Model abstractions of Device-Delay for Interactive Cyber-Physical Systems under Uncertainty

Krishnendu Ghosh
Miami University
Hamilton, Ohio 45011
ghoshk@miamioh.edu

ABSTRACT

Formal analysis of user-device interaction is key to create correct and usable interface design in cyber-physical systems. In this work, a novel formalism is created to address cognitive errors, in particular device-delay error arising during user-device interaction. The formalism describes a resource-based approach to fulfill the goals of a user under uncertainty. Computational feasibility of the formalism is analyzed using probabilistic model checking on a prototype of a vending machine.

KEYWORDS

Model abstractions, Probababilistic model checking, device-delay, uncertainty

ACM Reference format:
Krishnendu Ghosh. 1997. Model abstractions of Device-Delay for Interactive Cyber-Physical Systems under Uncertainty. In *Proceedings of ACM Woodstock conference, El Paso, Texas USA, July 1997 (WOODSTOCK'97),* 6 pages.
DOI: 10.475/123_4

1 INTRODUCTION

Cyber-physical systems are becoming omnipresent and found in applications related to avionics, automative, healthcare, traffic. Complexity of designing user interfaces of cyber-physical systems presents a challenge [11, 14]. Formal modeling of critical interactive systems such as medical infusion pumps, air traffic control is important in unraveling the potential design errors. The feedback loop with human integrated with physical processes are critical in the computations [13]. The interaction of human with the critical systems is a challenge for design of secure and usable interfaces. Formal verification methods have been addressed to ensure correct functioning of interactive computing systems [8]. Interactive systems is a combinatiuon of a human and an automated device. The correct functioning of interactive devices includes correct functioning of the automated device and human actions [8]. The minimization of human errors is critical in system design to prevent a catastrophe. Formal modeling of cognitive errors (slips) in embedded systems such as infusion pumps [4] has been addressed with the aim of designing correct user interfaces. Construction of correct user interfaces of any device is critical to ensure usability of the software. One of the errors in the user-device interaction is the device-delay error. The user performs an action and in the absence of alert mechanism for the completion of the action, the user repeats the action to execute the task again. The device-delay error [7] occurs when a device takes time to process the information entered by the user. An example of device-delay error is the device-delay occuring during the human-machine interaction of an infusion pump. In the infusion pump, the healthcare provider presses the button for a measured dose of drugs. In the absence of alerts, the healthcare provider presses the dosage button a second time thinking that the first execution of the dosage button was not executed by the device. The outcome for repetitive pressing of the button leads to increase of dosage in the infusion pump. The time that a device would take to process the information entered by user is not known beforehand. We address device-delay error in this work and to the best of our knowledge, this is the first time device-delay has been addressed under uncertainty.

Model checking [6] is a formal verification method that has been used to study the correctness of system design. Prior work has addressed model checking of interactive behavior and cognitive errors have been reported [2, 3]. A communicating sequential process(CSP) algebra model [2] and a rewriting-system based formalism [3] has been used to study different behavioral properties of the user. The goal of user-device interaction is to create a model for the verification of user actions integrated with the device processes. The contributions of this paper are: (i) A novel formalism is created using a resource based approach to model user interaction and incorporate device-delay error, (ii) An automated model abstraction method incorporating uncertainty in modeling device-delay and (iii) evaluate computational feasibility of the formalism using probabilistic model checking [1, 10] on a prototype of a vending machine. The goal of the formalism in this paper is to incorporate device-delay error in the formal verification of correct user-interfaces such that potential errors in user interactions. The potential causes of errors in human interaction are from incorrect system design or cognitive slips in human behaviour. A preliminary version of this work was reported [9].

