$\mathcal{HOTL}$: Hypotheses and Observations Testing Logic ⋆

Ismael Rodríguez, Mercedes G. Merayo and Manuel Núñez

Dept. Sistemas Informáticos y Computación
Universidad Complutense de Madrid, 28040 Madrid, Spain
e-mail: isrodrig@sip.ucm.es,mgmerayo@fdi.ucm.es,mn@sip.ucm.es

Abstract

To ensure the conformance of an *implementation under test* (in the following IUT) with respect to a specification requires, in general, the application of an infinite number of tests. In order to use finite test suites, most testing methodologies add some feasible hypotheses about the behavior of the IUT. Since these methodologies are designed for considering a fix set of hypotheses, they usually do not have the capability of dealing with other testing scenarios where the set of assumed hypotheses varies. In this paper we propose a logic to infer whether a set of *observations* (i.e., results of test applications) allows to claim that the IUT conforms to the specification *if* a specific set of hypotheses (taken from a repertory of hypotheses) is assumed. We show the soundness and completeness of our logic with respect to a general notion of conformance.

1 Introduction

Current systems have reached a high complexity. They usually have multiple components that can be very different, even built by different developers. As a result, none of the development team members knows the implementation so thoroughly to state, without any additional consideration, that the system is correct. These facts make it necessary to apply systematic techniques in order to check the system correctness. There are many alternatives to do this. One

⋆ This paper represents an extended and revised version of [19]. This research was partially supported by the Spanish MEC projects TIC2003-07848-C02-01 and TIN2006-15578-C02-01, the Junta de Castilla-La Mancha project PAC06-0008, the Comunidad de Madrid project to fund research groups CAM-910606, and the Marie Curie project MRTN-CT-2003-505121/TAROT.

of them consists in working with an abstract model (specification) showing the desirable behavior of the system. Then, we define the correctness of the IUT in terms of its comparison with the specification: We say that the IUT *conforms* to the specification if it shows a *similar* behavior to that of the model. In order to check the conformance of the implementation with respect to the specification, we may use formal testing techniques to extract tests from the specification, each test representing a desirable behavior that the IUT must fulfill. It is obvious that the more time we spend testing an IUT, the higher is the confidence in its correctness. However, software projects are usually bound to tight time constrains and the effort devoted to testing is limited. Actually, since most systems exhibit possibly infinite behaviors, it would take infinite time to assess the validity of all these behaviors. In order to overcome this problem, testers add some reasonable assumptions about the IUT regarding the knowledge about its construction. For example, the tester can assume that the implementation can be represented by means of a deterministic finite state machine, that it has at most n states, etc.

In this line, a wide range of testing methodologies have been proposed which, for a specific set of initial hypotheses, guarantee that a test suite extracted from the specification is correct and complete to check the conformance of the IUT with respect to the specification (e.g. [4,13,21,17]). However, a framework of hypotheses established in advance is very strict and limits the applicability of a specific testing methodology. For example, it could be desirable that, in a concrete environment, the tester assumes that the behavior in four specific states of the implementation is deterministic and that two of them represent equivalent states of the implementation. Furthermore, the tester could also make more complex assumptions such as "*non-deterministic states of the implementation cannot show outputs that the machine did not show once the state has been tested 100 times*." In a different scenario the tester could not believe this assumption but think that "*if she observes two sequences of length 200 and all their inputs and outputs coincide then they actually traverse the same IUT states*." Let us note that if the tester assumes the validity of a set of hypotheses to test a given IUT, then a specific test suite would be appropriate, while by using other hypotheses the test suite could not be so. It would be desirable to provide the tester with a tool to let her analyze the impact of considering a given set of hypotheses in the testing process, as well as the consequences of adding/eliminating hypotheses from the set. The goal of this methodology would be to ascertain if a given finite set of observations extracted by a test suite is *complete* in the case that the considered hypotheses hold, that is, we assess whether obtaining these observations from the IUT implies that the IUT conforms to the specification *if* the hypotheses hold.

In this paper we propose a *logic* called $\mathcal{HOTL}$ (*Hypotheses and Observations Testing Logic*). Its aim is to assess whether a given set of observations implies the correctness of the IUT under the assumption of a given set of hypotheses.

In order to allow the tester to compose sets of hypotheses, the logic provides a repertory of hypotheses, including hypotheses appearing in known testing methodologies. The final goal of the logic is to facilitate at least the following three tasks. First, a tester can use it to customize the testing process to her specific environment. By using the logic, she can infer not only the consequences of adding a new test, but also the consequences of adding a new hypothesis. In this way, the tester has control over a wide range of testing variables. In particular, the construction of test suites to extract observations and the definition of hypotheses can influence each other. This provides a dynamic testing scenario where, depending on the specification and the tester's knowledge of the IUT, different test suites and hypotheses can be considered. Second, such logic allows the tester to evaluate the *quality* of a test suite to discover errors in an implementation: If the observations that could be extracted by the test suite require (for their completeness) a set of hypotheses that is *harder* to be accepted than those required by another suite, then the latter suite should be preferred. This is because this suite could allow the tester to reach diagnostics in a less restrictive environment. Finally, let us also note that such a logic may provide a conceptual bridge between different testing approaches. In particular, we may use it to represent the (fix) sets of hypotheses considered by different approaches. Then, by considering the observations each test suite could obtain, a test suite that is complete in an approach (i.e., under some hypotheses) could be turned into a new suite that is complete in another (i.e., under the assumption of another set of hypotheses). Similarly, we can analyze how the size of test suites is affected by hypotheses. Moreover, we can use the logic to create intermediate approaches where sets of hypotheses are appropriately mixed. Let us note that to compare two methodologies might require to extend the logic: If a hypothesis cannot be expressed with the current repertory of hypotheses, then the logic might be extended to allow to represent it. Fortunately, the modularity of the logic will ease this task. As we will see, it is very likely that only a new *hypothesis predicate* and a new *deduction rule*, to handle this hypothesis, should be added.

Since the first of the previous tasks, that is, serving as core a of a (dynamic) testing methodology, enables the others, next we concentrate on how our logic is applied to perform it. The methodology consists of two phases. The first phase consists in the classical application of tests to the IUT. By using any of the available methods in the literature, a test suite will be derived from the specification. If the application of this test suite finds an unexpected result then the testing process stops: The IUT is not conforming. However, if such a wrong behavior is not detected then the tester cannot be sure that the IUT is correct. In this case, the second phase begins, that is, the tester applies the logic described in this paper to infer whether passing these tests *implies* that the IUT is correct if a given set of hypotheses is assumed. If it does then the IUT is assumed to be correct; otherwise, the tester may be interested in either applying more tests or in assuming more hypotheses (in the latter case, on

the cost of *feasibility*) and then applying the logic again until the correctness of the IUT is effectively granted. In order to appropriately apply the logic, the behavior of the IUT observed during the application of tests must be properly represented. For each application of a test to the IUT, we construct an *observation*, that is, a sequence of inputs and outputs denoting the test and the response produced by the IUT, respectively. Both observations and the assumed hypotheses will be represented by appropriate *predicates* of the logic. Then, the deduction rules of the logic will allow to infer whether we can claim that the IUT conforms to the specification. Actually, the logic will be used to check whether *all* the implementations that could produce these observations and fulfill the requirements of the hypotheses conform to the specification.

In the predefined repertory, hypotheses are split into two kinds: Hypotheses concerning specific parts (states) of the IUT and hypotheses concerning the whole IUT. In order to unambiguously denote the states regarded by the former, they will be attached to the corresponding observations that reached these states. For example, if the IUT was showing the sequence of outputs $o_1, o_2, \ldots, o_n$ as answer to the sequence of inputs $i_1, i_2, \ldots, i_n$, the tester may think that the state reached after performing i_1/o_1 is deterministic or that the state reached after performing the sequence $i_1/o_1, i_2/o_2$ is the same as the one reached after performing the whole sequence $i_1/o_1, i_2/o_2, \ldots, i_n/o_n$. Let us remark that these are *hypotheses* that the tester is assuming. Thus, she might be wrong and reach a wrong conclusion. However, this is similar to the case when the tester assumes that the implementation is deterministic or that it has at most n states and, in reality, this is not the case. In addition to using hypotheses associated to observations, the tester can also consider global hypotheses that concern the whole IUT. These are assumptions such as the ones that we mentioned before: Assuming that the IUT is deterministic, that it has at most n states, that it has a unique initial state, etc. In order to denote the assumption of this kind of hypotheses, specific logic predicates will be used.

Let us note that there are several papers where testing hypotheses are used to perform the testing process. For example, we may consider that the implementation is deterministic (e.g. [18,11]), that we are testing the coupling of several components by assuming that all of them are correct or that at most one of them is incorrect (e.g. [14]), etc. Our methodology provides a *generalization* of these frameworks because it allows to decide the specific hypotheses we will consider. In this line, we can compare the suitability of different test suites or test criteria in terms of the hypotheses that are considered (e.g. [15]); some formal relations to compare test suites have been defined [10]. Since our logic provides a mechanism to effectively compare sets of hypotheses, it may help to compute relations defined in these terms. Even though we work with rules and properties, our work is not related to *model checking* [7] since we do

not *check* the validity of properties: We assume that they hold and we infer results about the conformity of the IUT by using this assumption. In the same way, this work is not related to some recent work on passive testing where the validity of a set of properties (expressed by means of *invariants*) is checked by passively observing the execution of the system (e.g. [12,2,5,3]).

The rest of the paper is organized as follows. In order to illustrate our goals, in the next section we informally present some deductions we can make with our logic. In Section 3 we present some basic concepts related to the formalisms that we will use. In Section 4 we introduce the predicates of $\mathcal{HOTL}$, while in Section 5 we present the deduction rules. The main properties of the logic, soundness and completeness, are proved in Section 6. Next, in Section 7 we discuss some methods to extend $\mathcal{HOTL}$. In Section 8 we present our conclusions and some directions for further research. Finally, the repertory of hypotheses of $\mathcal{HOTL}$ is depicted in the appendix of the paper.

2 Preliminary Examples

Before concepts are formally described in forthcoming sections, let us present an informal overview of our methodology. Deduction rules will be informally applied to reach conclusions in two simple examples. Though rules are applied in a given order in $\mathcal{HOTL}$, in the following examples this order will be relaxed for the sake of clarity.

Let us consider the *spec* specification shown in Figure 1 (up left), defined in the form of a *finite state machine*. As usual, an arrow points to the initial state. We consider the following conformance testing relation: An IUT conforms to the specification if for any sequence of inputs that is considered by the specification, the sequence of outputs offered by the IUT is also offered by the specification. The following hypothesis will be assumed: Both specifications and implementations can be represented by finite state machines.

Let us suppose that we interact with the IUT by offering the sequence of inputs (a, a, a) and we obtain the sequence of outputs (d, e, f). Then, our knowledge about the IUT is that, from one of the initial states of the IUT (there could be more than one), the sequence $(a/d, a/e, a/f)$ can be produced. We create a model to represent our *knowledge* about the IUT. The model, denoted by $model_1$, is shown in Figure 1. Let us note that some states of this model could coincide in the IUT, but we do not know this fact *yet*. So, by now all steps in the observation are represented by transitions involving *different* states. At this point it is important to make clear that even though models will be depicted as `FSM`-like diagrams, they actually denote the *set of all* `FSMs` that can perform the transitions shown in the diagram and, perhaps, other non-

depicted transitions. In particular, the FSM that exactly coincides with the diagram is only *one of* the FSMs represented by the model.

Clearly, $model_1$ does not represent an IUT that necessarily conforms to *spec* since some FSMs represented by $model_1$ do not conform to *spec*. Since any of these FSMs could denote the IUT, the conformance is not guaranteed. This is consistent with our current knowledge about the IUT. For example, we do not know the reaction of the IUT to b and c in two situations that are considered by *spec*. Moreover, nothing guarantees that the reaction to (a, a, a) will *always* be (d, e, f). The IUT could be nondeterministic and the initial state could not be unique. Besides, we do not know whether further transitions allow to *cycle* the sequence $(a/d, a/e, a/f)$ as *spec* denotes. Hence, more information (i.e. observations and hypotheses) are required to validate the IUT. Obviously, if during the test application phase we observe an unexpected output then we stop and conclude that the IUT is not correct. For example, if after the sequence of inputs (a, b) we receive the sequence of outputs (d, f) then we know that the IUT is not conforming. Thus, during the rest of the section we will show examples where observations do not contain unexpected behavior.

Let us *assume* that we can bring the IUT back to (one of) its initial state(s), that is, that there exists a reliable *reset* button. After using it, we interact again with the IUT: We offer (a, b, b) and we obtain (d, d, d). This allows to extend our model of the IUT as shown in $model_2$. Since initial states in both observations could be different, both sequences depart from different initial states in the model.

Let us choose some other hypotheses from the available repertory and let us *assume* them. First, we will suppose that the initial state is *unique* in the IUT. This assumption, as any other, could be wrong. However, our study of the IUT will go on under the assumption that it actually holds. If there is a single initial state then our two observations share the initial state. Thus, the model can be refined as depicted in $model_3$. Let us introduce another hypothesis: If a given input is offered at an IUT state a high number of times, then all outputs that can be produced in response to this input are produced at least once, and all the states the IUT could move to are reached at least once. In our simple example, let us suppose that 2 is a *high* number. Since a has been offered two times at the initial state of the IUT, the only response that can be given to a in this state is d. Moreover, the destinations of both transitions with a/d, the states 2 and 6, are the only available destinations when a is offered from this state.

Let us introduce a few assumptions about the states reached in the IUT through the observations. Let us suppose that the states reached in the first observation after (a/d) and in the second after (a/d) coincide. In our model, it means that 2 and 6 are the same state. The new model is $model_4$. We can also

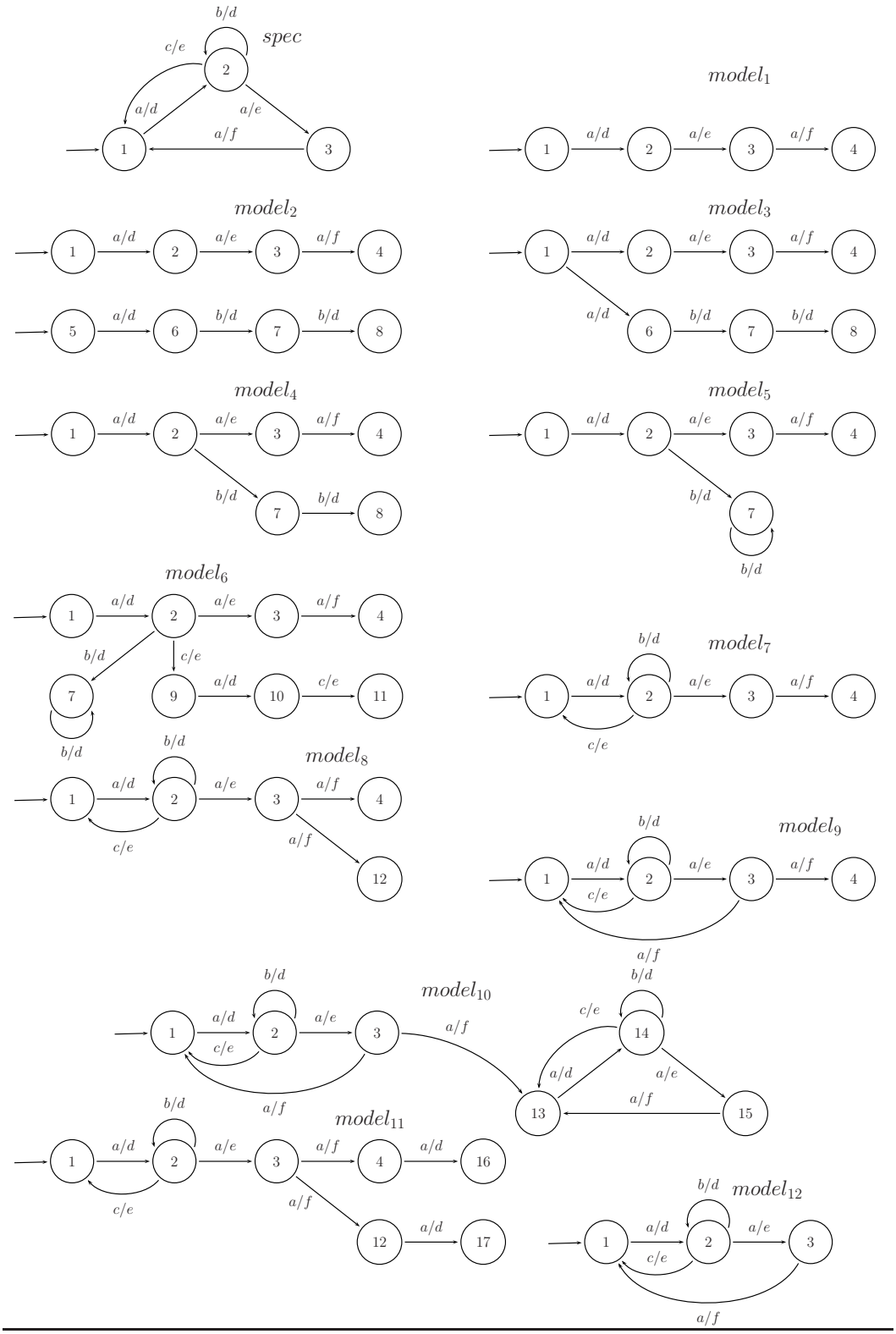

Fig. 1. Specifications and models (1/2).

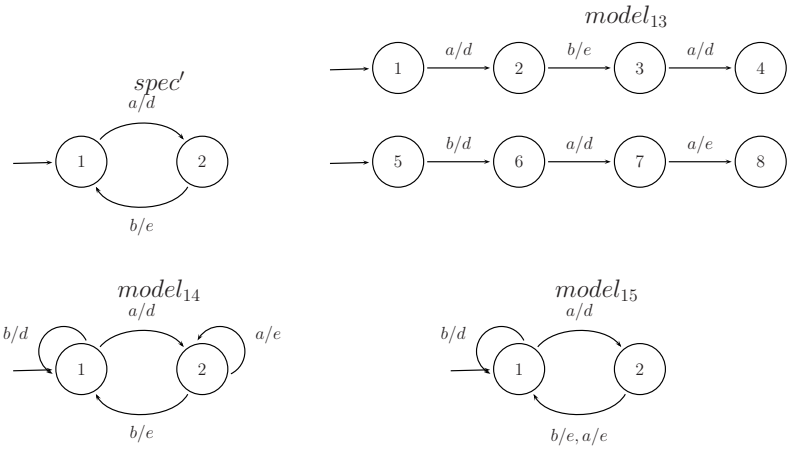

Fig. 2. Specifications and models (2/2).

use this assumption to relate states in the *same* observation. We assume that states in the second observation after $(a/d, b/d)$ and $(a/d, b/d, b/d)$ coincide. Then, 7 and 8 are the same state and we get $model_5$.

Conclusions about the IUT cannot be provided until the c/e transition of the specification is observed in the IUT. Hence, we need more observations. Let us suppose that we offer (a, c, a, c) and the IUT answers (d, e, d, e). Since we assumed that there is a single initial state, this observation begins at state 1. Besides, we deduced before that all the responses of this state to the input a are known. So, the observation traverses the known transition from 1 to 2. Next, input c is offered. Nothing about the response (or responses) in state 2 to c is known yet. Hence, from this point on all states are new. The resulting model is $model_6$.

Let us assume now that if the IUT performs a *long* sequence then, all times this sequence is taken, the same IUT states and transitions are traversed. In our simple example, we suppose that *long* means that the length of the sequence is higher than or equal to 2. Our model performs $1 \xrightarrow{a/d} 2 \xrightarrow{c/e} 9$ and $9 \xrightarrow{a/d} 10 \xrightarrow{c/e} 11$. Hence, we can assume that 1 is 9, 2 is 10, and 9 is 11. Moreover, we also have $2 \xrightarrow{b/d} 7 \xrightarrow{b/d} 7$ and $7 \xrightarrow{b/d} 7 \xrightarrow{b/d} 7$, so 2 is 7. The resulting model is $model_7$.

Let us suppose that the state 2 is deterministic. We need more information to infer whether the behavior of the IUT after $(a/d, a/e, a/f)$ is correct. Again, we propose (a, a, a) and we obtain the same result: (d, e, f). Since we assume the uniqueness of the initial state, the new observation also starts at 1. As we said before, the only possible transition from state 1 with input a produces d and moves the machine to state 2. Hence, the first transition of the new observation is known: $1 \xrightarrow{a/d} 2$. Since we assume that 2 is deterministic, the observation follows by performing the known transition $2 \xrightarrow{a/e} 3$. However,

nothing is known about state 3. So, the transition a/f leads to a new state. The resulting model is $model_8$. Let us note that a has already been offered two times from state 3. Due to our previous assumption that all the reactions are observed (at a given state and for a given input) if they are stimulated at least two times, the set of transitions that are available from the state 3 with the input a is completely known.

Let us assume now that the states 12 and 1 coincide in the IUT. We obtain $model_9$. Then, we introduce a new hypothesis: We suppose that from state 4 on, the behavior of the IUT is exactly the behavior defined by the specification from its state 1 on. For example, one might wish to include this hypothesis because the behavior from this state on represents a functionality/component that was tested/validated before. Thus, we can copy and paste the behavior of the specification *spec* and connect it to 4. The result is $model_{10}$. Let us note that this model cannot react to unspecified inputs in undesirable ways: Even though states 1 and 3 may be nondeterministic, our assumptions allowed to infer that the behavior when the input a is proposed is completely known in both cases. Let us also note that the reaction of these states when b or c are proposed is irrelevant for the correctness of the model with respect to our specification. We assumed that 2 is deterministic, and the behavior of states 13, 14, and 15 is supposed to exactly simulate the behavior of *spec*. So, the model does not allow to react wrongly when inputs concerned by *spec* are offered. Thus, the model is correct, that is, all the `FSMs` that fit into the model conform to the specification. Hence, if the hypotheses commented before are assumed then the IUT that produced these observations necessarily conforms to *spec*.

Let us note that many other sets of observations and hypotheses allow to deduce the correctness of the IUT. For instance, let us substitute our two observations of the sequence $(a/d, a/e, a/f)$ by two observations of the sequence $(a/d, a/e, a/f, a/d)$. If we consider all the hypotheses applied before up to the construction of $model_8$ then model $model_{11}$, also depicted in Figure 1, is obtained. Let us also assume the hypothesis that the IUT has, at most, 3 states. Among all states in $model_{11}$, no state in the set $\{1, 2, 3\}$ can be fused with any other state of the same set: If a is offered in 1 or 3 then the only allowed answer is d or f, respectively. Since the state 2 is deterministic, after a is offered only e is allowed. Hence, using at most 3 states requires that all states in the set $\{4, 12, 16, 17\}$ are fused with a state in $\{1, 2, 3\}$. Since 4 and 12 perform a/d, they must coincide with 1. Let us note that if, for example, states 4 and 2 are fused then 2 can produce either d or e when a is offered, which does not keep the condition that 2 is deterministic. Hence, 16 and 17 must coincide with 2. We deduce that the only model that consistently keeps the requirements imposed by observations and hypotheses is $model_{12}$. We also have that all the `FSMs` represented by this model conform *spec*. So, the new set of observations and hypotheses also imply the correction of the IUT.

Providing conclusions about the correctness of the IUT may be easier if we apply actual hypotheses *in depth*. For example, in $model_8$ we find the sequences of length two $2 \xrightarrow{a/e} 3 \xrightarrow{a/f} 4$ and $2 \xrightarrow{a/e} 3 \xrightarrow{a/f} 12$. Since both *long* sequences coincide, the hypotheses assumed in this case allow to infer that 4 and 12 coincide. Hence, there is no need to apply specific hypotheses to 4 and 12, as we did before. By similar arguments, in $model_{11}$ we can match 4 and 12, as well as 16 and 17. Hypotheses will be applied in depth in our logic.

It is worth to point out that, in the previous example, the conformance of the IUT was not guaranteed until suitable sets of observations and hypotheses were considered. For instance, if the sets of observations and hypotheses are exactly those we considered up to the construction of $model_5$ then the IUT is not *necessarily* conforming: After the sequence of inputs (a, b, c) is produced, it is not guaranteed that the sequence of outputs (d, d, e) will be obtained. That is, if we assume these hypotheses and tests provide us with these observations, then the IUT might still be incorrect.

