the MBT module: the *Controller* (6) and the *Observer* (7), and the *Bayesian analyzer* (8).

A. Controller and Observer

The aforementioned conformance game is carried out by the Controller and the Observer components. These software modules verify the existence of a *conformance relation* between the model and the SUT, formalized by means of the notions of *alternating simulation* [27] and *refinement* [24].

From an operational perspective, the conformance game starts from the initial state of the model, and it consists of a sequence of steps. For each step of the game, the Controller makes its own move: it chooses an available action in $A(s)$ from the current state s of the model, according to the adopted *test selection strategy*. Then, it uses the corresponding controllable API $\mathcal{H}(s)$ to supply the input $\mathcal{I}(a)$ to the running system. During execution, the test harness provides a serialized view of the observable behavior resulting from the execution of the SUT in response of the external stimuli. Thus, the observer, taking this information as input, makes its own move: it evaluates the pre-condition $\mathcal{P}re(s,a)$ on the supplied input. If the pre-condition holds, then it determines the target state s', such that the post-condition $\mathcal{P}ost(s,a)$, evaluated on the observed output, holds. Whenever a pre-condition does not hold or does not exist a target state s' such that the post-condition holds, there is a conformance failure. The game continues until the Controller decides to end the game (i.e., a termination condition has been reached) or a conformance failure is found (i.e., the output produced by the SUT is not predictable by the model).

B. Bayesian analyzer

During the conformance game, the Observer feeds the Bayesian analyzer to carry out statistical hypothesis testing. Namely, starting from the priors defined by the modeler, we incrementally update the knowledge about the uncertain parameters taking into account the evidence gathered during testing by applying Bayesian inference (see Eq. 1). In the following we provide a brief overview on the statistical machinery used to perform this activity, but we refer the reader to [17] for more details.

*Dirichlet distribution*s [28] are commonly used in Bayesian statistics as prior density functions. In particular, the Dirichlet distribution is the natural conjugate prior of the *categorical distribution*: a discrete probability distribution describing the possible outcomes of a random variable that can assume one of k possible values (i.e., categories), having each category associated with a specific probability. In our context, we use Dirichlet distributions as conjugate priors for the uncertain transition probabilities of a MDP model. Namely, the prior knowledge on transition probabilities $p_i^a = (p_{i,j}^a, ..., p_{i,k}^a)$, where $p_{i,j}^a$ is the probability to observe a transition from s_i to s_j when the action a is chosen, is described by letting p_i^a have a Dirichlet distribution with concentration parameters α_i as follows:

$$p_i^a \sim Dir(\alpha_i), \text{ where } \alpha_i = (\alpha_{i,j}, ..., \alpha_{i,k}) \quad (2)$$

The observer component collects statistics on the occurring model transitions in order to update the prior knowledge. More precisely, it collects a sample y that yields for each i, j, a, the occurrences $n_{i,j}^a$ from s_i to s_j, when the action a is selected. Given the sample y, the posterior distribution is also a Dirichlet distribution and can be computed very efficiently as follows:

$$p_i^a | y \sim Dir(\alpha_i'), \text{ where } \alpha_i' = (\alpha_{i,j}+n_{i,j}^a, ..., \alpha_{i,k}+n_{i,k}^a) \quad (3)$$

When little information is available, a natural choice is to use a *uninformative* prior with $\alpha_{i,j}^a = 1/2, \forall i, j, a$. Otherwise, when past experience is available, it is possible to use a prior having $\alpha_{i,j} = n_{i,j}^a$. For instance, considering the SafeHome case study (see Sect. III) we describe the hypothesis on θ_3 with a Dirichlet prior and concentration parameters equal to the following values: $(\alpha_{4,4} = 970, \alpha_{4,6} = 20, \alpha_{4,9} = 10)$, if in our past experience we observed 970 transitions from s_4 to s_4, 20 transitions from s_4 to s_6, and 10 transitions from s_4 to s_9, in a sample of $1k$ observations.

The online MBT process calibrates the uncertain θ parameters using the posterior mean and the HPD region, as introduced in Sect. II. The intuition behind our proposal is to take advantage of the specification of uncertain parameters to drive test case generation.

V. Test Selection Strategies

To describe our strategies, we firstly introduce the notion of *uncertainty-aware* reward structure, motivated by the practical need to identify model actions that maximize the probability

of exploring uncertain state transitions (associated with θ parameters). The uncertainty-aware reward structure is formally defined as follows:

Definition 1 (uncertainty-aware reward structure): Given a MDP model (S, s_0, A, δ) and a set of uncertain parameters $\theta_i \subseteq \theta$, the *uncertainty*-aware reward structure is $u = (u_s, u_a)$, s.t.,

- $u_s(s) = \begin{cases} k & \exists a \in A(s), s' \in S : \delta(s,a)(s') \in \theta_i \\ 0 & \text{otherwise} \end{cases}$
- $u_a(s,a,s') = \begin{cases} k & \delta(s,a)(s') \in \theta_i \\ 0 & \text{otherwise} \end{cases}$

where $k \in \mathbb{N}_{>0}$.

The rationale behind this definition is to assign a high and fixed reward value (k) to uncertain state transitions (and a low reward value to the other model elements). The best exploration policy that maximizes the expected cumulated uncertainty-aware rewards is then computed by applying dynamic programming, as anticipated in Sect. II.

Intuitively, parameters in θ_i for each i, compose the uncertain regions (θ_1, θ_2, etc. in Table. I) and the set of best policies (π_1^*, π_2^*, etc., respectively) maximizes the probability to reach each one of them. Thus, for each model state, we have in general multiple choices to act optimally towards different uncertain model regions. For example, from s_2 we might select either a_2 or a_3, depending on the target uncertain region, i.e., either θ_1 or θ_2, respectively. The number of alternative choices leading to different testing scenarios depends on the model complexity and/or the number of uncertain regions. Thus, multiple uncertainty-aware testing methods are likely to act differently in terms of delivered confidence.