BICT 2017, March 15-16, Hoboken, United States

2 RESOURCE-BASED INTERACTION MODEL

The motivation for the formalism is: the user interacts with the device to use it for fulfill the goals. The *goal* of user-device interaction is the reason for the user to interact with the device. In this paper, we use a vending machine as a running example for user-interaction. The goal for the user-device interaction for the vending machine is to receive products from the vending machine. The products are the *resources* in the vending machine. The *begin* state is the state in which the user begins interaction with the device to achieve his/her goals. The resources in the device decrease/increase from the *begin* state when the user interacts with the device. The resources that are produced/consumed by the device are *internal* resource. There are two types of *internal* resources, *concrete* and *abstract*. *Concrete* internal resources are physical objects such as chocolates , currency and *abstract* internal resources are qualitative objects such as time interval between start and end of an event occuring inside the device. Physical conditions that are not produced/consumed by the device but is changed during the interaction are called *external* resources. An example of an external resource is any physical condition such as temperature; another example includes the presence/absence of number of users or the time taken by users to perform a specific action on the device. External resources are important to address context or the location, where the device is being used. In our formalism in this paper, only *internal* resources are considered. Model abstractions for formal analysis using model checking [6] require formalization of transition systems. Formally,

Definition 2.1. A Kripke transition system $\mathcal{M}$ over a set AP of proposition letters is a tuple $\mathcal{M} = \langle S_0, S, R, L_s\rangle$ where, (i) S is a finite and non empty set of states ,(ii) $S_0 \subseteq S$ is the set of initial states ,(iii) R is a transition relation, $R \subseteq S \times S$ such that for each $s \in S$ there exists $s' \in S$ and $(s, s') \in R$ and (iv) $L_s : S \to 2^{AP}$ is the labeling function that labels $s \in S$ with the atomic propositions that are true in s.

A ede labeled Kripke (E-Kripke) transition system is a Kripke transition system that has labels on the transitions. Formally, $\mathcal{M}_e = \langle S_0, S, R, L_s, L_e\rangle$. where L_e is the edge labeling function given by $L_e : S \times S \to \mathcal{E}$ and $\mathcal{E}$ is the set of edge labels. We describe the representation of user-device interaction in the form of E-Kripke transition system is described as:
We are given (1) a set of resources, $\mathcal{R}esc$, (2) a set of quantities, $0, 1, 2, .., n$, $(n \in \mathbb{N})$ for each resource $resc \in \mathcal{R}esc$ is assigned and (3) a set of actions, $\mathcal{A}ct$. There is a transition from state s to s' modeled by change in the value of a resource for each action, $a \in \mathcal{E}$.

- AP is the set of all the atomic formulas, $resc_i = 0$, $resc_i = 1$, or $resc_i = n$ for ith. resource, $resc_i \in \mathcal{R}esc$ where $n, i \in \mathbb{N}$.
- S is the set of subsets s of AP where, for ith. resource, $resc \in \mathcal{R}esc$, exactly one of the formulas $resc_i = 0$, or $resc_i = 1$, or $resc_i = n$ is in s. The states contain quantities of all the resources in the system
- S_0 is the set of initial states of the E-Kripke structure. An initial state inherently is unique, containing only a single state as the resource level. Hence, $\mid S_0 \mid = 1$.
- The edge label on a transition is an action, e where $e \in \mathcal{A}ct$. The labeled transition is represented by a triple, $\langle s, e, s'\rangle$.
- For all $s, s' \in S$ and an action, $e \in \mathcal{A}ct$ given by: $r\hat{e}sc_1, r\hat{e}sc_2 \ldots r\hat{e}sc_n \overset{e}{\hookrightarrow} r\check{e}sc_1, r\check{e}sc_2, \ldots r\check{e}sc_n$.
  The notation for set of resources for the action, e before and after the action is given by,
  $$\{r\hat{e}sc_1, r\hat{e}sc_2, \ldots, r\hat{e}sc_n\}$$
  and
  $$\{r\check{e}sc_1, r\check{e}sc_2, \ldots, r\check{e}sc_n\}$$
  , respectively.

