Probabilistic CCRN: Reliability Analysis of Ubiquitous Computing Scenarios Using Probabilistic Model Checking

Reona Minoda, Masakazu Ishihata and Shin-ichi Minato

Graduate School of Information Science and Technology
Hokkaido University
Sapporo, Hokkaido 060–0814, Japan
Email: minoda@meme.hokudai.ac.jp, {ishihata.masakazu, minato}@ist.hokudai.ac.jp

***Abstract*—This paper proposes the method of reliability analysis of ubiquitous computing (UC) scenarios. In UC scenarios, various devices communicate with each other through wireless network, and this kind of communications sometimes break due to external interferences. To discuss the reliability in such situation, we introduce the notion of probability into *context catalytic reaction network* (CCRN), which is a description model of UC scenarios. This enables us to conduct quantitative analyses such as considerations of a trade-off between the reliability of UC scenarios and the costs which may be necessary for their implementations. To conduct a reliability analysis of UC scenarios, we use the technique of probabilistic model checking. We also evaluate our method experimentally by conducting a case study using a practical example assuming a museum.**

Keywords–Ubiquitous Computing; Catalytic Reaction Network; Probabilistic Model Checking; Smart Object.

I. INTRODUCTION

Nowadays, we are surrounded with various kinds of devices with computation and communication capabilities and we carry these devices every day. In this paper, we call these devices *"smart objects (SO)"*. SOs include personal computers (PCs), mobile phones, sensor devices, embedded computers and radio frequency identifier (RFID) tags. But we can also treat physical things like foods, medicine bottles and cups as SOs by embedding RFID tags in those. Here, we use the term *federation* to denote the definition and execution of interoperation among resources that accessible either through the Internet or through peer-to-peer ad hoc communication. For example, let us consider that there are a phone, a medicine bottle and food, and RFID tags are embedded in a medicine bottle and food. Imagine that this food and the medicine have a harmful effect when eaten together. If all these things are close to each other, a phone rings to inform a user to *warn not eat them together*. This phenomenon is a federation. Indeed, we can also consider federations related to other SOs and these federation may be involved in each other. We call these federation *"ubiquitous computing application scenarios (UC scenario)"* (see Figure 1).

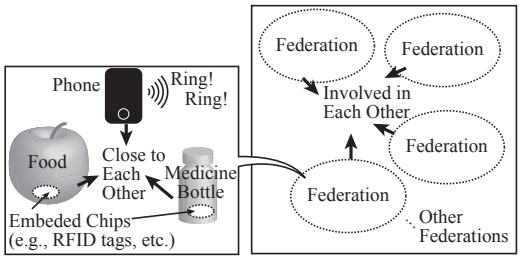


Figure 1. Example of Ubiquitous Computing Application Scenario

In our previous works, we showed an approach to verify UC scenarios by proposing context catalytic reaction network (CCRN) and its verification through model checking [1]. We also proposed more efficient method of this kind of verifications through symbolic model checking [2]. These contributions enabled us to verify the property of UC scenario, which is described in formal logic formulation and these verifications are conducted systematically thanks to various model checking verifiers, such as NuSMV2 [3].

However, there are still challenges of our approach. For example, UC scenarios are assumed that SOs typically communicate with each other by the wireless communication. This means that we need to consider these communications sometimes break due to various external causes. For this reason, it is important to discuss this kind of interference formally. In this paper, we show an approach to reliability analysis of UC scenarios by introducing a notion of probability to CCRN, which is a description model of UC scenarios. To analyze this kind of reliability, we use the technique of probabilistic model checking [4].

The rest of this paper is organized as follows. Section 2 introduces related works of our research. Section 3 provides preliminaries of this paper, such as basic definitions and notations including CCRN and probabilistic model checking. Using them, we introduce a notion of probability to CCRN in Section 4. In Section 5, we propose the method of reliability analysis of probabilistic CCRN using probabilistic model checking. Then, we evaluate our approach by conducting the case study assuming a practical scenario in Section 6. Finally, we conclude our contributions in Section 7.

II. RELATED WORKS

In this section, we introduce related works of our work.

A. Reconfigurable hardware verification for Ubiquitous Systems

In the field of implementations for ubiquitous systems, Guellouz, et al. use probabilistic model checking to verify the behavior of devices [5]. These devices have reconfigurable function blocks, which is standardized as IEC 61499 and a

 ISBN: 978-1-61208-598-2

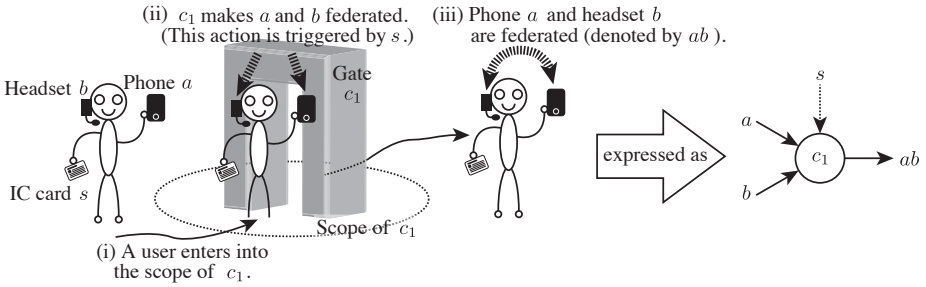


Figure 2. Example of a Catalytic Reaction

part of each blocks behaves probabilistically. Guellouz, et al. analyzed this behavior by using probabilistic model checking.