After computing the best policies, our MBT algorithm makes dynamic (on-the-fly) choices for exploring the uncertain regions, and these choices are regulated by the adopted *test selection strategy*. Such a strategy provides control over test scenarios by selecting actions during testing based on the following probabilistic function:

$$\mathcal{P}(s,a) = \begin{cases} 0 & \omega(s,a) = 0 \\ \omega(s,a) / \sum_{a' \in A(s)} \omega(s,a') & \textit{otherwise} \end{cases} \quad (4)$$

where ω represents a per-state weight function that maps a state s and an action a to a value in $\mathbb{R}_{\geq 0}$. The weight ω is used to drive the direction of the exploration, i.e., it can be used to selectively increase or decrease the probability of certain actions depending on different model-based exploration strategies.

In the following, we describe the strategies currently implemented in our framework. The flat strategy is used as baseline (Sect. V-A) and the novel strategies are: *history* (Sect. V-B); *distance* (Sect. V-C); *frequency* (Sect. V-D); and we also propose a combination of distance and frequency (Sect. V-E).

A. Flat Strategy

The flat strategy represents a pseudo-random test selection criterion that allows to select the actions depending on a statically defined weight function ω^{RT}, where *RT* is the acronym for random testing, and it maps a pair (s,a) to a fixed value. The idea is to choose among the available actions by using a discrete uniform distribution whereby all the available actions, leading to uncertain model regions, have equal weight. Given the set of best policies $\{\pi_i^*\}$, the function ω^{RT} is defined as follows:

$$\omega^{RT}(s,a) = \begin{cases} 1 & \exists i : \pi_i^*(s) = a \\ 0 & \textit{otherwise} \end{cases} \quad (5)$$

Intuitively, the weight function ω^{RT} makes the Controller able to stochastically sample the available actions increasing the likelihood of guiding the testing towards uncertain model regions.

On the one hand, the flat strategy (originally introduced in [16]) is guided by the awareness of uncertain model regions. On the other one hand, it does not take into account fine-grained information from updated beliefs and structural properties of the model. Furthermore, as described in [15], it has been shown superior to traditional MBT strategies. Thus, we selected this strategy as a baseline in our empirical evaluation to understand to what extent alternative uncertainty-aware strategies yield increased cost-effectiveness.

B. History Strategy

History-based test selection strategy (hist) has been introduced to take into account aging of the available actions. Specifically, we propose a strategy based on *global* information (i.e., considering the full history) which leverages the notion of *decrementing weight* commonly adopted when the tester wants to guide the direction of the exploration balancing the number of times actions are selected. This strategy selects among the available actions based on a weight function ω^{HT} (i.e., *HT* stands for history testing) defined as follows:

$$\omega^{HT}(s,a) = \begin{cases} 1/\#(s,a) & \exists i : \pi_i^*(s) = a \\ 0 & \textit{otherwise} \end{cases} \quad (6)$$

where $\#(s,a)$ denotes a *counter* function whose role is to keep balanced the number of times an uncertain region is visited during MBT. In this strategy, the more an uncertain model region is visited, the smaller is the probability to choose again actions leading to that region. The counter function denotes how many times the region θ_i has been visited during testing; e.g., from state s_2, the history strategy is likely to choose action a_2 (i.e., the choice given by π_1^*) if θ_1 has been visited less than θ_2 during past testing activity.

C. Distance Strategy

The distance strategy (dist) is introduced to consider the variability of uncertain parameters by calculating the magnitude of the HDR containing the credible values, as anticipated in Sect. II. Thus, dist selects actions depending on a weight function ω^{DT}, where *DT* stands for distance testing. Such a function maps a pair (s,a) to a value that quantifies the magnitude of the corresponding uncertain region. Namely, the weight ω^{DT} is $\|\text{HDR}_{\theta_i}\|$ if the best policy π_i^* maps the state s to the action a. This way parameters that show higher

possible variability are associated with a higher weight versus parameters whose uncertainty spans in a smaller range. Given the set of best policies $\{\pi_i^*\}$, the function ω^{DT} is defined as:

$$\omega^{DT}(s,a) = \begin{cases} \|\text{HDR}_{\theta_i}\| & \exists i : \pi_i^*(s) = a \\ 0 & \textit{otherwise} \end{cases} \quad (7)$$

The weight function ω^{DT} modifies the behavior of the Controller component that stochastically samples the available actions maximizing the probability to reach the parameters showing larger uncertainty in their specification. As a simple example, suppose the tester starts MBT from diverse prior knowledge on θ_1 and θ_2. High degree of uncertainty may be associated with θ_1 (rather than θ_2) if little information on the SafeHome behavior in alarm conditions is available. In this case the tester can adopt an uninformative prior for θ_1 (i.e., large $\|\text{HDR}_{\theta_1}\|$) and a prior expressing definite information for θ_2 (i.e., small $\|\text{HDR}_{\theta_2}\|$). This unbalanced confidence in beliefs affects the dist strategy during test case selection. Namely, the distance strategy is likely to choose those actions given by π_1^* to collect more evidence on region θ_1. Our hypothesis is that dist outperforms both flat and hist when beliefs on uncertain regions exhibit diverse variability. Thus, testing prioritizes model regions having higher uncertainty with the aim of delivering uniform posterior knowledge.

D. Frequency Strategy

The frequency strategy is introduced to consider how many times an uncertain parameter is involved in a computation, thus to prioritize the most frequent uncertain regions. Actions are selected depending on a weight function ω^{FT}, where ***FT*** stands for frequency testing. A pair (s,a) is associated to a value that quantifies the frequency of invoking such actions within the system actual running. For example, if the uncertain regions θ_1 and θ_2 have been visited n and m times, respectively, then the weight ω^{FT} can be calculated by using the ratio $n/(n+m)$, denoted as $\textit{freq}(\theta_i)$. This way, parameters that are invoked more frequently have a higher weight versus parameters that are less involved in the system running. Given the set of best policies $\{\pi_i^*\}$, the function ω^{DT} is defined as follows:

$$\omega^{FT}(s,a) = \begin{cases} \textit{freq}(\theta_i) & \exists i : \pi_i^*(s) = a \\ 0 & \textit{otherwise} \end{cases} \quad (8)$$

Intuitively, the weight function ω^{FT} modifies the behavior of the Controller component that stochastically samples the available actions maximizing the probability to reach the parameters showing a higher probability of being invoked. We expect that this strategy outperforms random choices when there exist uncertain regions frequently invoked during the system running. The subsequent actions under test are then affected by such system property.