There are two types of edges in the E-Kripke transition system, r-edge and ϵ-edge. A r(resource)-edge $\langle s, e, s'\rangle$ models a change in the quantity of at least a single resource in an action, $e \in \mathcal{A}ct$. The model is able to accomodate all the actions and each action changes the quantities in a resource. A labeled ϵ-edge models an action that has no change in internal resources. The ϵ-edge represents external resources that has no direct effects on any of the actions that change the quantities of internal resources. The formalism includes only those actions that change internal resource. Also, ϵ-edge models the scenario if the user leaves without the completion of the goals.

3 UNCERTAINTY IN THE FORMALISM

The values of device-delay time is uncertain. Hence, it is imperative that the resource-based interaction model to incoporate uncertainty in the values of device-delay. One way to address uncertainty is by using stochastic models, such as discrete time markov chain (DTMC) and markov decision processes (MDP) [15]. The set of values of device-delay are incorporated in the model and assigned probabilities. The objective is to evaluate the computational feasibility of the incorporated uncertainty in the resource-based formalism.

Definition 3.1. MDP is $\mathcal{M}_m\langle S_0, S_m, \mathcal{A}, P, \mathcal{L}\rangle$ where

- S_m is a finite set of states.
- S_0 is the initial state.
- $\mathcal{A}$ is the finite set of actions. Note:$\mathcal{A}$ can be a set of subsets.
- $P : S_m \times A \times S_m \to [0, 1]$ and $\forall a \in \mathcal{A}, \forall s \in S_m \sum_{s' \in S_m} P(s, a, s') = 1$
- $\mathcal{L} : S_m \to 2^{AP}$. $\mathcal{L}$ represents the labeling function and AP represents is the set of atomic propositions.

The E-Kripke transition system may then be transformed to a stochastic model such as markov decision process (MDP)

and discrete time markov chain (DTMC) is given. A MDP is the generalized form of DTMC where the probability distributions are selected nondeterministically. Assume $\mathcal{M}_e\langle S_0, S, R, L_s, L_e\rangle$ is the E-Kripke transition system. The transformation from a detrministic system to a MDP is denoted as $\mathcal{M}_e$ to $\mathcal{M}_m$ and is constructed by a fixed point algorithm. For simplicity, an action, e in the algorithm is a single action and not a set of actions as stated in the definition of MDP. In the construction of MDP, transition of E-Kripke transition system is transformed into multiple transitions representing same transition with different probability distributions.

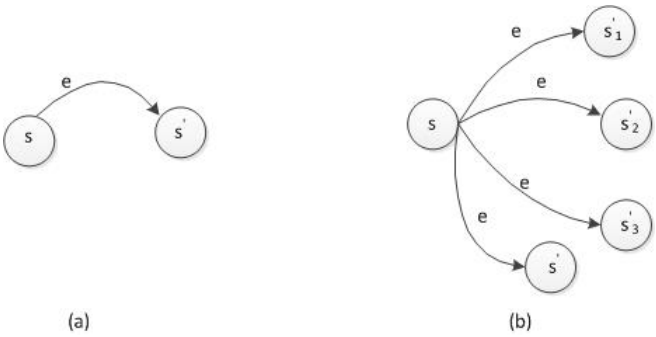

Figure 1: Transformation of a transition in an E-Kripke system to MDP (a) A labeled transition, $s \xrightarrow{e} s'$, in an E-Kripke system (b) A set of transitions formed from a given transition from the E-Kripke system in the construction of MDP.

We introduce the following constructs to address uncertainty in the resource-based formalism to evaluate computational feasibility.

Definition 3.2. (m-restricted state labels) Given a state, s with label, $L(s)$, a m-restricted state labels, $\check{L}_m(s)$ represents m different propositions from $L(s)$ and $m \in \mathbb{N}$.