B. Formal Verification of Cyber Physical Systems

Similarly to ubiquitous computing, a lot of devices, such as sensors measure physical phenomena, such as temperature, humidity, acceleration and so on, while actuators manipulate the physical world, like in automated robots. The combination of an electronic system with a physical process is called cyber physical system (CPS). In the field of CPS, Drechsler and Kühne use *timed automata* [6] as a state transition model to conduct formal verifications of given systems' properties [7].

C. Context Inconsistency Detection

In the field of ambient computing, Xu and Cheung propose a method of context inconsistency detection [8]. This method detects inconsistencies from a series of gathered events, such as "a user entered a room" and "the temperature of room is 30°C" by logical deduction. Unlike a formal verification, this method can be applied only after the system begins to work. Instead, a formal verification can find the failed cases from a given system *in advance*.

III. Preliminaries

In this section, we give definitions and notations which is necessary for this paper.

A. Basic Definitions and Notation

Let X and Y be any two sets, we use $\mathrm{X} \cup \mathrm{Y}$, $\mathrm{X} \cap \mathrm{Y}$ and $\mathrm{X} \setminus \mathrm{Y}$ to denote the union, intersection and difference of X and Y respectively. For a set X, we denote its power set (i.e., all subsets) by 2^{X} and its cardinality by $|\mathrm{X}|$. For a set X, we denote a set of k-elements subsets of X by $\binom{\mathrm{X}}{k}$. For a family M of sets (i.e., a set of sets), we denote the union and the intersection of all sets in M by $\bigcup \mathrm{M}$ and $\bigcap \mathrm{M}$ respectively. Let X be a set, we denote a set of all set partitions of X by $\mathfrak{P}(\mathrm{X})$. For example, given $\mathrm{X} = \{1, 2\}$, we have $\mathfrak{P}(\mathrm{X}) = \{\{\{1\}, \{2\}\},\ \{\{1, 2\}\}\}$.

B. Catalytic Reaction Network

A catalytic reaction network is originally proposed by Stuart Kauffman in the field of biology to analyze protein metabolism [9]. Based on this model, Tanaka applied it to the field of ubiquitous computing as the way to describe an application scenario involving mutually related multiple federations among SOs [10]. In this paper, we mean the latter by the term "catalytic reaction network".

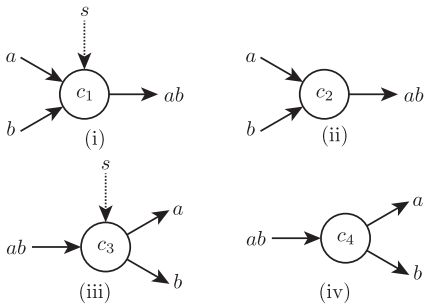


Figure 3. Four Types of a Catalytic Reactions

A catalytic reaction network is a set of catalytic reactions. Each catalytic reaction takes input materials and transforms them into output materials. And each catalytic reaction has a catalyst which is called *context*. It may be also possible to include a catalyst in input materials. We call this kind of catalyst *stimulus*. A catalytic reaction is occurred when all required SOs are in the proximity of each other. We use the term "*scope*" to denote the inside of the proximity area (we assume a range of Wi-Fi radiowave, and so on). The scope of a SO o is represented as a set of SOs which are accessible from the SO o. We assume that only the scopes of contexts are considered instead. In other words, we consider that the catalytic reaction is occurred if all required SOs just enter into the scope of the corresponding context.

Figure 2 shows an example of single catalytic reaction. In this example, there is a gate c_1 regarded as a context and a user has three SOs, i.e., a phone a, a headset b and an IC card s. If the user enters into the scope of c_1, c_1 makes a and b federated. This action is triggered by s. After that, phone a and headset b are federated. We denote federated SOs, such as a and b by a concatenation of a and b, i.e., ab. During this process, c_1 and s work as catalysts. In particular, s is a stimulus in this reaction. We express this reaction as the right hand side diagram of Figure 2.

In catalytic reaction networks, there are four types of catalytic reactions as we show in Figure 3. We categorize these four types of reactions into two groups. One group is the *composition* reaction group (Figure 3 (i) and (ii)), the other group is the *decomposition* reaction group (i.e., Figure 3 (iii) and (iv)). A catalytic reaction of Figure 2 is a type (i) catalytic reaction. We also consider the catalytic reaction without a stimulus, such as Figure 3 (ii). In type (ii), if a user who has SO a and SO b enters into the scope of context c_2,

 ISBN: 978-1-61208-598-2

c_2 makes a and b federated *without a stimulus*. In a similar way, we consider the decomposition reactions, such as Figure 3 (iii) and (iv). In type (iii), if a user who has two SOs that are federated into ab enters into the scope of context c_3, c_3 decomposes these SOs ab into a and b triggered by SO s. Type (iv) is a decomposition reaction without a stimulus.

The output SO of a reaction may promote other reactions as a stimulus or become an input SO of other reactions. In this way, catalytic reactions form a network of reactions.

Now we define these notions, contexts and SOs, formally. Basically, we regard that there are two types of SOs. One is a set of SOs which can federate with other smart objects denoted by O and the other one is a set of SOs which can only promote SOs in O to federate with each other denoted by C. Latter in this paper, SOs $c \in \mathrm{C}$ are called "context" and SOs $o \in \mathrm{O}$ are called just "SO" for convenience. We denote contexts and their corresponding reactions by c_i and r_i respectively.