E. Combined Strategy

The combined strategy is introduced to jointly consider the magnitude of the uncertainty and how many times an uncertain parameter is invoked, thus to prioritize the most large and frequent uncertain regions. This means that actions are selected depending on a weight function ω^{CT}, where ***CT*** is the acronym for combined testing. Such function maps a pair (s,a) to a value jointly quantifying the largeness and the frequency of uncertain regions. To this end, let us introduce two tuning constant values (i.e., c_d and c_f) that denote the importance of distance and frequency, respectively. The function ω^{CT} is defined by a weighted sum of the distance and the frequency strategies. If they are equally important, then $c_d = c_f = 0.5$. This way both the distance and the frequency of uncertainty are associated to parameters, thus to distinguish their influence in the MBT. Given the set of best policies $\{\pi_i^*\}$, the function ω^{CT} is defined as follows:

$$\omega^{CT}(s,a) = c_d \cdot \omega^{DT}(s,a) + c_f \cdot \omega^{FT}(s,a) \quad (9)$$

where c_d, c_f are real values in $[0,1]$ and $c_d + c_f = 1$.

Similarly to previous strategies, the weight function ω^{CT} modifies the behavior of the Controller component that maximizes the probability to reach the parameters showing a larger distance and a higher probability of being invoked. The importance of these two system properties is regulated by the tester that can set different values on the basis of her/his preference. We expect that this strategy outperforms the flat strategy whether there exist uncertain regions showing a large gap in their specification of uncertainty and, at the same time, frequently invoked during the system running.

VI. Evaluation

In this section we introduce our research questions (Sect. VI-A) for the evaluation of the proposed uncertainty-aware testing strategies. We describe the experiments in Sect. VI-B, and the results are presented in Sect. VI-C. We finally discuss threats to validity in Sect. VI-D.

A. Research Questions

The purpose of the evaluation is to study the effectiveness of our novel uncertainty-aware MBT methods under bounded effort. In case an unbounded number of tests is allowed, all the strategies may eventually converge to the optimal uncertainty mitigation. We are instead interested in investigating the ability to converge faster or slower for a bounded number of tests. In particular, we aim to answer three research questions:

- **RQ1:** What is the effectiveness of our strategies in terms of relative error of updated beliefs and detection rate of injected faults?
- **RQ2:** What is the practical relevance of our strategies in terms of HDR magnitude and their effect size?
- **RQ3:** How do our strategies compare in terms of HDR ratio and their occurrence as best choice?

B. Design of the Evaluation

The strategies under evaluation are discussed in Sect. V, specifically: *flat*, history (*hist*); distance (*dist*), and frequency

(*freq*)[2], and the combination of distance and frequency strategies by setting c_d-c_f with the following values: 20%-80% (*c-2-8*), 50%-50% (*c-5-5*), and 80%-20% (*c-8-2*).

To address RQ1, all the strategies have been experimented on three selected benchmarking examples from different application domains: the SafeHome cyber physical system [9], the Tele Assistant service-based system (TAS) [18], and an e-commerce web application (e-comm) [19].

To address RQ2 and RQ3, we generated a number of synthetic systems from pseudorandom MDP models. This setting allowed us to control structural properties of interest and to avoid possible biases of preselected and ad hoc case studies. Namely, we controlled: *#states*, *#actions per state*, and *#transitions*, that define the size (i.e., complexity) of the generated systems. For each size, we varied the *level of uncertainty* (i.e., percentage of transitions associated with θ parameters) between 20% and 80%. We also varied the *prior knowledge* by constructing two testing scenarios as follows: (i) the *balanced* case, where all the priors express same degree of confidence (in terms of HDR magnitude); and (ii) the *unbalanced* case, where half randomly selected θ parameters are more certain (i.e., smaller HDR magnitude) than others.

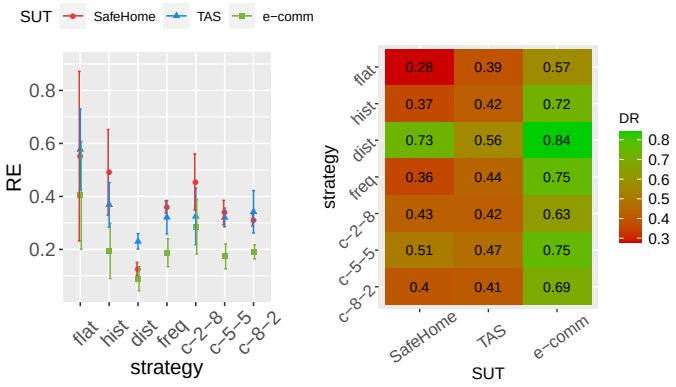
For all experiments (with both realistic and synthetic benchmarks), we compared our novel strategies with respect to the state-of-the-art baseline, i.e., the *flat* [15]. As anticipated in Sect. V, this choice is motivated by: (i) it embeds and leverages a coarse grained notion of uncertainty to select tests; and (ii) it has been shown superior to traditional MBT strategies. Testing strategies were executed 100 times for each benchmark to avoid bias in the results and consequently in the findings.

C. Results

RQ1. The MDPs and the uncertain regions of the selected benchmarks have been obtained following the specification of these systems presented in [9], [18], and [19], respectively. Their number of states varies from 9 to 12, the actions from 6 to 10, the transitions from 20 to 21, and the θ parameters from 4 to 7. Even though from the perspective of the structural complexity, the three MDP specifications show similar characteristics, these systems have very diverse behavior. For each benchmark we executed all the strategies by assuming bounded effort equal to $2k$ tests. We compared the effectiveness, under bounded effort, by measuring: the *Relative Error*[3] (RE) of the updated beliefs (point summarization of posteriors) with respect to actual values of θ parameters; and the *Detection Rate* (DR) of injected faults. The injection process has been carried out by applying perturbations to the θ values, i.e., uniform sampling between 0.02 and 0.08. Here, we define detection as the ability to recognize that updated beliefs (interval summarization of posteriors) exclude initial beliefs that were set to meet requirements. Thus, the DR can be interpreted as the ability to spot requirements violations.