An example of an m-restricted state label of $L(s) = \{resc_1 = 1, resc_2 = 3, resc_3 = 1\}$, is $\check{L}_1(s) = \{resc_1 = 1, resc_2 = 3, resc_3 = 2\}$. Here, $m = 1$, since only a proposition, $resc_3 = 1$ is changed to $resc_3 = 2$.
The set of m-restricted state labels of $L(s)$ is given by $\check{\mathfrak{L}}_m(s)$. For a 1-restricted state label and given the possible values for resource, $resc_3$ are 2,3 and 4, hence $\check{\mathfrak{L}}_1(s) = \{\{resc_1 = 1, resc_2 = 3, resc_3 = 2\}, \{resc_1 = 1, resc_2 = 3, resc_3 = 3\}, \{resc_1 = 1, resc_2 = 3, resc_3 = 4\}\}$. The number of different values $resc_3$ in $\check{\mathfrak{L}}_1(s)$ can be assigned is k which is 3. k is defined as the *fan-out* number of a m-restricted state label and $k =\mid \check{\mathfrak{L}}_m(s) \mid$. The fan-out number represents the probable values for a resource and hence, address imprecision for a resource after an action.

A r-edge, $s \xrightarrow{e} s'$ in a E-Kripke is transformed to multiple transitions in a MDP. Each of the multiple transitions are of the form, $s \xrightarrow{e} \check{s}, \forall \check{s} \in \check{S}$. $\check{S}$ is the set of states that are 1-restricted labeled states for the labels of the state, s' and includes the state, s'. In Figure 1, the transition in a E-Kripke system is transformed into four transitions from state s to the set of states, $\check{S} = \{s'_1, s'_2, s'_3, s'\}$. A transition from state, s to any state in $\check{S}$ form a transition of MDP. The states represented by in the Figure 1, s'_1, s'_2 and s'_3 are m-restricted state labels of s' with a fan-out number,4 and $m = 1$. A DTMC can be modeled as the four transitions for each fan-out number representing a probability distribution. If more than one transitions are assigned different probability distributions, then it is exclusively MDP where the probability distributions are selected nondeterministically. A fixed point algorithm on the number of states of an E-Kripke system, *AutomatedMDP* automates the construction of MDP. The input of the algorithm is the E-Kripke transition system and number of potential values of device-delay, *TimeDelayInt*. In the model, the restricted label is only for the resource, $resc_i$ that represents device-delay. In Figure 1, *TimeDelayInt* = 3, there are four transitions in the MDP for a single transition in the E-Kripke transition system. The four transitions represent different possible values of device-delay for an action in the model. Each transition in the constructed MDP is assigned a probability. The probabilities on each transition represent the probability that a device-delay occurs for a given action.
The m-restricted labels and the fan-out numbers are used to abstract the uncertain values of device-delay. The increase in the number of transitions and states in the model addresses the potential values of device-delay related to an action.

The algorithm constructs a stochastic model, $\mathcal{M}_m$ for every action in the edge labeled Kripke structure, $\mathcal{M}_e$. The variable, $TimeDelayInt$ is the number of possible device-delay time intervals allowed in the model. The fixed point construction is based on the set of states of $\mathcal{M}_e$. Clearly, $S \subseteq S_m$. The complexity of the algorithm is $O(\mid S^2 \mid) * (\mid E \mid)$.

LEMMA 3.3. *The algorithm terminates after finite number of steps.*

PROOF. There are finite number of steps in the algorithm because there are finite number of states in $\mathcal{M}_e$. For each state, $s \in S$, there are finite number of edges representing actions. □

LEMMA 3.4. *There is transition in $\mathcal{M}_e$ iff there exists a transition in $\mathcal{M}_m$.*

PROOF. Assume there is a transition, (s, s') in $\mathcal{M}_e$. The transition, s, s' represents an action, e. By construction, there is atleast a transition representing action, e in $\mathcal{M}_m$ starting from s to a state, $\check{s}$ with a probability distribution, P such that $\forall \check{s} \in \check{S} \sum_{\check{s} \in \check{S}} P(s, e, \check{s}) = 1$ where $\check{S}$ is the set of states that has a transition from s for action, e.