Next, we define catalytic reactions formally by following definition

Definition 1 (Catalytic Reaction): Let O be a set of SOs, a catalytic reaction r is defined as an action tuple $\langle pre(r), add(r), del(r)\rangle$. Every catalytic reaction r satisfies following conditions:

- $pre(r) \subseteq 2^{\mathrm{O}}$, $add(r) \subseteq 2^{\mathrm{O}}$ and $del(r) \subseteq 2^{\mathrm{O}}$,
- $pre(r)$, $add(r)$ and $del(r)$ are pairwise disjoint sets respectively,
- $del(r) \subseteq pre(r)$, and
- $\bigcup add(r) = \bigcup del(r)$.

$pre(r)$, $add(r)$ and $del(r)$ of catalytic reaction r correspond to the precondition for the reaction application, the addition of federation after the reaction, and the deletion of federation after the reaction respectively. We give some examples of catalytic reactions. Given $\mathrm{O} = \{a, b, s\}$, a catalytic reaction of Figure 3 (i) and (iii) can be defined by $r_1 \triangleq \langle\{\{a\}, \{b\}, \{s\}\}, \{\{a, b\}\}, \{\{a\}, \{b\}\}\rangle$ and $r_3 \triangleq \langle\{\{a, b\}, \{s\}\}, \{\{a\}, \{b\}\}, \{\{a, b\}\}\rangle$ respectively. Note that Definition 1 is more general definition of catalytic reactions compared to four types of catalytic reactions in Figure 2. If we represent only these catalytic reactions by this definition, we just set cardinality constraints of $pre(r)$, $add(r)$ and $del(r)$. If r is a composition reaction (i.e., Figure 2 (i) and (ii)), r should satisfy $|pre(r)| = 2$ or 3, $|add(r)| = 1$ and $|del(r)| = 2$; otherwise, if r is a decomposition reaction (i.e., Figure 2 (iii) and (iv)), r should satisfy $|pre(r)| = 1$ or 2, $|add(r)| = 2$ and $|del(r)| = 1$.

Finally, a catalytic reaction network is defined as follows:

Definition 2 (Catalytic Reaction Network): A catalytic reaction network R is a set of catalytic reactions.

C. Context Catalytic Reaction Network

This section describes a segment graph and a CCRN.

1) Segment Graph: As we discussed in previous section, a catalytic reaction is occurred when required SOs enter into the scope of the corresponding context. To analyze the property of a given catalytic reaction network as a state transition system, it is necessary to formalize the movement of SOs. For example, in Figure 4 (i), there are contexts c_1 and c_2 and these scopes have an *overlap*. A user can walk around the path $\alpha\beta$ shown

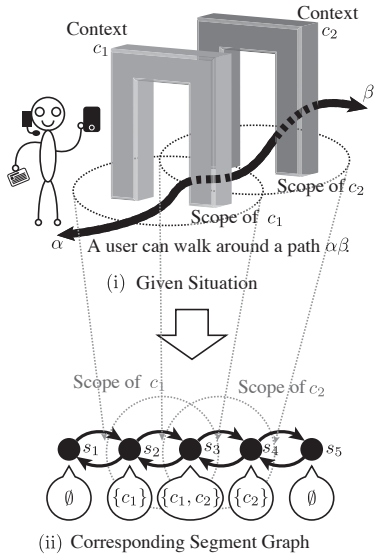


Figure 4. Example of Segment Graph

in Figure 4 (i). This situation can be represented as a segment graph shown in Figure 4 (ii). We consider that the user walk around this segment graph and the user is always located at one of the nodes of this segment graph. Each node of a segment graph has a corresponding set of scopes of contexts. In this way, the given situation like Figure 4 (i) including overlaps of scopes of contexts can be represented as a discrete structure. Now we define a segment graph as follows.

Definition 3 (Segment Graph): Let C be a set of contexts, a segment graph $\mathbf{G}$ is a tuple $(\mathrm{S}, \mathrm{E}, F)$, where

- S is a finite set of segments,
- $\mathrm{E} \subseteq \mathrm{S} \times \mathrm{S}$ is a set of directed edges between two segments, and
- $F : \mathrm{S} \to 2^{\mathrm{C}}$ is a function returning scopes of contexts at corresponding segments.

2) Context Catalytic Reaction Network: A context catalytic reaction network (CCRN) is a discrete structure of a situation involving SOs in a catalytic reaction network. A CCRN is defined as a conbination of a segment graph and a catalytic reaction network.

Definition 4 (Context Catalytic Reaction Network): Let O be a set of SOs, a CCRN $\mathbf{C}$ is a tuple $(\mathrm{R}, \mathbf{G}, f_i, l_0)$, where

- R is a set of catalytic reactions (i.e., CRN),
- $\mathbf{G}$ is a segment graph $(\mathrm{S}, \mathrm{E}, F)$,
- $f_i \subseteq \mathrm{O}$ is a set of SOs fixed to segment $s_i \in \mathrm{S}$, and
- $l_0 \in \mathrm{S}$ is the initial segment locating mobile SOs (mobile SOs can be represented as $\mathrm{O} \setminus \bigcup f_i$).

D. Model Checking

A model checking is a method to verify a property of a state transition system. It has been often used in various fields, which ranges from electronic-circuit-design verification [11] to secure-network-protocol (e.g., Secure Sockets Layer (SSL) protocol) design verification [12]. In the model checking, it is typically assumed to use a Kripke structure as a state transition system. The property of a Kripke structure is described by a

modal logic. There are two kind of commonly used modal logics, such as *linear temporal logic (LTL)* and *computational tree logic (CTL)*. In this paper, we use LTL to describe the property of the Kripke structure.

1) Kripke Structure: Before we look on the detail of a model checking, we give the definition of a Kripke structure [13], which is necessary for a modal logic and a model checking.

Definition 5 (Kripke Structure): Let *AP* be a set of atomic propositions, *a Kripke structrue* M is a tuple (S, I, R, L), where

- S is a finite set of states,
- $I \subseteq S$ is a set of initial states,
- $R \subseteq S \times S$ is a set of transition relation such that R is left-total, i.e., $\forall s \in S,\ \exists s' \in S$ such that $(s, s') \in R$, and
- $L : S \to 2^{AP}$ is a labeling function.

2) Linear Temporal Logic: Linear temporal logic (LTL) is one of the most well-known modal logic. LTL was first proposed for the formal verification of computer programs by Amir Pnueil in 1977 [14]. First, we give a definition of LTL syntax.

Definition 6 (Linear Temporal Logic Syntax): Let *AP* be a set of atomic propositions, a linear temporal logic formula ϕ is defined by the following syntax recursively.

$$\phi ::= \top \mid \bot \mid p \mid \neg\phi \mid \phi \vee \phi \mid \mathbf{X}\ \phi \mid \mathbf{G}\ \phi \mid \mathbf{F}\ \phi \mid \phi\ \mathbf{U}\ \phi \quad (1)$$

where $p \in AP$.

These right-hand terms denote true, false, p, negation, disjunction, next time, always, eventually and until respectively.

Next, we define a transition path π of a Kripke structure M.

Definition 7 (Transition Path): Let M be a Kripke structure, $\pi = (\pi_0, \pi_1, \pi_2, \dots)$ is a transition path in M if it respects M's transition relation, i.e., $\forall i.(\pi_i, \pi_{i+1}) \in R$. π^i denotes π's ith suffix, i.e., $\pi^i = (\pi_i, \pi_{i+1}, \pi_{i+2}, \dots)$.

Also it can be shown that

$$\begin{aligned}(\pi^i)^j &= (\pi_i, \pi_{i+1}, \pi_{i+2}, \dots)^j \\ &= (\pi_{i+j}, \pi_{i+j+1}, \pi_{i+j+2}, \dots) \\ &= \pi^{i+j}. \end{aligned} \quad (2)$$

Now, we focus on the semantics of linear temporal logic. First, we define the binary satisfaction relation, denoted by $\models$, for LTL formulae. This satisfaction is with respect to a pair – $\langle M, \pi \rangle$, a Kripke structure and a transition path. Then, we enumerate LTL semantics as follows:

- $M, \pi \models \top$ (true is always satisfied)
- $M, \pi \not\models \bot$ (false is never satisfied)
- $(M, \pi \models p)$ iff $(p \in L(\pi_0))$ (atomic propositions are satisfied when they are members of the path's first element's labels)

And there are two LTL semantics of boolean combinations as follows:

- $(M, \pi \models \neg\phi)$ iff $(M, \pi \not\models \phi)$
- $(M, \pi \models \phi \vee \psi)$ iff $[(M, \pi \models \phi) \vee (M, \pi \models \psi)]$

And there are four LTL semantics of temporal operators as follows:

- $(M, \pi \models \mathbf{X}\ \phi)$ iff $(M, \pi^1 \models \phi)$
- $(M, \pi \models \mathbf{F}\ \phi)$ iff $\left[\exists i.(M, \pi^i \models \phi)\right]$
- $(M, \pi \models \mathbf{G}\ \phi)$ iff $\left[\forall i.(M, \pi^i \models \phi)\right]$
- $(M, \pi \models \phi\ \mathbf{U}\ \psi)$ iff $\left[(\exists i.(M, \pi^i \models \psi)) \wedge (\forall j < i.(M, \pi^j \models \phi))\right]$

3) Model Checking Problem: Intuitively saying, a model checking problem is to judge whether a given Kripke structure M satisfies a given property described in a modal logic formula ϕ. A model checking problem is formally stated as follows.

Definition 8 (Model Checking Problem): Given a desired property described by a modal logic formula ϕ (in this paper, we use LTL) and a Kripke structure M, a model checking problem is a decision problem whether the following formula

$$\forall \pi.(M, \pi \models \phi) \quad (3)$$

is satisfied or not. Note that a set $\{\pi \mid (M, \pi \not\models \phi)\}$ is particularly called a set of *counterexamples*.

It is known that a model checking problem can be reduced to a graph search if M has finite states.

There are several implementations of the model checking verifier, such as **S**imple **P**romela **IN**terpreter (SPIN) [15] and **L**abel **T**ransition **S**ystem **A**nalyzer (LTSA) [16].

E. Probabilistic Model Checking

Probabilistic model checking is another method of reliability analysis for given Kripke structure. In original model checking, the reachability between two states of Kripke structure is defined as the existence of the directed edge between these states. Instead, in probabilistic model checking, the reachability between two states of Kripke structure is represented as a probability. This probability of the reachability is normalized with respect to each states as follows

$$\sum_{s', (s,s') \in R} P(s' \mid s) = 1 \text{ for all } s \in S. \quad (4)$$

1) Probabilistic LTL: Probabilistic LTL is extended property description language of LTL for probabilistic model checking. In probabilistic LTL, temporal operators $\mathbf{G}$ and $\mathbf{F}$ has an additional bound parameter k denoted by $\mathbf{G}_{\leq k}$ and $\mathbf{G}_{\leq k}$. These temporal operators have following semantics:

- $(M, \pi \models \mathbf{F}_{\leq k}\ \phi)$ iff $\left[\exists i \leq k.(M, \pi^i \models \phi)\right]$
- $(M, \pi \models \mathbf{G}_{\leq k}\ \phi)$ iff $\left[\forall i \leq k.(M, \pi^i \models \phi)\right]$

To discuss the probability of the transition path, probabilistic LTL also has quantitative operator $\mathbf{P}_{=?}$. Given a LTL property ϕ, $\mathbf{P}_{=?}\ \phi$ evaluates the existence probability of transition paths which satisfies the LTL property ϕ.

2) Probabilistic Model Checking Problem: Intuitively saying, a probabilistic model checking problem evaluates the existence probability of transition paths with length k which satisfies the property ϕ described by probabilistic LTL. This transition assumes discrete-time Markov chain. A probabilistic model checking problem is defined as follows:

Definition 9: Given a Kripke structure M, $\tau_{s,s'} = P(s' \mid s)$, a probabilistic LTL ϕ and a length of bound k, a

probabilistic model checking problem evaluates the following probability:

$$\sum_{\forall \pi.(M,\pi \models \phi \wedge |\pi|=k)} \prod_{t=1}^{k} P(\pi_t \mid \pi_{t-1}) \tag{5}$$

One of the most famous implementation of probabilistic model checkers is Probabilistic Symbolic Model Checker (PRISM) [4]. In this paper, we use PRISM to conduct a case study in Section 6.

IV. Probabilistic CCRN

In this section, we propose probabilistic CCRN (P-CCRN), which is the extended model of CCRN by adding a notion of probability. To introduce a notion of probability, we consider two kinds of probability in CCRN. One is the probability of a user behavior, and the other one is the probability of each of catalytic reactions. We denote the probability of users' moving from segment i to segment j by $\tau_{i,j}$ and for all segments $i \in \mathrm{S}$ of a given segment graph, it is satisfied that

$$\sum_{j,(i,j)\in E} \tau_{i,j} = 1 \tag{6}$$

where E is a set of edges in a given segment graph. We denote the probability of the occurrence of catalytic reaction r by $\theta_r = [0, 1]$. By getting them together, we define a probabilistic function P as follows.

Definition 10 (Probabilistic Component): A probabilistic component $\mathbf{P}$ is a tuple $\langle \mathrm{T}, \Theta \rangle$ where $\mathrm{T} = \{\tau_{i,j} \in [0,1] \mid (i,j) \in \mathrm{E}\}$, $\Theta = \{\theta_r \in [0,1] \mid r \in \mathrm{R}\}$, E is a set of edges in a segment graph and R is a set of catalytic reactions.

Definition 11 (Probabilictic CCRN): Let $\mathbf{C}$ and $\mathbf{P}$ be a CCRN and a probabilistic component respectively. A probabilistic CCRN (P-CCRN) is a tuple of $\langle \mathbf{C}, \mathbf{P} \rangle$.

V. Formulation of P-CCRN Reliability Analysis

This section shows a reliability analysis method by using P-CCRN. To do so, we define states of P-CCRN and represent transitions between two states as a probability function. Finally, we propositionize states of P-CCRN to conduct probabilistic model checking.

A. State Representation

Let U be a set of states included in CCRN. Each of states of given CCRN u can be represented as a pair of states of the user's location S_{seg} and states of SOs' federation S_{fed} denoted by $\langle S_{\text{seg}}, S_{\text{fed}} \rangle$ where $S_{\text{seg}} \in \mathrm{S}$ and $S_{\text{fed}} \in \mathfrak{P}(\mathrm{O})$ (i.e., S_{fed} is a partition set of O). We also assume the independence between two events a user behavior and catalytic reactions' success or failure.

B. Transition Representation

A transition (u, u^{next}) between two states are occurred when a user who has SOs moves along with a directed edge in given segment graph. S_{seg} is changed directly when the user moves and S_{fed} is changed by catalytic reactions of corresponding contexts located at the segment that the user aims to go. We define the function of $r_i(S_{\text{fed}})$ to represent applications of catalytic reactions.

Definition 12 (Catalytic Reaction Application): Let r_i and S_{fed} be a catalytic reaction and a state of SOs' federation respectively. $r_i : \mathfrak{P}(\mathrm{O}) \to \mathfrak{P}(\mathrm{O})$ is a function of catalytic reaction application which is a procedure of following update of given S_{fed}:

$$r_i(S_{\text{fed}}) = \begin{cases} S_{\text{fed}} \setminus \mathit{del}(r_i) \cup \mathit{add}(r_i) & \text{if } \mathit{pre}(r) \subseteq S_{\text{fed}} \\ S_{\text{fed}} & \text{otherwise} \end{cases} \tag{7}$$

When the state of CCRN is $\langle S_{\text{seg}}, S_{\text{fed}} \rangle$ and a transition $(\langle S_{\text{seg}}, S_{\text{fed}} \rangle, \langle S_{\text{seg}}^{\text{next}}, S_{\text{fed}}^{\text{next}} \rangle)$ occurs, $\exists S_{\text{seg}}^{\text{next}}.(S_{\text{seg}}, S_{\text{seg}}^{\text{next}}) \in E$ and $\exists r.(r \in R(S_{\text{seg}}^{\text{next}}, S_{\text{fed}}))$ are selected probabilistically where $R(s, \mathrm{A}) = \{r_i \mid c_i \in F(s) \wedge \mathit{pre}(r_i) \subseteq \mathrm{A}\}$.

C. Assigning the probability of a transition between two states

Now we assign the probability of a transition $(\langle S_{\text{seg}}, S_{\text{fed}} \rangle, \langle S_{\text{seg}}^{\text{next}}, S_{\text{fed}}^{\text{next}} \rangle)$ between two states of given probabilistic CCRN. This probability is represented as $P(\langle S_{\text{seg}}^{\text{next}}, S_{\text{fed}}^{\text{next}} \rangle \mid \langle S_{\text{seg}}, S_{\text{fed}} \rangle)$. Using the assumption of the independence between two events a user behavior and catalytic reactions' success or failure, we can rewrite this probability as follows:

$$\begin{aligned} &P(\langle S_{\text{seg}}^{\text{next}}, S_{\text{fed}}^{\text{next}} \rangle \mid \langle S_{\text{seg}}, S_{\text{fed}} \rangle) \\ &\quad = P(S_{\text{seg}}^{\text{next}} \mid S_{\text{seg}}) P(S_{\text{fed}}^{\text{next}} \mid S_{\text{seg}}^{\text{next}}, S_{\text{fed}}) \end{aligned} \tag{8}$$

$P(S_{\text{seg}}^{\text{next}} \mid S_{\text{seg}})$ can be defined from T directly, namely,

$$P(S_{\text{seg}}^{\text{next}} \mid S_{\text{seg}}) \triangleq \tau_{S_{\text{seg}}, S_{\text{seg}}^{\text{next}}}. \tag{9}$$

On the other hand, $P(S_{\text{fed}}^{\text{next}} \mid S_{\text{seg}}^{\text{next}}, S_{\text{fed}})$ can be defined by several ways. For example, if there are more than one catalytic reaction that can be applied when a user enters into a segment $S_{\text{seg}}^{\text{next}}$ with federated devices S_{fed}, at first, we evaluate the probability of all catalytic reactions independently like we try a coin flip with the number of these reactions of coins and assume that heads are reactions that are applied. In this paper, we give three strategy to deal with this kind of situation.

1) If there are more than one head, we choose one of them uniformly. This represents *mutual exclusion* of concurrent processes among multiple devices.
2) If there are more than one head, we do *not* choose any of these. This assumes that the mutual exclusion does not work and of course this kind of situation should be avoided properly.
3) Let all the catalytic reactions be indexed in order of reaction rate and if there are more than one head, we choose the catalytic reaction with lowest number from them. This represents that fastest catalytic reaction is applied at the highest priority.

In this paper, we use the first strategy to conduct case studies in Section 6. This strategy can be represented as follows:

$$\begin{aligned} &P(S_{\text{fed}}^{\text{next}} \mid S_{\text{seg}}^{\text{next}}, S_{\text{fed}}) = \\ &\begin{cases} \prod_{i \in \mathrm{R}'} (1 - \theta_i) & \text{if } S_{\text{fed}}^{\text{next}} = S_{\text{fed}} \\ \sum_{j=1}^{|\mathrm{R}'|} \sum_{\chi \in X_{ij}} \frac{1}{j} \prod_{k \in \chi} \theta_k \ \prod_{\ell \in \mathrm{R}' \setminus \chi} (1 - \theta_\ell) & \text{otherwise} \end{cases} \\ &\qquad \text{such that } S_{\text{fed}}^{\text{next}} \setminus S_{\text{fed}} = \mathit{add}(r_i) \text{and } r_i \in \mathrm{R}', \\ &\text{where } \mathrm{X}_{ij} = \{i\} \cup \binom{\mathrm{R}' \setminus \{i\}}{j-1} \text{ and } \mathrm{R}' = R(S_{\text{seg}}^{\text{next}}, S_{\text{fed}}). \end{aligned} \tag{10}$$

 ISBN: 978-1-61208-598-2

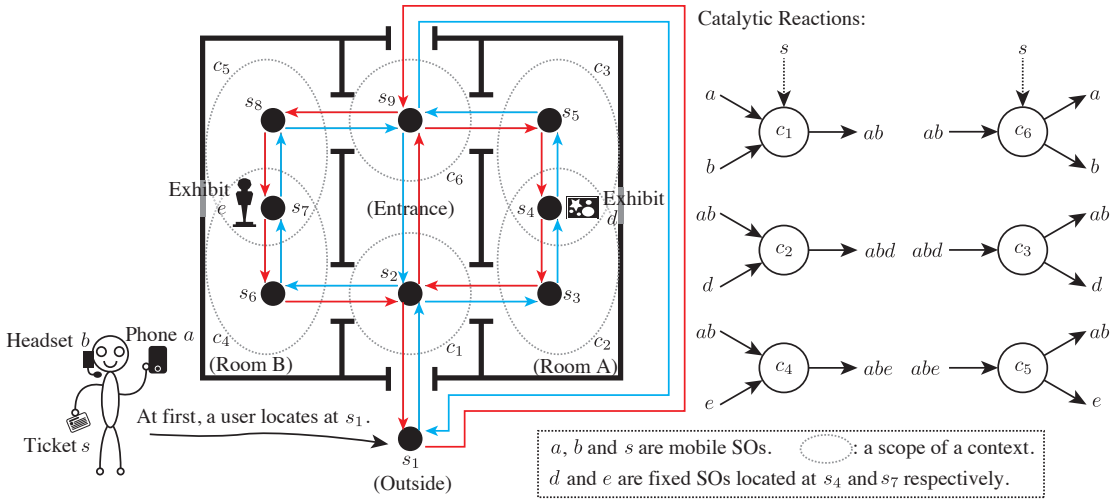


Figure 5. A CCRN assuming a museum

D. Propositionizing

To conduct probabilistic model checking, we assign two kinds of propositions $\mathrm{fed}(\mathrm{O}' \subseteq \mathrm{O})$ and $\mathrm{seg}(s \in \mathrm{S})$ to each states of given P-CCRN. Given a state $\langle S_{\mathrm{seg}}, S_{\mathrm{fed}} \rangle$, semantics of these propositions are defined as follows:

- $\mathrm{seg}(s \in \mathrm{S}) \models \top$ iff $s = S_{\mathrm{seg}}$ (a user locates at segment s)
- $\mathrm{fed}(\mathrm{O}' \subseteq \mathrm{O}) \models \top$ iff $\mathrm{O}' \in S_{\mathrm{fed}}$ (a federation O' exists)

VI. Case Study of Reliability Analysis

We have conducted a case study of a reliability analysis of a given P-CCRN, using probabilistic model checking. We assume that a CCRN is given by the designer who intend to design applications of ubiquitous computing. Here we use a CCRN of a museum example as shown in Figure 5. Left hand side and right hand side of this figure represent the segment graph $\mathbf{G}$ and the catalytic reaction network R of this CCRN respectively. In this example, a user enters the entrance of a museum, carrying a phone a, a headset b and a ticket s. Once the user entered the entrance, the phone a and the headset b are federated by a reaction associated with the scope of c_1, which is triggered by the ticket s. Then, the federated SOs ab are worked as a voice guide of the museum. Next, if the user enters into room A, the federated SO ab and an exhibit d are federated by a reaction associated with the scope of c_2. By the federated SO abd, an explanation of the exhibit d can be provided to the user. After this, the user leaves the room A and the federated SO abd is decomposed and becomes ab again by a reaction associated with the scope of c_3. The similar reactions occur in the room B, which is for an explanation of an exhibit e. If the user leaves one of the exhibition rooms and returns to the entrance, the federated SO ab is decomposed before leaving the museum.

Next we assign the probability to the user movement and catalytic reactions. In Figure 5, every directed edges of the segment graph is colored with blue or red. Blue edges assume the regular route of the museum to tour and red edges assume the opposite (i.e., wrong) way. The user can move along with these edges but here we use parameter $\alpha \in [0, 1]$ to decide how frequent he or she tends to go along with the regular route. More precisely, in every segments, the user chooses blue edges with a probability of α, otherwise, he or she chooses red edges with a probability of $1 - \alpha$. Then, if there are more than one edge after he or she chooses color of edge, he or she chooses an one edge uniformly from them. For all $(i, j) \in \mathrm{E}$, we can set $\tau_{i,j}$ as follows:

$$\tau_{i,j} = \begin{cases} \alpha / |\mathrm{BLUE}_i| & \text{if } \tau_{i,j} \text{ is a blue edge} \\ (1-\alpha) / |\mathrm{RED}_i| & \text{if } \tau_{i,j} \text{ is a red edge} \end{cases} \quad (11)$$

where BLUE_i is a set of $\{(i, j) \in \mathrm{E} \mid (i, j) \text{ is a blue edge}\}$ and RED_i is a set of $\{(i, j) \in \mathrm{E} \mid (i, j) \text{ is a red edge}\}$. In regards to catalytic reactions, we assign the same probability $\beta \in [0, 1]$ to occurrences of all catalytic reactions. Namely, we set $\theta_r = \beta$ for all $r \in \mathrm{R}$.

In this configuration, we conducted an experiment of reliability analysis of P-CCRN. We use PRISM to evaluate the probability of following properties with the bound parameter $k = 20$.

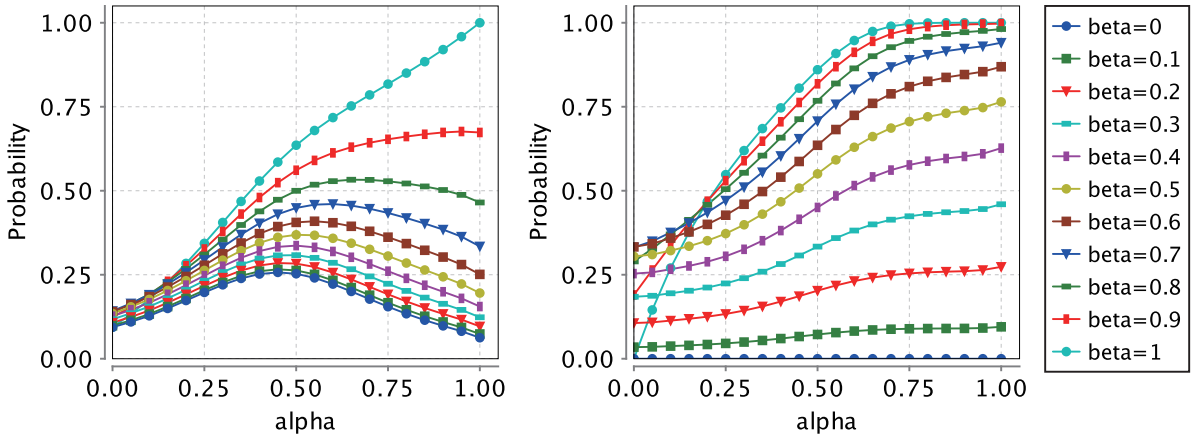
$$\phi_1 = \mathbf{P}_{=?}\left[\mathbf{G}_{\leq k}\left(\neg \mathrm{seg}(s_3) \vee \mathrm{fed}(\{a, b, d\})\right)\right] \quad (12)$$

$$\phi_2 = \mathbf{P}_{=?}[\mathbf{F}_{\leq k}((\mathrm{seg}(s_3) \wedge \mathrm{fed}(\{a, b, d\})) \vee (\mathrm{seg}(s_6) \wedge \mathrm{fed}(\{a, b, e\}))\ )] \quad (13)$$

Intuitively, ϕ_1 means that how frequent the user can be always provided the explanation of exhibition d when he or she is at segment s_2 and ϕ_2 means that how frequent the user can be provided the explanation of exhibition d or e even just once when he or she enters the corresponding room of the exhibition.

Figure 6 shows results of these probability evaluation of property ϕ_1 and ϕ_2 by changing parameters α and β from 0 to 1. When α and β are 1, this is the most ideal case. In other words, the user always moves along with the regular route only and catalytic reactions always react when the conditions of them satisfy. In this case, both of properties ϕ_1 and ϕ_2 are satisfied with a probability of 1. However, if α and β are decreased (i.e., the user behaves unpleasantly and catalytic reactions do not fire even if the conditions of them satisfy), probabilities of properties ϕ_1 and ϕ_2 are also decreased. The

 ISBN: 978-1-61208-598-2



(i) Property ϕ_1 (ii) Property ϕ_2

Figure 6. Results of Experiments

most important aspect of this reliability analysis is that we can evaluate precisely and quantitatively how reliable this kind of UC scenarios are. For a particular example, quantitative evaluation of UC scenarios help us to consider trade-offs between the reliability of UC scenarios and the cost of implementation for the satisfaction of the reliability by changing parameters of probabilities, such as α and β in this case study. In this case, if β is closer to 1, this means we may need more costs for the implementations.

VII. Conclusion

In this paper, we proposed the method of reliability analysis for UC scenarios described by P-CCRN. By our method, we can discuss the reliability of UC scenarios even if these scenarios are in rather practical situation than ideal cases. Reliability analyses are important because these analyses are quantitative, and this means we can discuss about trade-offs between the reliability and the cost for the satisfaction of the reliability. Once we design a UC scenario by P-CCRN, we may actually implement this which usually takes the cost. In that sense, our approach for reliability analysis is not only theoretical but also practical. In future work, we analyze various kinds of UC scenarios assuming more various interferences including possible situations in real places.

Acknowledgment

This work was partly supported by JSPS KAKENHI (S) Grant Number 15H05711.

References

[1] R. Minoda, Y. Tanaka, and S.-i. Minato, "Verifying Scenarios of Proximity-based Federation among Smart Objects through Model Checking," in Proceedings of UBICOMM 2016 The Tenth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, no. c, 2016, pp. 65–71.

[2] R. Minoda and S.-i. Minato, "Efficient Scenario Verification of Proximity-based Federations among Smart Objects through Symbolic Model Checking," in Proceedings of the 7th International Joint Conference on Pervasive and Embedded Computing and Communication Systems (PECCS2017), 2017, pp. 13–21.

[3] A. Cimatti, E. Clarke, and E. Giunchiglia, "Nusmv 2: An opensource tool for symbolic model checking," Computer Aided Verification, vol. 2404, 2002, pp. 359–364.

[4] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of Probabilistic Real-Time Systems," in Proc. 23rd International Conference on Computer Aided Verification (CAV'11), ser. LNCS, G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806. Springer, 2011, pp. 585–591.

[5] S. Guellouz, A. Benzina, M. Khalgui, and G. Frey, "ZiZo : A Complete Tool Chain for the Modeling and Verification of Reconfigurable Function Blocks ZiZo : A Complete Tool Chain for the Modeling and Verification of Reconfigurable Function Blocks," International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, no. c, 2016, pp. 144–151.

[6] R. Alur and D. L. Dill, "A theory of timed automata," Theoretical Computer Science, vol. 126, no. 2, apr 1994, pp. 183–235.

[7] R. Drechsler and U. Kühne, Eds., Formal Modeling and Verification of Cyber-Physical Systems. Wiesbaden: Springer Fachmedien Wiesbaden, 2015.

[8] C. Xu and S. C. Cheung, "Inconsistency Detection and Resolution for Context-aware Middleware Support," Proceedings of the 10th European Software Engineering Conference Held Jointly with 13th ACM SIGSOFT International Symposium on Foundations of Software Engineering, 2005, pp. 336–345.

[9] S. Kauffman, Investigations. Oxford New York: Oxford University Press, 2002.

[10] Y. Tanaka, "Proximity-based federation of smart objects: liberating ubiquitous computing from stereotyped application scenarios," in Knowledge-Based and Intelligent Information and Engineering Systems. Springer, 2010, pp. 14–30.

[11] J. R. Burch, E. M. Clarke, K. L. McMillan, and D. L. Dill, "Sequential circuit verification using symbolic model checking," in Proceedings of the 27th ACM/IEEE Design Automation Conference, ser. DAC '90. New York, NY, USA: ACM, 1990, pp. 46–51.

[12] J. C. Mitchell, V. Shmatikov, and U. Stern, "Finite-state Analysis of SSL 3.0," in Proceedings of the 7th Conference on USENIX Security Symposium - Volume 7, ser. SSYM'98. Berkeley, CA, USA: USENIX Association, 1998, p. 16.

[13] S. A. Kripke, "Semantical Analysis of Modal Logic I Normal Modal Propositional Calculi," Zeitschrift für Mathematische Logik und Grundlagen der Mathematik, vol. 9, no. 5-6, 1963, pp. 67–96.

[14] A. Pnueli, "The temporal logic of programs," 18th Annual Symposium on Foundations of Computer Science (sfcs 1977), 1977, pp. 46–57.

[15] G. Holzmann, "The model checker SPIN," IEEE Transactions on Software Engineering, vol. 23, no. 5, may 1997, pp. 279–295.

[16] J. Magee and J. Kramer, Concurrency State Models and Java Programs. New York, New York, USA: John Wiley and Sons, 1999.

 ISBN: 978-1-61208-598-2