We present another small example where more subtle features of our methodology are explored. Let us consider the $spec'$ specification depicted in Figure 2. We interact with the IUT and we obtain the observations $(a/d, b/e, a/d)$ and $(b/d, a/d, a/e)$. Let us note that the second observation performs actions that are not concerned by $spec'$: No requirement is given for input b from its first state. Contrarily to what may be expected, this observation will provide significant information about the conformance of the IUT to $spec'$. As we will see, the fact that the IUT answers d to input b from its first state will allow to *distinguish* this state from another deterministic IUT state where e is answered to b. This will allow to associate other *concerned* observed actions with different states accordingly. It will turn out that all `FSMs` that fulfill the deduced conditions actually conform to $spec'$. The reader may think that a tester would never apply an input that is not specified in the specification and that this example is artificial. We just say that *if* the tester applies these *strange* inputs then in some situations she can obtain valuable information about the IUT. However, this application could also provide useless information. Anyway, if only expected inputs are applied, then the same knowledge can also be obtained but, possibly, with more testing effort. Let us consider how the facts previously commented are deduced.

After the observations $(a/d, b/e, a/d)$ and $(b/d, a/d, a/e)$ are considered, the $model_{13}$ is obtained. Let us consider only two assumptions: All IUT states are deterministic and the IUT has at most 2 states. All states of $model_{13}$ must collapse into only 2 states. Let us consider the models fulfilling this requirement in such a way that the states are deterministic. Since 1 and 7 react to a in a different way, they cannot coincide. For the same reasons, 6 and 7 must be different. The fact that only 2 states can be used implies that 1 and 6 must coincide. Moreover, since both 1 and 6 are deterministic,

they move to the same state when the input a is offered (and the output d is answered): 2 and 7 must coincide. The state 2 reacts to b by producing e, but 5 answers d to this input. Hence, 5 cannot be fused with 2 and 7 and, by elimination, it must be fused with 1 and 6. Similarly, 3 produces d when a is offered, so it cannot be fused with 2 and 7 and it must join 1, 6, and 5. Since 3 and 1 are the same state, the destination of a/d from 3, the state 4, must coincide with the destination of 1, that is, 2 (and 7). Summarizing, $1, 3, 5, 6$ and $2, 4, 7$ must be grouped into different states, though 8 can join any of both groups. Depending on the election of 8, two models are possible: $model_{14}$ and $model_{15}$. Since all the `FSMs` represented by these two models conform to $spec'$, any IUT that can produce the two observations conforms to $spec'$, provided that the considered hypotheses hold. In the next sections we will present the logic allowing to make these deductions. In order to illustrate formal concepts, a more complex running example will be used in the following sections.

3 Formal Model

In this section we introduce some basic concepts that will be used along the paper to formally present our methodology. Specifically, we introduce the notion of finite state machine and a conformance relation. Let us note that machines are allowed to be non-deterministic. It will be the tester, by adding the corresponding hypothesis, who can assume that some/all states of the machine are deterministic.

Definition 1 A *finite state machine*, in short `FSM`, is a tuple of five elements $F = (\mathcal{S}, \texttt{inputs}, \texttt{outputs}, \mathcal{I}, \mathcal{T})$ where $\mathcal{S}$ is the set of *states*, `inputs` is the set of *input actions*, `outputs` is the set of *output actions*, $\mathcal{I} \subseteq \mathcal{S}$ is the set of *initial states*, and $\mathcal{T}$ is the set of *transitions*.

A transition is a tuple $(s, i, o, s') \in \mathcal{T}$ where $s, s' \in \mathcal{S}$ are the *initial* and *final* state of the transition, respectively, $i \in \texttt{inputs}$ is the *input* that activates the transition, and $o \in \texttt{outputs}$ is the *output* produced in response. A transition $(s, i, o, s') \in \mathcal{T}$ is also denoted by $s \xrightarrow{i/o} s'$.

We say that $(i_1/o_1, \ldots, i_n/o_n)$ is a *trace* of F if there exists an initial state $s_1 \in \mathcal{I}$ and states $s_2, \ldots, s_{n+1} \in \mathcal{S}$ such that the transitions $s_1 \xrightarrow{i_1/o_1} s_2$, $s_2 \xrightarrow{i_2/o_2} s_3$, $\ldots$, $s_n \xrightarrow{i_n/o_n} s_{n+1}$ belong to $\mathcal{T}$. The set of all traces of F is denoted by $\texttt{traces}(F)$. Let us consider $s_1, s_2 \in \mathcal{S}$. We say that the state s_2 is *reachable* from s_1, denoted by $\texttt{isReachable}(F, s_1, s_2)$, if either $s_1 = s_2$ or there exist s', i, o with $s_1 \xrightarrow{i/o} s' \in \mathcal{T}$ and $\texttt{isReachable}(F, s', s_2)$; $\texttt{reachableStates}(F, s)$ contains all the states s' such that $\texttt{isReachable}(F, s, s')$ holds.

Let $s \in \mathcal{S}$ and $i \in \texttt{inputs}$. Then, $\texttt{outs}(F, s, i)$ denotes the set of outputs that can be produced at state s in response to i, that is, $\{o \mid \exists\, s' : s \xrightarrow{i/o} s' \in \mathcal{T}\}$.

Let $s \in \mathcal{S}$. We say that s is a *deterministic state of* F, denoted by $\texttt{isDet}(F, s)$, if there do not exist two transitions $s \xrightarrow{i/o'} s'$, $s \xrightarrow{i/o''} s'' \in \mathcal{T}$ such that $o' \neq o''$ or $s' \neq s''$. We say that F is *deterministic* if for all state $s \in \mathcal{S}$ we have $\texttt{isDet}(F, s)$.

The set of all FSMs is denoted by FsmSet. □

Given an FSM, due to technical reasons and without loss of generality, we will assume that for all $s \in \mathcal{S}$ and $i \in \texttt{inputs}$ there exists $o \in \texttt{outputs}$ such that $\nexists\, s' \in \mathcal{S} : s \xrightarrow{i/o} s' \in \mathcal{T}$. Making an FSM to fulfill this property is trivial: It is enough to extend outputs with a new fresh output symbol. In order to fix the kind of formalisms our logic will deal with, the following hypothesis will be imposed: Both implementations and specifications can be represented by appropriate FSMs. As a consequence, we have that when an input is offered to an IUT it always produces an observable response (that is, *quiescent* states not producing any output are not considered). In order to simplify the model, we implicitly assume the existence of a *reliable reset button* in the IUT allowing us to obtain *separated* observations, that is, sequences where each of them departs from one of the initial states.

Next we present the basic conformance relation that will be considered in our framework. This relation is similar to ioco [21] but in the framework of FSMs. It has been introduced in [16] as a preliminary step to define timed conformance relations for FSMs. Intuitively, an IUT is conforming to a specification if it does not *invent* behaviors for those traces that can be executed by the specification. As usual, we will assume that the IUT is *input-enabled*, that is, for all state s and input i there exist o, s' such that $s \xrightarrow{i/o} s'$ belongs to the set of transitions of the IUT. During the rest of the paper, and when no confusion arises, we will assume that the FSM representing a generic specification is given by $spec = (\mathcal{S}_{spec}, \texttt{inputs}_{spec}, \texttt{outputs}_{spec}, \mathcal{I}_{spec}, \mathcal{T}_{spec})$.

Definition 2 Let S and I be two FSMs. We say that I *conforms to* S, denoted by I conf S, if for all $\rho_1 = (i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o_n) \in \texttt{traces}(S)$, with $n \geq 1$, we have $\rho_2 = (i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o'_n) \in \texttt{traces}(I)$ implies $\rho_2 \in \texttt{traces}(S)$. □

Example 1 A running example, adapted from [8], will be used along the paper to illustrate our framework. A medical ray beaming system is controlled by using three buttons: A button for charging the machine (a single button press increases the voltage by 10 mV), another one for the beam activation, and the last one for resetting the machine at any time. The system will only

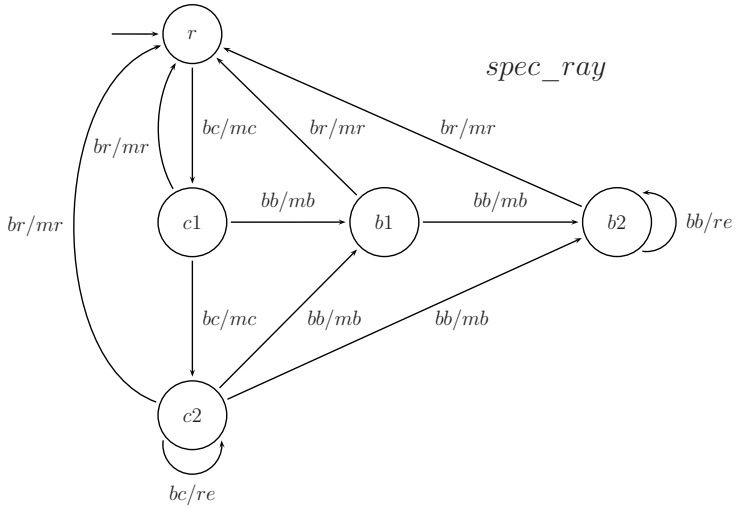

Fig. 3. Finite State Machine *spec_ray*.

charge the machine twice (increasing the voltage up to 20 mV) and it only lets to beam twice. Any further attempt to either increase the charge of the machine or to activate the beaming will be rejected because there is a danger of seriously injuring the patient. The components of *spec_ray* are:

- $\mathcal{S}_{spec\_ray} = \{r, c1, c2, b1, b2\}$, where r denotes the *ready* state, $c1/c2$ denote the states where the beamer has been charged one/two times, and $b1/b2$ denote the states where the first/second beaming is performed.
- $\texttt{inputs}_{spec\_ray} = \{br, bc, bb\}$, where $br/bc/bb$ respectively denote that the reset/charging/beaming button has been pressed.
- $\texttt{outputs}_{spec\_ray} = \{mr, mc, mb, re\}$, where $mr/mc/mb$ respectively denote that the machine is ready/charging/beaming while re denotes that the command has been rejected.
- $\mathcal{I}_{spec\_ray} = \{r\}$, that is, the initial state is *ready*.

Figure 3 shows the FSM corresponding to *spec_ray*. □

4 Predicates of the Logic

In this section we present the predicates that will be part of $\mathcal{HOTL}$. These predicates will encode our knowledge and assumptions about the IUT. In particular, they will allow us to represent the *observations* that we have obtained from the IUT during the preliminary *classical* testing phase. *Observations* denote that, in response to a given sequence of inputs, the IUT produced a given sequence of outputs. Our notion of observation will include some assumptions about the IUT as well as the observed behavior. Let us remark that if one of

the sequences shows a behavior that is forbidden by the specification, then the IUT does not conform to the specification and no further analysis is required, that is, there is no need to apply our logic.

4.1 Manipulating Observations

During the rest of the paper, $\texttt{Obs}$ denotes the set of all the observations collected during the preliminary interaction with the IUT, while $\texttt{Hyp}$ denotes the set of *hypotheses* the tester has assumed. In this latter set, we will not consider the hypotheses that are implicitly introduced by means of observations.

Observations follow the form $ob = (a_1, i_1/o_1, a_2, \ldots, a_n, i_n/o_n, a_{n+1}) \in \texttt{Obs}$, where ob is a unique identification name. It denotes that when the sequence of inputs $i_1, \ldots, i_n$ was proposed from the initial configuration of the implementation, the sequence $o_1, \ldots, o_n$ was obtained as response. In addition, for all $1 \leq j \leq n+1$, a_j represents a set of *special attributes* concerning the state of the implementation that we reached after performing $i_1/o_1, \ldots, i_{j-1}/o_{j-1}$ in *this* observation. Attributes denote our assumptions about this state. For all $1 \leq j \leq n+1$ the attributes in the set a_j are of the form $\texttt{imp}(q)$ or $\texttt{det}$, where $\texttt{imp}(q)$ denotes that the implementation state reached after $i_1/o_1, \ldots, i_{j-1}/o_{j-1}$ is associated to a *state identifier name* q and $\texttt{det}$ denotes that the implementation state reached after $i_1/o_1, \ldots, i_{j-1}/o_{j-1}$ in this observation is deterministic. State identifier names are used to match equal states: If two states are associated with the same state identifier name then they represent the *same* state of the implementation.[1] The set of all state identifier names will be denoted by $\mathcal{Q}$. Besides, attributes belonging to a_{n+1} can also be of the form $\texttt{spec}(s)$, with $s \in \mathcal{S}_{spec}$, denoting that the implementation state reached after $i_1/o_1, \ldots, i_n/o_n$ is such that the subgraph that can be reached from it is isomorphic to the subgraph that can be reached from the state s of the specification. We assume that attributes of the form $\texttt{spec}(s)$ can appear only at the end of the observation, meaning that the behavior of the implementation from that point on is known and there is no need to check its correctness.

Example 2 For our case study we will consider that the following set of observations $\texttt{Obs}=\{ob_i | 1 \leq i \leq 7\}$ was obtained after applying some tests in a preliminary phase:

[1] Let us remark that, since we consider the IUT to be a black-box, a tester cannot always be sure of the state where the IUT is placed. However, she may still *hypothesize* that the reached states after performing two subsequences are in fact the same.

$$ob_1 = (\emptyset, bc/mc, \{\texttt{imp}(q_1)\}, bc/mc, \emptyset, bc/re, \{\texttt{imp}(q_2)\}, br/mr, \emptyset, bc/mc, \{\texttt{imp}(q_1)\})$$

$$ob_2 = (\emptyset, bc/mc, \emptyset, bc/mc, \emptyset, bc/re, \{\texttt{imp}(q_2)\}, bb/mb, \{\texttt{imp}(q_3)\}, br/mr, \emptyset)$$

$$ob_3 = (\emptyset, bc/mc, \{\texttt{imp}(q_1), \texttt{det}\}, bc/mc, \{\texttt{imp}(q_2)\}, br/mr, \emptyset)$$

$$ob_4 = (\emptyset, bc/mc, \emptyset, bc/mc, \emptyset, bb/mb, \{\texttt{imp}(q_3)\}, bb/mb, \{\texttt{imp}(q_4), \texttt{det}\}, br/mr, \emptyset)$$

$$ob_5 = (\emptyset, bc/mc, \emptyset, bb/mb, \{\texttt{imp}(q_3)\}, br/mr, \emptyset, bc/mc, \{\texttt{imp}(q_1)\})$$

$$ob_6 = (\emptyset, bc/mc, \emptyset, bb/mb, \emptyset, bb/mb, \{\texttt{imp}(q_4)\}, bb/re, \emptyset, bb/re, \{\texttt{det}\}, br/mr, \emptyset)$$

$$ob_7 = (\{\texttt{det}\}, bc/mc, \emptyset, br/mr, \emptyset)$$

For example, ob_3 denotes that we initially do not make assumptions about the state the IUT selected as *initial* for this observation (its set of attributes is $\emptyset$). After pressing the charging button bc, the beaming system is charged and the state reached is identified by a specific state identifier name (denoted by q_1). Moreover, we assume that this state is deterministic. Next, we press the charge button bc again and the response is the same as before. This time, we assume that a state with identifier q_2 is reached. Finally, we press the reset button br and the machine is reset. We make no assumptions about the state reached this time. □

4.2 Model Predicates

Observations will allow to create *model predicates*. A model predicate denotes our knowledge about the implementation. Models will be constructed according to the observations and hypotheses we consider. In particular, they induce a graph consistent with the observations and hypotheses considered so far. As more information is retrieved, models will be refined and particularized. We denote model predicates by $\texttt{model}\,(m)$, where $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \mathcal{O})$ is a *model*. The meaning of the different components of the tuple is:

- $\mathcal{S}$ (*states*): It is the set of states that appear in the graph of the model. Despite the fact that this graph attempts to represent (a part of) the behavior of the implementation, any name belonging to $\mathcal{S}$ is fresh and by no means related to the corresponding state of the implementation. Let us note that after more information is considered, it could turn out that some states belonging to $\mathcal{S}$ coincide.
- $\mathcal{T}$ (*transitions*): Set of transitions appearing in the graph of the model.
- $\mathcal{I}$ (*initial states*): Set of states that are initial in the model. Two additional symbols may appear in $\mathcal{I}$. The first special symbol, α, denotes that any state in $\mathcal{S}$ could be initial. The second symbol, β, denotes that not only

states belonging to $\mathcal{S}$ could be initial (that is, it implies α), but also that other states not explicitly represented in $\mathcal{S}$ could be initial (recall that $\mathcal{S}$ denotes a *part* of the implementation states).

- $\mathcal{A}$ (*accounting*): Set of *accounting registers*. Each register in the set is a tuple $(s, i, outs, f, n)$ denoting that in state $s \in \mathcal{S}$ the input i has been offered n times and we have obtained the outputs belonging to the set $outs$. Besides, for each transition $t \in \mathcal{T}$ departing from state s with input i, the function $f : \mathcal{T} \longrightarrow \mathbb{N}$ denotes the number of times the transition t has been observed. This information allows the tester to handle some hypotheses about nondeterminism. If, due to the hypotheses that we consider, we infer that the number of times we observed an input is high enough to believe that the implementation cannot react to that input in a way that has not happened before (that is, either with an output that was not produced before or leading to a state that was not taken before), then the value n is set to $\top$. In this case, we say that the behavior at state s for the input i is *closed*.
- $\mathcal{E}$ (*equality relations*): Set of equalities relating states in $\mathcal{S}$. Equalities have the form s `is` q, where $s \in \mathcal{S}$ is a state and $q \in \mathcal{Q}$ is a state identifier name. As we said before, the purpose of state identifier names is allowing to match states of the model. For example, if s_1 `is` $q_1 \in \mathcal{E}$ and s_2 `is` $q_1 \in \mathcal{E}$ then we infer that $s_1 = s_2$ and that one of the states could be eliminated afterwards.
- $\mathcal{D}$ (*deterministic states*): Set of states that are deterministic (according to the hypotheses considered so far).
- $\mathcal{O}$ (*used observations*): Set of observations we have used so far for the construction of this model. The aim of recording them is to avoid considering the same observation several times, which could ruin the information codified, for instance, in $\mathcal{A}$.

In $\mathcal{HOTL}$, conclusions about the conformance of a model (that is, of the possible IUTs it represents) with respect to a specification will be established only after the full set of observations `Obs` has been considered. Besides, we will require that no other rule concerning hypotheses in `Hyp` can be applied. In Section 5 we introduce the hypotheses a tester might consider in this set. These hypotheses include usual ones such as to assume an upper bound on the number of states of the IUT, the uniqueness of the initial state, the determinism of the IUT, etcetera.

4.3 Other Predicates

Models can be labelled by some *tags* to denote special characteristics. The *correct* tag denotes that the model is *correct* with respect to the specification. That is, the `model` (*correct*(m)) predicate denotes that m is a correct model, that is, a model denoting a behavior that cannot be nonconforming to the

specification. A different predicate, `allModelsCorrect`, represents a set of correct models. This predicate is the *goal* of the logic: If it holds then all the IUTs that could produce the observations in `Obs` and meet all the requirements in `Hyp` conform to the specification. Another tag that may decorate models is the *consistent* tag. The `model` (*consistent*(m)) predicate means that the model m does not include any *inconsistency*. Note that the requirements imposed by `Obs` and `Hyp` could lead to inconsistent models. For example, let us consider a model where a state s is assumed to be deterministic, s is equal to another state s', and s' produces either o_1 or o_2 when i is offered, with $o_1 \neq o_2$. There is no `FSM` that meets the requirements of this model. Since a user of the logic can create a set of observations and hypotheses leading to that model, inconsistent models may indeed appear. As we will see, the rules of the logic will eliminate inconsistent models by deducing an *empty* set of models from them. In addition, the logic will provide rules that allow to *guarantee* the consistency of a model.

In general, several models can be constructed from a set of observations and hypotheses. Hence, our logic will deal with *sets* of models. If $\mathcal{M}$ is a set of models then the `models` ($\mathcal{M}$) predicate denotes that, according to the observations and hypotheses considered, $\mathcal{M}$ contains all the models that are valid candidates to properly describe the implementation. Sometimes it will be convenient to handle only a *subset* of these models. If we want to denote that $\mathcal{M}'$ is a subset of the models that may describe the implementation, then we will write `modelsSubset` ($\mathcal{M}'$).

The formal semantics of predicates, which is defined in terms of the set of `FSMs` that fulfill each predicate, is introduced in Section 6. In the same section, these concepts will be used to prove the soundness and completeness of the logic.

5 Deduction Rules of $\mathcal{HOTL}$

Rules will follow the format $\frac{premises}{conclusion}$. If B can be deduced from A, by using one of these rules, then we write $A \vdash B$. If B is deduced from A by using the rule r, we also write $A \vdash_r B$. The goal of the rules of the logic is to deduce the *conformance* of a set of observations `Obs` and hypotheses `Hyp`, that is, whether *all* the `FSMs` that meet these conditions conform to the specification. Since inconsistent models may appear, conformance will be granted only if there exists at least one *consistent* model that meets these premises. For the sake of readability, some formal definitions and rules have been moved to the forthcoming Section 5.2. The aim is to allow the reader to have an overview of $\mathcal{HOTL}$ before some technical concepts are described in detail. In these cases, brief informal explanations of these concepts will be provided in this section.

$\mathcal{HOTL}$ considers observations and hypotheses in two phases. First, observations, as well as the hypotheses they can implicitly express, are collected. Once all of them have been considered (i.e., we have constructed a model predicate with $\mathcal{O} = \texttt{Obs}$) a second phase, to add the rest of hypotheses, starts. All rules of the second phase will include the requirement $\mathcal{O} = \texttt{Obs}$.

First, we present a rule denoting how a model can be constructed from a simple observation. Given a predicate denoting that an observation was collected, the rule deduces some details about the behavior of the implementation. These details are codified by means of a *model* that shows this behavior. Basically, *new* states and transitions will be created in the model so that it can produce the observation. Even though some model states could actually coincide, we will not consider this fact yet. Thus, we take fresh states to name all of them. Besides, the hypotheses denoted by the attributes of the observation will affect the information associated to the corresponding model states. In particular, if the tester assumes that the last state of the observation is isomorphic to a state of the specification (i.e., $\texttt{spec}(s)$, for some $s \in \mathcal{S}_{spec}$) then the sets of states, transitions, accounting registers, and deterministic states will be extended with some extra elements taken from the specification and denoted by $\mathcal{S}'$, $\mathcal{T}'$, $\mathcal{A}'$, and $\mathcal{D}'$, respectively. The new states and transitions $\mathcal{S}'$ and $\mathcal{T}'$, respectively, will copy the structure existing among the states that can be reached from s in the specification. The new accounting, $\mathcal{A}'$, will denote that the knowledge concerning the new states is *closed* for all inputs, that is, the only transitions departing from these states are those we copy from the specification and no other transitions will be added in the future. Finally, those model states that correspond to deterministic specification states will be included in the set $\mathcal{D}'$ of deterministic states of the model.

$$(obser)\frac{ob = (a_1, i_1/o_1, a_2, \ldots, a_n, i_n/o_n, a_{n+1}) \in \texttt{Obs} \ \wedge \ s_1, \ldots, s_{n+1} \text{ are fresh states}}{\texttt{model}\left(\begin{array}{c} \{s_1, \ldots, s_{n+1}\} \cup \mathcal{S}', \\ \{s_1 \xrightarrow{i_1/o_1} s_2, \ldots, s_n \xrightarrow{i_n/o_n} s_{n+1}\} \cup \mathcal{T}', \{s_1, \beta\}, \\ \{(s_j, i_j, \{o_j\}, f_{s_j}, 1) \mid 1 \leq j \leq n\} \cup \mathcal{A}', \\ \{s_j \ \texttt{is} \ q_j | 1 \leq j \leq n+1 \ \wedge \ \texttt{imp}(q_j) \in a_j\}, \\ \{s_j \mid 1 \leq j \leq n+1 \ \wedge \ \texttt{det} \in a_j\} \cup \mathcal{D}', \{ob\} \end{array}\right)}$$

where $f_{s_j}(t) = 1$ if $t = s_j \xrightarrow{i_j/o_j} s_{j+1}$ and $f_{s_j}(t) = 0$ otherwise.