Case a: $\check{s} = s_0$.
There are two transitions, an r-edge and ϵ-edge. The r-edge represents the change in the resources from the final state to the initial state. Hence, $\mid \check{s} \mid = 2$.

Case b: $\check{s} \neq s_0$.
There are $TimeDelayInt + 1$ transitions for given by

Algorithm 1 AutomatedMDP

Input: Edge labeled Kripke transition system $\mathcal{M}_e\langle S_0, S, R, L_s, L_e\rangle$, *TimeDelayInt* is the number of possible time values.
Output: MDP, $\mathcal{M}_m\langle S_0, S_m, \mathcal{A}, P, L\rangle$
$S_0 = \{s_0\}, S' = \{s_0\}, \hat{S} = \{s_0\}, S_m = \emptyset$;
while $(\hat{S} \neq S)$ **do**
$\hat{S} = S$
for each $s \in \hat{S}$ **do**
for each action, $e \in E$ starting from $s, s \xrightarrow{e} s'$ **do**
if $s' = s_0$ **then**
$\check{S} = \{s_0, s\}$. {If the final state is the initial state, then there are two edges, one ϵ edge and the other, a r-edge}
else
$\check{S} = \{\check{s} : \forall \check{s}, L(\check{s}) \in \mathfrak{L}(s')$ where $\mathfrak{L}(s') = \check{\mathfrak{L}}(s') \cup L(s')\}$.
end if
Construct a r-edge, $s \rightarrow \check{s}$ where $\check{s} \in \check{S}$.
Construct an ϵ-edge, $s \rightarrow s$.
$S_t = \check{S} \cup s$,$\exists P$ where P is a probability distribution such that $\forall s_t \in S_t \sum_{s'_t \in S_t} P(s_t, e, s'_t) = 1, \mid \check{S} \mid =$ *TimeDelayInt* $+ 1$. {$\check{S}$ is the set of states that form the MDP from s}
$S' = \{s' : s \rightarrow s'\}$
end for
end for
$S_m = S_m \cup \check{S}$ {Set of States in $\mathcal{M}_m$}
$S = S \cup S'$
end while
$S_\infty = \hat{S}$

TimeDelayInt r-edges and an ϵ edge. Hence, $\mid \check{s} \mid =$ *TimeDelayInt* $+ 1$.

Conversely, pick any state, $\check{s} \in S_m \setminus S_0$ in $\mathcal{M}_m$.

Case a: $\check{s} = s_0$.
For an action, e find s such that $s \rightarrow \check{s}$ and there exists a probability distribution, P such that $\forall \check{s}, s \rightarrow \check{s}, \sum_{\check{s} \in \check{S}} P(s, e, \check{s}) = 1$, $\check{S}$ is the set of states from s for a given action, e. For the action, e starting from state s , then there exists a state $s' \in \check{S}$ such that $s \rightarrow s'$. Hence, (s, s') is a transition in $\mathcal{M}_e$ for action e.
Similar reasoning is given for any transition with action, ϵ.

Case b: $\check{s} \neq s_0$.
Similar reasoning given for case a.

□

4 CASE STUDY: A VENDING MACHINE

Cerone et al [5] used a chocolate machine as a case study of their model. The vending machine contains different objects such as candies and food for selection. The quantities for the objects can be selected by the user. The vending machine operates when money (currency) is inserted and dispenses the objects from the vending machine that was selected by the user. In our formalism, we do not allow the quantities of the objects to be selected rather provide a mechanism that any object is selected. Here money (currency), objects and device-delay are *internal* resources. The time taken by user to perfom interaction after a completion of a subgoal and the beginning of the next subgoal, *user-time* is an *external* resource. The set of actions that changes the values of resources are vending machine are accept currency, select quantities (of objects in the vending machine), dispense change and dispense resource (selection from the vending machine). We state the difference in the value of the resources that change with any of the following action. In our model, there are three types of internal resources for the vending machine. The resources are (i) currency (money) (ii) objects (to be selected) and (iii) device-delay. Here, *objects* are tangible resources other than currency to be selected from the vending machine. The actions that are modeled to accomplish the goal of a user are given by:

(1) *Accept Currency:* changes the quantity of the currency in the vending machine. Currency is added to the device and the quantity of currency is increased from the initial state of zero.
(2) *Select Quantities* is an action which represents the selection of the objects.
(3) *Dispense Change:* is the action when the change in the form of currency is given back to the user. This action reduces the currency in the device.
(4) *Dispense Resource:* is the action that constructs when there is decrease in the number of objects in the vending machine with the dispense of a object to a lower level than previous level.
(5) *Begin:* is the action when the vending machine returns to the initial state after the dispense of the objects before the another user can start interacting with the system. It is the state where the out-edge is labeled as *Accept Currency*

Device-delay occurs in each of the four aforementioned actions. Each action takes a time interval for completion. The construction of the state machine starts from the initial state, s_o representing an initial value of currency and objects in the vending machine. The state machine uses high value of currency and low value of objects to represent increase and descrease of currency and objects, respectively. A transition, (s, s') is constructed with the action *Accept Currency* where the currency on the state, s' is increased to a high value. The next transition is constructed with the representation of the change of device-delay for the action *Select Quantities*. We assume there are adequate quantites of objects to be selected and the amount of currency entered is adeuquate for the purchase. With the action *Dispense Change*, there is a change in value of currency in the machine. Once the user takes the change, the objects are dispensed and the value of the objects in the machine becomes $low - value$. *Dispense*

Resource is the action that shows the value(number) of the objects decrease with the user receiving the objects. The action, *Begin* creates a transition back to the initial state for the next usage of the machine and there is change of the objects back to the initial state. The ϵ-edge (self loop) on each state represents the external resources such as the user leaving without the completion of his goals.

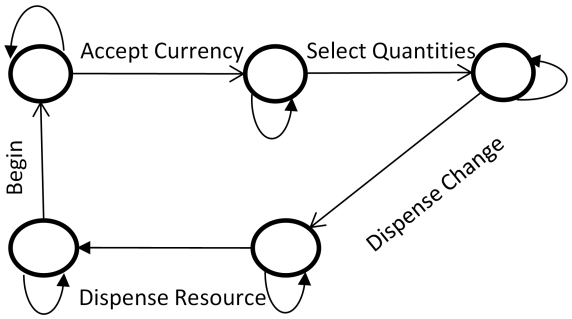

Figure 2: A finite state machine representing user-device interaction in a vending machine

The objective of the simulation of the formalism is to get an estimate of the computational feasibility of the model and how the formalism scales in deterministic model as in DTMC and non deterministic model represented by MDP. The increase in the size of the problem is represented by increasing the $fan-out$ number of a propositions. The number of resources are represented symbolically. In the model, only one proposition changes its values, hence it is 1-restricted state label. The implementation of the resource based model was constructed using PRISM (www.prismmodelcheker.com) [12] on a Sun machine with processor of 502 Mhz with 1152 Mb memory. The sample PCTL queries were posed on the DTMC and MDP and the times for execution were recorded. The currency is user input that is the amount of money that is entered in the vending machine. The objects or commodities are in the vending machine that is sought by the used. In the model, there are two boolean variables for currency, $curr$ and $high-currency$. The $objects$ and $low-objects$ represents the internal tangible resources. In the model, we assume there are adequate objects for the user to select from the vending machine. The initial state contains three boolean variables, $curr$, $object$ and $init-time$ to be true. The $init-time$ represents the start time for the interaction of the user and the vending machine. The $high-currency$ boolean value occurs in the model for the action, *Accept Currency*. The $curr$ is true when $high-currency$ is false with the action, *Dispense Change*. Simlarly, the $object$ is true initially and then it becomes false when the $low-object$ is true with the action, *Dispense Resource*. For completion of each action, there are n boolean values representing $device-delay$. Hence, each action from a state has $n+1$ transitions representing n boolean values for time and one self loop. In the DTMC model, there is a distribution on the transitions from a state. In our simulation of DTMC, we assume the probabilities on the transitions are equally likely for a specific device-delay time and hence, the probability on each transition is $\frac{1}{p}$ where p is the number of transitions coming out of a state. The assumption of equally likely is drastic simplication to reality but it is a way to incorporate potential values of device-delay. We use the assumption equally likely events because the objective of the simulation is to evalute the computational feasibility of the model. The MDP model incorporates nondeterminism and the self loop on the beginning state of the action has a probability distribution different than for other transitions. The size of the simulation model is increased with the increase of number of boolean values for $device-delay$. For example, if there are $five$ boolean values representing $device-delay$, it implies that an action completes itself after one of five boolean variables representing $device-delay$. A $device-delay$ that takes values t_1, t_2, t_3, t_4 and t_5 seconds meaning each action can complete in any of the times. Therefore, there are 5 transitions to different states from the beginning state of the action. The fan-out number for $device-delay$ are $5, 10, 15$ and 20. The following PCTL queries were posed on the model:

Query 1. Compute the maximum probability to reach a state where currency is at high level. The PCTL translation for the query is $Pmax? = [true\ \mathbf{U}\ (high-currency = true)]$.

Query 2. Compute the probability that either "curr = false " or "object = false" at some point. The PCTL formula for the query is $P =?[\mathbf{F}(currency = false)|(object = false)]$.

Query 3. The probability that high-currency is true is less than 0.1. The PCTL formula, $P_{<0.1}[\mathbf{F}(high - currency = true)]$.

Query 4. From an initial state, the probability that *low-object* is true before *object* is true is greater than equal to 0.99. PCTL formula, "$init$" $=> P_{\geq 0.99}[(low-object = true)\ \mathbf{U}\ (object = true)]$

Query 5.The probability that *low-object* = true is within 3 steps from *initial time*, $init-time = true$ is atleast 0.98. The PCTL formula, $P_{\leq 0.98}[(init - time = true)\ \mathbf{U}^{<=3}(low - object = true)]$.

The results in the Table 1 elucidate that the formalism is computational feasible for the prototype with larger problem sizes. All of the models required less than two minutes for model construction and model checking. Clearly, the times for DTMC model construction for different sizes of the model was significantly efficient than MDP model construction. The nondeterminism in the MDP models made computationally intensive in comparison to the DTMC models. Additionally, some of the interesting user design properties can be expressed in PCTL queries:

(1) Completion of the goal of the user: The PCTL query, "$init$" $=> P_{\geq 1}[\mathbf{F}(init - time = true)]$ represents if an interaction has started, then the probability of reaching or culmination is greater 0.99. The query

Query	*DTMC*				*MDP*			
	Fan-out numbers							
	5	10	15	20	5	10	15	20
1.	0.284	1.067	3.75	30.747	0.967	6.72	30.652	104.738
2.	0.67	1.15	3.704	30.654	0.982	6.743	30.684	104.9
3.	0.254	1.147	3.701	30.308	0.969	6.858	30.393	106.314
4.	0.276	1.14	3.618	30.687	0.967	6.664	30.61	104.2
5.	0.246	1.139	3.596	29.646	0.984	6.757	30.597	103.813

Table 1: Execution times(in seconds) for PCTL queries on DTMC and MDP models.

checks if the initial state represented by $init - time$ can be reached after the actions.

(2) Ordering of the actions by the user : The formula $P_{\geq 1}[(low{-}currency = true)\mathbf{U}(low{-}object = true)]$ represent the change is dispensed first before the object from the vending machine. The value of currency is low before the value of object is low.

Query (1) is useful for the system designers to evaluate whether the goal(s) can be accomplished by the user. If the probability is low, then the system is potentially faulty and would need to be redesigned. Query (2) evaluates sequence of actions by the user to accomplish the goal(s). The query evaluates whether the end of a action or a sequence of actions, accomplishes a goal. For example, if the change is dispensed later than the retrieval of the object, there is potential that the user may not take the change because the goal of the user interaction was to retrieve the object from the vending machine. The query provides insights of the sequence of actions by the user to minimize error.