[2]The frequency strategy makes use of an operational profile which assigns values proportional to the HDR magnitude of priors.

[3]The RE is computed as the magnitude of the difference between the exact value and the estimation divided by the magnitude of the exact value.



(a) Relative Error (RE). (b) Detection Rate (DR).

Fig. 3: Effectiveness of strategies on realistic benchmarks.

Fig. 3a reports mean and standard deviation of RE results. Common patterns can be observed in all the three target systems. The flat strategy is likely to exhibit the worst RE values. On average, it is above 50% with values above 90% at peak. The difference between maximum and minimum values is 0.69, i.e., on average 34% more compared to the other strategies. This means that the flat strategy yields less predictability. The lowest RE values correspond to the distance strategy. On average, we observed mean RE values between 0.09 (e-comm) and 0.23 (TAS). The history is likely to score better than flat but worse than distance. The usage of operational profiles (i.e., frequency strategy and combination of frequency and distance) is not likely to decrease the RE with respect to distance in all the three target systems.

Fig. 3b reports the DR results. Consistently with our previous findings, the distance strategy yields the highest DR values across the three benchmarks. With this strategy we were able to detect on average 30% more faults compared to the flat strategy. The flat strategy yields the worst DR values. The DR values measured by using the history strategy are close to the one obtained with frequency: the average difference across the three benchmarks is 2%. The frequency strategy and combinations of frequency and distance do not yield better DR values.

Summary: By using representative benchmarks, we found the distance strategy as the most effective one in terms of RE and DR. The flat baseline strategy is always less effective compared to our strategies.

RQ2. To answer this question, we measured the accuracy of the inference process (i.e., uncertainty quantification capability) through the HDR magnitude of the posteriors, i.e., a traditional metric for this purpose in Bayesian statistics (see Sect. II). Table II shows the results obtained by testing the synthetic systems mdp_1 to mdp_9, having complexity ranging from 250 to $10k$ structural elements, respectively. To ensure a fair comparison among the strategies, the testing campaign on the synthetic systems have been conducted by assuming equal

TABLE II: HDR magnitude using flat, history, distance, frequency, and combined strategies for 9 synthetic systems varying: structural compexity, %uncertainty, and balanced/unbalanced prior knowledge.

		balanced							*unbalanced*						
size (states; actions; transitions)	%uncertainty	*flat*	*hist*	*dist*	*freq*	*c-2-8*	*c-5-5*	*c-8-2*	*flat*	*hist*	*dist*	*freq*	*c-2-8*	*c-5-5*	*c-8-2*
mdp_1 (10; 5; 250)	20	0.367	0.144	0.159	0.174	0.150	0.149	0.150	0.215	0.108	0.086	0.089	0.088	0.089	**0.084**
	50	0.316	0.256	0.243	0.258	0.269	0.247	**0.240**	0.212	0.194	0.138	0.153	0.158	0.161	0.147
	80	0.411	0.382	0.344	0.383	0.362	0.359	0.347	0.317	0.317	0.352	0.276	0.324	0.307	0.352
mdp_2 (10; 10; 500)	20	0.312	0.108	0.108	0.132	0.121	0.111	**0.106**	0.175	0.076	0.063	0.063	0.064	0.063	0.065
	50	0.357	0.194	0.217	0.233	0.226	0.206	0.216	0.227	0.142	0.125	0.132	0.132	0.129	0.125
	80	0.381	0.282	0.258	0.254	0.251	**0.246**	**0.246**	0.305	0.226	0.183	0.197	0.190	0.181	**0.173**
mdp_3 (10; 20; 1k)	20	0.387	0.064	0.062	0.073	0.070	0.066	0.062	0.189	0.050	0.046	0.045	0.044	0.046	0.045
	50	0.341	0.124	0.134	0.149	0.142	0.136	0.133	0.249	0.097	0.079	0.095	0.094	0.091	0.089
	80	0.327	0.250	0.215	0.217	0.208	**0.211**	0.206	0.247	0.196	0.141	0.146	0.150	0.141	0.134
mdp_4 (20; 5; 1k)	20	0.241	0.154	0.145	0.169	0.155	0.152	0.150	0.180	0.125	0.122	0.119	0.120	0.124	0.130
	50	0.226	0.234	0.216	0.228	0.218	0.212	**0.207**	0.166	0.193	0.148	0.153	0.146	0.146	**0.143**
	80	0.231	0.293	0.225	0.227	0.222	0.230	**0.222**	0.183	0.226	0.196	0.182	0.189	0.182	0.189
mdp_5 (20; 10; 2k)	20	0.212	0.100	0.107	0.122	0.111	0.111	0.106	0.161	0.091	0.080	0.078	**0.077**	**0.077**	0.082
	50	0.229	0.209	0.178	0.209	0.190	0.189	0.178	0.166	0.154	0.131	0.126	0.127	0.126	0.127
	80	0.211	0.184	0.184	0.200	0.192	0.191	**0.183**	0.167	0.146	0.146	0.140	0.148	0.140	0.148
mdp_6 (20; 20; 4k)	20	0.260	0.091	0.089	0.105	0.093	0.090	0.089	0.222	0.075	0.069	0.068	0.069	0.068	0.068
	50	0.225	0.164	0.127	0.133	0.128	0.127	0.128	0.170	0.134	0.098	0.099	0.099	0.100	0.100
	80	0.222	0.156	0.155	0.166	0.163	**0.153**	0.155	0.184	0.132	0.123	0.123	0.121	**0.118**	0.120
mdp_7 (30; 5; 2.5k)	20	0.174	0.156	0.129	0.157	0.144	0.130	**0.123**	0.101	0.100	0.060	0.096	0.096	0.096	0.095
	50	0.174	0.227	0.169	0.171	0.173	**0.162**	0.165	0.143	0.175	0.123	0.129	0.131	0.125	0.125
	80	0.186	0.179	0.175	0.180	0.181	0.175	0.175	0.169	0.169	0.156	0.151	0.154	0.161	0.165
mdp_8 (30; 10; 5k)	20	0.178	0.127	0.105	0.101	0.099	0.102	**0.098**	0.124	0.092	0.071	0.074	0.075	**0.070**	0.071
	50	0.161	0.166	0.141	0.148	0.144	0.138	**0.136**	0.126	0.138	0.118	0.122	0.124	**0.112**	0.125
	80	0.173	0.232	0.167	0.173	0.173	**0.166**	0.167	0.143	0.213	0.147	0.138	0.139	0.142	0.149
mdp_9 (30; 20; 10k)	20	0.209	0.081	0.082	0.093	0.087	0.085	0.084	0.148	0.069	0.066	0.065	0.065	**0.064**	0.065
	50	0.166	0.108	0.110	0.116	0.112	0.109	0.112	0.139	0.093	0.086	0.085	0.086	0.086	0.088
	80	0.167	0.150	0.121	0.119	0.119	0.119	0.121	0.134	0.121	0.095	0.097	0.098	0.096	0.095

effort proportional to the model size[4]. Detailed results for each single θ parameter within a specific experiment are provided in the publicly available dataset paired with this paper. Boldface entries in Table II highlight the best results (i.e., smallest HDR magnitude) when considering all the strategies. Gray cells instead emphasize the comparison between flat, history, distance, and frequency strategies, i.e., excluding their combination. Results show that the flat strategy (i.e., the baseline) is often associated with high HDR magnitude. It exhibits the worst behavior in 85% of the experiments. Few exceptions have been observed within high level of uncertainty (80%) and small number of actions (5).

To deepen our investigation we compared each individual strategy with the baseline by following the practical guidelines introduced in [29]. Namely, we used the standardized non-parametric Vargha and Delaney's $\hat{A}_{12}$ effect size to measure practical value of the HDR magnitude. In our context, the $\hat{A}_{12}$ indicates the probability that a selected strategy yields increased confidence compared to the flat one. Results are shown in Table III. Values represent the effect size by varying three major factors: %uncertainty, model structural complexity (#actions per state), and the prior knowledge (balanced vs unbalanced). Similarly to Table II, the best values are highlighted. We can observe that all the strategies in both balanced and unbalanced conditions outperform the flat strategy (i.e., $\hat{A}_{12} > 0.5$). In most cases, the distance method is associated with the highest value. On the contrary, the history is the method exhibiting the lowest values (e.g., 0.53 with 80% uncertainty in the balanced case, and 5 actions in the unbalanced scenario). Consistently with our initial intuition, see Sect. V, those strategies that take into account prior knowledge (i.e., distance, frequency, and combinations) improve their performance for high level of uncertainty (80%) in the unbalanced conditions w.r.t. the same strategies in balanced conditions.

[4]The total effort is $N \times$ *#transitions*, with N constant value equal to 4 in our experimental campaign.

TABLE III: Efect size as measured by $\hat{A}_{12}$.

	%uncertainty			**#actions**		
balanced	20	50	80	5	10	20
hist	1.000	0.716	0.531	0.617	0.704	0.926
dist	1.000	0.790	0.679	0.741	0.802	0.951
freq	1.000	0.728	0.630	0.704	0.753	0.951
c-2-8	1.000	0.765	0.654	0.741	0.753	**0.975**
c-5-5	1.000	**0.815**	0.667	0.728	**0.852**	0.951
c-8-2	1.000	**0.815**	**0.691**	**0.753**	0.802	**0.975**
unbalanced	20	50	80	5	10	20
hist	0.963	0.716	0.556	0.531	0.642	0.901
dist	0.988	0.951	0.691	0.741	0.741	0.975
freq	0.988	0.926	0.716	0.741	0.840	0.975
c-2-8	0.988	0.926	0.667	0.728	0.790	0.951
c-5-5	0.988	0.926	**0.741**	0.741	0.790	0.975
c-8-2	0.975	0.926	0.704	0.728	0.704	**1.000**

When increasing structural complexity (i.e., increasing the degrees of freedom during MBT) all the new strategies increase the magnitude of the improvement. Such a trend can be understood by reading the values in both the balanced and unbalanced conditions. The effect size of combined strategies has been the highest in 90% of the experiments. However, they are not likely to achieve the highest confidence, especially in unbalanced conditions. Overall, we can observe the highest confidence without combination in 75% of our experiments.

TABLE IV: Gain in terms of HDR ratio.

	%uncertainty			#actions		
balanced	20	50	80	5	10	20
hist	2.549	1.399	1.111	1.232	1.383	2.305
dist	2.622	1.478	1.253	1.331	1.537	2.385
freq	2.283	1.378	1.214	1.222	1.462	2.164
c-2-8	2.469	1.421	1.240	1.292	1.510	2.256
c-5-5	2.565	1.485	1.257	1.336	1.565	2.333
c-8-2	**2.649**	**1.492**	**1.270**	**1.358**	**1.588**	2.374
unbalanced	20	50	80	5	10	20
hist	2.084	1.317	1.093	1.125	1.271	1.910
dist	2.357	1.617	1.272	1.389	1.560	2.190
freq	2.419	1.524	1.308	1.329	1.527	2.140
c-2-8	2.363	1.523	1.272	1.301	1.516	2.154
c-5-5	2.359	1.550	1.306	1.301	**1.578**	2.167
c-8-2	2.352	1.560	1.305	1.316	1.546	2.175

Summary: Our testing methods are likely to outperform the flat strategy to a large extent in both balanced and unbalanced scenarios. We observed that the magnitude of the improvement increases by increasing the structural complexity of the SUT.

RQ3. To answer this question, we studied the gain obtained out of the testing activity in terms of accuracy of the inference process. Such a gain is calculated as the ratio of the HDR magnitude obtained using a target strategy vs the baseline (i.e., the flat strategy). Table IV shows the results when varying: %uncertainty, structural complexity, and prior knowledge.

Values are always greater than 1.0, meaning that our strategies deliver more confidence than the flat one. Consistently with the discussion for RQ2, the gain is smaller when decreasing the level of uncertainty. Nevertheless, we can observe a decreased gain loss in the unbalanced case. On average, the gain loss is 43% in the balanced case when passing from 20% to 50% uncertainty level. Such a value is reduced to 34% in the unbalanced case. On the contrary, the gain increases when scaling up the model complexity (#actions per state). In the balanced case, the gain increases on average from 16% to 78% when increasing the #actions from 5 to 10 and from 10 to 20, respectively. In the unbalanced condition the same values are 15% and 65%, respectively. Overall, the gain of the distance and frequency strategies is always larger than the history one. More specifically, the distance exhibits the largest one in the balanced scenario, whereas the frequency scores better in the unbalanced scenario, with 20% and 80% uncertainty level. Combined strategies, instead, show higher effectiveness mostly in the balanced condition, largest value is 2.65 obtained with c-8-2 and 20% uncertainty, see Table IV. Here they achieve highest gain values in 85% of our experiments, whereas in the unbalanced case this percentage is lower (30%).

To determine the best strategy (between history, distance, and frequency) depending on the characteristics of the SUT, we measured the frequency of occurrence of best confidence gain for all the experiments on the synthetic systems. Table V shows the results summarized by prior knowledge (balanced vs unbalanced), level of uncertainty, and model complexity. Insights extracted from data follow. We can observe that by increasing the uncertainty level, the history strategy undergoes

TABLE V: Best choice frequency.

	%uncertainty			#actions		
balanced	20	50	80	5	10	20
hist	0.33	0.33	0.00	0.22	0.33	0.33
dist	0.55	0.67	0.78	0.78	0.56	0.44
freq	0.11	0.00	0.22	0.00	0.11	0.22
unbalanced	20	50	80	5	10	20
hist	0.00	0.00	0.00	0.00	0.00	0.00
dist	0.33	0.78	0.33	0.56	0.44	0.33
freq	0.67	0.22	0.67	0.44	0.56	0.67

a degradation in favor of the distance one. With high level of uncertainty (80%) the distance has been the best choice in 78% of the experiments. Considering the unbalanced condition, history is always worse than both distance and frequency. We found that the distance is likely to be the most effective choice when the level of uncertainty is $\simeq 0.5$ (i.e., the number of certain vs uncertain regions is almost equal). On the contrary, when the number of θ parameters is very high ($\geq 80\%$) or very low ($\leq 20\%$), the usage of operational profiles that selectively increase/decrease chances to hit uncertain regions (depending on the degree of uncertainty), reveals a substantial effectiveness. In these cases, the frequency strategy has been the best choice in 67% of the performed experiments. Taking into account the #actions factor, we can observe that both distance and frequency are always better than history. This trend is even more evident with unbalanced condition. Furthermore, by increasing the #actions, the effectiveness of distance decreases whereas the effectiveness of frequency increases. Despite this trend, in the unbalanced case, the distance strategy always results the best choice the tester can do. In the unbalanced condition instead, frequency is better than distance with high #actions, e.g., frequency increased by 34% with 20 #actions per state.

Summary: In terms of accuracy of the inference process, our strategies yield a gain up to $2.65\times$. The distance is the best choice in case of balanced scenarios. The frequency is better in the unbalanced scenario with an increasing structural complexity.

D. Threats to Validity

Generalization of results is a typical threat to *external validity* in empirical evaluations. We mitigated such a threat by conducting a large testing campaign on several case studies showing different structural complexity. Furthermore, we detailed all the factors controlled in our experiments (i.e., model structural characteristics, uncertain regions, prior knowledge).

To mitigate threats to *internal validity*, we designed our experimental environment to have direct manipulation of the factors of interest. In particular, we controlled both true values of θ parameters and design-time beliefs expressed by priors. This setting has been crucial to assess cause-effect relations between external factors and effectiveness of our strategies. This fine-grained access to independent variables provides a greater internal validity based on an association observed

without manipulation. Direct manipulation enables also the replication of the same experimental setting when varying test generation strategy.

We addressed threats to *conclusion validity* by reducing the possibility of producing results by chance. We repeated experiments 100 times and using for each experiment a very large sample size (between 200 and $4k$). We followed the guidelines introduced in [29] to detect statistical differences. Namely, we conducted a pairwise comparison among selected strategies using the Mann-Whitney U test to calculate p-value with significance level $\alpha = 0.05$. In addition to statistical differences, we used the standardized Vargha and Delaney's $\hat{A}_{12}$ non-parametric effect size measure.

We handled major *construct validity* threats by assessing the validity of the metrics used during our experimental campaign. The effort has been measured by considering the total number of executed tests that represents a traditional choice to assess randomized testing algorithms [29]. The effectiveness has been measured by adopting the RE ad the DR that represent sensible choices to measure the precision of updated beliefs as reported in [15]. The HDR magnitude yields instead the highest possible accuracy in estimating the θ parameters. As described in [17], this is a traditional measure in Bayesian inference to assess the confidence of the posterior knowledge.

VII. Related Work

A survey on MBT approaches is reported in [30] where the strategy for test case generation is highlighted as challenging. In [31] the idea of variability-aware testing is fostered, test cases are generated with the goal of minimizing the effort and maximizing the accuracy. In [32] testing is supported by behavioral coverage using machine learning algorithms to augment standard syntactic testing. In [33] reachability information is used to generate test cases for different goals and/or program variants. All these MBT methodologies propose optimized test case generation but they do not consider system uncertainties for such a scope.

Several approaches have been defined to measure the variation of uncertain input parameters and system output [34]. A taxonomy of potential sources of uncertainty is presented in [35] where a distinction is made for the different phases of software development, but testing is almost neglected. Uncertainty propagation for dependability has been investigated in analytical models [36], and there exist approaches embedding the specification of uncertain parameters for performance and reliability [37], [38]. However, to the best of our knowledge, there is no approach modifying the very analysis process.

Probabilistic models and their adaptation is proposed by: (i) [39], i.e., time-varying transition probabilities of Markov models are continuously updated; (ii) [19], i.e., runtime quantitative verification and sensitivity analysis are used to support self-adaptive systems; (iii) [40], i.e., queueing networks include adaptation knobs dynamically set to fulfill performance goals. However, all these works modify the system models to react to runtime changes, whereas our approach exploits uncertainties to deeper analyze specific parts of such a model.

Uncertainty awareness in MBT recently gained attention due to the potential of increasing the level of assurance of delivered software [41], [42]. In [43] uncertainty sampling is used to generate test data and it outperforms conventional random testing. An uncertainty-wise modeling framework has been proposed in [23] to create test-ready models and support MBT of uncertain CPSs. Discovering uncertainties (occurred with unknown sources) of CPSs is tackled in [9] where test cases are generated to guarantee coverage of models. Uncertainty is considered as first-class concern also in [16], however all uncertainties are equally treated, there is no distinction on their peculiar characteristics. In [10] the test case generation process takes into account the uncertainty in timing properties (e.g., the detection time of external events), and statistical model checking is adopted to verify timing constraints. In [11] an approximation-refinement loop (consisting of incrementing the training data and refining a target system's model) is introduced in combination with testing to detect requirements violations. A domain specific language to deal with the uncertainty affecting physical behavior of CPSs has been introduced in [12] where sampling and machine learning techniques are adopted to generate appropriate test cases.

Summarizing, this paper differs from the state-of-the-art since it leverages on fine-grained characteristics of the uncertain model regions to drive MBT exploration strategies.

VIII. Conclusion

In this paper we presented novel MBT strategies that exploit awareness on sources of uncertainty to drive the testing process. Our fine-grained strategies are based on past knowledge (*History*), magnitude of the variability of the uncertain parameters (*Distance*), and operational profiles (*Frequency*). We empirically evaluated the effectiveness of these strategies on three representative benchmarks from different domains and nine synthetic systems with increasing structural complexity up to $10k$ model transitions, and varying percentage of uncertainty, and balanced/unbalanced prior knowledge. We show that our novel strategies outperform the flat baseline in terms of relative error, detection rate of injected faults, and the accuracy of the estimations.

Summarizing, the *Distance* strategy greedily optimizes the local density regions and resulted to be the best choice, with few exceptions in the unbalanced scenario where the *Frequency* strategy instead scores better in the presence of a high number of model actions. As future work, we plan to further investigate the trade-off between these two strategies, in fact when a limited number of samples is available the greediness of *Distance* may reduce its overall performance, due to limited coverage of some parameters. Moreover, we plan to conduct additional assessment of our uncertainty-aware testing methods in industrial case studies.

IX. Acknowledgments

This work has been partially funded by MIUR PRIN project 2017TWRCNB SEDUCE.

REFERENCES

[1] A. Pretschner, "Model-based testing," in *Proceedings of theInternational Conference on Software Engineering (ICSE)*, 2005, pp. 722–723.

[2] E. Bringmann and A. Krämer, "Model-based testing of automotive systems," in *Proceedings of theInternational Conference on Software Testing, Verification, and Validation*, 2008, pp. 485–493.

[3] M. Utting and B. Legeard, *Practical model-based testing: a tools approach*. Elsevier, 2010.

[4] D. Garlan, "Software engineering in an uncertain world," in *International Workshop on Future of software engineering research*, 2010, pp. 125–128.

[5] A. C. Dias-Neto and G. H. Travassos, "A picture from the model-based testing area: Concepts, techniques, and challenges," in *Advances in Computers*, ser. Advances in Computers, M. V. Zelkowitz, Ed. Elsevier, 2010, vol. 80, pp. 45 – 120.

[6] K. Meinke and N. Walkinshaw, "Model-based testing and model inference," in *Leveraging Applications of Formal Methods, Verification and Validation. Technologies for Mastering Change*. Springer Berlin Heidelberg, 2012, pp. 440–443.

[7] B. K. Aichernig, W. Mostowski, M. R. Mousavi, M. Tappler, and M. Taromirad, *Model Learning and Model-Based Testing*. Cham: Springer International Publishing, 2018, pp. 74–100. [Online]. Available: https://doi.org/10.1007/978-3-319-96562-8_3

[8] S. Elbaum and D. S. Rosenblum, "Known unknowns: testing in the presence of uncertainty," in *Proceedings of the International Symposium on Foundations of Software Engineering*, 2014, pp. 833–836.

[9] M. Zhang, S. Ali, and T. Yue, "Uncertainty-wise test case generation and minimization for cyber-physical systems," *Journal of Systems and Software*, vol. 153, pp. 1 – 21, 2019.

[10] C. Wang, F. Pastore, and L. C. Briand, "Oracles for testing software timeliness with uncertainty," *ACM Trans. Softw. Eng. Methodol.*, vol. 28, no. 1, pp. 1:1–1:30, 2019.

[11] C. Menghi, S. Nejati, L. Briand, and Y. I. Parache, "Approximation-refinement testing of compute-intensive cyber-physical models: An approach based on system identification," in *Proceedings of the International Conference on Software Engineering*, 2020, pp. 372–384.

[12] S. Y. Shin, K. Chaouch, S. Nejati, M. Sabetzadeh, L. C. Briand, and F. Zimmer, "Uncertainty-aware specification and analysis for hardware-in-the-loop testing of cyber-physical systems," *Journal of Systems and Software*, vol. 171, p. 110813, 2020.

[13] N. Esfahani, E. Kouroshfar, and S. Malek, "Taming uncertainty in self-adaptive software," in *Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2011, pp. 234–244.

[14] N. Esfahani, S. Malek, and K. Razavi, "Guidearch: guiding the exploration of architectural solution space under uncertainty," in *International Conference on Software Engineering*, 2013, pp. 43–52.

[15] M. Camilli, A. Gargantini, and P. Scandurra, "Model-based hypothesis testing of uncertain software systems," *Software Testing, Verification and Reliability*, vol. 30, no. 2, p. e1730, 2020.

[16] M. Camilli, C. Bellettini, A. Gargantini, and P. Scandurra, "Online model-based testing under uncertainty," in *International Symposium on Software Reliability Engineering*, 2018, pp. 36–46.

[17] D. Insua, F. Ruggeri, and M. Wiper, *Bayesian Analysis of Stochastic Process Models*, ser. Wiley Series in Probability and Statistics. Wiley, 2012.

[18] D. Weyns and R. Calinescu, "Tele assistance: A self-adaptive service-based system examplar," in *International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2015, pp. 88–92.

[19] A. Filieri, G. Tamburrelli, and C. Ghezzi, "Supporting self-adaptation via quantitative verification and sensitivity analysis at run time," *IEEE Transactions on Software Engineering*, vol. 42, no. 1, pp. 75–99, 2016.

[20] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

[21] V. Forejt, M. Kwiatkowska, G. Norman, and D. Parker, *Automated Verification Techniques for Probabilistic Systems*. Springer Berlin Heidelberg, 2011, pp. 53–113.

[22] C. P. Robert, *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation*, 2nd ed. Springer, May 2007.

[23] M. Zhang, S. Ali, T. Yue, R. Norgren, and O. Okariz, "Uncertainty-wise cyber-physical system test modeling," *Software & Systems Modeling*, pp. 1–40, 2017.

[24] M. Veanes, C. Campbell, W. Schulte, and N. Tillmann, "Online testing with model programs," *SIGSOFT Softw. Eng. Notes*, vol. 30, no. 5, pp. 273–282, 2005.

[25] M. Broy, B. Jonsson, J.-P. Katoen, M. Leucker, and A. Pretschner, *Model-Based Testing of Reactive Systems: Advanced Lectures (Lecture Notes in Computer Science)*. Springer-Verlag New York, Inc., 2005.

[26] M. Eysholdt and H. Behrens, "Xtext: Implement your language faster than the quick and dirty way," in *International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion*, 2010, pp. 307–309.

[27] L. de Alfaro, *Game Models for Open Systems*. Springer Berlin Heidelberg, 2003, pp. 269–289.

[28] P. Diaconis and D. Ylvisaker, "Conjugate priors for exponential families," *Ann. Statist.*, vol. 7, no. 2, pp. 269–281, 1979.

[29] A. Arcuri and L. Briand, "A practical guide for using statistical tests to assess randomized algorithms in software engineering," in *International Conference on Software Engineering*, 2011, pp. 1–10.

[30] A. C. Dias Neto, R. Subramanyan, M. Vieira, and G. H. Travassos, "A survey on model-based testing approaches: a systematic review," in *International Workshop on Empirical Assessment of Software Engineering Languages and Technologies*, 2007, pp. 31–36.

[31] C. Kästner, A. Von Rhein, S. Erdweg, J. Pusch, S. Apel, T. Rendel, and K. Ostermann, "Toward variability-aware testing," in *Proceedings of the International Workshop on Feature-Oriented Software Development*, 2012, pp. 1–8.

[32] G. Fraser and N. Walkinshaw, "Assessing and generating test sets in terms of behavioural adequacy," *Software Testing, Verification and Reliability*, vol. 25, no. 8, pp. 749–780, 2015.

[33] J. Bürdek, M. Lochau, S. Bauregger, A. Holzer, A. von Rhein, S. Apel, and D. Beyer, "Facilitating reuse in multi-goal test-suite generation for software product lines," in *Fundamental Approaches to Software Engineering*, 2015, pp. 84–99.

[34] S. H. Lee and W. Chen, "A comparative study of uncertainty propagation methods for black-box-type problems," *Structural and Multidisciplinary Optimization*, vol. 37, no. 3, p. 239, 2009.

[35] A. J. Ramirez, A. C. Jensen, and B. H. Cheng, "A taxonomy of uncertainty for dynamically adaptive systems," in *International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, 2012, pp. 99–108.

[36] A. Devaraj, K. Mishra, and K. S. Trivedi, "Uncertainty propagation in analytic availability models," in *Symposium on Reliable Distributed Systems*, 2010, pp. 121–130.

[37] C. Trubiani, I. Meedeniya, V. Cortellessa, A. Aleti, and L. Grunske, "Model-based performance analysis of software architectures under uncertainty," in *International Conference on Quality of Software Architectures*, 2013, pp. 69–78.

[38] I. Meedeniya, A. Aleti, and L. Grunske, "Architecture-driven reliability optimization with uncertain model parameters," *Journal of Systems and Software*, vol. 85, no. 10, pp. 2340–2355, 2012.

[39] A. Filieri, L. Grunske, and A. Leva, "Lightweight adaptive filtering for efficient learning and updating of probabilistic models," in *International Conference on Software Engineering*, 2015, pp. 200–211.

[40] E. Incerto, M. Tribastone, and C. Trubiani, "Software performance self-adaptation through efficient model predictive control," in *International Conference on Automated Software Engineering*, 2017, pp. 485–496.

[41] M. Camilli, A. Gargantini, R. Madaudo, and P. Scandurra, "Hyppotest: Hypothesis testing toolkit for uncertain service-based web applications," in *Integrated Formal Methods*, W. Ahrendt and S. L. Tapia Tarifa, Eds. Cham: Springer International Publishing, 2019, pp. 495–503.

[42] M. Camilli and B. Russo, "Model-based testing under parametric variability of uncertain beliefs," in *Software Engineering and Formal Methods*, F. de Boer and A. Cerone, Eds. Cham: Springer International Publishing, 2020, pp. 175–192.

[43] N. Walkinshaw and G. Fraser, "Uncertainty-driven black-box test data generation," in *International Conference on Software Testing, Verification and Validation*, 2017, pp. 253–263.