The formal definition of $\mathcal{S}'$, $\mathcal{T}'$, $\mathcal{A}'$, and $\mathcal{D}'$, denoting the additions due to an attribute of the form $\texttt{spec}(s)$, follows. If there does not exist s' such that $\texttt{spec}(s') \in a_{n+1}$ then $(\mathcal{S}', \mathcal{T}', \mathcal{A}', \mathcal{D}') = (\emptyset, \emptyset, \emptyset, \emptyset)$. Otherwise, that is, if $\texttt{spec}(s) \in a_{n+1}$ for some $s \in \mathcal{S}_{spec}$, let us consider the following set of states:

$$U = \{u_j \mid u_j \text{ is a fresh state } \wedge\ 1 \leq j < |\texttt{reachableStates}(spec, s)|\}$$

and a bijective function $g : \texttt{reachableStates}(spec, s) \longrightarrow U \cup \{s_{n+1}\}$ such that $g(s) = s_{n+1}$. Then, $(\mathcal{S}', \mathcal{T}', \mathcal{A}', \mathcal{D}')$ is equal to

$$\left(\begin{array}{c} U, \\ \{g(s') \xrightarrow{i/o} g(s'') \mid s' \xrightarrow{i/o} s'' \in \mathcal{T}_{spec} \ \wedge\ \texttt{isReachable}(spec, s, s')\}, \\ \left\{ \left(\begin{array}{c} u, i, \\ \texttt{outs}(spec, g^{-1}(u), i), \\ f^i_u, \top \end{array} \right) \middle| \begin{array}{l} u \in U \cup \{s_{n+1}\} \ \wedge\ i \in \texttt{inputs}_{spec} \wedge \\ \exists u' \in U, o \in \texttt{outputs}_{spec} :\ u \xrightarrow{i/o} u' \in \mathcal{T}' \end{array} \right\}, \\ \{g(s') | \texttt{isReachable}(spec, s, s') \ \wedge\ \texttt{isDet}(spec, s')\} \end{array} \right)$$

where $f^i_u(t) = 1$ for all t such that there exists o', u' with $t = u \xrightarrow{i/o'} u' \in \mathcal{T}'$ and $f^i_u(t) = 0$ otherwise.

Example 3 If we apply the *obser* deduction rule to the observations ob_1 and ob_3 given in Example 2, we obtain the following models:

$m_1 = (\mathcal{S}_1, \mathcal{T}_1, \mathcal{I}_1, \mathcal{A}_1, \mathcal{E}_1, \mathcal{D}_1, \mathcal{O}_1)$, where

$$\begin{aligned} \mathcal{S}_1 &= \{s_1, s_2, s_3, s_4, s_5, s_6\} \\ \mathcal{T}_1 &= \left\{ s_1 \xrightarrow{bc/mc} s_2, s_2 \xrightarrow{bc/mc} s_3, s_3 \xrightarrow{bc/re} s_4, s_4 \xrightarrow{br/mr} s_5, s_5 \xrightarrow{bc/mc} s_6 \right\} \\ \mathcal{I}_1 &= \{s_1, \beta\} \\ \mathcal{A}_1 &= \left\{ \begin{array}{l} (s_1, bc, \{mc\}, f_{s_1}, 1), (s_2, bc, \{mc\}, f_{s_2}, 1), (s_3, bc, \{re\}, f_{s_3}, 1), \\ (s_4, br, \{mr\}, f_{s_4}, 1), (s_5, bc, \{mc\}, f_{s_5}, 1) \end{array} \right\} \\ \mathcal{E}_1 &= \{s_2 \texttt{ is } q_1, s_4 \texttt{ is } q_2, s_6 \texttt{ is } q_1\},\ \mathcal{D}_1 = \emptyset, \text{ and } \mathcal{O}_1 = \{ob_1\} \end{aligned}$$

$m_3 = (\mathcal{S}_3, \mathcal{T}_3, \mathcal{I}_3, \mathcal{A}_3, \mathcal{E}_3, \mathcal{D}_3, \mathcal{O}_3)$, where

$$\mathcal{S}_3 = \{s_{13}, s_{14}, s_{15}, s_{16}\}$$

$$\mathcal{T}_3 = \left\{ s_{13} \xrightarrow{bc/mc} s_{14}, s_{14} \xrightarrow{bc/mc} s_{15}, s_{15} \xrightarrow{br/mr} s_{16} \right\}$$

$$\mathcal{I}_3 = \{s_{13}, \beta\}$$

$$\mathcal{A}_3 = \left\{ (s_{13}, bc, \{mc\}, f_{s_{13}}, 1), (s_{14}, bc, \{mc\}, f_{s_{14}}, 1), (s_{15}, br, \{mr\}, f_{s_{15}}, 1) \right\}$$

$$\mathcal{E}_3 = \{s_{14} \texttt{ is } q_1, s_{15} \texttt{ is } q_2\},\ \mathcal{D}_3 = \{s_{14}\}, \text{ and } \mathcal{O}_3 = \{ob_3\}$$

where f_{s_i} returns 1 for the unique transition departing from s_i and 0 otherwise.

Similarly, for all $1 \leq i \leq 7$ we can obtain a model m_i by applying the deduction rule *obser* to ob_i. □

We will be able to join different models created from different observations into a single model. The components of the new model will be the union of the components of each model.

$$(fusion)\frac{\begin{array}{c}\texttt{model}\,(\mathcal{S}_1, \mathcal{T}_1, \mathcal{I}_1, \mathcal{A}_1, \mathcal{E}_1, \mathcal{D}_1, \mathcal{O}_1)\ \wedge \\ \texttt{model}\,(\mathcal{S}_2, \mathcal{T}_2, \mathcal{I}_2, \mathcal{A}_2, \mathcal{E}_2, \mathcal{D}_2, \mathcal{O}_2)\ \wedge\ \mathcal{O}_1 \cap \mathcal{O}_2 = \emptyset\end{array}}{\texttt{model}\left(\mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{T}_1 \cup \mathcal{T}_2, \mathcal{I}_1 \cup \mathcal{I}_2, \mathcal{A}_1 \cup \mathcal{A}_2, \mathcal{E}_1 \cup \mathcal{E}_2, \mathcal{D}_1 \cup \mathcal{D}_2, \mathcal{O}_1 \cup \mathcal{O}_2\right)}$$

The condition $\mathcal{O}_1 \cap \mathcal{O}_2 = \emptyset$ appearing in the previous rule avoids to include the same observation in a model more than once, which would be inefficient. Besides, since models in the second phase must fulfill $\mathcal{O} = \texttt{Obs}$, we avoid to use the previous rule in the second phase.

By iteratively applying these two first rules, we will eventually obtain a model where $\mathcal{O}$ includes all the observations belonging to the set `Obs`.

Example 4 The deduction rule *fusion* allows to join all the models obtained after applying the deduction rule *obser* to the set of observations given in Example 2. After it, we have $\texttt{model}\,(m_T)$ for a model m_T defined as follows:

$$m_T = \left(\bigcup_{j=1}^{7} \mathcal{S}_j, \bigcup_{j=1}^{7} \mathcal{T}_j, \bigcup_{j=1}^{7} \mathcal{I}_j, \bigcup_{j=1}^{7} \mathcal{A}_j, \bigcup_{j=1}^{7} \mathcal{E}_j, \bigcup_{j=1}^{7} \mathcal{D}_j, \texttt{Obs}\right)$$

□

At this point, the inclusion of those hypotheses that are not covered by observations will begin. During this new phase, we will usually need several models

to represent all the FSMs that are compatible with a set of observations and hypotheses. The next simple rule allows to represent a single model by means of a set containing a single element. Since the forthcoming rules will concern only the second phase, in all cases we will have $\mathcal{O} = \texttt{Obs}$.

$$(set)\frac{\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})}{\texttt{models}\,(\{(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\})}$$

In order to reflect how a rule that applies to a single model affects the set including this model, we provide the following rule. Let φ denote any logical predicate (in particular, it could be the predicate `true`) and m be a model. Then,

$$(propagation)\frac{\texttt{models}\,(\mathcal{M}\cup\{m\})\wedge\varphi\wedge((\texttt{model}\,(m)\wedge\varphi)\vdash\texttt{modelsSubset}\,(\mathcal{M}'))}{\texttt{models}\,(\mathcal{M}\cup\mathcal{M}')}$$

By using the previous rule, we will be able to use other rules that apply to a *single* model and then propagate its change to the set where the model is included as expected: As the previous rule states, the considered model is modified while other models belonging to the set remain unchanged. Actually, most of the forthcoming rules will apply to single models. After each of them is used, the rule *propagation* will be applied to propagate its effect to the corresponding set of models.

Our logic will allow to discover that a state of the model coincides with another one. In this case, we will eliminate one of the states and will allocate all of its constraints to the other one. This will modify all the components that define the model. This functionality is provided by the `modelElim` function. Specifically, $\texttt{modelElim}(m, s_1, s_2)$ denotes the elimination of the state s_2 and the transference of all its responsibilities to the state s_1 in the model m. This function returns a *set* of models. If the transference of responsibilities creates an *inconsistency* in the rest of the model, an empty set of models is returned. Sometimes we will use a *generalized* version of this function to perform the consecutive elimination of several states: $\texttt{modelElim}(m, s, \{s_1, \ldots, s_n\})$ represents the substitution of s_1 by s, followed by the substitution of s_2 by s, and so on up to s_n. The formal definitions of both variants of the `modelElim` function are given in Section 5.2.

Next we present some rules that use this function. In the first one, we join two states if the set of equalities allows to deduce that both coincide.

$$(equality)\frac{\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\ \ s_1,s_2\in\mathcal{S}\ \wedge\ \{s_1\ \texttt{is}\ q, s_2\ \texttt{is}\ q\}\subseteq\mathcal{E}}{\texttt{modelsSubset}\,(\texttt{modelElim}((\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs}),s_1,s_2))}$$

Example 5 Let us denote by $\mathcal{E}_T$ the set of equality relations of the model m_T. By iteratively applying the *equality* deduction rule to all pairs of elements of the form $\{s'\ \text{is}\ q, s''\ \text{is}\ q\}$ in $\mathcal{E}_T$ until there are no more pairs like this, we reduce the number of states in the model. For example, let us consider the states s_2 and s_{14}, extracted from ob_1 and ob_3, respectively. We have $\{s_2\ \texttt{is}\ q_1, s_{14}\ \texttt{is}\ q_1\}\subseteq\mathcal{E}_T$. After applying the *equality* rule to this pair, we obtain a new model where all the occurrences of s_{14} are replaced by s_2. The state s_{14} is removed from $\mathcal{S}_T$, the transitions $s_{13}\xrightarrow{bc/mc}s_{14}$, $s_{14}\xrightarrow{bc/mc}s_{15}\in\mathcal{T}_T$ are transformed into $s_{13}\xrightarrow{bc/mc}s_2$, $s_2\xrightarrow{bc/mc}s_{15}$, and the set of initial states remains unchanged. The set of observations is $\texttt{Obs}$. Regarding the set of accounting registers, the element $(s_{14},bc,\{mc\},f_{s_{14}},1)$ is removed, while the former element $(s_2,bc,\{mc\},f_{s_2},1)$ is substituted by a new element $(s_2,bc,\{mc\},f'_{s_2},2)$, where $f'_{s_2}(t)=1$ if t is either $s_2\xrightarrow{bc/mc}s_3$ or the new transition $s_2\xrightarrow{bc/mc}s_{15}$ and $f'_{s_2}(t)=0$ otherwise. Let us note that also the function $f_{s_{13}}$ of the register $(s_{13},bc,\{mc\},f_{s_{13}},1)$ must change: A new function $f'_{s_{13}}$ substitutes $f_{s_{13}}$, where $f'_{s_{13}}(s_{13}\xrightarrow{bc/mc}s_{14})=0$ and $f'_{s_{13}}(s_{13}\xrightarrow{bc/mc}s_2)=1$. Finally, s_{14} is removed from the set of deterministic states $\mathcal{D}$, and s_2 is introduced in this set. □

Another situation where two states can be fused appears when a deterministic state shows two transitions labelled by the same input. Since the state is deterministic, they must also be labelled by the same output. The determinism of the state implies that both destinations are actually the same state. Hence, these two reached states can be fused. Note that if both outputs are different then the model is inconsistent, because the determinism of the state is not preserved. In this case, an empty set of models is produced.

$$(determ)\frac{\begin{array}{c}\texttt{model}\,(m)\ \wedge\ m=(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\\ s,s_1,s_2\in\mathcal{S}\ \wedge\ s\in\mathcal{D}\ \wedge\ \{s\xrightarrow{i/o_1}s_1, s\xrightarrow{i/o_2}s_2\}\subseteq\mathcal{T}\end{array}}{\texttt{modelsSubset}\,(\mathcal{M}')}$$

where $\mathcal{M}'=\texttt{modelElim}(m,s_1,s_2)$ if $o_1=o_2$ and $\mathcal{M}'=\emptyset$ otherwise.

Example 6 We apply the *determ* deduction rule to the deterministic state s_2 and the transitions $s_2\xrightarrow{bc/mc}s_3$ and $s_2\xrightarrow{bc/mc}s_{15}$. Using this rule implies

the application of the `modelElim` function to the states s_3 and s_{15}. After it, we obtain a new model where the state s_{15} is removed and its occurrences are replaced by s_3. □

Next we present the first rule dealing with an hypothesis that is not implicitly given by an observation. Thus, we will consider an element belonging to the set `Hyp`. This hypothesis allows to assume that the initial state of the implementation is *unique*. In this case, all initial states will be fused. Besides, any symbol in $\mathcal{I}$ denoting that other states could be initial, that is, α and β, will be eliminated.

$$(singleInit)\frac{\begin{array}{c}\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\ \mathcal{I}\cap\mathcal{S}=\{s_1,\ldots,s_n\}\ \wedge\\ \texttt{singleInit}\in\texttt{Hyp}\ \wedge\ m'=(\mathcal{S},\mathcal{T},\mathcal{I}\backslash\{\alpha,\beta\},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\end{array}}{\texttt{modelsSubset}\,(\texttt{modelElim}(m',s_1,\{s_2,\ldots,s_n\}))}$$

Example 7 At this point we can include the predicate `singleInit` $\in$ `Hyp` into our running example. This indicates that the tester assumes the uniqueness of the initial state. As a result, all the states that were supposed to be initial collapse into one. In our case, we will consider that this state is s_1 (it could be any other initial state). By applying the `modelElim` function to the model, we obtain a new model where all the occurrences of the initial state s_{13} are replaced by s_1. Similarly, the rest of initial states are substituted by s_1 until the set of initial states $\{s_1\}$ is obtained. □

If the tester adds the hypothesis that all the states are deterministic then the complete set of states $\mathcal{S}$ coincides with the set of deterministic states $\mathcal{D}$.

$$(allDet)\frac{\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\ \texttt{allDet}\in\texttt{Hyp}}{\texttt{modelsSubset}\,(\{(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{S},\texttt{Obs})\})}$$

The logic $\mathcal{HOTL}$ allows to consider other hypotheses about the IUT. For example, the predicate `allTranHappenWith`(n) assumes that for all state s and input i such that the IUT behavior has been observed n times, *all* the outgoing transitions from the state s having as input i have been observed at least once during these n times. This means that the IUT state s cannot react to i with an output that has not produced so far or moving to a state it has not moved before. If the hypothesis is assumed then some accounting registers of the model will be set to $\top$, denoting that our knowledge about this state and input is *closed*. Depending on the compatibility of the hypothesis with the current model, several models can be produced by this rule. If no model is returned then we infer that the resulting model is inconsistent with the current model

requirements. The $\texttt{upperBoundOfStates}(n)$ hypothesis allows to assume that the IUT uses at most n states. The reduction of states, based on the identification of several states with the same state identifiers, will be performed by means of new equalities $s \texttt{ is } q \in \mathcal{E}$. The $\texttt{longSequencesSamePath}(n)$ hypothesis assumes that if two sequences of n transitions produce the same inputs and outputs, then they actually go through the same states. The set $\mathcal{E}$, containing the assumed equalities between states, will be also used in this case. Given $i \in \texttt{inputs}_{spec}$ and $o \in \texttt{outputs}_{spec}$, the $\texttt{uniqueOrigin}(i, o)$ hypothesis allows to assume that the departing state coincides in all IUT transitions labelled by the pair i/o, that is, there is a single origin for all these transitions in the IUT. Similarly, the $\texttt{uniqueDestination}(i, o)$ hypothesis assumes that the destination of all transitions labelled by i/o is unique. In Section 5.2 we present the formal definition of the rules that allow to consider the $\texttt{allTranHappenWith}(n)$, $\texttt{upperBoundOfStates}(n)$, $\texttt{uniqueDestination}(i, o)$, $\texttt{uniqueOrigin}(i, o)$, and $\texttt{longSequencesSamePath}(n)$ hypotheses. The repertory of hypotheses of $\mathcal{HOTL}$ presented in this paper is compiled in an appendix at the end of the paper.

Example 8 Let us assume that we apply the *equality* and *determ* deduction rules to our model as far as we can. Since we have $\{s_4 \texttt{ is } q_2, s_{15} \texttt{ is } q_2\} \subseteq \mathcal{E}_T$, s_{15} collapses into s_4. Similarly, other model states are fused to s_4 as well. Since none of them is deterministic, s_4 is not assumed to be so. Let us suppose that $\texttt{allTranHappenWith}(2)$ is included in $\texttt{Hyp}$. Since we had $s_4 \xrightarrow{br/mr} s_5$ and $s_{15} \xrightarrow{br/mr} s_{16}$ before s_4 and s_{15} were fused, after joining them we obtain $(s_4, br, \{mr\}, f'_{s_4}, 2) \in \mathcal{A}_T$, where $f'_{s_4}(t) = 1$ if t is either $s_4 \xrightarrow{br/mr} s_5$ or the new transition $s_4 \xrightarrow{br/mr} s_{16}$ and $f'_{s_4}(t) = 0$ otherwise. Hence, by $\texttt{allTranHappenWith}(2)$ we have that $(s_4, br, \{mr\}, f'_{s_4}, \top) \in \mathcal{A}_T$, that is, the behavior of s_4 when *br* is received is closed (see the definition of the *allTran* rule in the next section). Thus, the only available destinations from s_4 when *br* is produced are s_5 and s_{16}.

Let us assume $\texttt{uniqueDestination}(br, mr) \in \texttt{Hyp}$. This implies, in particular, that the destinations of the transitions $s_4 \xrightarrow{br/mr} s_5$ and $s_4 \xrightarrow{br/mr} s_{16}$ coincide (see *destination* in the next section). Hence, s_{16} collapses into s_5 and we deduce that the destination from s_4 when *br* is produced is unique: s_5. □

We have seen some rules that may lead to inconsistent models. In some of these cases, an empty set of models is produced, that is, the inconsistent model is eliminated. Before granting conformance, we need to be sure that at least one model belonging to the set of models is consistent. Next we provide a rule that labels a model as consistent. Let us note that inconsistencies created by the application of a rule are sometimes detected by the application of subsequent rules. For instance, the *determ* rule can detect that a previous rule matched

a deterministic state with another state in such a way that both react to the input i with a different output. Similarly, the *equality* rule may detect, by means of the `modelElim` function, that fusing two states as required is not consistent. Actually, as we will see in Section 6, all inconsistencies can be detected by applying suitable rules. Thus, a model is free of inconsistencies if any other rule is either not applicable to the model or the application does not modify the model (that is, it deduces the same model). Next we introduce this concept. In the following definition, $\mathcal{R}$ denotes the set of all rules in $\mathcal{HOTL}$ that follow the form required to apply the *propagation* rule. In particular, this set consists of all previous rules from *equality* up to the forthcoming *correct* rule.

Definition 3 We denote the set of all rules in $\mathcal{HOTL}$ that follow the form $(\texttt{model}\,(m)\ \wedge\ \varphi) \vdash \texttt{modelsSubset}\,(\mathcal{M})$ by $\mathcal{R}$.

Let $r = (\texttt{model}\,(m)\ \wedge\ \varphi) \vdash \texttt{modelsSubset}\,(\mathcal{M}) \in \mathcal{R}$ be a rule and m' be a model. The *unable predicate* for m' and r, denoted by $\texttt{unable}(m', r)$, is defined by the expression

$$\texttt{unable}(m', r) = \neg\varphi\ \vee\ ((\texttt{model}\,(m')\ \wedge\ \varphi) \vdash_r \texttt{modelsSubset}\,(\{m'\}))$$

We extend this predicate to deal with sets of rules G as expected:

$$\texttt{unable}(m', G) = \bigwedge\{\texttt{unable}(m', r) | r \in G\}$$

□

The next rule detects that a model is consistent. It requires that no other rule that manages hypotheses can modify the model. Specifically, these rules are all the rules belonging to $\mathcal{R}$ we have seen so far.

$$(consistent)\frac{\begin{array}{c} m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs})\ \wedge\ \texttt{model}\,(m)\ \wedge \\ \texttt{unable}(m, \mathcal{R}\backslash\{consistent, correct\}) \end{array}}{\texttt{modelsSubset}\,(\{consistent(m)\})}$$

Since a model is a (probably incomplete) representation of the IUT, in order to check whether a model conforms to the specification, two aspects must be taken into account. First, only the conformance of consistent models will be considered. Second, given a consistent model, we will check its conformance with respect to the specification by considering the *worst* instance of the model, that is, if this instance conforms to the specification then any other instance extracted from the model does so. This worst instance (an `FSM`) is constructed as follows: For each state s and input i such that the behavior of s for i is not closed *and* either s is not deterministic or no transition with input i

exists in the model, a new *malicious* transition is created. The new transition is labelled with a special output *error*, that does not belong to $\texttt{outputs}_{spec}$. This transition leads to a new state $\perp$ that produces the same output *error* for any input. Since the specification cannot produce the output *error*, this worst instance will conform to the specification only if the unspecified parts of the model are not relevant for the correctness of the IUT it represents (see Section 6).

Definition 4 Let $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs})$ be a model. We define the *worst instance* of the model m with respect to the considered specification *spec*, denoted by $\texttt{worstCase}(m)$, as the FSM

$$\left(\begin{array}{l} \mathcal{S} \cup \{\perp\}, \texttt{inputs}_{spec}, \texttt{outputs}_{spec} \cup \{error\}, \\ \mathcal{T} \cup \left\{ s \xrightarrow{i/error} \perp \;\middle|\; \begin{array}{l} s \in \mathcal{S} \cup \{\perp\} \;\wedge\; i \in \texttt{inputs}_{spec} \wedge \\ \nexists\, outs, f : (s, i, outs, f, \top) \in \mathcal{A} \wedge \\ (s \notin \mathcal{D} \vee \nexists\, s', o : s \xrightarrow{i/o} s' \in \mathcal{T}) \end{array} \right\}, \\ \mathcal{I}' \end{array} \right)$$

where $\mathcal{I}'$ is defined as

$$\mathcal{I}' = \begin{cases} \mathcal{I} & \text{if } \mathcal{I} \cap \{\alpha, \beta\} = \emptyset \\ \mathcal{S} & \text{if } \alpha \in \mathcal{I} \\ \mathcal{S} \cup \{\perp\} & \text{otherwise} \end{cases}$$

□

Thus, the rule for indicating the correctness of a model is

$$(correct)\frac{\begin{array}{c} m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs}) \wedge \\ \texttt{model}\,(consistent(m)) \;\wedge\; \texttt{worstCase}(m)\;\texttt{conf}\;spec \end{array}}{\texttt{modelsSubset}\,(\{correct(m)\})}$$

Now we can consider the conformance of a set of models. A set conforms to the specification if all the elements do so and the set contains at least one element. Note that an empty set of models denotes that all the models were inconsistent. Hence, granting the conformance of an empty set would imply accepting models that do not represent any implementation. [2]

[2] If an empty set of models is deduced, we do not provide a final diagnostic of conformance since we have actually detected that it is impossible that an FSM produces the observations in Obs and fulfills the hypotheses assumed by the tester, denoted by Hyp. Even though *false implies anything*, accepting inconsistent models is not useful for a tester.

$$(allCorrect)\frac{\texttt{models}\,(\mathcal{M})\ \wedge\ \mathcal{M}\neq\emptyset\ \wedge\ \mathcal{M}=\{correct(m_1),\ldots,correct(m_n)\}}{\texttt{allModelsCorrect}}$$

Example 9 Next we complete our running example. In addition to the observations $ob_1,\ldots,ob_7$ and the hypotheses `singleInit`, `allTranHappenWith`(2), and `uniqueDestination`(br, mr) considered before, only one more hypothesis is assumed in our example: `uniqueDestination`(bb, re). Let us consider the model m_R obtained after applying the *determ*, *equality*, *singleInit*, *allTran*, and *destination* deduction rules.

These rules are applied as follows. The *singleInit* rule is applied once and the *destination* rule is used twice (once for each assumed input/output pair), while *determ*, *equality*, and *allTran* are repeatedly applied as long as any of them returns a different model. Let us recall that after each of these rules is used, the *propagation* rule must be applied as well. When the *determ*, *equality*, and *allTran* rules cannot be applied any more, our model cannot be further manipulated to produce new inconsistencies. Then, we can use the *consistent* and *propagation* rules to deduce `models` $(\{consistent(m_R)\})$.

We build an `FSM` by applying the function `worstCase` to m_R and we verify its conformance with respect to the specification. The obtained `FSM`, denoted by $worst_{spec\_ray}$, is graphically depicted in Figure 4. For the sake of clarity, we have included two states $\perp$, even though they correspond to only one state.

We have $worst_{spec\_ray}$ `conf` $spec\_ray$ and, by applying the *correct* deduction rule (followed by *propagation*), we obtain `models` $(\{correct(m_R)\})$ and deduce, by means of the *allCorrect* deduction rule, `allModelsCorrect`. That is, if these observations are obtained from the IUT and these hypotheses are assumed, then the IUT necessarily conforms to the specification. □

Now that we have presented the set of deduction rules, we introduce a correctness criterion. In the next definition, in order to uniquely denote observations, fresh names are assigned to them. Observation predicates follow the form '$ob = o \in$ `Obs`', where o belongs to `Obs` and ob is the name of the observation. Hypothesis predicates follow the form '$h \in$ `Hyp`' for some h belonging to `Hyp`.

Definition 5 Let *spec* be an `FSM`, `Obs` be a set of observations, and `Hyp` be a set of hypotheses. Let P_{obs} and P_{hyp} be the sets of predicates defined as follows:

$$P_{obs} = \{ob = o \in \texttt{Obs} \mid ob \text{ is a fresh name } \wedge\ o \in \texttt{Obs}\}$$

$$P_{hyp} = \{h_1 \in \texttt{Hyp},\ldots,h_n \in \texttt{Hyp}\},\quad \texttt{Hyp} = \{h_1,\ldots,h_n\}$$

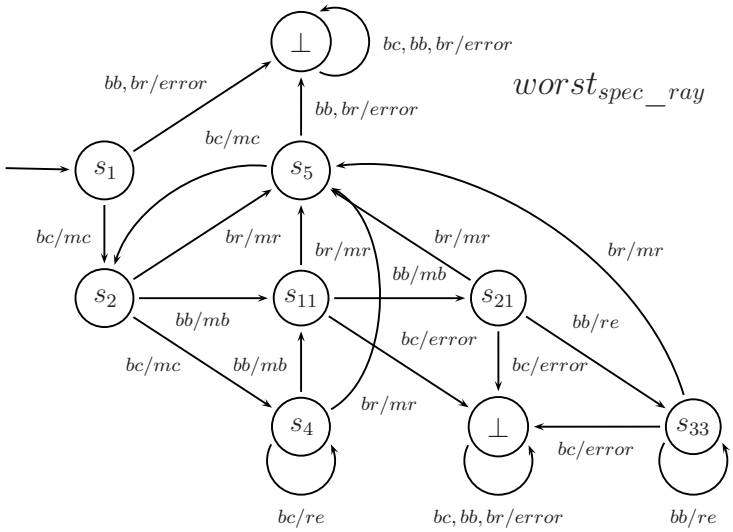

Fig. 4. Finite State Machine $worst_{spec\_ray}$.

If the deduction rules allow to infer a predicate p from the set of predicates $P = P_{obs} \cup P_{hyp}$ then we say that p is deduced from P and we denote it by $P \vdash^* p$.

If we have $P \vdash^* \texttt{allModelsCorrect}$ then we say that P *logically conforms to spec* and we denote it by P `logicConf` *spec*. □

5.2 Additional definitions

In this section we present some formal concepts that were used, but not formally defined, in the previous section. The definition of the `modelElim` function is constructed in two steps. When we eliminate a state model and transfer its responsibilities to another state, we have to modify all the components defined by the model. First, we show how to modify one of them, the *accounting*. The next function shows how an accounting $\mathcal{A}$ is updated when a state s_2 is modified because it is equal to another state s_1. Basically, we move all the accounting information from s_2 to s_1. In the definition, the new accounting set is constructed by joining two sets. The first set denotes the accounting for all states different from s_1 and s_2. A register $(s, i, outs, f, n) \in \mathcal{A}$, with $s \neq s_1$ and $s \neq s_2$, will change only if there is a registered transition from s to s_2 with input i, that is, if $f(s \xrightarrow{i/o} s_2) > 0$ for some o. In this case, the information provided by f must denote the change of s_2 by s_1: We set $f(s \xrightarrow{i/o} s_2) = 0$ and we increase the value of $f(s \xrightarrow{i/o} s_1)$. The second set denotes the new registers of s_1. They add all the information taken from both s_1 and s_2. Finally, all

information concerning s_2 is removed.

Definition 6 Let $\mathcal{A}$ be a set of *accounting registers* and s_1, s_2 be states. Then, we have

$$\texttt{countElim}(\mathcal{A}, s_1, s_2) =$$

$$\left\{ (s, i, outs, f', n) \;\middle|\; \begin{array}{l} s \notin \{s_1, s_2\} \wedge (s, i, outs, f, n) \in \mathcal{A} \wedge \\ f'(s \xrightarrow{i/o} s') = \begin{cases} f(s \xrightarrow{i/o} s') & \texttt{if } s' \neq s_1, s_2 \\ f(s \xrightarrow{i/o} s_1) + f(s \xrightarrow{i/o} s_2) & \texttt{if } s' = s_1 \\ 0 & \texttt{if } s' = s_2 \end{cases} \end{array} \right\}$$

$$\bigcup$$

$$\left\{ \begin{pmatrix} s_1, i, \\ outs_1 \cup outs_2, \\ f', n \end{pmatrix} \;\middle|\; \begin{array}{l} \exists\, p, q, g, h : \\ \quad ((s_1, i, outs_1, g, p) \in \mathcal{A} \;\vee\; (s_2, i, outs_2, h, q) \in \mathcal{A}) \wedge \\ n = \sum\{m \mid (s, i, outs, f, m) \in \mathcal{A}, s \in \{s_1, s_2\}\} \wedge \\ f'(t) = \sum\{f(t) | (s, i, outs, f, m) \in \mathcal{A}, s \in \{s_1, s_2\}\} \end{array} \right\}$$

We assume that for all $n \in \mathbb{N}$ we have $n + \top = \top$. □

The previous function is auxiliary to create another one that defines how to eliminate a state s when we discover that this state is equal to another state s'. As we said before, we will transfer all the responsibilities of the state to be eliminated to the other state. Let us note that sometimes this could be *inconsistent*. For example, if for the state s we have that the behavior with input i is *closed*, that is, $(s, i, outs, f, \top) \in \mathcal{A}$, and for the state s' we have an outgoing transition labelled with i/o, being $o \notin outs$, then the resulting model would be inconsistent because it would not preserve the closed behavior of s. In this case, an empty set of models will be returned by the function (see case (a) of the following definition). Otherwise, the new model is obtained by *substituting* all occurrences of the state to be eliminated by the state that will stay (case (b)). We will denote by $[x/y]$ the renaming of any occurrence of x by y. In the next definition, we use the following property: For all index $j \in \{1, 2\}$, the expression $3 - j$ always denotes the other number of the set.

Definition 7 Let $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \mathcal{O})$ be a model and $s_1, s_2 \in \mathcal{S}$. We define the predicate $\texttt{modelElim}(m, s_1, s_2)$ as $\texttt{models}\,(\mathcal{M})$, where $\mathcal{M}$ is constructed as follows:

(a) If there exist $i, outs, f, o, s$, and $j \in \{1, 2\}$ such that $s_{3-j} \xrightarrow{i/o} s \in \mathcal{T}$, $(s_j, i, outs, f, \top) \in \mathcal{A}$, and $o \notin outs$, then $\mathcal{M} = \emptyset$.

(b) Otherwise, $\mathcal{M} = \left\{ \left(\begin{array}{c} \mathcal{S}\backslash\{s_2\}, \mathcal{T}[s_2/s_1], \mathcal{I}[s_2/s_1], \\ \texttt{countElim}(\mathcal{A}, s_1, s_2), \mathcal{E}[s_2/s_1], \\ \mathcal{D}[s_2/s_1], \mathcal{O} \end{array} \right) \right\}$

The previous function can be generalized to operate over *sets* of states as follows. Let $S \subseteq \mathcal{S}$ be a set of states. We have

$$\texttt{modelElim}(m, s, S) = \begin{cases} \{m\} & \texttt{if } S = \emptyset \\ \bigcup_{j=1}^{k} \texttt{modelElim}(m'_j, s, \{s_2, \ldots, s_n\}) & \texttt{if } S = \{s_1, \ldots, s_n\} \end{cases}$$

where $\{m'_1, \ldots, m'_k\} = \texttt{modelElim}(m, s, s_1)$. □

Next we present the formal definitions of the deduction rules that allow to consider the remaining hypotheses. The $\texttt{allTranHappenWith}(n)$ hypothesis allows to assume that if an input is produced n times at a given state, then we observe all the outputs that can be produced at this state in response to this input, and we move to all the states we can move from this state with this input. In particular, we assume that all *transitions* leaving this state with this input are observed. Let us note that we could assume this hypothesis in a model that actually does *not* fulfill it. For example, let T' be a set of transitions leaving the state s with the input i such that the number of times we have observed occurrences of transitions belonging to T' is n. According to the hypothesis, the transitions in T' are *all* transitions leaving s with i. However, other transitions from s with i could have been observed and they would be part of the model as well. If the hypothesis holds then these transitions should be *repeated*, that is, each one should coincide with one of the transitions in T'. However, if one of them produces an output that cannot be produced by one of the transitions of T' then this transition is necessarily *different* and the hypothesis does not hold (see case (a) in the following rule). This inconsistency will be indicated by returning an *empty* set of models. Otherwise, the destinations of these transitions will be fused with the destinations of the transitions of T'. In this way, they will be transformed into *repeated* transitions (case (b) in the following rule). These fusions will be denoted by introducing new elements in the set of equalities $\mathcal{E}$. The set K appearing in the rule consists of all the sets $\mathcal{E}'$ showing a possible way to match states to have this effect. For each $\mathcal{E}'$, a different model is built. We assume that $\top \geq n$ for all $n \in \mathbb{N}$.

$$(allTran)\frac{\begin{array}{l}\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\\ \mathcal{M}'=\{(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A}',\mathcal{E}\cup\mathcal{E}',\mathcal{D},\mathcal{O})\mid\mathcal{E}'\in K\}\ \wedge\\ n\in\mathbb{N}\ \wedge\ \texttt{allTranHappenWith}(n)\in\texttt{Hyp}\end{array}}{\texttt{modelsSubset}\,(\mathcal{M}')}$$

where

$$\mathcal{A}'=\ \{(s,i,outs,f,n')|(s,i,outs,f,n')\in\mathcal{A},n'<n\}$$
$$\bigcup$$
$$\{(s,i,outs,f,\top)|(s,i,outs,f,n')\in\mathcal{A},n'\geq n\}$$

and K is defined as follows. For all state s and input i, let T_s^i denote the set of transitions outgoing from s with i, that is, $T_s^i=\{t\mid\exists\,o,s':t{=}s\xrightarrow{i/o}s'\in\mathcal{T}\}$. Then,

(a) Let $(s,i,outs,f,n')\in\mathcal{A}$ such that $|T_s^i|>n$. Let us suppose that there exist $T'\subseteq T_s^i$, with $\sum\{f(t)|t\in T'\}\geq n$, and a transition $s\xrightarrow{i/o}s'\in T_s^i\backslash T'$ such that for all $s\xrightarrow{i/o'}s''\in T'$ we have $o\neq o'$. Then, $K=\emptyset$ (i.e., $\mathcal{M}'=\emptyset$).

(b) Otherwise, K is the set containing all the sets $\mathcal{E}'$ fulfilling:
- For all $(s,i,outs,f,n')\in\mathcal{A}$ such that $|T_s^i|>n$ and set $T'\subseteq T_s^i$, with $T'=\{s\xrightarrow{i/o_1}s_1,\ldots,s\xrightarrow{i/o_q}s_q\}$ such that $\sum\{f(t)|t\in T'\}\geq n$, let us consider the set $\{u_i\mid 1\leq i\leq q'\}=\{u|\exists\ o:s\xrightarrow{i/o}u\in T_s^i\backslash T'\}$. Then, for some set of q fresh state identifier names $\{w_i\mid 1\leq i\leq q\}$ we have $\{s_i\ \texttt{is}\ w_i\mid 1\leq i\leq q\}\subseteq\mathcal{E}'$ and $\{u_i\ \texttt{is}\ w_{r_i}\mid 1\leq i\leq q'\}\subseteq\mathcal{E}'$, where for all $1\leq k\leq q'$ there exists o_k such that $s\xrightarrow{i/o_k}u_k\in T_s^i\backslash T'$ and $s\xrightarrow{i/o_k}s_{r_k}\in T'$, with $1\leq r_k\leq q$.
- There are not more elements belonging to $\mathcal{E}'$.

If there does not exist such a set $\mathcal{E}'$ fulfilling the previous properties then we consider $K=\{\emptyset\}$ (i.e., $\mathcal{M}'=\{(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A}',\mathcal{E},\mathcal{D},\texttt{Obs})\}$).

In the remaining rules, when we talk about *all* the sets that fulfill a condition (like the previous one in case (b)), we will consider them *up to* renaming of fresh states, that is, α-conversion. For example, we consider that $\{s_1\ \texttt{is}\ w_1,s_2\ \texttt{is}\ w_1,s_3\ \texttt{is}\ w_2\}$ and $\{s_1\ \texttt{is}\ w_2,s_2\ \texttt{is}\ w_2,s_3\ \texttt{is}\ w_3\}$ (where w_1,w_2,w_3 are fresh state identifier names) are in fact the same sets.

If we assume an upper bound n on the number of states, that is, we suppose $\texttt{upperBoundOfStates}(n)$, forthcoming fusions of states will lead to a model where at most n states will be used. In particular, each state will be assigned to a state identifier name taken from a set of n names. Each way to assign these names will yield a different model that is included in the set of deduced models.

$$(upper)\frac{\begin{array}{l}\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\\ \mathcal{M}'=\{(\mathcal{S},\mathcal{T},\mathcal{I}',\mathcal{A},\mathcal{E}\cup\mathcal{E}',\mathcal{D},\texttt{Obs})\mid\mathcal{E}'\in K\}\ \wedge\\ n\in\mathbb{N}\ \wedge\ \texttt{upperBoundOfStates}(n)\in\texttt{Hyp}\end{array}}{\texttt{modelsSubset}\,(\mathcal{M}')}$$

where $\mathcal{I}' = (\mathcal{I}\backslash\{\beta\})\cup\{\alpha\}$ if $\beta\in\mathcal{I}$ and $\mathcal{I}'=\mathcal{I}$ otherwise. In order to define the set K, we have $K=\{\emptyset\}$ if $|\mathcal{S}|\leq n$; otherwise, let $\mathcal{S}=\{s_1,\ldots,s_q\}$ and $\{w_1,\ldots,w_n\}$ be a set of n fresh state identifier names. Then, K is the set of all the sets $\mathcal{E}'$ fulfilling $\{s_i\ \texttt{is}\ w_{r_i}\mid 1\leq i\leq q\}\subseteq\mathcal{E}'$, where for all $1\leq i\leq q$ we have $1\leq r_i\leq n$, and $\mathcal{E}'$ does not contain other elements.

The $\texttt{longSequencesSamePath}(n)$ hypothesis assumes that all sequences having at least n transitions and producing the same sequence of inputs and outputs actually traverse the same implementation states. In this case, we will match those states that are at the same points of each of these sequences. In order to do that, we will bind the states that correspond to equivalent points by adding new equalities in the set of equalities.

$$(long)\frac{\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\ n\in\mathbb{N}\ \wedge\ \texttt{longSequencesSamePath}(n)\in\texttt{Hyp}}{\texttt{modelsSubset}\,(\{(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E}\cup\mathcal{E}',\mathcal{D},\texttt{Obs})\})}$$

where

$$\mathcal{E}'=\left\{\begin{array}{l|l} s_i\ \texttt{is}\ w_i, & s_1\xrightarrow{i_1/o_1}s_2\in\mathcal{T},\ldots,s_n\xrightarrow{i_n/o_n}s_{n+1}\in\mathcal{T}\ \wedge\\ s'_i\ \texttt{is}\ w_i & s'_1\xrightarrow{i_1/o_1}s'_2\in\mathcal{T},\ldots,s'_n\xrightarrow{i_n/o_n}s'_{n+1}\in\mathcal{T}\ \wedge\\ & 1\leq i\leq n+1\ \wedge\ s_i\neq s'_i\ \wedge\ w_i\text{ fresh state identifier name}\end{array}\right\}$$

The $\texttt{uniqueOrigin}(i,o)$ hypothesis allows to assume that the departing state coincides for all the IUT transitions labelled by the pair i/o. We identify these transitions in the IUT and we match all the departing states by assigning them the same state identifier name.

$$(origin)\frac{\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\ \texttt{uniqueOrigin}(i,o)\in\texttt{Hyp}}{\texttt{modelsSubset}\,(\{(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E}\cup\mathcal{E}',\mathcal{D},\texttt{Obs})\})}$$

where in order to define $\mathcal{E}'$, let us consider a fresh state identifier w and let $Y=\{s\ \texttt{is}\ w\mid\exists\ s':s\xrightarrow{i/o}s'\in\mathcal{T}\}$. Then, $\mathcal{E}'=\emptyset$ if $|Y|<2$ and $\mathcal{E}'=Y$ otherwise.

Similarly, $\texttt{uniqueDestination}(i, o)$ assumes that all IUT transitions labelled by i/o lead to the same IUT state.

$$(\textit{destination})\frac{\texttt{model}\,(\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs}) \;\wedge\; \texttt{uniqueOrigin}(i, o) \in \texttt{Hyp}}{\texttt{modelsSubset}\,(\{(\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E} \cup \mathcal{E}', \mathcal{D}, \texttt{Obs})\})}$$

where in order to define $\mathcal{E}'$, let us consider a fresh state identifier w and let $Y = \{s \;\texttt{is}\; w \mid \exists\; s' : s' \xrightarrow{i/o} s \in \mathcal{T}\}$. Then, $\mathcal{E}' = \emptyset$ if $|Y| < 2$ and $\mathcal{E}' = Y$ otherwise.

6 Soundness and Completeness of $\mathcal{HOTL}$

In this section we study the validity of the method, that is, we check the *soundness* and *completeness* of $\mathcal{HOTL}$. We have to relate the deductions we make by using our logic with the notion of conformance introduced in Definition 2. Let us note that dealing with this issue requires to provide a formal meaning to each concept in the logic, that is, we must provide all these concepts with some semantics *outside* the logic. The formal meaning of a predicate will be defined in terms of the set of FSMs that fulfill its requirements. This set will be denoted by its characteristic function, that is, given a predicate of the logic p, we consider that the *characterization* of p, denoted by $\alpha(p)$, is a function $\alpha(p) : \texttt{FsmSet} \longrightarrow \texttt{Bool}$ such that when applied to an FSM returns `true` if the FSM is included in those represented by the predicate p. In this section the semantics of all predicates is formally defined by means of the function α. Let us consider that P is the conjunction of all the considered observation and hypothesis predicates. Then, the function $\alpha(P)$ characterizes all the FSMs that can produce these observations and fulfill these hypotheses, that is, it denotes all the FSMs that, according to our knowledge, can *define* the IUT. So, if our logic deduces that all of these FSMs conform to the specification (i.e., `allModelsCorrect` is obtained) then the IUT actually conforms to the specification.

First, we define the *characterization* of predicates of the form $\texttt{model}\,(m)$. We will find out whether a given FSM fits into the model requirements by considering two requirements. On the one hand, we will check whether the FSM follows the *structure* of states and transitions of the model. That is, the FSM will be required to have a core of states and transitions derived from those of the model. We will not consider the full set of states $\mathcal{S}$ of the model but only those states that remain after the equalities given in the set $\mathcal{E}$ are fully exploited. A *mapping function* will allow to map each state of the model into its corresponding FSM state. Let us note that the model imposes some fusions, but FSMs where more states are fused could fit into the model as well. Hence, the

mapping function will be allowed to collapse other model states into coincident FSM states. In addition, we consider whether other states and transitions of the FSM not derived from the model also preserve the conditions imposed by the model. In order to allow sets of states and transitions in the FSM to include more elements than those imposed in the first requirement, in the next definition we will denote these sets with $\subseteq$ relations instead of with $=$. As we did before, we will implicitly refer to the components of the corresponding specification by $spec = (\mathcal{S}_{spec}, \texttt{inputs}_{spec}, \texttt{outputs}_{spec}, \mathcal{I}_{spec}, \mathcal{T}_{spec})$.

Definition 8 Let $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \mathcal{O})$ be a model. Let $\uparrow \subseteq \mathcal{S} \times \mathcal{S}$ be a binary relation defined as $s_1 \uparrow s_2$ iff there exists q such that $\{s_1 \texttt{ is } q, s_2 \texttt{ is } q\} \subseteq \mathcal{E}$. Let $\uparrow^*$ be the reflexive, symmetric, and transitive closure of $\uparrow$. Let $U = \mathcal{S}/[\uparrow^*]$ and $f : \mathcal{S} \longrightarrow U$ be a function such that for all $s_1, s_2 \in \mathcal{S}$ we have $s_1 \uparrow^* s_2$ implies $f(s_1) = f(s_2)$. Under these conditions we say that f is a *model mapping function* for m. Let $F = (\mathcal{S}', \texttt{inputs}', \texttt{outputs}', \mathcal{I}', \mathcal{T}')$ be an FSM, $\mathcal{S}'' \subseteq \mathcal{S}'$ be a set of states, and $f' : U \longrightarrow \mathcal{S}''$ be a function. Under these conditions we say that $g = f \circ f'$ is a *model-to-FSM mapping function* for m and F.

We say that $\alpha(\texttt{model}\ (m)) : \texttt{FsmSet} \longrightarrow \texttt{Bool}$ is the *characterization* of $\texttt{model}\ (m)$ if for each FSM $F = (\mathcal{S}', \texttt{inputs}', \texttt{outputs}', \mathcal{I}', \mathcal{T}')$ we have $\alpha(\texttt{model}\ (m))(F) = \texttt{true}$ iff $\texttt{inputs}' = \texttt{inputs}_{spec}$, $\texttt{outputs}' = \texttt{outputs}_{spec}$, and there exists a model-to-FSM mapping function g for m and F such that the following conditions hold:

- *(initial states)* Let $Q = \{u | \exists\ s : s \in \mathcal{I}\ \wedge\ g(s) = u\}$. If $\{\alpha, \beta\} \cap \mathcal{I} = \emptyset$ then $\mathcal{I}' = Q$; if $\alpha \in \mathcal{I}$ then $Q \subseteq \mathcal{I}' \subseteq \{u | \exists\ s : s \in \mathcal{S}\ \wedge\ g(s) = u\}$; otherwise, that is $\beta \in \mathcal{I}$, $Q \subseteq \mathcal{I}'$.
- *(transitions)* $\{u \xrightarrow{i/o} u' \mid \exists\ s, s' : s \xrightarrow{i/o} s' \in \mathcal{T}\ \wedge\ g(s) = u\ \wedge\ g(s') = u'\} \subseteq \mathcal{T}'$.
- *(deterministic states)* For all $s \in \mathcal{S} \cap \mathcal{D}$, we have that $g(s)$ is *deterministic*. That is, for all $i \in \texttt{inputs}'$ there do not exist in the set $\mathcal{T}'$ two transitions $g(s) \xrightarrow{i/o_1} u_1$ and $g(s) \xrightarrow{i/o_2} u_2$ such that $u_1 \neq u_2$ or $o_1 \neq o_2$.
- *(closed states)* For all $u \in \mathcal{S}'$ we have that if there exist s, i, $outs$, and f such that $g(s) = u$ and $(s, i, outs, f, \top) \in \mathcal{A}$ then the behavior of u is *closed* for i. That is, for all o, u' with $u \xrightarrow{i/o} u' \in \mathcal{T}'$ there exists s' such that $g(s') = u'$ and $s \xrightarrow{i/o} s' \in \mathcal{T}$.

□

For the sake of clarity, and without loss of generality, from now on we will assume that for all F such that $\alpha(\texttt{model}\ (m))(F)$ holds, the states of F that are derived from the states of m are named as in m. More formally, in the previous definition we assume that the range of g is included in its domain.

We consider that the characterization of a *tagged* model such as *consistent*(m)

or *correct*(m) coincides with the characterization of m. In particular, we will assume $\alpha(\texttt{model}\,(\mathit{consistent}(m))) = \alpha(\texttt{model}\,(\mathit{correct}(m))) = \alpha(\texttt{model}\,(m))$. Let us note that a tag denotes that the FSMs described by a model actually fulfill some property. That is, a tag does not indicate that among all FSMs denoted by a model we must take only those ones that fulfill the property. So, tags are unnecessary in characterization terms. In fact, their usefulness consists in controlling the correct application of rules to reach (or not) a allModelsCorrect diagnostic.

Next we define the characterization of *observation* predicates, that is, predicates following the form $ob = (a_1, i_1/o_1, a_2, \ldots, a_n, i_n/o_n, a_{n+1}) \in \texttt{Obs}$. Given such observation, we could construct from scratch the function that characterizes the set of FSMs that can produce $o_1, \ldots, o_n$ in response to $i_1, \ldots, i_n$ and fulfill the assumptions implicitly introduced by $a_1, \ldots, a_{n+1}$. However, the definition of the characterization of this predicate will be easier if we take advantage of the previous definition. Let us note that, as a result, the correctness of the *obser* deduction rule is trivially achieved.

Definition 9 Let $ob = (a_1, i_1/o_1, a_2, \ldots, a_n, i_n/o_n, a_{n+1}) \in \texttt{Obs}$, p be a predicate, and $\texttt{model}\,(m)$ be the predicate that results after applying the *obser* rule to p. The *characterization* of p, denoted by $\alpha(p)$, is given by the expression $\alpha(p) = \alpha(\texttt{model}\,(m))$. □

In order to define the characterization of hypothesis predicates, we consider only those FSMs that fulfill the requirement given by the hypothesis. Next we define these characterizations. The case of $\texttt{allTranHappenWith}(n) \in \texttt{Hyp}$ deserves additional explanations. It requires that for all FSM that can perform all input/output sequences denoted by Obs through some of its states, if an input i is offered n times at a state s along the execution of Obs then the n transitions produced as response are *all* the transitions of the FSM allowing to leave s when i is offered. Let us note that the FSM could execute all the sequences of Obs through different paths of states if it were non-deterministic. An FSM will be characterized by $\texttt{allTranHappenWith}(n)$ if the previous condition holds for at least some paths allowing to execute Obs. [3]

Definition 10 Let p be a predicate of the form $h \in \texttt{Hyp}$ and let us consider the set $\texttt{Obs} = \{(a_{j1}, i_{j1}/o_{j1}, a_{j2}, \ldots, a_{jr_j}, i_{jr_j}/o_{jr_j}, a_{j\,r_j+1}) | 1 \leq j \leq l\}$ of observations. The *characterization* of p is a function $\alpha(p) : \texttt{FsmSet} \longrightarrow \texttt{Bool}$ such that for all FSM $F = (\mathcal{S}', \texttt{inputs}', \texttt{outputs}', \mathcal{I}', \mathcal{T}')$ we have that $\alpha(p)(F)$ holds iff $\texttt{inputs}' = \texttt{inputs}_{spec}$, $\texttt{outputs}' = \texttt{outputs}_{spec}$, and

[3] The semantics of this hypothesis depends on the set of actual observations Obs because if the hypothesis were assumed for *all* set of observations then no non-deterministic FSM would fulfill it: No non-deterministic FSM shows all transitions from a state for *all* set of observations where this state is reached n times.

- If p is the predicate $\texttt{allDet} \in \texttt{Hyp}$ then there do not exist two transitions $u \xrightarrow{i/o_1} u_1$, $u \xrightarrow{i/o_2} u_2 \in \mathcal{T}'$ such that $u_1 \neq u_2$ or $o_1 \neq o_2$.
- If p is $\texttt{singleInit} \in \texttt{Hyp}$ then $|\mathcal{I}'| = 1$.
- If p is $\texttt{upperBoundOfStates}(n) \in \texttt{Hyp}$ then $|\mathcal{S}'| \leq n$.
- If p is $\texttt{allTranHappenWith}(n) \in \texttt{Hyp}$ then let us firstly suppose that F can perform all input/output sequences in $\texttt{Obs}$, that is, for all $1 \leq j \leq l$ there exists a trace $s_{j1} \xrightarrow{i_{j1}/o_{j1}} s_{j2} \ldots s_{jr_j} \xrightarrow{i_{jr_j}/o_{jr_j}} s_{j\,r_j+1}$, with $s_{j1}, \ldots, s_{jr_j+1} \in \mathcal{S}'$ and $s_{j1} \in \mathcal{I}'$. Let $V = \{s_{jk} | 1 \leq j \leq l \ \wedge \ 1 \leq k \leq r_j\}$ denote these states, $T(j,k) = s_{jk} \xrightarrow{i_{jk}/o_{jk}} s_{j(k+1)}$ denote the transition observed from s_{jk}, and $I(j,k) = i_{jk}$. Let $(j_1,k_1), \ldots, (j_n,k_n)$ be n different pairs of indexes denoting the *same* state (that is, $s_{j_1k_1}, \ldots, s_{j_nk_n} \in V$ and $s_{j_1k_1} = \ldots = s_{j_nk_n}$) such that for some $i \in \texttt{inputs}'$ we have $I(j_q,k_q) = i$ for all $1 \leq q \leq n$. Then, $\{t | o \in \texttt{outputs}' \wedge s' \in \mathcal{S}' \wedge t = s_{j_1k_1} \xrightarrow{i/o} s' \in \mathcal{T}'\} = \{T(j_q,k_q) | 1 \leq q \leq n\}$, that is, the transitions of F departing from the state $s_{j_1k_1} = \ldots = s_{j_nk_n}$ labelled with the input i are those belonging to $\{T(j_q,k_q) | 1 \leq q \leq n\}$. If F cannot perform all the sequences belonging to $\texttt{Obs}$ then no condition is required.
- If p is $\texttt{longSequencesSamePath}(n) \in \texttt{Hyp}$ then there do not exist in F two traces $s_1 \xrightarrow{i_1/o_1} s_2 \ldots s_n \xrightarrow{i_n/o_n} s_{n+1}$ and $s'_1 \xrightarrow{i_1/o_1} s'_2 \ldots s'_n \xrightarrow{i_n/o_n} s'_{n+1}$ such that $s_i \neq s'_i$ for some $1 \leq i \leq n+1$.
- If p is $\texttt{uniqueOrigin}(i,o) \in \texttt{Hyp}$ then there do not exist two transitions $s_1 \xrightarrow{i/o} s_2, s'_1 \xrightarrow{i/o} s'_2 \in \mathcal{T}'$ with $s_1 \neq s'_1$.
- If p is $\texttt{uniqueDestination}(i,o) \in \texttt{Hyp}$ then there do not exist two transitions $s_1 \xrightarrow{i/o} s_2, s'_1 \xrightarrow{i/o} s'_2 \in \mathcal{T}'$ with $s_2 \neq s'_2$.

□

The characterization of a predicate $p_1 \ \wedge \ p_2$ is easily defined as expected: $\alpha(p_1 \ \wedge \ p_2)(F)$ holds iff both $\alpha(p_1)(F)$ and $\alpha(p_2)(F)$. Let us remark that when a new hypothesis is added to a model we make the *conjunction* of the model predicate and the hypothesis predicate. So, the new set of represented $\texttt{FSMs}$ is a subset of the former set. Therefore, if the resulting model is still consistent then adding hypotheses preserves *conformance*: If all the $\texttt{FSMs}$ are conforming before the hypothesis is added then the $\texttt{FSMs}$ of the new model, which are a subset of them, must be so. For the same reason, *non-conformance* is not preserved when hypotheses are added. Actually, the objective of adding new hypotheses is to facilitate to ensure conformance. However, it is worth to point out that consistency may be lost by adding a new hypothesis.

The characterization of a $\texttt{models}\,(\mathcal{M})$ predicate is also simple: We have that $\alpha(\texttt{models}\,(\mathcal{M}))(F)$ holds iff there exists $m \in \mathcal{M}$ such that $\alpha(\texttt{model}\,(m))(F)$ holds. The characterization of $\texttt{modelsSubset}\,(\mathcal{M})$ is defined in the same way. The $\texttt{allModelsCorrect}$ predicate, which is the goal of the logic, will be con-

sidered apart from the rest of predicates. Instead of considering its characterization, we will explicitly prove that reaching this predicate implies that all FSMs fitting into the observations and hypotheses conform to the specification.

Next we consider the *correctness* of our deduction rules. The correctness of a rule is assessed in terms of the sets of FSMs that fulfill the premise and the conclusion. In order to denote these sets, we introduce the following notation: The *evaluation* of a predicate p, denoted by $\nu(p)$, is the set of FSMs fulfilling p, that is $\{F|F \in \texttt{FsmSet} \ \wedge \ \alpha(p)(F)\}$. If the evaluations of the premise and the conclusion of a rule contain the same sets of FSMs then we trivially obtain the property that all the FSMs that fit into the requirements of the premise conform to a specification *iff* all the FSMs that fit into the conclusion do so. So, (non)conformance is preserved by the rule. However, let us note that some rules return a *superset* of the FSMs that fulfill the premises, instead of the same set. There are two different reasons for that:

(a) Let $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs})$ be a model. The model m represents those FSMs whose set of states includes $\mathcal{S}$ and whose set of transitions includes $\mathcal{T}$. Adding an hypothesis to m can affect states belonging to $\mathcal{S}$ and transitions belonging to $\mathcal{T}$. States and transitions which are not *explicitly* represented in the model expression m but may extend $\mathcal{S}$ and $\mathcal{T}$, are not affected (and they should). For instance, by applying *allDet* we label all states belonging to $\mathcal{S}$ as deterministic, but no condition is explicitly added to other states. Fortunately, in order to check the (non)conformance of all FSMs that are deterministic in *all* states and include $\mathcal{S}$ and $\mathcal{T}$ we will be able to focus only on the behavior of $\mathcal{S}$ and $\mathcal{T}$. Moreover, as we will see, the same idea applies to the rest of hypotheses. We introduce the following notation. Given a model $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \mathcal{O})$, we consider that $\texttt{st}(m)$ denotes its set of states, that is $\texttt{st}(m) = \mathcal{S}$. Then, the set of FSMs represented by m whose set of states is a subset of $\texttt{st}(m)$, denoted by $\langle m \rangle$ and called *constrained evaluation* of m, is defined as the set of FSMs

$$\{F|F = (\mathcal{S}', \texttt{inputs}', \texttt{outputs}', \mathcal{I}', \mathcal{T}') \in \nu(\texttt{model}\,(m)) \ \wedge \ \mathcal{S}' \subseteq \texttt{st}(m)\}$$

Given a set of models $\mathcal{M}$, we denote by $\langle \mathcal{M} \rangle$ the set $\{\langle m \rangle | m \in \mathcal{M}\}$. We apply the previous notion to sets of FSMs as follows: Given a set of FSMs $\mathcal{F}$, the subset of FSMs whose set of states is a subset of $\mathcal{S}$ is denoted by $\langle \mathcal{F}, \mathcal{S} \rangle$. We call it the *constraint of $\mathcal{F}$ to $\mathcal{S}$*. As we will see, in most of our rules the constrained evaluations associated with the premises and the conclusion of the rule will coincide.

(b) However, in some situations the previous condition does not hold. In particular, adding some hypotheses to the model may require to apply the same rule *several* times. For instance, let us suppose that we apply the *long* rule to include the $\texttt{longSequencesSamePath}(n)$ hypothesis. After applying it, we discover that two states coincide. After matching them, a new path of

n states appears. So, *long* can be applied again. Intuitively, the *consistent* rule is responsible of forcing a *complete* application of hypotheses: Since the *consistent* rule is enabled only when no other rule can modify the model (and applying it is required to reach `allModelsCorrect` afterwards), we will have that, at the end, hypotheses will be properly considered.

Let us consider the correctness of the deduction rules. By the definition of the α function for observation predicates, given in Definition 9, we trivially deduce that the set of `FSMs` fulfilling the premise and the conclusion of the *obser* rule coincide. Next we consider some properties concerning the rest of rules. For each property, we show in italics the rule this property is concerned with.

Lemma 1 The following properties hold:

- *(fusion)* Let m_1, m_2, and m_3 be models where each component of m_3 is the union of the corresponding components of m_1 and m_2. We have

$$\nu(\texttt{model}\,(m_1)\ \wedge\ \texttt{model}\,(m_2)) = \nu(\texttt{model}\,(m_3))$$

- *(set)* Let m be a model. We have

$$\nu(\texttt{model}\,(m)) = \nu(\texttt{models}\,(\{m\}))$$

- *(propagation)* If $\langle m\rangle \subseteq \langle \mathcal{M}'\rangle$ then for all $\mathcal{M}$ we have

$$\langle \mathcal{M}\cup\{m\}\rangle \subseteq \langle \mathcal{M}\cup\mathcal{M}'\rangle$$

  Besides, if $\nu(\texttt{model}\,(m)) \subseteq \nu(\texttt{models}\,(\mathcal{M}'))$ then for all $\mathcal{M}$ we also have

$$\nu(\texttt{models}\,(\mathcal{M}\cup\{m\})) \subseteq \nu(\texttt{models}\,(\mathcal{M}\cup\mathcal{M}'))$$

- *(equality)* Let $m = (\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})$ and $s_1, s_2 \in \mathcal{S}$. If there exists q such that $\{s_1\ \texttt{is}\ q, s_2\ \texttt{is}\ q\} \subseteq \mathcal{E}$ then we have

$$\nu(\texttt{model}\,(m)) = \nu(\texttt{modelsSubset}\,(\texttt{modelElim}(m, s_1, s_2)))$$

- *(determ)* Let $m = (\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})$. Let $s, s_1, s_2 \in \mathcal{S}$ and $s \in \mathcal{D}$ be such that $\{s \xrightarrow{i/o_1} s_1, s \xrightarrow{i/o_2} s_2\} \subseteq \mathcal{T}$. Let $\mathcal{M}'$ be equal to $\texttt{modelElim}(m, s_1, s_2)$ if $o_1 = o_2$ and equal to $\emptyset$ otherwise. We have

$$\nu(\texttt{model}\,(m)) = \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$$

- *(singleInit)* Let $m = (\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})$, $\mathcal{I}' = \mathcal{I}\backslash\{\alpha,\beta\}$, and $m' = (\mathcal{S},\mathcal{T},\mathcal{I}',\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})$. Let us consider $\mathcal{I}\cap\mathcal{S} = \{s_1,\ldots,s_n\}$. We have

$$\nu(\texttt{model}\,(m)) = \nu(\texttt{modelsSubset}\,(\texttt{modelElim}(m', s_1, \{s_2,\ldots,s_n\})))$$

- *(allDet)* Let $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs})$ and $m' = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{S}, \texttt{Obs})$. We have both

$$\langle \nu(\texttt{model}\,(m) \ \wedge\ \texttt{allDet} \in \texttt{Hyp}), \mathcal{S} \rangle = \langle \{m'\} \rangle$$

and

$$\nu(\texttt{model}\,(m) \ \wedge\ \texttt{allDet} \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\{m'\}))$$

- *(allTran)* Let $n \in \mathbb{N}$, m be a deduced model, and $\mathcal{M}'$ be obtained from m as defined by *allTran*. We have

$$\nu(\texttt{model}\,(m) \ \wedge\ \texttt{allTranHappenWith}(n) \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$$

- *(upper)* Let m be a model and $n \in \mathbb{N}$. Let $\mathcal{M}'$ be obtained from m as defined in the *upper* rule. We have both

$$\langle \nu(\texttt{model}\,(m) \ \wedge\ \texttt{upperBoundOfStates}(n) \in \texttt{Hyp}), \texttt{st}(m) \rangle = \langle \mathcal{M}' \rangle$$

and

$$\nu(\texttt{model}\,(m) \ \wedge\ \texttt{upperBoundOfStates}(n) \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$$

- *(long)* Let $n \in \mathbb{N}$, m be a model, and $\mathcal{M}'$ be obtained from m as defined by *long*. We have

$$\nu(\texttt{model}\,(m) \wedge \texttt{longSequencesSamePath}(n) \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$$

- *(origin)* Let m be a model, i be an input, o be an output, and $\mathcal{M}'$ be obtained from m as defined by *origin* applied to (i, o). We have both

$$\langle \nu(\texttt{model}\,(m) \ \wedge\ \texttt{uniqueOrigin}(i, o) \in \texttt{Hyp}), \texttt{st}(m) \rangle = \langle \mathcal{M}' \rangle$$

and

$$\nu(\texttt{model}\,(m) \ \wedge\ \texttt{uniqueOrigin}(i, o) \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$$

- *(destination)* Let m be a model, i be an input, o be an output, and $\mathcal{M}'$ be obtained from m as defined by *destination* applied to (i, o). We have both

$$\langle \nu(\texttt{model}\,(m) \ \wedge\ \texttt{uniqueDestination}(i, o) \in \texttt{Hyp}), \texttt{st}(m) \rangle = \langle \mathcal{M}' \rangle$$

and

$$\nu(\texttt{model}\,(m) \ \wedge\ \texttt{uniqueDestination}(i, o) \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$$

- *(consistent)* Let m be a model such that $\texttt{unable}(m, \mathcal{R} \backslash \{\mathit{consistent}, \mathit{correct}\})$. We have

$$\nu(\texttt{model}\,(m)) = \nu(\texttt{modelsSubset}\,(\{\mathit{consistent}(m)\}))$$

- *(correct)* Let m be a model such that $\texttt{worstCase}(m)$ $\texttt{conf}$ *spec*. We have

$$\nu(\texttt{model}\,(\mathit{consistent}(m))) = \nu(\texttt{modelsSubset}\,(\{\mathit{correct}(m)\}))$$

Proof: We prove the results described in *(allDet)*, *(long)*, and *(allTran)*. The rest of results use very similar arguments or are straightforward.

- *(allDet)* Let $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs})$ and $m' = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{S}, \texttt{Obs})$. First, let us prove $\langle \nu(\texttt{model}\,(m) \wedge \texttt{allDet} \in \texttt{Hyp}), \mathcal{S}\rangle = \langle\{m'\}\rangle$. Let us note that, due to the definition of the ν function from α, the set $\nu(\texttt{model}\,(m) \wedge \texttt{allDet} \in \texttt{Hyp})$ is equal to $\nu(\texttt{model}\,(m)) \cap \nu(\texttt{allDet} \in \texttt{Hyp})$. For all FSM $F = (\mathcal{S}', \texttt{inputs}', \texttt{outputs}', \mathcal{I}', \mathcal{T}')$ belonging to this set we have that $F \in \nu(\texttt{model}\,(m))$ and, according to Definition 10, there do not exist two transitions $u \xrightarrow{i/o_1} u_1$, $u \xrightarrow{i/o_2} u_2 \in \mathcal{T}'$ such that $u_1 \neq u_2$ or $o_1 \neq o_2$. Consequently, the set $\langle \nu(\texttt{model}\,(m) \wedge \texttt{allDet} \in \texttt{Hyp}), \mathcal{S}\rangle$ denotes all deterministic FSMs belonging to $\nu(\texttt{model}\,(m))$ whose set of states is a subset of $\mathcal{S}$. In addition, let us consider the elements of $\langle\{m'\}\rangle = \langle m'\rangle$. We have that $\nu(\texttt{model}\,(m'))$ is the set of FSMs belonging to $\nu(\texttt{model}\,(m))$ such that all states belonging to $\mathcal{S}$ are deterministic (but the rest of states might not be so). Thus, $\langle m'\rangle$ denotes all FSMs belonging to $\nu(\texttt{model}\,(m))$ such that, on the one hand, all states belonging to $\mathcal{S}$ are deterministic and, on the other hand, the set of states is a subset of $\mathcal{S}$. That is, $\langle m'\rangle$ denotes all deterministic FSMs belonging to $\nu(\texttt{model}\,(m))$ whose set of states is a subset of $\mathcal{S}$. Thus, we deduce that $\langle \nu(\texttt{model}\,(m) \wedge \texttt{allDet} \in \texttt{Hyp}), \mathcal{S}\rangle = \langle\{m'\}\rangle$.

  Now we prove $\nu(\texttt{model}\,(m) \wedge \texttt{allDet} \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\{m'\}))$. Let us consider $F \in \nu(\texttt{model}\,(m)) \cap \nu(\texttt{allDet} \in \texttt{Hyp})$. That is, F is a deterministic FSM belonging to $\nu(\texttt{model}\,(m))$. Let us note that we have $\nu(\texttt{modelsSubset}\,(\{m'\})) = \nu(\texttt{models}\,(\{m'\})) = \nu(\texttt{model}\,(m'))$. The term $\nu(\texttt{model}\,(m'))$ denotes all FSMs belonging to $\nu(\texttt{model}\,(m))$ such that all states included in $\mathcal{S}$ are deterministic (but the rest might not). Clearly, we have $F \in \nu(\texttt{model}\,(m'))$. Thus, $\nu(\texttt{model}\,(m) \wedge \texttt{allDet} \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\{m'\}))$.
- *(long)* Let $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs})$ be a model, $n \in \mathbb{N}$, and $\mathcal{M}'$ be obtained from m as defined by the rule *long*. We have to prove

  $$\nu(\texttt{model}\,(m) \wedge \texttt{longSequencesSamePath}(n) \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$$

  Let $F = (\mathcal{S}', \texttt{inputs}', \texttt{outputs}', \mathcal{I}', \mathcal{T}')$ be an FSM such that we have $F \in \nu(\texttt{model}\,(m) \wedge \texttt{longSequencesSamePath}(n) \in \texttt{Hyp})$. Thus, we also have $F \in \nu(\texttt{model}\,(m))$ and, according to Definition 10, there do not exist in F two traces $s_1 \xrightarrow{i_1/o_1} s_2 \ldots s_n \xrightarrow{i_n/o_n} s_{n+1}$ and $s'_1 \xrightarrow{i_1/o_1} s'_2 \ldots s'_n \xrightarrow{i_n/o_n} s'_{n+1}$ such that $s_i \neq s'_i$ for some $1 \leq i \leq n+1$. According to the *long* rule, $\mathcal{M}'$ consists of a single model $m' = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E} \cup \mathcal{E}', \mathcal{D}, \texttt{Obs})$ where $\mathcal{E}'$ denotes a set of equalities. These equalities are added to match states in equal positions along the same sequence of n inputs/outputs. We have

$\nu(\texttt{modelsSubset}\,(\mathcal{M}')) = \nu(\texttt{modelsSubset}\,(\{m'\})) = \nu(\texttt{model}\,(m'))$. Let us note that, according to Definition 8, the FSMs contained in $\nu(\texttt{model}\,(m'))$ are actually constrained by the fusions induced by the set of equalities of m', $\mathcal{E} \cup \mathcal{E}'$ (see how the *model-to-FSM function* is constructed from the set of equalities in Definition 8). Comparing this set with $\texttt{model}\,(m)$, the effect of these fusions is that some FSMs in $\texttt{model}\,(m)$ where there exist $s_1 \xrightarrow{i_1/o_1} s_2 \ldots s_n \xrightarrow{i_n/o_n} s_{n+1}$ and $s'_1 \xrightarrow{i_1/o_1} s'_2 \ldots s'_n \xrightarrow{i_n/o_n} s'_{n+1}$ such that $s_i \neq s'_i$ for some $1 \leq i \leq n+1$ are missing in $\texttt{model}\,(m')$. In fact, $\nu(\texttt{model}\,(m'))$ consists of all FSMs in $\nu(\texttt{model}\,(m))$ but them. Since F fulfills $\texttt{longSequencesSamePath}(n)$, F is not one of the FSMs that is missing in $\texttt{model}\,(m')$. Hence, we have $F \in \nu(\texttt{model}\,(m'))$ and we finally infer that $\nu(\texttt{model}\,(m) \ \wedge \ \texttt{longSequencesSamePath}(n) \in \texttt{Hyp})$ is indeed a subset of $\nu(\texttt{modelsSubset}\,(\mathcal{M}'))$.

- *(allTran)* Let $n \in \mathbb{N}$, $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \texttt{Obs})$ be the model, and $\mathcal{M}'$ be obtained from m as defined by the application of *allTran*. We will prove that $\nu(\texttt{model}\,(m) \ \wedge \ \texttt{allTranHappenWith}(n) \in \texttt{Hyp}) \subseteq \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$. Let us suppose that $F \in \nu(\texttt{model}\,(m) \ \wedge \ \texttt{allTranHappenWith}(n) \in \texttt{Hyp})$. Then, $F \in \nu(\texttt{model}\,(m))$ and, according to Definition 10, if F can perform all sequences in Obs then, for all set $T'_{s,i}$ of transitions traversing the sequences of Obs where the input i is offered n times from the state s, we have that all transitions in F leaving s with i are included in $T'_{s,i}$. Let us note that the application of rules preserves the capability of models to perform all sequences of Obs. Moreover, the number of occurrences of each transition through Obs is provided by $\mathcal{A}$ (specifically, by f for each $(s, i, outs, f, n) \in \mathcal{A}$). In order to check both properties, it is enough to see how transitions are preserved when a state is eliminated by the modelElim function, given in Definition 7, and how it updates $\mathcal{A}$, given in Definition 6. Since the set of observations of m is Obs, m performs all sequences of Obs and the set $\mathcal{A}$ of m keeps the number of occurrences of each transition through Obs. After the *allTran* rule is applied, a set of models $\mathcal{M}'$ is provided. Since $F \in \nu(\texttt{allTranHappenWith}(n))$, there do not exist $t_{s,i} \in \mathcal{T}$ leaving a state s with an input i and a set $T'_{s,i}$ of transitions where i is offered n or more times from s through the sequences of Obs such that $t_{s,i} \notin T'_{s,i}$. Thus, the case (b) of the *allTran* rule is taken and we have $\mathcal{M}' \neq \emptyset$. A new model $m' \in \mathcal{M}'$ is provided for each way to add new equalities $\mathcal{E}'$ in such a way that, for all state s, input i, and set of transitions $T''_{s,i}$ from s with i such that the cumulated number of occurrences of each transition is at least n, all transitions from s with i are included in $T''_{s,i}$. Hence, there must exist $m' \in \mathcal{M}'$ such that $F \in \nu(\texttt{model}\,(m'))$ and we have $F \in \nu(\texttt{modelsSubset}\,(\mathcal{M}'))$.

□

Thus, the set of FSMs that fulfill the premises in the *obser*, *fusion*, *set*, *equality*, *determ*, *singleInit*, *consistent*, and *correct* deduction rules coincides with the set of FSMs that fulfill the conclusion. In the *allDet*, *upper*, *origin*, and *destina-*

tion rules, the constrained evaluations of the premises and conclusion coincide, and the evaluation of the conclusion is, in general, a superset of the constrained evaluation of the premise. The latter property also holds in the *allTran* and *long* rules. The correctness of the *allCorrect* rule, not considered in the previous result, will be analyzed in terms of the meaning of the `allModelsCorrect` diagnostic rather than on the basis of the set of represented `FSMs`. Besides, let us note that the relevant meaning of the *consistent* and *correct* rules does not concern which `FSMs` are represented but whether they are all consistent or correct, respectively. Later we will address these issues. Regarding the *propagation* rule, it always preserves the only two properties preserved by all targeted rules, that is, the inclusion of evaluations and the inclusion of constrained evaluations. Let us note that, as we will see in the next result, the *inclusion of evaluations* implies the *inclusion of constrained evaluations*. Moreover, the *equality of evaluations* implies the *equality of constrained evaluations*. This implies, in turn, the *inclusion of constrained evaluations*. In the following result, let us remind that $\mathcal{R}$ is the set of rules where the *propagation* rule can be used (see Definition 3).

Lemma 2 Let us consider $\texttt{model}\,(m) \ \wedge \ \varphi \vdash_r \texttt{modelsSubset}\,(\mathcal{M})$, with $r \in \mathcal{R}$. We have the following results:

(a) If $\nu(\texttt{model}\,(m) \ \wedge \ \varphi) = \nu(\texttt{models}\,(\mathcal{M}))$ then $\langle \nu(\texttt{model}\,(m) \ \wedge \ \varphi), \texttt{st}(m)\rangle = \langle \mathcal{M}\rangle$.
(b) If $\langle \nu(\texttt{model}\,(m) \wedge \varphi), \texttt{st}(m)\rangle = \langle \mathcal{M}\rangle$ then $\langle \nu(\texttt{model}\,(m) \wedge \varphi), \texttt{st}(m)\rangle \subseteq \langle \mathcal{M}\rangle$.
(c) If $\nu(\texttt{model}\,(m) \wedge \varphi) \subseteq \nu(\texttt{models}\,(\mathcal{M}))$ then $\langle \nu(\texttt{model}\,(m) \wedge \varphi), \texttt{st}(m)\rangle \subseteq \langle \mathcal{M}\rangle$.
(d) If $\nu(\texttt{model}\,(m) \wedge \varphi) = \nu(\texttt{models}\,(\mathcal{M}))$ then $\nu(\texttt{model}\,(m) \wedge \varphi) \subseteq \nu(\texttt{models}\,(\mathcal{M}))$.

Proof: The properties (b) and (d) are straightforward. We consider the proof of (a). We have $F \in \nu(\texttt{model}\,(m) \ \wedge \ \varphi)$ iff $F \in \nu(\texttt{models}\,(\mathcal{M}))$. Thus, the `FSMs` belonging to $\nu(\texttt{model}\,(m) \wedge \varphi)$ whose set of states is included in $\texttt{st}(m)$ coincides with the ones belonging to $\nu(\texttt{models}\,(\mathcal{M}))$ fulfilling the same condition. Let us note that for all $m' \in \mathcal{M}$ we have $\texttt{st}(m') \subseteq \texttt{st}(m)$: In all rules in $\mathcal{R}$, each resulting model m' either keeps the states of the original model m or loses some of them. So, for all $m' \in \mathcal{M}$ we have that taking only states in $\texttt{st}(m)$ is equivalent to taking only states in $\texttt{st}(m')$. For similar reasons we have (c). □

Next we consider some additional properties of the rules. First we show that, for all rules, the constrained evaluation of the conclusion is a superset of the constrained evaluation of the premises. This property also holds if we consider evaluations. Then, we show that if a rule preserves the `FSMs` that fulfill an hypothesis h, then it also preserves the `FSMs` that fulfill h and any other hypothesis h'. Next we prove that adding some information to a model reduces, in general, the set of `FSMs` it represents. It is so because the model is *particularized*. Finally, we show that if a hypothesis currently holds in a model

then its corresponding rule returns a singleton with that model.

Lemma 3 Let us consider $\texttt{model}\,(m)\ \wedge\ \varphi \vdash_r \texttt{modelsSubset}\,(\mathcal{M}) \in \mathcal{R}$, with $r \in \mathcal{R}$. We have the following results:

(a) If $\varphi = h \in \texttt{Hyp}$ then $\langle \nu(\texttt{model}\,(m)\ \wedge\ h \in \texttt{Hyp}), \texttt{st}(m)\rangle \subseteq \langle \mathcal{M}\rangle$.
(b) If $\varphi = h \in \texttt{Hyp}$ then $\nu(\texttt{model}\,(m)\ \wedge\ h \in \texttt{Hyp}) \subseteq \nu(\texttt{models}\,(\mathcal{M}))$.
(c) Let $\varphi = h \in \texttt{Hyp}$. Then, $\langle \nu(\texttt{model}\,(m)\ \wedge\ h \in \texttt{Hyp}\ \wedge\ h' \in \texttt{Hyp}), \texttt{st}(m)\rangle \subseteq \bigcup_{m' \in \mathcal{M}} \langle \nu(\texttt{model}\,(m')\ \wedge\ h \in \texttt{Hyp}\ \wedge\ h' \in \texttt{Hyp}), \texttt{st}(m'))\rangle$, where h' is any hypothesis of $\mathcal{HOTL}$.
(d) We have $\langle m\rangle \supseteq \langle \mathcal{M}\rangle$.
(e) Let $\varphi = h \in \texttt{Hyp}$. If we have that $\texttt{model}\,(m) \wedge h \in \texttt{Hyp} \vdash_r \texttt{models}\,(\{m\})$ and $\texttt{unable}(m, \{\mathit{determ}, \mathit{equality}\})$ then $\langle (\texttt{model}\,(m)\ \wedge\ h \in \texttt{Hyp}), \texttt{st}(m)\rangle = \langle m\rangle$. Moreover, if we have both $\langle (\texttt{model}\,(m)\ \wedge\ h \in \texttt{Hyp}), \texttt{st}(m)\rangle = \langle m\rangle$ and $\texttt{unable}(m, \{\mathit{determ}, \mathit{equality}\})$ then $\texttt{model}\,(m) \wedge h \in \texttt{Hyp} \vdash_r \texttt{models}\,(\{m\})$.

Proof: The results (a) and (b) are a direct consequence of the properties presented in Lemma 1 for all rules in $\mathcal{R}$ and the implications given in Lemma 2. Besides, (c) is easy to prove by using (a): If constrained evaluations properly preserve all FSMs where h holds then they also preserve the FSMs where both h and h' hold. This is because $\nu(h \in \texttt{Hyp}\ \wedge\ h' \in \texttt{Hyp}) = \nu(h \in \texttt{Hyp}) \cap \nu(h' \in \texttt{Hyp}) \subseteq \nu(h \in \texttt{Hyp})$. Next, (d) is a consequence of the fact that constraining m with additional requirements reduces, in general, the set of FSMs that fit into the resulting model(s). All rules in $\mathcal{R}$ either mark some states as deterministic (i.e., $\mathcal{D}$ is enlarged) or match some pairs of states (i.e., $\mathcal{E}$ is enlarged). In the former case, the model loses all FSMs where states marked as deterministic are actually nondeterministic. In the latter case, new equalities match some pairs of states, say s_1 and s_2. Transitions reaching or leaving either s_1 or s_2 stay in the new model. Moreover, the fusion of s_1 and s_2 allows s_1 to behave as s_2 in the new model by means of transitions inherited from s_2, and viceversa (i.e. more transitions are imposed in the new model, being all FSMs that do *not* include them lost). If the new model allows FSMs to extend model transitions with these new transitions, then the former model allows it as well. That is, all FSMs where these transitions are added *also* belong to the former model. Finally, let us consider the first statement of (e). Let us suppose that $\langle (\texttt{model}\,(m)\ \wedge\ h \in \texttt{Hyp}), \texttt{st}(m)\rangle \neq \langle m\rangle$. By (a), we have $\langle (\texttt{model}\,(m)\ \wedge\ h \in \texttt{Hyp}), \texttt{st}(m)\rangle \subset \langle m\rangle$. In this situation, we have that either $\texttt{model}\,(m)\ \wedge\ h \in \texttt{Hyp} \vdash_r \texttt{models}\,(\{m\})$ holds or it does not. If it does not hold then the property that we are proving trivially holds. So let us consider that it holds. The *allDet*, *singleInit*, *allTran*, *upper*, *long*, *origin*, and *destination* rules apply a given hypothesis h by searching for states or groups of states that do not fulfill h and manipulating them. In the *allTran* and *long* rules, states might not completely exhibit their characteristics regarding each of these hypotheses only if pairs of states that can be matched have not been matched yet: In *allTran*, some accounting registers could reach a threshold

n only after more states are fused; in *long*, new paths of length n could be created after some states are fused. In these cases, either *determ* or *equality* can be applied. So, $\texttt{unable}(m, \{determ, equality\})$ does not hold. In the rest of rules, it is easy to see that the required manipulations are always visible in the model. Thus, in all these cases, if $\langle(\texttt{model}\,(m) \ \wedge \ h \in \texttt{Hyp}), \texttt{st}(m)\rangle \subset \langle m\rangle$ then $\texttt{model}\,(m) \ \wedge \ h \in \texttt{Hyp} \vdash_r \texttt{models}\,(\{m\})$ is false. The second statement also derives from the rules definition. □

Next we show that the *consistent* rule can be applied to a model if and only if its constrained evaluation already considers all assumed hypotheses and the *determ* and *equality* rules cannot be applied.

Lemma 4 Let P be a set of predicates and $P_{hyp} = \{h_i \in \texttt{Hyp} \,|\, 1 \leq i \leq n\}$, for some $n \in \mathbb{N}$, be the set of predicates of the form $h \in \texttt{Hyp}$ belonging to P. Let $\mathcal{M}$ be a set of models and m be a model such that $P \vdash^* \texttt{models}\,(\mathcal{M} \cup \{m\})$, Then, we have $\texttt{model}\,(m) \vdash_{consistent} \texttt{models}\,(consistent(m))$ iff we have both $\texttt{unable}(m, \{determ, equality\})$ and

$$\langle\nu(\texttt{model}\,(m) \ \wedge \ h_1 \in \texttt{Hyp} \ \wedge \ \ldots \ \wedge \ h_n \in \texttt{Hyp}), \texttt{st}(m)\rangle = \langle\{consistent(m)\}\rangle$$

Proof: We prove the implication from left to right by contrapositive. Hence, we consider that the right hand side statement does not hold. In this case, either $\texttt{unable}(m, \{determ, equality\})$ holds or it does not. If this condition holds then we have $\langle\nu(\texttt{model}\,(m) \ \wedge \ h_1 \in \texttt{Hyp} \ \wedge \ \ldots \ \wedge \ h_n \in \texttt{Hyp}), \texttt{st}(m)\rangle \neq \langle\{consistent(m)\}\rangle$. Since $\langle m\rangle = \langle consistent(m)\rangle = \langle\{consistent(m)\}\rangle$ and for all predicates p and q we have $\nu(p \ \wedge \ q) \subseteq \nu(p)$, we immediately deduce $\langle\nu(\texttt{model}\,(m) \ \wedge \ h_1 \in \texttt{Hyp} \ \wedge \ \ldots \ \wedge \ h_n \in \texttt{Hyp}), \texttt{st}(m)\rangle \subset \langle\{consistent(m)\}\rangle = \langle m\rangle$. This implies that there exists $h_i \in \texttt{Hyp}$ such that $\langle\nu(\texttt{model}\,(m) \ \wedge \ h_i \in \texttt{Hyp}), \texttt{st}(m)\rangle \subset \langle m\rangle$. That is, $\langle m\rangle$ does not fulfill h_i. Let r_i be the rule that allows to consider h_i. Let us note that rules concerning hypotheses can always be applied to any model, provided that the hypothesis belongs to $\texttt{Hyp}$. Since $\texttt{unable}(m, \{determ, equality\})$ holds, by Lemma 3 (e) we have that if r_i returns $\{m\}$ then m actually fulfills h_i. Since m does not do it, we deduce that r_i does not return $\{m\}$ and, since r_i can be applied, $\texttt{unable}(m, r_i)$ is false. So, the *consistent* rule cannot be applied. In addition, if $\texttt{unable}(m, \{determ, equality\})$ does not hold then the *consistent* rule cannot be applied.

Next we consider the implication from right to left. First, let us note that if $\texttt{model}\,(m) \vdash_{consistent} \texttt{models}\,(consistent(m))$ does not hold then we have that $\texttt{unable}(\mathcal{R}\backslash\{consistent, correct\})$ is false. Hence, there exists a rule r in the set $\mathcal{R}\backslash\{consistent, correct\}$ such that r can be applied to m and a set of models different to $\{m\}$ is returned. If $r \in \{determ, equality\}$ then we obtain the desired result because $\texttt{unable}(m, \{determ, equality\})$ is false. Otherwise, r involves the application of a hypothesis, say h. Since m is modified by the application of r, by Lemma 3 (e) either $\texttt{unable}(m, \{determ, equality\})$ is false (and we have again the previous case) or $\langle\nu(\texttt{model}\,(m) \ \wedge \ h \in \texttt{Hyp}), \texttt{st}(m)\rangle \neq \langle m\rangle$.

In the latter case, by Lemma 3 (a) we have $\langle \nu(\texttt{model}\,(m) \wedge h \in \texttt{Hyp}), \texttt{st}(m)\rangle \subset \langle m\rangle$. Hence, $\langle \nu(\texttt{model}\,(m) \wedge h_1 \in \texttt{Hyp} \wedge \ldots \wedge h_n \in \texttt{Hyp}), \texttt{st}(m)\rangle \subset \langle\{\mathit{consistent}(m)\}\rangle$. □

Next we show that the application of rules never returns *less* FSMs that those that fulfill the actual set of observations and hypotheses. We consider this result in two phases. First, we show that the model m reached after considering all observations and mixing the corresponding models exactly denotes the FSMs that can produce these observations. Second, we show that the constrained evaluation of any set of models $\mathcal{M}$ obtained after considering some hypotheses is a superset of the constrained evaluation of m. This result is also considered in the context of evaluations.

Lemma 5 Let $P = P_{obs} \cup P_{hyp}$ be a set of predicates where $P_{obs} \neq \emptyset$ is a set of observation predicates and P_{hyp} is a set of hypothesis predicates. Let us suppose we have the derivations $P \vdash^* \texttt{model}\,(m) \vdash_{set} \texttt{models}\,(\{m\})$ and $P \vdash^* \texttt{models}\,(\mathcal{M})$. We have:

(a) $\nu(\bigwedge_{p\in P_{obs}}) = \nu(\texttt{models}\,(\{m\}))$
(b) $\langle \nu(\texttt{models}\,(\{m\}) \wedge \bigwedge_{p\in P_{hyp}}), \texttt{st}(m)\rangle \subseteq \langle\mathcal{M}\rangle$
(c) $\nu(\texttt{models}\,(\{m\}) \wedge \bigwedge_{p\in P_{hyp}}) \subseteq \nu(\texttt{models}\,(\mathcal{M}))$

Proof: We consider (a). Since the *set* rule requires that the set of observations is Obs, we have that $\texttt{models}\,(\{m\})$ is the model obtained after applying the *obser*, *fusion*, and *set* rules until they cannot be applied anymore, and before any other rule is applied. Let us consider the behavior of the *obser* rule, commented before, and the properties of the *fusion* and *set* rules given in Lemma 1. The *obser* rule preserves the set of represented FSMs when each predicate $p \in P_{obs}$ is considered. The other rules also preserve the sets considered in the premises. Let us note that all observations belonging to P_{obs} must be considered to reach $\texttt{models}\,(\{m\})$. Thus, we have $\nu(\texttt{models}\,(\{m\})) = \nu(\bigwedge_{p\in P_{obs}})$ and (a) holds.

Let us consider (b). By (a), any predicate obtained by applying *obser*, *fusion*, and *set* until they cannot be applied anymore is such that its evaluation is equal to $\nu(\bigwedge_{p\in P_{obs}})$. In order to reach a $\texttt{models}\,(\mathcal{M})$ predicate, a predicate like that must be reached before (recall that other rules cannot be applied until $\mathcal{O} = \texttt{Obs}$). Let $p = \texttt{models}\,(\{m'\})$ be such predicate. The $\texttt{models}\,(\mathcal{M})$ predicate is obtained from p in some steps (say n) as follows. For some models $m_1, \ldots, m_n$, with $m_1 = m'$ and $m_n \in \mathcal{M}$, sets of models $\mathcal{M}_1, \ldots, \mathcal{M}_n$, rules $r_1, \ldots, r_n \in \mathcal{R}$, and conditions $\varphi_1, \ldots, \varphi_n$, we have $\texttt{model}\,(m_i) \wedge \varphi_i \vdash_{r_i} \texttt{models}\,(\mathcal{M}_i)$, for all $1 \leq i \leq n$, in such a way that $m_i \in \overrightarrow{\mathcal{M}_{i-1}}$, where $\overrightarrow{\mathcal{M}_{i-1}}$ denotes the set of models that are obtained after sequentially applying the rules $r_1, \ldots, r_{i-1}$ (and, after each application, the *propagation* rule). The combination of r_i and *propagation* has the effect of substituting the model where r_i is applied by a new set of models. Thus, $((\ldots(((\mathcal{M}_1 \backslash \{m_2\}) \cup \mathcal{M}_2) \backslash \{m_3\}) \ldots \backslash \{m_i\}) \cup \mathcal{M}_i) = \overrightarrow{\mathcal{M}_i}$. Then,

$\mathcal{M} = \overrightarrow{\mathcal{M}_n}$. On the one hand, if φ_i is a condition of the form $h_i \in \texttt{Hyp}$ then, by Lemma 3 (a), r_i provides a new constrained evaluation that is a superset of the constrained evaluation of the premises. On the other hand, if $\varphi_i = \texttt{true}$ then $r_i \in \{determ, equality\}$ and, by Lemma 1, we also conclude the same (let us recall that equality of evaluations implies inclusion of constrained evaluations). Hence, $\langle \nu(\texttt{model}\,(m_i) \ \wedge \ \varphi_i), \texttt{st}(m_i)\rangle \subseteq \langle \mathcal{M}_i\rangle$.

Next we prove the property $\mathcal{P} \equiv \langle \nu(\texttt{model}\,(m_1) \wedge \bigwedge_{1\leq j\leq i} \varphi_j), \texttt{st}(m_1)\rangle \subseteq \langle \overrightarrow{\mathcal{M}_i}\rangle$ by induction over i. In the anchor case we consider $i = 1$. We have $\overrightarrow{\mathcal{M}_1} = \mathcal{M}_1$, so $\mathcal{P}$ trivially holds. Let us assume that $\mathcal{P}$ holds for $i = k$, and let us prove it for $k+1$. By induction hypothesis, $\langle \nu(\texttt{model}\,(m_1) \wedge \bigwedge_{1\leq j\leq k} \varphi_j), \texttt{st}(m_1)\rangle \subseteq \langle \overrightarrow{\mathcal{M}_k}\rangle$. If all FSMs included in $\langle \nu(\texttt{model}\,(m_1) \ \wedge \ \bigwedge_{1\leq j\leq k} \varphi_j), \texttt{st}(m_1)\rangle$ are also included in $\langle \overrightarrow{\mathcal{M}_k}\rangle$, then the FSMs belonging to $\langle \nu(\texttt{model}\,(m_1) \ \wedge \ \bigwedge_{1\leq j\leq k} \varphi_j), \texttt{st}(m_1)\rangle$ that *also* fulfill the condition φ_{k+1} are included in $\langle \overrightarrow{\mathcal{M}_k}\rangle$ as well. So, we have $\langle \nu(\texttt{model}\,(m_1) \ \wedge \ \bigwedge_{1\leq j\leq k} \varphi_j), \texttt{st}(m_1)\rangle \cap \nu(\varphi_{k+1}) \subseteq \langle \overrightarrow{\mathcal{M}_k}\rangle \cap \nu(\varphi_{k+1})$. Let us note that $\langle \nu(\texttt{model}\,(m_{k+1}) \ \wedge \ \varphi_{k+1}), \texttt{st}(m_{k+1})\rangle \subseteq \langle \mathcal{M}_{k+1}\rangle$. Hence, if we consider the set of FSMs $\langle \overrightarrow{\mathcal{M}_k}\rangle \cap \nu(\varphi_{k+1})$, we remove all the FSMs belonging to the set $\langle \nu(\texttt{model}\,(m_{k+1}) \ \wedge \ \varphi_{k+1}), \texttt{st}(m_{k+1})\rangle$, and add the FSMs belonging to $\langle \mathcal{M}_{k+1}\rangle$ then no FSM is lost, that is, we have $\langle \overrightarrow{\mathcal{M}_k}\rangle \cap \nu(\varphi_{k+1}) \subseteq ((\langle \overrightarrow{\mathcal{M}_k}\rangle \cap \nu(\varphi_{k+1})) \setminus \langle \nu(\texttt{model}\,(m_{k+1}) \wedge \varphi_{k+1}), \texttt{st}(m_{k+1})\rangle) \cup \langle \mathcal{M}_{k+1}\rangle$. Let us note that $\langle \nu(\texttt{model}\,(m_{k+1}) \ \wedge \ \varphi_{k+1}), \texttt{st}(m_{k+1})\rangle = \langle m_{k+1}\rangle \cap \nu(\varphi_{k+1})$. So, the former expression is equivalent to $((\langle \overrightarrow{\mathcal{M}_k}\rangle \cap \nu(\varphi_{k+1})) \setminus (\langle m_{k+1}\rangle \cap \nu(\varphi_{k+1}))) \cup \langle \mathcal{M}_{k+1}\rangle$. By the distributive property, we have that this expression is equal to $((\langle \overrightarrow{\mathcal{M}_k}\rangle \setminus \langle m_{k+1}\rangle) \cap \nu(\varphi_{k+1})) \cup \langle \mathcal{M}_{k+1}\rangle$. All elements in this set are included in $(\langle \overrightarrow{\mathcal{M}_k}\rangle \setminus \langle m_{k+1}\rangle) \cup \langle \mathcal{M}_{k+1}\rangle$, which is equivalent to $\langle \overrightarrow{\mathcal{M}_{k+1}}\rangle$. Summarizing, we deduce the following relation: $\langle \nu(\texttt{model}\,(m_1) \ \wedge \ \bigwedge_{1\leq j\leq k+1} \varphi_j), \texttt{st}(m_1)\rangle = \langle \nu(\texttt{model}\,(m_1) \ \wedge \ \bigwedge_{1\leq j\leq k} \varphi_j), \texttt{st}(m_1)\rangle \cap \nu(\varphi_{k+1}) \subseteq \langle \overrightarrow{\mathcal{M}_{k+1}}\rangle$. Hence, $\mathcal{P}$ holds.

By $\mathcal{P}$, we deduce $\langle \nu(\texttt{model}\,(m_1) \ \wedge \ \bigwedge_{1\leq j\leq n} \varphi_j), \texttt{st}(m_1)\rangle \subseteq \langle \overrightarrow{\mathcal{M}_n}\rangle$. Let us remind that $m_1 = m'$ and $\overrightarrow{\mathcal{M}_n} = \mathcal{M}$. Let $P'_{hyp} = \{\varphi_1, \ldots, \varphi_n\}$. Predicates in P'_{hyp} either follow the pattern $h \in \texttt{Hyp}$ or are the true predicate. Hence, we have $\{p \mid p \in P'_{hyp} \ \wedge \ p \neq \texttt{true}\} \subseteq P_{hyp}$ and $\nu(P_{hyp}) \subseteq \nu(P'_{hyp})$. Since we have $\langle \nu(\texttt{model}\,(m') \ \wedge \ \bigwedge_{p\in P_{hyp}}), \texttt{st}(m')\rangle \subseteq \langle \nu(\texttt{model}\,(m') \ \wedge \ \bigwedge_{p\in P'_{hyp}}), \texttt{st}(m')\rangle$, we obtain $\langle \nu(\texttt{models}\,(\{m'\}) \wedge \bigwedge_{p\in P_{hyp}}), \texttt{st}(m')\rangle \subseteq \langle \mathcal{M}\rangle$. Let us also remind that $\nu(\texttt{models}\,(\{m\})) = \nu(\texttt{models}\,(\{m'\})) = \nu(\bigwedge_{p\in P_{obs}})$. Moreover, due to the definition of the *obser*, *fusion*, and *set* rules, we have that m and m' only differ in state names, that is, a bijective state renaming allows to obtain m from m' and viceversa. Hence, $\langle \nu(\texttt{models}\,(\{m'\})), \texttt{st}(m')\rangle = \langle \nu(\texttt{models}\,(\{m\})), \texttt{st}(m)\rangle$. As a direct consequence, we deduce $\langle \nu(\texttt{models}\,(\{m'\}) \ \wedge \ \bigwedge_{p\in P_{hyp}}), \texttt{st}(m')\rangle = \langle \nu(\texttt{models}\,(\{m\}) \wedge \bigwedge_{p\in P_{hyp}}), \texttt{st}(m)\rangle$. Thus, we obtain the desired result, that is, $\langle \nu(\texttt{models}\,(\{m\}) \wedge \bigwedge_{p\in P_{hyp}}), \texttt{st}(m)\rangle \subseteq \langle \mathcal{M}\rangle$.

The validity of the result (c) is proved by using similar arguments. □

Moreover, if all models belonging to a set of models pass the *consistent* rule then the FSMs represented by this set coincides with the set of FSMs represented by the *conjunction* of the model obtained after processing the observations *and* all the hypotheses.

Lemma 6 Let $P = P_{obs} \cup P_{hyp}$ be a set of predicates where $P_{obs} \neq \emptyset$ is a set of observation predicates and P_{hyp} is a set of hypothesis predicates. Let $\mathcal{M}$ be such that $P \vdash^* \texttt{models}(\mathcal{M})$ and $\mathcal{M} = \{\mathit{consistent}(m_1), \ldots, \mathit{consistent}(m_n)\}$, with $n \geq 1$. Finally, let $\texttt{models}(\{m\})$ be a predicate obtained from P after applying the *obser*, *fusion*, and *set* rules until they cannot be applied anymore. Then, we have $\langle \nu(\texttt{model}(m) \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle = \langle \mathcal{M} \rangle$.

Proof: Let us consider the set $\mathcal{M}' = \{m_1, \ldots, m_n\}$. By Lemma 4, we have that the predicate $\texttt{models}(\mathcal{M}')$ is such that for all $m' \in \mathcal{M}'$ the property $\texttt{model}(m') \vdash_{\mathit{consistent}} \texttt{models}(\{\mathit{consistent}(m')\})$ holds iff for all $m' \in \mathcal{M}'$ we have $\langle \nu(\texttt{model}(m') \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m') \rangle = \langle \{\mathit{consistent}(m')\} \rangle = \langle m' \rangle$ and the *determ* and *equality* rules cannot be applied to m'. Since $\texttt{models}(\mathcal{M}')$ is reached from $\texttt{model}(m)$, by applying Lemma 5 (b) we obtain the property $\langle \nu(\texttt{model}(m) \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle \subseteq \langle \mathcal{M}' \rangle$.

In order to prove the remaining inclusion, let us note first that $\langle \mathcal{M}' \rangle = \bigcup_{m' \in \mathcal{M}'} \langle m' \rangle = \bigcup_{m' \in \mathcal{M}'} \langle \nu(\texttt{model}(m') \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m') \rangle$. Let us assume that there exists $m' \in \mathcal{M}'$ and $F \in \langle \nu(\texttt{model}(m') \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m') \rangle$ such that we have that $F \notin \langle \nu(\texttt{model}(m) \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle$. Since m' is obtained from m after applying some rules, by Lemma 3 (d) we have $\langle m \rangle \supseteq \langle m' \rangle$. Thus, such an FSM F does not exist. Hence, we deduce that we have $\langle \nu(\texttt{model}(m) \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle \supseteq \bigcup_{m' \in \mathcal{M}'} \langle \nu(\texttt{model}(m') \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m') \rangle$. From this last result we can conclude $\langle \nu(\texttt{model}(m) \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle = \langle \mathcal{M}' \rangle = \langle \mathcal{M} \rangle$. □

By taking into account Lemmas 5 (a) and 6 we deduce that if a set of models passes the *consistent* rule then this set contains all FSMs that can produce the observations and fulfill the hypotheses *within* the states considered by each model of the set.

Corollary 1 Let $P = P_{obs} \cup P_{hyp}$ be a set of predicates, where $P_{obs} \neq \emptyset$ is a set of observation predicates and P_{hyp} is a set of hypothesis predicates. Let $\mathcal{M}$ be such that $P \vdash^* \texttt{models}(\mathcal{M})$ and $\mathcal{M} = \{\mathit{consistent}(m_1), \ldots, \mathit{consistent}(m_n)\}$, with $n \geq 1$. Then, there exists a model m such that

(a) $\nu(\bigwedge_{p \in P_{obs}}) = \nu(\texttt{model}(m))$
(b) $\langle \nu(\texttt{model}(m) \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle = \langle \mathcal{M} \rangle$

□

If the set of observations and hypotheses is such that at least one FSM fulfills

them then, after the successive application of rules, each model belonging to the set of models will be able to pass the *consistent* rule.

Lemma 7 Let $P = P_{obs} \cup P_{hyp}$ be a set of predicates, where $P_{obs} \neq \emptyset$ is a set of observation predicates and P_{hyp} is a set of hypothesis predicates. We have $\nu(\bigwedge_{p \in P}) \neq \emptyset$ iff there exists $\mathcal{M}$ such that $P \vdash^* \texttt{models}\,(\mathcal{M})$ and $\mathcal{M} = \{\mathit{consistent}(m_1), \ldots, \mathit{consistent}(m_n)\}$, with $n \geq 1$.

Proof: We consider the implication from left to right. By Lemma 5 (a) we have $\nu(q) = \nu(\bigwedge_{p \in P_{obs}})$, for the predicate $q = \texttt{model}\,(m)$ reached after applying *obser*, *fusion*, and *set* until they cannot be applied anymore. Let us show that there exists a predicate $\texttt{models}\,(\mathcal{M})$, with $\mathcal{M} = \{m_1, \ldots, m_n\}$, such that it is reached from q and for all $1 \leq i \leq n$ we have $\texttt{model}\,(m_i) \vdash_{\mathit{consistent}} \texttt{models}\,(\{\mathit{consistent}(m_i)\})$. Let us note that if this holds then, by applying n times the *propagation* rule, we can immediately deduce $P \vdash^* \texttt{models}\,(\mathcal{M}')$, with $\mathcal{M}' = \{\mathit{consistent}(m_1), \ldots, \mathit{consistent}(m_n)\}$.

By the definition of the *consistent* rule, a predicate $\texttt{models}\,(\mathcal{M}'')$ is such that for all $m'' \in \mathcal{M}''$ we have $\texttt{model}\,(m'') \vdash_{\mathit{consistent}} \texttt{models}\,(\{\mathit{consistent}(m'')\})$ iff for all $m'' \in \mathcal{M}''$ we have $\texttt{unable}(m'', \mathcal{R}')$, where $\mathcal{R}' = \mathcal{R} \backslash \{\mathit{consistent}, \mathit{correct}\}$. We will show that such a predicate $\texttt{models}\,(\mathcal{M}'')$ is reached from q by operating as follows: First, the *singleInitial*, *allDet*, *origin*, *destination*, and *upper* rules are applied once for each corresponding hypothesis in `Hyp`. Next, the *allTran* and *long* rules are repeatedly applied in any order. These seven rules are applied in such a way that, after each one is applied, the rules *determ* and *equality* rules are applied until they cannot be applied anymore.

Let us note that the *singleInit* and *allDet* rules can modify each model only once, that is, if we assume that there is a single initial state or that all states are deterministic, further application of these rules will return a singleton with the same model. This is also the case of the *origin* and *destination* rules: After *equality* and *determ* exploit all new equalities added to $\mathcal{E}$ by these rules, in each case either a single origin/destination state will remain in the model, or an inconsistency will be found. A single application of *upper* is enough because the number of model states never increases. Thus, after only models with n states are returned, the hypothesis will always hold. The *allTran* and *long* rules are such that if they do not return a singleton with the same model, then they return some models where each one contains new equalities in its corresponding set $\mathcal{E}$. After each of them is applied, the rules *determ* and *equality* will exploit these new equalities to either find an inconsistency (i.e., the model is eliminated) or match some states (i.e., the number of states is reduced). Rules are not applied to eliminated models and the elimination of states cannot be performed indefinitely because the number of states is finite. Let us consider

the following expression:

$$b(\mathcal{M}) = \left\{ (i,n) \left| \begin{array}{l} 1 \leq i \leq max\{\mathtt{st}(m) | m \in \mathcal{M}\} \wedge \\ n = |\{m | m \in \mathcal{M} \ \wedge \ \mathtt{st}(m) = i\}| \end{array} \right. \right\}$$

We consider the following order relation between sets of models: $\mathcal{M} > \mathcal{M}'$ iff

$$max \left\{ l \left| \begin{array}{c} \exists\, n\, ((l,n) \in b(\mathcal{M})) \wedge \nexists\, m\, ((l,m) \in b(\mathcal{M}')) \vee \\ \exists\, n,m\, ((l,n) \in b(\mathcal{M}) \ \wedge \ (l,m) \in b(\mathcal{M}') \ \wedge \ n > m) \end{array} \right. \right\}$$

$$\bigvee$$

$$max \left\{ l \left| \begin{array}{c} \exists\, n\, ((l,n) \in b(\mathcal{M}')) \wedge \nexists\, m\, ((l,m) \in b(\mathcal{M})) \vee \\ \exists\, n,m\, ((l,n) \in b(\mathcal{M}') \ \wedge \ (l,m) \in b(\mathcal{M}) \ \wedge \ n > m) \end{array} \right. \right\}$$

For example, if $\mathcal{M}_1$ has one model with 3 states and three models with 2 states then $b(\mathcal{M}_1) = \{(3,1),(2,3),(1,0)\}$; if $b(\mathcal{M}_2) = \{(2,4),(1,8)\}$ then $\mathcal{M}_1 > \mathcal{M}_2$ because $3 > 2$.

Let $\mathtt{models}\,(\mathcal{M})$ and $\mathtt{models}\,(\mathcal{M}')$ be predicates such that $\mathtt{models}\,(\mathcal{M}')$ is obtained from $\mathtt{models}\,(\mathcal{M})$ after applying a *allTran* or *long* rule and then *determ* and *equality* until they cannot be applied anymore (the application of each requires applying *propagation* afterwards). Then, we have $\mathcal{M} > \mathcal{M}'$ because if the *allTran* or *long* rules modify a model then new returned models have less states than the original model. Let us note that there exists a *minimal* set of models: $\emptyset$. However, if $\nu(\bigwedge_{p \in P}) \neq \emptyset$ then this predicate cannot be reached. By Lemma 5 (c), for all $\mathcal{M}$ such that $P \vdash^* \mathtt{models}\,(\mathcal{M})$ we have $\nu(q \ \wedge \ \bigwedge_{p \in P_{hyp}}) \subseteq \nu(\mathtt{models}\,(\mathcal{M}))$. As we said before, we have $\nu(q) = \nu(\bigwedge_{p \in P_{obs}})$. So, $\nu(\bigwedge_{p \in P_{obs}} \ \wedge \ \bigwedge_{p \in P_{hyp}}) \subseteq \nu(\mathtt{models}\,(\mathcal{M})$. Thus, $\nu(\bigwedge_{p \in P}) \neq \emptyset$ implies $\nu(\mathtt{models}\,(\mathcal{M})) \neq \emptyset$. That is, if $\nu(\bigwedge_{p \in P}) \neq \emptyset$ then the minimal set of models is $\mathtt{models}\,(\{m\})$, where $\mathtt{st}(m) = 1$. Hence, the proposed process terminates and it is done in such a way that some FSMs are represented: A set of models $\mathcal{M}''$, with $\nu(\mathtt{models}\,(\mathcal{M}'')) \neq \emptyset$, where no rule in $\mathcal{R}'$ can modify any model $m'' \in \mathcal{M}''$ is finally reached. That is, for all $m'' \in \mathcal{M}''$ we have $\mathtt{unable}(m'', \mathcal{R}')$ and the *consistent* rule can be applied to all models in $\mathcal{M}''$.

Next we consider the implication from right to left. Let $\mathcal{M}$ be a set of models such that $P \vdash^* \mathtt{models}\,(\mathcal{M})$ and $\mathcal{M} = \{\mathit{consistent}(m_1), \ldots, \mathit{consistent}(m_n)\}$, with $n \geq 1$. Let us note that if the *determ* and *equality* rules cannot be applied to a model m_i then $\nu(\mathtt{model}\,(m_i)) \neq \emptyset$. A model represents an empty set of FSMs only if the components of the tuple denoting the model contradict each other (let us note that a model without states represents *all* FSMs). Let $m_i = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \mathcal{O})$. By the manipulation of $\mathcal{T}$ and $\mathcal{I}$ in rules, these

sets never refer to states not included in $\mathcal{S}$. The set $\mathcal{A}$ cannot contradict other sets: If $(s, i, outs, f, \top) \in \mathcal{A}$ then no *more* outputs are allowed from s after performing i, but all elements in *outs* are allowed. By the construction of $\mathcal{A}$, the set *outs* coincides with the current set of available outputs from s after i, and the function f denotes the number of observed occurrences of each transition from s with i. The set $\mathcal{O}$ does not affect $\nu(\texttt{model}\,(m_i))$. So, inconsistencies can only be produced by the contents of $\mathcal{E}$ and $\mathcal{D}$ (with respect to $\mathcal{T}$). If *determ* cannot be applied then all states in the set $\mathcal{D}$ are actually deterministic according to $\mathcal{T}$. Besides, if *equality* cannot be applied then $\mathcal{E}$ does not allow to match more pairs of states. So, (nondeterministic) states cannot be added to the set $\mathcal{D}$. Hence, if $\texttt{unable}(m_i, \{equality, determ\})$ then $\nu(\texttt{model}\,(m_i)) \neq \emptyset$. Since $P \vdash^* \texttt{models}\,(\mathcal{M})$, by Lemmas 5 (a) and 6 we have $\langle(\bigwedge_{p\in P_{obs}} \wedge \bigwedge_{p\in P_{hyp}}), \texttt{st}(m)\rangle = \langle\mathcal{M}\rangle$, where $q = \texttt{model}\,(m)$ is again the predicate reached after applying *obser*, *fusion*, and *set*. Since $\nu(\texttt{model}\,(m_i)) \neq \emptyset$, we have $\nu(\texttt{models}\,(\mathcal{M})) \neq \emptyset$ and $\langle\mathcal{M}\rangle \neq \emptyset$. So, $\langle(\bigwedge_{p\in P_{obs}} \wedge \bigwedge_{p\in P_{hyp}}), \texttt{st}(m)\rangle \neq \emptyset$ and $\nu(\bigwedge_{p\in P}) \neq \emptyset$. □

Let us note that the proof of the previous lemma implicitly provides a suitable criterion to select which rule to apply next: By following the proposed order, a diagnostic is reached after applying rules a finite number of times.

Next we show that a model passes the *correct* rule if and only if all the FSMs it represents conform to the specification.

Lemma 8 Let *spec* be an FSM and m be a model. We have $\texttt{worstCase}(m)\,\texttt{conf}$ *spec* iff for all $F \in \nu(\texttt{model}\,(m))$ we have $F\ \texttt{conf}$ *spec*.

Proof: Let $m = (\mathcal{S}, \mathcal{T}, \mathcal{I}, \mathcal{A}, \mathcal{E}, \mathcal{D}, \mathcal{O})$ be a model. First we prove the implication from left to right. Let us suppose $\texttt{worstCase}(m)\ \texttt{conf}$ *spec* and let us assume that there exists $F \in \nu(\texttt{model}\,(m))$ such that $F\ \texttt{conf}$ *spec* does not hold. By the definition of the conf relation (see Definition 2), there exists $\rho_1 = (i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o_n) \in \texttt{traces}(spec)$, with $n \geq 1$, such that $\rho_2 = (i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o'_n) \in \texttt{traces}(F)$ and $\rho_2 \notin \texttt{traces}(spec)$. Next we show that there actually exist two sequences $\rho'_1 = (i_1/o_1, \ldots, i_{j-1}/o_{j-1}, i_j/o_j)$ and $\rho'_2 = (i_1/o_1, \ldots, i_{j-1}/o_{j-1}, i_j/o'_j)$, with $j \leq n$, such that $\rho'_1 \in \texttt{traces}(spec)$, $\rho'_2 \in \texttt{traces}(\texttt{worstCase}(m))$, but $\rho'_2 \notin \texttt{traces}(spec)$. In this case, we have $\texttt{worstCase}(m)\ \texttt{conf}$ *spec* is false, which is a contradiction.

Let $F = (\mathcal{S}', \texttt{inputs}', \texttt{outputs}', \mathcal{I}', \mathcal{T}')$ and $\rho' = (i_1/o_1, \ldots, i_p/o_p)$ be the maximal prefix of $\rho = (i_1/o_1, \ldots, i_{n-1}/o_{n-1})$ such that both F and $\texttt{worstCase}(m)$ can perform ρ' by traversing the same states from the same initial state. We consider the following possibilities: Either $p = 0$ (i.e. ρ' is empty) or $p \geq 1$. If $p = 0$ then the initial state used in F to perform ρ either does not belong to $\mathcal{S}$ or it does but it is not an initial state belonging to $\mathcal{I}$. By the construction of F, these cases imply $\beta \in \mathcal{I}$ and $\{\alpha, \beta\} \cap \mathcal{I} \neq \emptyset$, respectively. If $\beta \in \mathcal{I}$ then,

by the construction of $\texttt{worstCase}(m)$ given in Definition 4, $\perp$ is an initial state of $\texttt{worstCase}(m)$. We also have that $\perp \xrightarrow{i_1/error} \perp$ is a transition in $\texttt{worstCase}(m)$. Since $(i_1/o_1) \in \texttt{traces}(spec)$ and $(i_1/error) \notin \texttt{traces}(spec)$, we deduce $\texttt{worstCase}(m)$ $\texttt{conf}$ $spec$ does not hold. If $\beta \notin \mathcal{I}$ then, taking into account that $\{\alpha, \beta\} \cap \mathcal{I} \neq \emptyset$, we have $\alpha \in \mathcal{I}$. This implies that all states of $\texttt{worstCase}(m)$ are initial, which contradicts the fact that F performs ρ from a state in $\mathcal{S}$ that is not initial. Hence, if $p = 0$ then $\alpha \in \mathcal{I}$ is not possible.

Let us consider the case: $1 \leq p < n-1$. For some state $v \in \mathcal{S} \cup \mathcal{S}'$ we have $v \xrightarrow{i_{p+1}/o_{p+1}} w \in \mathcal{T}'$ and $v \xrightarrow{i_{p+1}/o_{p+1}} w \notin \mathcal{T}$. According to the construction of F we deduce that $v \notin \mathcal{D}$ and there does not exist $outs$ and f such that $(v, i_{p+1}, outs, f, \top) \in \mathcal{A}$. Hence, by the construction of $\texttt{worstCase}(m)$ from m, in this case we have that $v \xrightarrow{i_{p+1}/error} \perp$ is a transition in $\texttt{worstCase}(m)$. So, $\rho'' = (i_1/o_1, \ldots, i_{p+1}/error) \in \texttt{traces}(\texttt{worstCase}(m))$. Besides, we have $(i_1/o_1, \ldots, i_{p+1}/o_{p+1}) \in \texttt{traces}(spec)$, but $\rho'' \notin \texttt{traces}(spec)$ because $error \notin \texttt{outputs}_{spec}$. Thus, $\texttt{worstCase}(m)$ does not conform to $spec$.

Finally, let us consider the case when $p = n-1$. If we have $s \xrightarrow{i_n/o'_n} s'' \in \mathcal{T}$, for some s'', then we trivially obtain that $\texttt{worstCase}(m)$ does not conform to $spec$ because of ρ_2 (see the beginning of the proof for the definition of ρ_2). Let us suppose that there does not exist s'' such that $s \xrightarrow{i_n/o'_n} s'' \in \mathcal{T}$. Since there exists s''' such that $s \xrightarrow{i_n/o'_n} s''' \in \mathcal{T}'$, by the construction of F, we have that there does not exist $outs$ and f such that $(s, i_n, outs, f, \top) \in \mathcal{A}$. Moreover, either $s \notin \mathcal{D}$ or there do not exist s'' and o such that $s \xrightarrow{i_n/o} s'' \in \mathcal{T}$. Hence, by the construction of $\texttt{worstCase}(m)$, we have $(i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/error) \in \texttt{traces}(\texttt{worstCase}(m))$. So, $\texttt{worstCase}(m)$ is again incorrect with respect to $spec$.

Now we consider the implication from right to left. Again, we reason by contrapositive: We suppose that $\texttt{worstCase}(m)$ does not conform to $spec$ and we show that there exists $F \in \nu(\texttt{model}\,(m))$ such that F does not conform to $spec$. There exists $\rho_1 = (i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o_n) \in \texttt{traces}(spec)$ with $n \geq 1$ such that $\rho_2 = (i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o'_n) \in \texttt{traces}(\texttt{worstCase}(m))$ and $\rho_2 \notin \texttt{traces}(spec)$. We have two possibilities: Either $o'_n = error$ or not. Let us suppose that $o'_n \neq error$. Then, $\texttt{worstCase}(m)$ can perform the transition that allows to execute i_n/o'_n from m. By the construction of each machine, for all $F \in \nu(\texttt{model}\,(m))$ we have $\rho_2 \in \texttt{traces}(F)$. So, F $\texttt{conf}$ $spec$ does not hold. Let us consider now the second case, that is, $o'_n = error$, and let $s \in \mathcal{S}$ be the state reached in $\texttt{worstCase}(m)$ after executing $(i_1/o_1, \ldots, i_{n-1}/o_{n-1})$ and before performing $i_n/error$. By the construction of $\texttt{worstCase}(m)$, there does not exist $outs$ and f such that $(s, i_n, outs, f, \top) \in \mathcal{A}$ and either $s \notin \mathcal{D}$ or there do not exist $s'' \in \mathcal{S}$ and o such that $s \xrightarrow{i_n/o} s'' \in \mathcal{T}$. Actually, by the

construction of FSMs from m, there exists an FSM $F \in \nu(\texttt{model}\,(m))$ for each possible behavior we can *invent* from $s \in \mathcal{S}'$ with input i_n. Since there exists an output $o \in \texttt{outputs}_{spec}$ such that o cannot be produced in *spec* in response to the input i from the state s, we have $(i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o) \in \texttt{traces}(F)$ but $(i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o) \notin \texttt{traces}(spec)$. Hence, F does not conform to *spec*. □

Next we show that, in terms of conformance, it is equivalent to consider evaluations or constrained evaluations.

Lemma 9 Let *spec* be an FSM and m be a model. For all $F \in \nu(\texttt{model}\,(m))$ we have F conf *spec* iff for all $F' \in \langle m \rangle$ we have F' conf *spec*.

Proof: Clearly, $\langle m \rangle \subseteq \nu(\texttt{model}\,(m))$. So, we only need to consider the implication from left to right. By contrapositive, let us suppose that there exist $F \in \nu(\texttt{model}\,(m))$ and two traces $\rho_1 = (i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o_n)$ and $\rho_2 = (i_1/o_1, \ldots, i_{n-1}/o_{n-1}, i_n/o'_n)$, with $n \geq 1$, such that $\rho_1 \in \texttt{traces}(spec)$, $\rho_2 \in \texttt{traces}(F)$, but $\rho_2 \notin \texttt{traces}(spec)$. Let F' be the FSM that results from F by eliminating all states that are not in $\texttt{st}(m)$ and all transitions involving them. We have $F' \in \langle m \rangle$. In order to show that F' conf *spec* does not hold we use similar arguments to those used in the proof of Lemma 8, though we refer to F' instead of to $\texttt{worstCase}(m)$. Let $\rho'_1 = (i_1/o_1, \ldots, i_j/o_j)$ be the longest prefix of ρ_1 such that $\rho'_1 \in \texttt{traces}(F')$ and both F and F' can execute this sequence by traversing the same states. Let us note that the transitions that allow F' to execute ρ'_1 are those of m. If F performs (i_{j+1}/o_{j+1}) after ρ_1 then it means that m *allows* to extend its set of transitions to do it. According to the construction of F, this implies that the corresponding state is not closed for i_{j+1} and either it is not deterministic or no current transition describes the reaction of this state to i_{j+1}. Then, there exists $F'' \in \langle m \rangle$ such that F'' performs (i_{j+1}/o) at this point and moves to a state belonging to $\texttt{st}(m)$. However, $\rho'_2 = (i_1/o_1, \ldots, i_j/o_j, i_{j+1}/o) \notin \texttt{traces}(spec)$. Since we have $(i_1/o_1, \ldots, i_j/o_j, i_{j+1}/o_{j+1}) \in \texttt{traces}(spec)$ and $\rho'_2 \in \texttt{traces}(F'')$, we deduce that F'' does not conform to *spec*. □

A straightforward application of Lemmas 8 and 9 leads to the following corollary: We can assess models in terms of their constrained evaluations.

Corollary 2 Let *spec* be an FSM and m be a model. Then, we have that $\texttt{worstCase}(m)$ conf *spec* iff for all $F \in \langle m \rangle$ we have F conf *spec*. □

In the next result we relate the terms obtained by the application of rules and the set of FSMs that actually fulfill the observations and hypotheses. On the one hand, if an FSM belongs to the *constrained evaluation* of the set of models obtained after applying the *consistent* rule then it fulfills the observations and hypothesis. On the other hand, if an FSM fulfills the observations and hypotheses then it belongs to the *evaluation* of the set obtained after applying

the *consistent* rule.

Lemma 10 Let $P = P_{obs} \cup P_{hyp}$ be a set of predicates where $P_{obs} \neq \emptyset$ is a set of observation predicates and P_{hyp} is a set of hypothesis predicates. Let us suppose $P \vdash^* \texttt{models}\,(\mathcal{M})$, where $\mathcal{M} = \{\mathit{consistent}(m_1), \ldots, \mathit{consistent}(m_n)\}$. Then,

(a) For all $F \in \langle \mathcal{M} \rangle$ we have $F \in \nu(\bigwedge_{p \in P_{obs}} \wedge \bigwedge_{p \in P_{hyp}})$.
(b) For all $F \in \nu(\bigwedge_{p \in P_{obs}} \wedge \bigwedge_{p \in P_{hyp}})$ we have $F \in \nu(\texttt{models}\,(\mathcal{M}))$.

Proof: First, let us consider (a). By Corollary 1, we have that there exists m such that $\nu(\bigwedge_{p \in P_{obs}}) = \nu(\texttt{model}\,(m))$ and $\langle \nu(\texttt{model}\,(m) \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle = \langle \mathcal{M} \rangle$. Thus, we have $\langle \mathcal{M} \rangle = \langle \nu(\bigwedge_{p \in P_{obs}} \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle$. Since we have $\langle \nu(\bigwedge_{p \in P_{obs}} \wedge \bigwedge_{p \in P_{hyp}}), \texttt{st}(m) \rangle \subseteq \nu(\bigwedge_{p \in P_{obs}} \wedge \bigwedge_{p \in P_{hyp}})$, we deduce that (a) holds.

Next we consider (b). By Lemma 5 (c), for all $\mathcal{M}$ obtained from m we have $\nu(\texttt{model}\,(m) \wedge \bigwedge_{p \in P_{hyp}}) \subseteq \nu(\texttt{models}\,(\mathcal{M}))$. Hence, $\nu(\bigwedge_{p \in P_{obs}} \wedge \bigwedge_{p \in P_{hyp}}) \subseteq \nu(\texttt{models}\,(\mathcal{M}))$ and so we conclude that (b) holds. □

Finally, we use all the previous machinery to prove the soundness and completeness of $\mathcal{HOTL}$.

Theorem 1 *(Soundness and Completeness Theorem)* Let *spec* be an `FSM` and P be a set of predicates including at least one observation predicate. Then, P `logicConf` *spec iff* for all `FSM` $F \in \nu(\bigwedge_{p \in P})$ we have that F `conf` *spec* and $\nu(\bigwedge_{p \in P}) \neq \emptyset$.

Proof: Let us consider the implication from left to right. If P `logicConf` *spec* then $P \vdash^*$ `allModelsCorrect`. Since a set of correct models is required to achieve `allModelsCorrect` and, before this, a set of consistent models is also required, we have $P \vdash^* \texttt{models}\,(\mathcal{M})$, for some set of consistent models $\mathcal{M} = \{\mathit{consistent}(m_1), \ldots, \mathit{consistent}(m_n)\}$, with $n \geq 1$. By Lemma 7 we have $\nu(\bigwedge_{p \in P}) \neq \emptyset$. We also have that each model m_i passes the *correct* rule, that is, $\texttt{worstCase}(m_i)$ `conf` *spec*. Let us consider any $F \in \nu(\bigwedge_{p \in P})$ such that F `conf` *spec* does not hold. Due to Lemma 10 (b), $F \in \nu(\mathcal{M})$. So, there exists m_i such that $F \in \nu(m_i)$. If F `conf` *spec* does not hold then, by Lemma 9, there exists $F' \in \langle m_i \rangle$ such that F' `conf` *spec* does not hold. Then, by Corollary 2, $\texttt{worstCase}(m_i)$ `conf` *spec* does not hold and at least one model of $\mathcal{M}$ will not pass the *correct* rule. So, we do not achieve `allModelsCorrect`, which makes a contradiction.

We consider the implication from right to left. If $\nu(\bigwedge_{p \in P}) \neq \emptyset$ then, by Lemma 7, we have $P \vdash^* \texttt{models}\,(\mathcal{M})$ for some set of consistent models $\mathcal{M} = \{\mathit{consistent}(m_1), \ldots, \mathit{consistent}(m_n)\}$, with $n \geq 1$. Let us suppose that for all `FSM` $F \in \nu(\bigwedge_{p \in P})$ we have F `conf` *spec* and let us prove that from $\texttt{models}\,(\mathcal{M})$ we can achieve `allModelsCorrect`. If we do not, then there ex-

ists $m_i \in \mathcal{M}$ such that $\texttt{worstCase}(m_i)$ $\texttt{conf}$ *spec* does not hold. By Lemma 8, this also means that for some $F \in \nu(\texttt{model}\ (m_i))$ we have that $F\,\texttt{conf}\,spec$ does not hold. Moreover, by Lemma 9 there exists $F' \in \langle m_i \rangle$ such that $F'\,\texttt{conf}\,spec$ does not hold. Since $F' \in \langle \mathcal{M} \rangle$, by Lemma 10 (a) we deduce $F' \in \nu(\bigwedge_{p \in P})$. So, we get a contradiction. □

Corollary 3 Let IUT and *spec* be FSMs and P be a set of predicates including at least one observation predicate. If $IUT \in \nu(\bigwedge_{p \in P})$ then P $\texttt{logicConf}$ *spec* implies $IUT\,\texttt{conf}\,spec$. Moreover, if there exists $F \in \nu(\bigwedge_{p \in P})$ such that $F\,\texttt{conf}$ *spec* does not hold then P $\texttt{logicConf}$ *spec* does not hold. □

7 Extending $\mathcal{HOTL}$: General Guidelines

In this section we consider several possibilities to extend $\mathcal{HOTL}$ with new capabilities. We present two different directions of improvement: Improvements oriented to add the capability to express new hypotheses, and improvements oriented to ease the transition from the logic, considered as a theoretical framework, to a tool/prover allowing to make the same deductions computationally.

Regarding the representation of new hypotheses, next we consider some guidelines to extend $\mathcal{HOTL}$ with additional predicates and rules. If a user desires to extend the current repertory of available hypotheses then she has to take into account the way properties are represented and analyzed in the logic. $\mathcal{HOTL}$ is based on the idea that in order to analyze the (infinite) set of FSMs that fulfill some properties, it is enough to analyze the worst particularization of a *model* representing these properties. This particularization is, in turn, a single FSM. The treatment of any new hypothesis must fit into this scheme. In order to add a new hypothesis to the repertory, the user of the logic has to follow the following steps:

(1) Check whether the tuple representing models has to be extended with new components to denote additional information. Let us note that a new hypothesis could not need such an additional information. For example, if we want to add the hypothesis "*all the times a given input is proposed everywhere in the IUT, a given fix output is produced*" then no new information is required in models: Adding this hypothesis just consists in adding new transitions (in the set $\mathcal{T}$) and *closing* the behavior of all states with respect to this input (in the set $\mathcal{A}$). Similarly, if we want to add the hypothesis "*after n observations are obtained, all initial states are observed at least once*" then it is enough to consider the number of observations in Obs. So, no additional information is required. However, if we need to add a hypothesis such as "*if an IUT state is reached n times then the behavior of all inputs in this state is closed*" then we need an additional component in models

to denote the number of times each model state is reached. In general, if new components are added to models then we have to define how these components are managed by the `modelElim` function, which transfers the responsibilities of some states to others when a state is eliminated.

(2) By means of the characterization function α, the *semantics* of the new hypothesis predicate must be provided. This fact will allow to check the correctness of the new rule with respect to the rest of rules of the logic (see the next step).

(3) A new rule allowing to assume the hypothesis in a given model has to be added. For the sake of homogeneity, it is very convenient that the new rule follows the form required to apply the *propagation* rule. In order to guarantee a correct behavior of the logic after the new rule is added, it must fulfill some properties. As we saw in the previous section, the correctness and completeness of $\mathcal{HOTL}$ is based on the fulfillment of some properties by all rules in the logic. Two results summarize the dependence of the proof constructed in the previous section on the behavior of each specific rule: Lemmas 3 and 7. If rules allow to infer these two results then the rest of results considered in the section, which are a consequence of these properties, hold and the soundness and completeness of the logic is obtained. Regarding Lemma 3, in the new rule we have to check that
   - the constrained evaluation of the premises is included in the constrained evaluation of the conclusion (Lemma 3, property (a)) and that the evaluation of the premises is included in the evaluation of the conclusion (b);
   - the constrained evaluation of the conclusion is included in the constrained evaluation of the model given in the premise (property d); and
   - if the constrained evaluation actually fulfills the hypothesis then the constrained evaluation of the conclusion coincides with the one corresponding to the premise (property e).

   The property (c) of Lemma 3 does not have to be considered because it does not depend on the definition of each rule. Regarding Lemma 7, we have to prove that there always exists a finite sequence of applications of rules leading to a model where no more rules can be applied. Concerning the new rule, checking this consists in proving that either the new rule does not need to be applied more than once, or that its application returns a *smaller* set of models according to a given termination criterion defined in the proof of Lemma 7. Besides, if new components are added to the formal representation of models in step (1) then we have to prove that any inconsistency can be detected by applying the *determ* and *equality* rules.

Let us note that extending $\mathcal{HOTL}$ may also consist in providing the capability to *refuse* some hypotheses that are implicitly assumed in the current framework. The current logic assumes that specifications and implementations are given by finite state machines. A more general framework would allow to suppose that systems are represented by using other formalisms. As a *particular* case, we could assume the hypothesis that systems actually are finite state

machines. Let us consider a few examples. Currently, the only method to denote that the IUT does not produce any output as response to an input is to use a special output action to denote a *mute* response. If systems are represented by using *labelled transition systems*, where inputs and outputs are not paired, this situation can be dealt with in a more natural way. Other interesting formalisms such as probabilistic automata, probabilistic finite state machines, timed automata, etcetera, could be considered. In each case, new specific hypotheses should deal with the peculiarities of each framework. For example, we may consider properties such as "*no action has a probability lower than p,*" or "*if an action is observed at time t then we assume that it is also produced in all times belonging to the interval* $[t - \delta, t + \delta]$."

At this point, let us consider how the logic can be improved to ease the transition from the current theoretical framework to a computational system performing the same deductions. Let us note that some rules of our logic potentially lead to an *explosion* of models. For example, the *upper* rule returns a model for each way to map the states of the model into n states at most. In spite of the fact that doing this is theoretically correct, applying this rule is not feasible in a computational environment if a model has a high number of states. Similarly, other rules consider all the ways to map some states in such a way that a property holds. In these cases, there are several possibilities to limit the number of models that are constructed and analyzed:

- Though hypothesis rules can be applied in any order, the application order influences the number of steps and models that are needed to provide conclusions. For example, a given order may lead to quickly discover that some models are *inconsistent* and can be discarded from the calculus, while a different order could make us to consider and maintain these models longer. In order to take advantage of the elimination of inconsistent models, we can split the application of a hypothesis into *several* steps in such a way that inconsistent models are systematically eliminated after each step. For instance, in the *upper* rule, the operation of mapping m states into n states can be split into $m - n$ operations, consisting each of them in mapping two states into a single state. After each step is done, an exhaustive search for inconsistencies can be performed. For instance, let us suppose that 10 original states must be mapped into 5 states. Besides, let us assume that two deterministic states s_1 and s_2 answer b and c, respectively, when the input a is proposed. Clearly, matching s_1 and s_2 leads to an inconsistent model. Thus, there is no need to consider any mapping where, in particular, we match s_1 and s_2. If states are iteratively matched in such a way that inconsistencies are searched after each match, then we avoid to consider any mapping where this pair is included: All models where the pair (s_1, s_2) is chosen are immediately discarded. A new version of the *upper* rule follows. It matches a single pair of states each time as well as requires a complete application of other hypothesis rules to detect inconsistencies after each

step.

$$(upper)\frac{\begin{array}{c}\texttt{model}\,(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E},\mathcal{D},\texttt{Obs})\ \wedge\ \texttt{upperBoundOfStates}(n)\in\texttt{Hyp}\ \wedge \\ |\mathcal{S}|>n\ \wedge\ \texttt{unable}(m,\mathcal{R}\backslash\{upper,consistent,correct\})\end{array}}{\texttt{modelsSubset}\left(\bigcup_{\mathcal{E}'\in K}(\mathcal{S},\mathcal{T},\mathcal{I},\mathcal{A},\mathcal{E}\cup\mathcal{E}',\mathcal{D},\texttt{Obs})\right)}$$

where K contains all the sets $\mathcal{E}' = \{u_1\ \texttt{is}\ w, u_2\ \texttt{is}\ w\}$, being w a fresh state identifier name and $u_1, u_2 \in \mathcal{S}$.

- In some situations, the explosion of models could be avoided by using known test derivation algorithms. There exist in the literature several algorithms to construct finite test suites which are complete to detect faults in the IUT, that is, such that any faulty IUT is detected by a test of the suite (see for example [6,22,20,9,1]). Usually, the completeness of these suites lies on the assumption of an upper bound in the number of states of the IUT (our `upperBoundOfStates`(n) hypothesis). Though $\mathcal{HOTL}$ deals with a more general framework where the set of hypotheses is chosen by the tester, in particular cases where the assumed hypotheses coincide with those actually required by one of such algorithms we could stop the application of rules and apply the algorithm as follows. First, we use the algorithm to extract a (complete) test suite from the specification. Then, we use the test suite to analyze our *knowledge* about the IUT: If all `FSMs` represented by the current *model* pass all tests in the test suite provided by the algorithm then the IUT is necessarily correct. In order to check this, it is enough to check whether the *worst particularization* of the model (a single `FSM`) does so. If this `FSM` passes the test suite then the IUT is correct. Thus, no more rules have to be applied (in particular, *upper* does not have to be applied) and the explosion of models is avoided. If the `FSM` does not pass a test of the suite then no conclusions are obtained since we still have to apply the logic rules to further refine the model in order to reach a diagnostic. Going one step further, it would be desirable that the algorithm can take advantage of the knowledge given in the model so that, based on this information, smaller test suites can be considered. Integrating $\mathcal{HOTL}$ with known test derivation algorithms in such way requires a deeper research that is outside the scope of this paper.

There are other issues that may help to improve the efficiency of a computational calculus based on $\mathcal{HOTL}$. As we saw in the previous section, the `allModelsCorrect` diagnostic is reached iff all `FSMs` that produce the considered observations and fulfill the hypotheses conform to the considered specification. Hence, if the `allModelsCorrect` predicate cannot be obtained then we know that the conformance of the IUT cannot be guaranteed. Let us note that if a model tagged as *consistent* does not pass the *correct* rule then there is no need to analyze any other model belonging to the set of models: The

IUT is not necessarily conforming. So, a simple rule denoting that the worst particularization of a consistent model is not conforming avoids to perform a lot of unnecessary calculations.

We can also provide a rule denoting that a consistent model is necessarily *incorrect*. This information is specially relevant for the tester, who can use it to provide a safe diagnostic of incorrectness. This rule would be similar to the *correct* rule. This time, the *best* particularization would be considered. In order to build the best particularization of a model, unknown behaviors would be filled by taking the *specification* behavior (being correct by definition), instead of by adding the worst possible behavior. For all model states with non-closed inputs (i.e., unknown transitions), we consider all traces leading to this state in the model. Let us note that there could be infinite such traces. Then, each unknown is filled with a behavior that must be *common* to all the specification states that are reached in the specification by any of these traces (the number of these specification states is, obviously, finite). If the resulting `FSM` is incorrect then the model represents an `FSM` that is *necessarily* incorrect.

Let us note that in order to guarantee that the IUT is incorrect, *all* models in the set of models must be incorrect. In particular, if a model in the set is not necessarily incorrect, then the IUT could be still correct. Besides, let us note that considering the incorrectness of a model that has not been labelled as *consistent* yet is pointless. In fact, if this model turns out to be inconsistent then it will be ignored because fulfilling the requirements it represents is impossible. Hence, its incorrectness is irrelevant: This model does *not* represent a possible IUT.

8 Conclusions and Future Work

In this paper we have presented a logic to infer whether a collection of observations obtained by testing an IUT together with a set of hypotheses allow to deduce that the IUT conforms to the specification. A repertory of heterogeneous hypotheses providing a tester with expressivity to denote a wide range of testing scenarios has been presented. By considering those observations and hypotheses that better fit into her necessities, the tester can obtain diagnosis results about the conformance of an IUT in a flexible range of situations. Besides, our logic allows her to iteratively add observations (i.e., the results of the application of tests) and/or hypotheses until the complete set of predicates guarantees the conformance. In this sense, our logic can be used to dynamically *guide* the steps of a testing methodology.

As future work we will study some ways to improve our logic. Some of them are described in detail in the previous section: Extending the repertory of hy-

potheses, allowing more expressive formalisms for representing specifications and implementations, integrating $\mathcal{HOTL}$ with known test derivation algorithms as a way to avoid the model explosion, providing an incorrectness rule, etc. In particular, dealing with efficiency issues is required prior to developing a prover allowing to make the same deductions computationally. Besides, we want to develop a more complex application example in the context of Internet protocols. This would allow to *test* the versatility of our framework with real applications. We would also like to introduce a *feasibility* score for each of the rules. For example, for a given framework, we can consider that assuming that all the states are deterministic is harder than assuming that the implementation has less that 50 states. In this case, a lower feasibility score will be assigned to the first hypothesis. By accounting the feasibility of all the hypotheses that we have to add before ensuring conformance we will obtain a measure of the suitability of the considered observations and, indirectly, of the tests that we used to obtain them. Hence, our logic can help a tester to choose her tests so that more *trustable* diagnosis results are obtained.

Acknowledgements. We would like to thank the anonymous reviewer of this paper for the careful reading, the valuable comments, and his/her enthusiasm regarding $\mathcal{HOTL}$. His/her review has definitively improved the quality of the paper. We would also like to thank the anonymous reviewers of [19]. In particular, they provided very interesting ideas for continuing our work (some of them are commented above). Certainly, these ideas will be considered in the future to continue our research on $\mathcal{HOTL}$.

Appendix: Repertory of Hypotheses of $\mathcal{HOTL}$

Hypothesis	**Source**	**Rule**	**Meaning**
`det`	`Obs`	*obser*	The corresponding IUT state is deterministic
`imp`(q)	`Obs`	*obser*	The corresponding IUT state coincides with the one having as state identifier q
`spec`(s)	`Obs`	*obser*	From the corresponding IUT state on, the behavior coincides with the one given by the specification from state s
`singleInit`	`Hyp`	*singleInit*	There is a single initial state in the IUT
`allDet`	`Hyp`	*allDet*	All IUT states are deterministic
`allTranHappenWith`(n)	`Hyp`	*allTran*	For each state and input, all transitions departing from that state labelled by this input are observed after offering the input n times
`upperBoundOfStates`(n)	`Hyp`	*upper*	The IUT has at most n states
`longSequencesSamePath`(n)	`Hyp`	*long*	Every time that a sequence of length n is performed, the same IUT states are traversed
`uniqueOrigin`(i, o)	`Hyp`	*origin*	All IUT transitions labelled with i/o depart from the same state
`uniqueDestination`(i, o)	`Hyp`	*destination*	All IUT transitions labelled with i/o lead to the same state

References

[1] A.V. Aho, A.T. Dahbura, D. Lee, and M.Ü. Uyar. An optimization technique for protocol conformance test generation based on UIO sequences and Rural Chinese Postman Tours. *IEEE Transactions on Communications*, 39(11):1604–1615, 1991.

[2] J.A. Arnedo, A. Cavalli, and M. Núñez. Fast testing of critical properties through passive testing. In *15th Int. Conf. on Testing Communicating Systems, TestCom'03, LNCS 2644*, pages 295–310. Springer, 2003.

[3] E. Bayse, A. Cavalli, M. Núñez, and F. Zaïdi. A passive testing approach based on invariants: Application to the WAP. *Computer Networks*, 48(2):247–266, 2005.

[4] B.S. Bosik and M.Ü. Uyar. Finite state machine based formal methods in protocol conformance testing. *Computer Networks & ISDN Systems*, 22:7–33, 1991.

[5] A. Cavalli, C. Gervy, and S. Prokopenko. New approaches for passive testing using an extended finite state machine specification. *Journal of Information and Software Technology*, 45:837–852, 2003.

[6] T.S. Chow. Testing software design modelled by finite state machines. *IEEE Transactions on Software Engineering*, 4:178–187, 1978.

[7] E.M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 2000.

[8] G. Eleftherakis and P. Kefalas. Towards model checking of finite state machines extended with memory through refinement. In *Advances in Signal Processing and Computer Technologies*, pages 321–326. World Scientific and Engineering Society Press, 2001.

[9] S. Fujiwara, G. Bochmann, F. Khendek, M. Amalou, and A. Ghedamsi. Test selection based on finite-state models. *IEEE Transactions on Software Engineering*, 17(6):591–603, 1991.

[10] R. Hierons. Comparing test sets and criteria in the presence of test hypotheses and fault domains. *ACM Transactions on Software Engineering and Methodology*, 11(4):427–448, 2002.

[11] R.M. Hierons and M. Harman. Testing conformance of a deterministic implementation to a non-deterministic stream X-machine. *Theoretical Computer Science*, 323(1–3):191–233, 2004.

[12] D. Lee, D. Chen, R. Hao, R. Miller, J. Wu, and X. Yin. A formal approach for passive testing of protocol data portions. In *10th IEEE Int. Conf. on Network Protocols, ICNP'02*, pages 122–131. IEEE Computer Society Press, 2002.

[13] D. Lee and M. Yannakakis. Principles and methods of testing finite state machines: A survey. *Proceedings of the IEEE*, 84(8):1090–1123, 1996.

[14] L.P. Lima and A. Cavalli. A pragmatic approach to generating tests sequences for embedded systems. In *10th Workshop on Testing of Communicating Systems*, pages 288–307. Chapman & Hall, 1997.

[15] S.C. Ntafos. A comparison of some structural testing strategies. *IEEE Transactions on Software Engineering*, 14:868–874, 1988.

[16] M. Núñez and I. Rodríguez. Encoding `PAMR` into (timed) `EFSMs`. In *22nd IFIP WG 6.1 Int. Conf. on Formal Methods for Networked and Distributed Systems, FORTE'02, LNCS 2529*, pages 1–16. Springer, 2002.

[17] A. Petrenko. Fault model-driven test derivation from finite state models: Annotated bibliography. In *4th Summer School on Modeling and Verification of Parallel Processes, MOVEP 2000, LNCS 2067*, pages 196–205. Springer, 2001.

[18] A. Petrenko, N. Yevtushenko, and G. von Bochmann. Testing deterministic implementations from their nondeterministic FSM specifications. In *9th IFIP Workshop on Testing of Communicating Systems, IWTCS'96*, pages 125–140. Chapman & Hall, 1996.

[19] I. Rodríguez, M.G. Merayo, and M. Núñez. A logic for assessing sets of heterogeneous testing hypotheses. In *18th Int. Conf. on Testing Communicating Systems, TestCom'06, LNCS 3964*, pages 39–54. Springer, 2006.

[20] D.P. Sidhu and T.-K. Leung. Formal methods for protocol testing: A detailed study. *IEEE Transactions on Software Engineering*, 15(4):413–426, 1989.

[21] J. Tretmans. Test generation with inputs, outputs and repetitive quiescence. *Software – Concepts and Tools*, 17(3):103–120, 1996.

[22] S.T. Voung, W.L. Chan, and M.R. Ito. The UIOv-method for protocol test sequence generation. In *2nd IFIP TC6 Int. Workshop on Protocol Test Systems, IWPTS'89*, pages 161–175, North-Holland, 1990.