5 CONCLUSION AND FUTURE WORK

In this work, a formal framework model to study device-delay errors during user-device interaction in an interactive system such as vending machine has been proposed. The computational feasibility (scalability) of the formalism is evaluated on different problem sizes represented by introducing the number of possible device-delay values. The stochastic modeling is a way to incorporate uncertainty. The probabilities on the transitions is an important feature in modeling interactive systems because computation of probability of certain events cannot provide enough information for user interface designers to correct devic-delay errors. Particularly, this analysis would be invaluable even when there is small chance of error that can potentially lead to a catastrophe. We seek also to propose a way where the probabilities are better estimated on the transitions of the model using bayesian models. In future, a rigorous evaluation of the resource based formalism on an interactive system such as infusion pump will be performed.

REFERENCES

[1] A. Aziz, V. Singhal, F. Balarin, R. Brayton, and A. Sangiovanni-Vincentelli. 1995. It usually works: The temporal logic of stochastic systems. In *Computer Aided Verification*. Springer, 155–165.

[2] T.A. Basuki. 2007. Model Checking Interface Design to Reduce User Errors. In *The Pre-proceedings of the 2nd International Workshop on Formal Methods for Interactive Systems (FMIS 2007)*. 1.

[3] T.A. Basuki, A. Cerone, A. Griesmayer, and R. Schlatte. 2009. Model-checking user behaviour using interacting components. *Formal aspects of computing* 21, 6 (2009), 571–588.

[4] J.C. Campos and M.D. Harrison. 2011. Modelling and analysing the interactive behaviour of an infusion pump. In *Proceedings of the Fourth International Workshop on Formal Methods for Interactive Systems. EASST, Potsdam.*

[5] A. Cerone and P. Curzon. 2008. Formal methods for interactive systems. *Innovations in Systems and Software Engineering* 4, 2 (2008), 123–123.

[6] E.M. Clarke, O. Grumberg, and D. Peled. 1999. Model checking. (1999).

[7] P. Curzon and A. Blandford. Detecting multiple classes of user errors. In *Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction(EHCI'01)*.

[8] Paul Curzon, Rimvydas Rukšėnas, and Ann Blandford. 2007. An approach to formal verification of human–computer interaction. *Formal Aspects of Computing* 19, 4 (2007), 513–550.

[9] Krishnendu Ghosh. 2012. Formal Analysis of Device-Delay Error in User-Device Interaction under Uncertainty. In *Software Reliability Engineering Workshops (ISSREW), 2012 IEEE 23rd International Symposium on*. IEEE, 23–24.

[10] H. Hansson and B. Jonsson. 1994. A logic for reasoning about time and reliability. *Formal aspects of computing* 6, 5 (1994), 512–535.

[11] Bernhard Kölmel, Rebecca Bulander, Uwe Dittmann, Alfred Schätter, and Günther Würtz. 2014. Usability Requirements for Complex Cyber-Physical Systems in a Totally Networked World. In *Working Conference on Virtual Enterprises*. Springer, 253–258.

[12] M. Kwiatkowska, G. Norman, and D. Parker. 2011. PRISM 4.0: Verification of Probabilistic Real-time Systems. In *Proc. 23rd International Conference on Computer Aided Verification (CAV'11) (LNCS)*, G. Gopalakrishnan and S. Qadeer (Eds.), Vol. 6806. Springer, 585–591.

[13] Edward A Lee. 2008. Cyber physical systems: Design challenges. In *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*. IEEE, 363–369.

[14] Volker Paelke and Carsten Röcker. 2015. User Interfaces for Cyber-Physical Systems: Challenges and Possible Approaches. In *International Conference of Design, User Experience, and Usability*. Springer, 75–85.

[15] M.L. Puterman. 1994. *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc.